CONTROL METHODS FOR ROBOTIC APPLICATIONS

Lecture notes

Leonid B. Freidovich

(June 9, 2017)

UMEÅ UNIVERSITY

Contents

Part I

Introduction and Kinematics

Chapter 1

Introduction and Preview

We start with a brief introduction and a preview of all the other chapters using a simple academic example.

1.1 Disclosure and Acknowledgments

Below we present mostly a compendium of an excellent comprehensive introduction to mathematical methods for (model-based) control of industrial robotic manipulators based on the following textbook:

M.W. Spong, S. Hutchinson, and M. Vidyasagar, **Robot Modeling and Control**, *John Wiley & Sons*, Inc. ISBN 0-471-64990-2.

To the best of our knowledge, nowadays, this book and

Introduction to robotics: Mechanics and Control by J.J. Craig, *Pearson Education International*, ISBN 0-33-123629-6

are the most commonly used for university engineering education in English on robot control.

It should also be acknowledged that for the preparation of this notes, we have used lecture slides prepared by Professor Anton S. Shiriaev for an advanced course on robot dynamics and control. Using these slides and the textbook by Spong et al., the course has been taught several times by A.S. Shiriaev, L.B. Freidovich, and U. Mettin at Umeå university, Sweden and at the Norwegian University of Science and Technology (NTNU), Norway. This notes have been also used by A. Robertsson at Lund University, Sweden. The current presentation has been shaped by feedback from the instructors and many students. Finally, we would like to acknowledge that many figures have been prepared by Oleg Borisov and Vladislav Gromov with some inspiration from the book by M.W. Spong et al. using Ipe by O. Cheong (http://ipe.otfried.org/).

The first version of these notes has been published by NRO ITMO and therefore this text should be cited as

Leonid B. Freidovich, ***Control Methods for Robotic Applications: Lecture Notes***, Saint Petersburg National Research University of Information Technologies Mechanics and Optics, St. Petersburg, Russia, 2013.

1.2 Industrial Robotic Manipulators

Here, the notion **industrial robot** is understood, following ISO 8373, as

> "an automatically controlled, reprogrammable, multipurpose manipulator programmable in three or more axes, which may be either fixed in place or mobile for use in industrial automation applications,"

where, according to the International Federation of Robotics (IFR),

- *Reprogrammable* means that its programmed executable motions or auxiliary functions may be changed without physical alterations;
- *Multipurpose* means capable of being adapted to a different application with physical alterations;
- *Axis* refer to direction used to specify the robot motion in a linear or rotary mode.

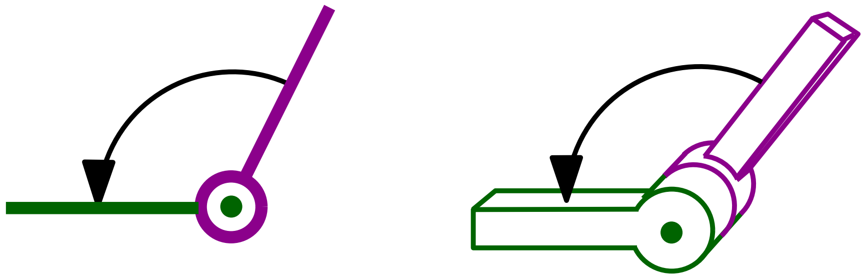
In essence, an industrial robot is an externally powered mechanical device, motion of which is controlled by a computer following a changeable program. The motions must be planned in advance for a particular industrial task, such as an automated assembly. They should be executed repeatably and typically with high precision. To achieve this, one needs to develop a reliable mathematical model for the electro-/hydraulico-/pneumatico-mechanical systems.

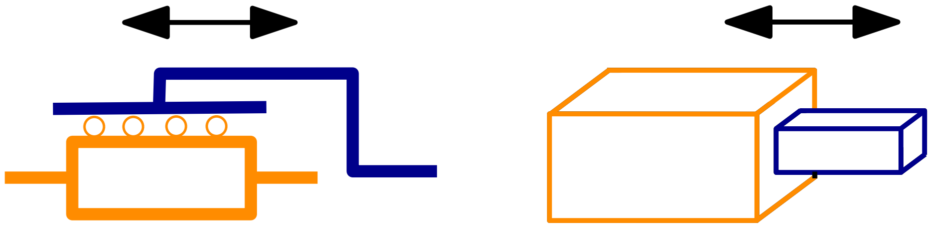
Below, we mostly concentrate on modeling the mechanical part, review basic terminology and useful concepts, describe elements of sensing the environment and self configuration, as well as present basic approaches for control design and most common methods for relevant analysis.

1.3 Terminology and Examples

Mechanical part of a manipulating robot is a kinematic chain, which is called **an arm**. It consists of several (rigid) bodies, called **links**, that are connected by various **joints** and move relative to each other.

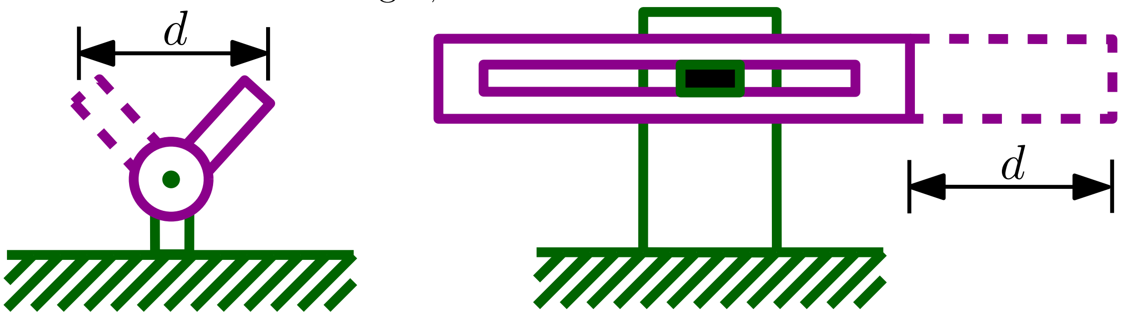
Below (in these notes), without loss of generality, we consider only so-called **single-degree-of-freedom (1-DOF) joints** that are either **revolute** or **prismatic**. Revolute joint means that one link rotates around an axis fixed on the other one and a single angular variable fully defines their relative orientation. Prismatic joint means that one link moves along a straight line fixed on the other one and a single distance variable fully defines their relative displacement that corresponds either to an extension or a retraction. They are schematically shown below.





The other types of joints can be easily modeled by combinations of these two by introducing auxiliary zero-mass zero-length links.

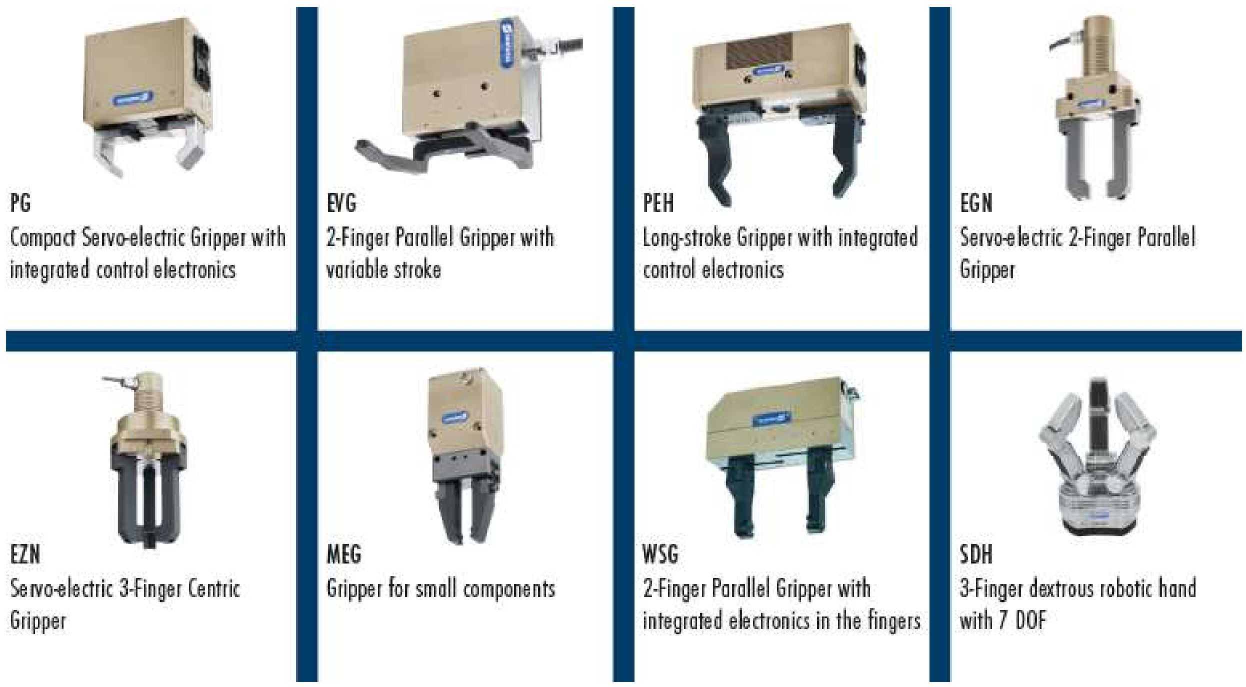
It should be noticed that prismatic joints are much easier for analysis and control design to achieve certain goals of manipulation. However, revolute joints have certain advantages; one of them is illustrated next.



It is clear from the above picture that much larger links are necessary to allow covering the same distance $\boldsymbol{d}$ when they are connected with a revolute joint instead of a prismatic one. As a result, most of the currently used industrial robotic manipulators posses some revolute joints. Note that there are hardly any prismatic joints in nature! Manipulators with only revolute joints are called **articulated**.

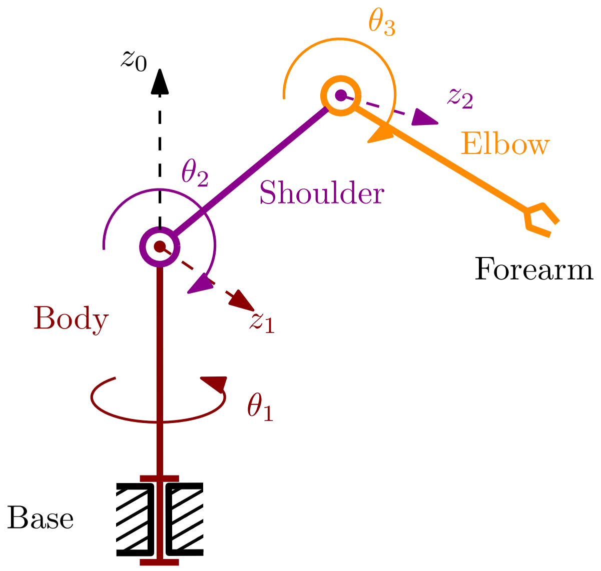
Typically, manipulators are classified by the number of single-degree-of-freedom (1-DOF) joints, that is by the number of mechanical degrees of freedom (DOF), and by the type of joints connecting the first three links counting from the base. For example, an RRP manipulator means that the first joint and the second one are revolute (R), while the third is prismatic (P) and the type of the other joints, if any, is not specified.

At the end of the arm, an industrial manipulator has an end-effector that can be a tool or an active (or passive) gripper. Often, the end-effectors can be changed, either automatically or manually, according to the current task. A few examples of tools and end-effectors are shown below (courtesy of Schunk)

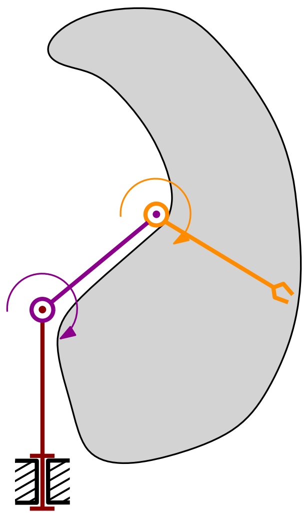

PG
Compact Servo-electric Gripper with integrated control electronics
EVG
2-Finger Parallel Gripper with variable stroke
PEH
Long-stroke Gripper with integrated control electronics
EGN
Servo-electric 2-Finger Parallel Gripper
EZN
Servo-electric 3-Finger Centric Gripper
MEG
Gripper for small components
WSG
2-Finger Parallel Gripper with integrated electronics in the fingers
SDH
3-Finger dextrous robotic hand with 7 DOF

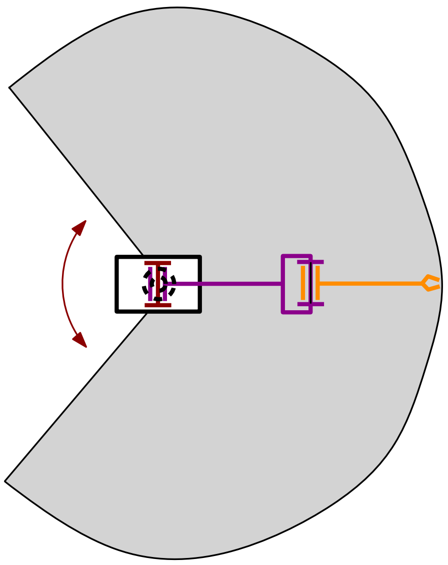
To use a tool, manipulator needs to reach a target object. The region, within which it can operate, is called **workspace**.

Consider, for example, the PBA manipulator by Schunk that is shown below (courtesy of Schunk) together with a symbolic representation for the first **3** DOF.



This is an articulated robotic manipulator, which has 6-DOF and is of RRR-type. The workspace for a hypothetical **3**-DOF ("elbow") manipulator that corresponds to the shown first **3** DOFs is shown below.





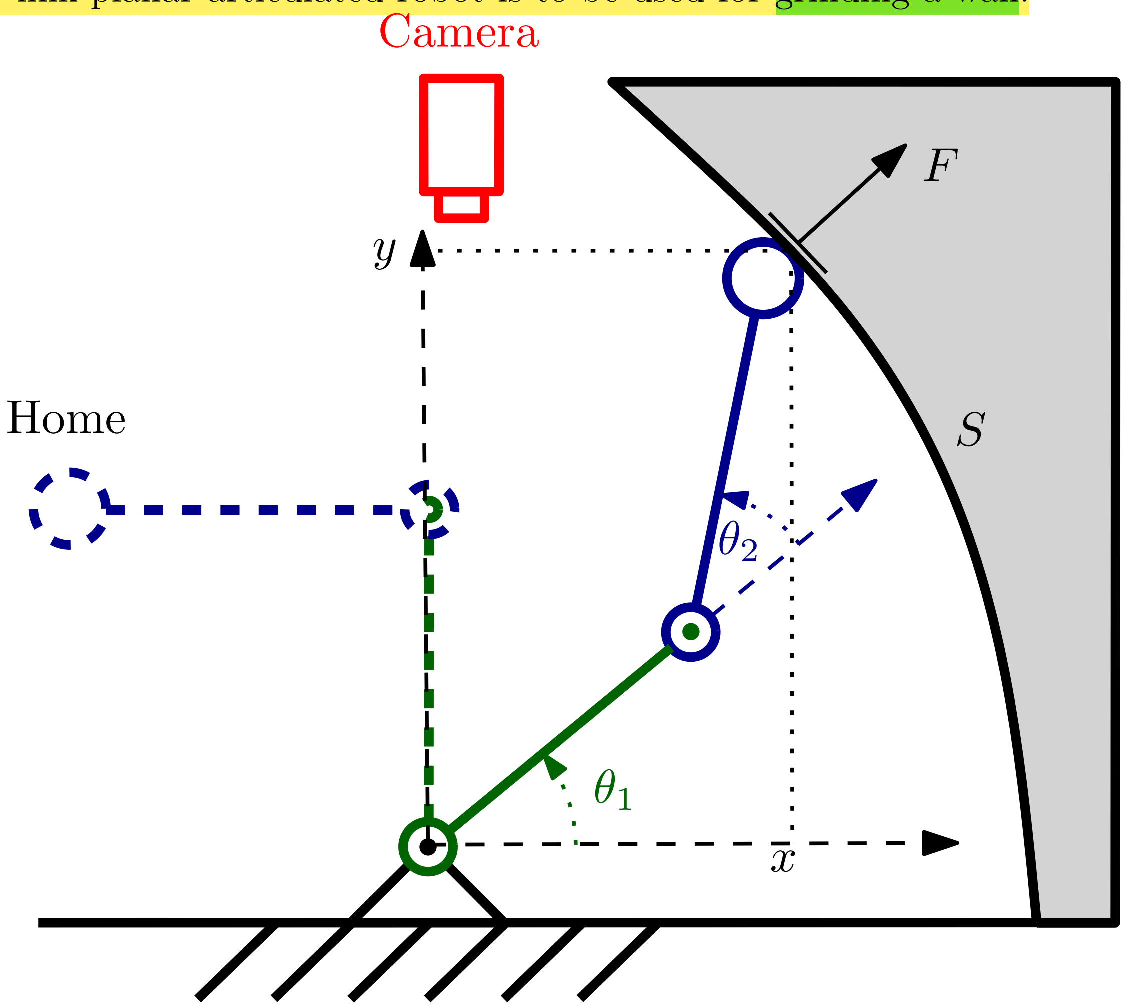
Our goal for the rest of this text is to understand how to force an industrial robotic manipulator to perform a required task. In the next section we introduce a few conceptual problems that should be solved in order to achieve this on a simple example. Each chapter after that is to cover one of these and a few other problems in details.

1.4 Introduction of the Conceptual Problems

The following conceptual problems must be resolved to make a robot succeed in performing a typical task:

- Forward Kinematics (see Chapter 2),
- Inverse Kinematics (see Chapter 2),
- Velocity Kinematics (see Chapter 3),
- Dynamics (see Chapter 5),
- Path Planning and Trajectory Generation (see Chapter 4),
- Motion Control (see Chapters 6 and 7),
- Force Control (see Chapter A),
- Computer Vision and Vision Based Control (see Chapters B and C).

Let us briefly introduce these tasks on the following example: Suppose a two-link planar articulated robot is to be used for grinding a wall.



Let us assume that we can assign the torques applied to the links by the actuators at the two joints. If we manage to produce these torques appropriately, we should be able to control the two corresponding angles. How do we choose these angles to perform grinding?

Obviously, to answer this question, we need to know

- A parametrization of the wall in the plane,
- Configurations of the robot (i.e. the values of the angles) that correspond to touching the wall.

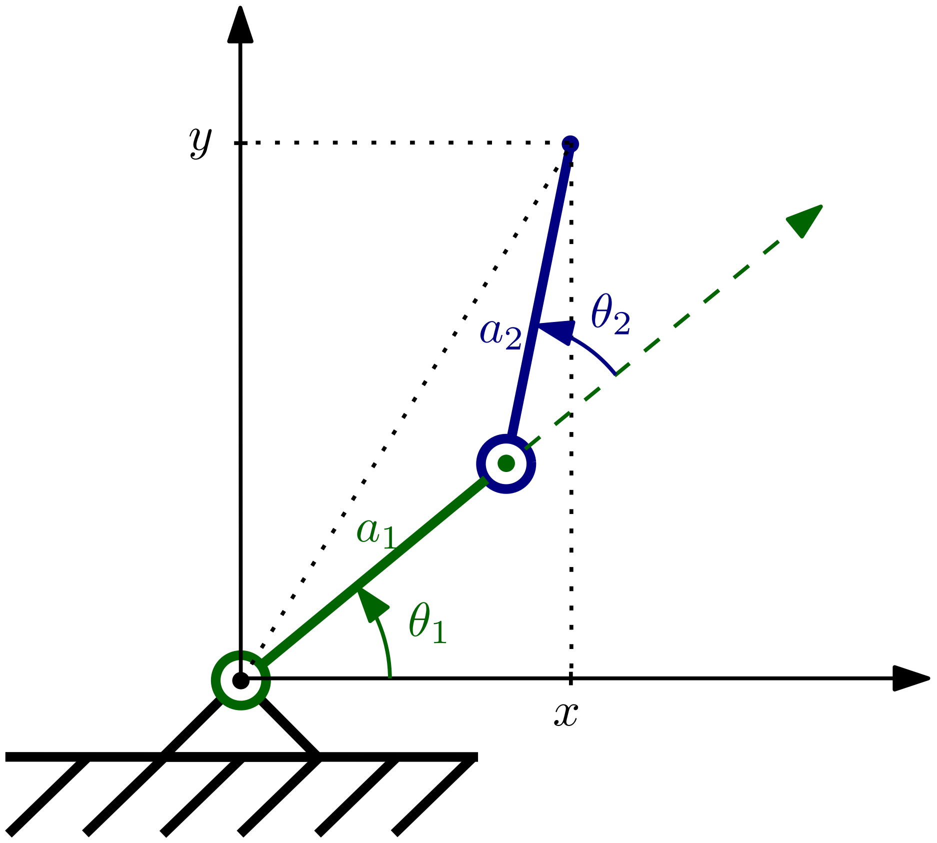
To know the result of assigning the joint angles in terms of the position and the orientation of the end-effector one must solve the **forward kinematics** problem: Find the position (x, y) of the end-effector and the orientation of the tool as functions of joint angles (θ_1, θ_2).

At the same time, to define the control task, we need to know the required values of the joint angles to touch the wall, i.e. we must solve the **inverse kinematics** problem: Find the angles (θ_1, θ_2) as functions of coordinates of the end-effector (x, y).

Let us proceed with these two tasks.

1.4.1 Forward Kinematics

Denote by a_1 and a_2 the lengths of the links; the drawings needed to compute the position of the end-effector is given next.



Let us denote by (x_1, y_1) and (x_2, y_2) coordinates of the endings of the links with respect to their beginnings. Then, using elementary geometry

$$\begin{aligned} x &= x_1 + x_2 = a_1 \cos\theta_1 + a_2 \cos(\theta_1 + \theta_2) \\ y &= y_1 + y_2 = a_1 \sin\theta_1 + a_2 \sin(\theta_1 + \theta_2) \end{aligned}$$

It is also quite easy to find the orientation of the tool assuming it is attached to the ending of the last link and directed along it. Clearly, it should make the angle

$$\theta = \theta_1 + \theta_2$$

with the horizontal line. However, we will soon discover that it is more convenient to define the tool orientation (especially in 3D) as a direction of an appropriate unit vector, $\vec{x}_2$, together with a couple of the other vectors, $\vec{y}_2$ and $\vec{z}_2$, defining an (orthogonal) coordinate frame rigidly attached to the tool. In our case, we can easily compute a coordinate representation of the direction vector $\vec{x}_2$ with respect to the fixed frame, which is rigidly attached to the base:

$$x_2^0 = \left[\ \cos(\theta),\ \sin(\theta),\ 0\right]^T$$

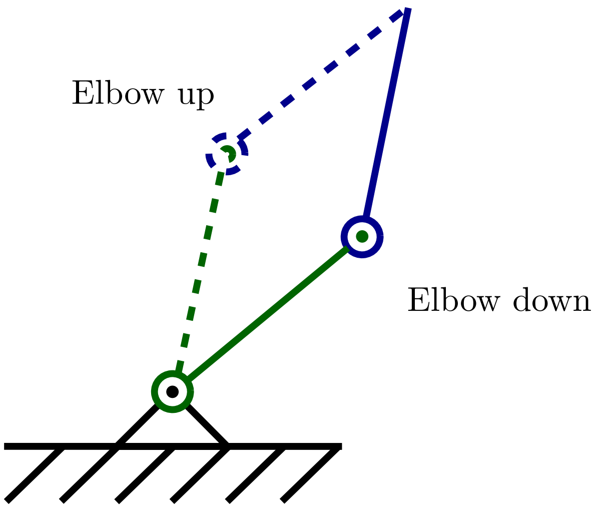
Similarly, we can define coordinate representations the other two vectors:

$$\begin{aligned} y_2^0 &= \left[-\sin(\theta),\ \cos(\theta),\ 0\right]^T \\ z_2^0 &= \left[0,\ 0,\ 1\right]^T \end{aligned}$$

The components of these three representations of direction vectors define a special 3×3 matrix, called **rotation matrix**, that is even more useful for specification of the end-effector orientation.

For a more complex multi-link not planar robot finding appropriate triangles in the 3D-space in order to use similar arguments is hardly possible. This is why we will adopt a more systematic approach in Chapter 2: We will attach to each link a coordinate system (frame) and compute pairwise transformations between them. After that, we will be able to inductively compute the position and the orientation of the end-effector moving step by step from the base in a straightforward way.

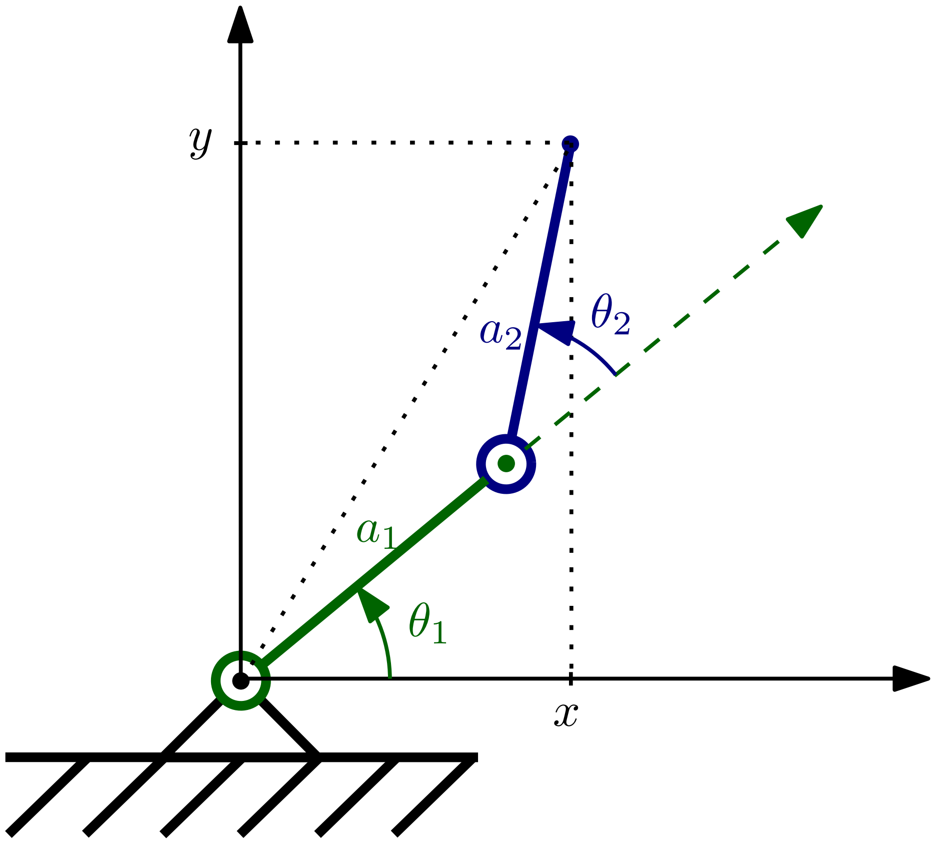
Returning to our example, it is useful to note that if the tool position (x, y) is given, but the orientation is not defined, then, except for a couple of particular cases, there are two possible configurations, called "Elbow Up" and "Elbow Down". They are shown below.



Now, from the solution of the forward kinematics problem we know the result of assigning particular values for the joint angles on the position and orientation of the tool for wall grinding in our example. To find the required angles from a description of the wall we need to solve the inverse problem, which is often harder.

1.4.2 Inverse Kinematics

In our simple case, instead of trying to solve the computed above equations for the forward kinematics we can just use the cosine theorem for the triangle made by the links



We immediately obtain

$$\cos\theta_2 = \frac{x^2+y^2-a_1^2-a_2^2}{2a_1a_2} =: D, \quad \left\{\Rightarrow \sin\theta_2 = \pm\sqrt{1-D^2}\right\}$$

$$\Rightarrow \quad \theta_2 = \tan^{-1}\left(\frac{\sin\theta_2}{\cos\theta_2}\right) = \tan^{-1}\left(\frac{\pm\sqrt{1-D^2}}{D}\right)$$

$$\Rightarrow \quad \theta_1 = \tan^{-1}\left(\frac{y}{x}\right) - \tan^{-1}\left(\frac{a_2\sin\theta_2}{a_1+a_2\cos\theta_2}\right)$$

In a more complex case, finding a solution of the inverse kinematics problem may be challenging and may require using numerical methods. We will discuss this in Chapter 2.

1.4.3 Velocity Kinematics

In many situations it is necessary to compute (and to assign via an appropriate feedback action) not only the position and orientation but also the velocity of the tool. The corresponding forward problem and, surprisingly, the inverse problem are simpler.

Clearly, the geometrical relations between (x, y) and (θ_1, θ_2)

$$x = a_1\cos\theta_1 + a_2\cos(\theta_1+\theta_2), \quad y = a_1\sin\theta_1 + a_2\sin(\theta_1+\theta_2)$$

immediately imply the following relations between the velocities

$$\begin{array}{rcl} \frac{d}{dt}x(t) & = & -a_1\sin\theta_1\frac{d}{dt}\theta_1 - a_2\sin(\theta_1+\theta_2)\left(\frac{d}{dt}\theta_1+\frac{d}{dt}\theta_2\right) \\ \frac{d}{dt}y(t) & = & a_1\cos\theta_1\frac{d}{dt}\theta_1 + a_2\cos(\theta_1+\theta_2)\left(\frac{d}{dt}\theta_1+\frac{d}{dt}\theta_2\right) \end{array}$$

In a more compact (matrix) form this relation can be written as follows

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = \underbrace{\left[\begin{array}{c|c} -a_1\sin\theta_1 - a_2\sin(\theta_1+\theta_2) & -a_2\sin(\theta_1+\theta_2) \\ a_1\cos\theta_1 + a_2\cos(\theta_1+\theta_2) & a_2\cos(\theta_1+\theta_2)\end{array}\right]}_{=:\, J(\theta_1,\theta_2)} \begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\end{bmatrix}$$

The matrix $J(\cdot)$ above is called **Jacobian** of the manipulator. It is introduced for a general case in Chapter 3. Whenever this matrix is not singular, the computed relation between the joint velocities and the tool velocity

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = J(\theta_1,\theta_2)\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\end{bmatrix}$$

allows to compute the joint velocities $\dot{\theta}_1(t)$ and $\dot{\theta}_2(t)$ that are required to achieve a particular velocity of the tool.

Indeed, $\dot{\theta}_1(t)$ and $\dot{\theta}_2(t)$ are found by

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = J^{-1}(\theta_1, \theta_2) \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}$$

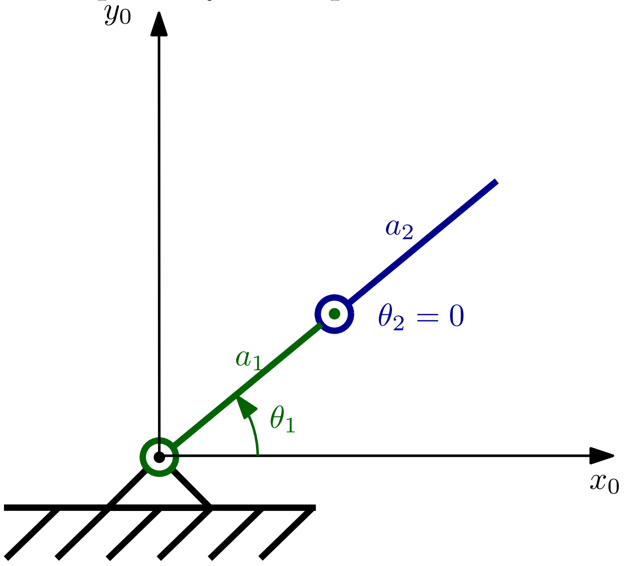
with

$$J^{-1}(\theta_1, \theta_2) = \frac{-1}{a_1 a_2 \sin(\theta_2)} \left[\begin{array}{c|c} a_2 \cos(\theta_1 + \theta_2) & a_2 \sin(\theta_1 + \theta_2) \\ a_1 \cos\theta_1 + a_2 \cos(\theta_1 + \theta_2) & a_1 \sin\theta_1 + a_2 \sin(\theta_1 + \theta_2) \end{array} \right]$$

It is clear that the inverse of the Jacobian is not defined if and only if

$$\sin\theta_2 = 0 \quad \{ \Rightarrow \ \theta_2 = \pi \cdot k, \ k = \cdots - 1, \ 0, \ 1, \ldots \}$$

The corresponding postures of the manipulator are called **Singular Configurations**; in our example they correspond to the case of full arm extension:



If $\theta_2 = 0$, then the Jacobian $J(\theta_1, \theta_2)$ looses the rank and cannot be inverted. It means that the tool velocity, defining direction of motion of the end-effector, cannot be any, but should belong to a particular line irrespective of the joint velocities. Since near the singularity very large joint velocities are needed to direct the motions of the tool, it is common practice to avoid motions in their vicinity.

Now, with a solution for kinematics at hand, one can define a task of achieving certain time evolution of the joint angles. In order to understand how to induce the corresponding motion with the help of the actuators, equations of dynamics should be considered.

1.4.4 Dynamics of Motions

Clearly, position, orientation, and velocity kinematics define motions of the manipulator irrespective of an actuation and laws of mechanics. However, in order to understand how to introduce the effect of control action and to describe feasible trajectories, equations of motion should be derived. In Chapter 5 we will introduce a systematic approach for deriving such equations using formalism of the Euler-Lagrange equations.

With their help, under certain assumptions, one would easily derive for example the following equations describing dynamics of our double pendulum when it moves towards the wall

$$\begin{bmatrix} p_1 + p_2 + 2p_3\cos\theta_2 & p_2 + p_3\cos\theta_2 \\ p_2 + p_3\cos\theta_2 & p_2 \end{bmatrix}\begin{bmatrix}\ddot\theta_1\\ \ddot\theta_2\end{bmatrix} + p_3\sin\theta_2\begin{bmatrix} -\dot\theta_2 & -\dot\theta_2-\dot\theta_1 \\ \dot\theta_1 & 0\end{bmatrix}\begin{bmatrix}\dot\theta_1\\ \dot\theta_2\end{bmatrix} + \begin{bmatrix} p_4 g\cos\theta_1 + p_5 g\cos(\theta_1+\theta_2) \\ p_5 g\cos(\theta_1+\theta_2)\end{bmatrix} = \begin{bmatrix} u_1\\ u_2\end{bmatrix}$$

where

- p_1-p_5 are constants defined by physical parameters of the links (masses, moments of inertia, lengths, and locations of the centers of masses);
- u_1, u_2 are control torques applied at the joints.

These equations can be used to plan a required time evolution of the joint angles necessary for the tool to follow the wall together with the required torques. Also, they can be used to propose and to rigorously justify an appropriate feedback control law to apply in order to achieve the target motion using the sensors and the actuators installed on the robot.

It should be noticed, however, that just barely following the wall is not enough for grinding. One needs to introduce specifications and to plan the motion with certain care. It is also important to realize that the equations above are derived ignoring the interaction forces that may include impact forces, appearing when the tool touches the wall, as well as friction and resistance forces, appearing during execution of grinding. So, appropriate modifications may be required.

1.4.5 Path Planning and Trajectory Generation

Planning a motion to perform a task, such as grinding a wall, is typically done in two steps. First a path is planned and after that it is enhanced with certain velocities to follow. The next should be noticed.

- **Path** is a curve in the configuration space of the robot: It defines geometric evolution of the tool (to follow the wall) without timing.

- **Trajectory** is the path augmented with the information on velocities, with which the tool and the links will travel along the path.

- Planning trajectory might also include specifications on the accelerations and the induced interaction forces with the wall.

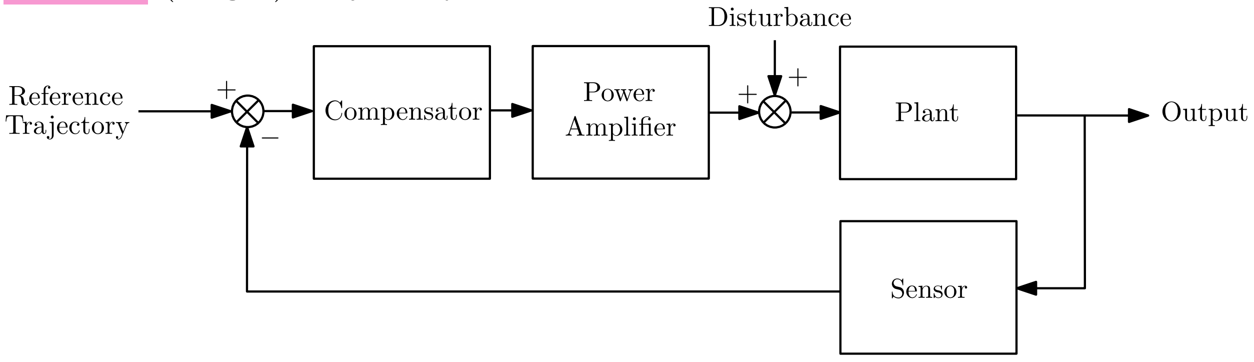
Chapter 4 is devoted to an introductory discussion on the issue, covering the basic approach relying only on kinematics. We will not discuss here modern approaches based on dynamic models that are currently under active development.

As soon as a target trajectory is generated, one needs to suggest a feedback control law to implement it.

1.4.6 Motion Control

Several mathematics-based and engineering-intuition-based approaches are discussed in Chapter 6 and Chapter 7 respectively.

Possible basic structure of a feedback control system assuming a generated reference (target) trajectory is shown below



1.4.7 Force Feedback

It is clear that as soon as the tool touches the wall the equations of motion for the free robotic arm presented above are invalid. They should be modified by introducing the wall reaction forces acting on the tool. These forces should be controlled in order to prevent the tool from breaking and to achieve necessary quality of grinding. An introduction to the current commonly used approaches is presented in Chapter A.

1.4.8 Computer Vision and Vision Based Control

Performing the task of grinding the wall requires the control system to react on various factors detectable by cameras. Also, the wall profile may be unknown in advance and detected by a camera. In this case, a trajectory and an appropriate feedback action should be generated "on fly" or using camera signals for feedback. Signals from cameras cannot be used in feedback directly as, say, signals from encoders giving the values of the joint angles. The obtained video images must be processed. The issue is touched in Chapters B and C.

1.5 Conclusions

In this Chapter we have introduced some basic terminology used in robotics and have briefly given a preview of the challenges and of the contents of the next Chapters. We proceed next with introducing necessary mathematical tools and discussing the way of solving the problem of control design for industrial robotic manipulators.

Chapter 2

Kinematics

In order to describe kinematics of a robotic arm of arbitrary complexity we proceed as follows. The arm is a chain of links, which are assumed to be rigid bodies, connected at (1-DOF) joints. Starting from the base of the manipulator, at each joint two parts of the arm are moving with respect to each other. The problem of kinematics is essentially to describe positions and velocities of every point of each link. As will be clear later, we are mostly interested in the mutual orientations of the links and positions of the points associated with the joints, with the centers of masses of the links, and with the end point of the tool and the orientation of the end-effector.

A systematic procedure that we will follow is to attach to each rigid body a coordinate system, called a **frame**. Each frame consists of an origin and **2** axes (for planar motions) or **3** axes (for general motions). Every point of each rigid body is uniquely defined by the coordinates in the attached frame that do not change irrespective to whether the body is moving or not. However, whenever the arm is set in motion, the positions of the origins of the frames and their pairwise orientations are changing. As soon as we define such relations between each couple of the frames, we will be able to easily describe positions and velocities of all the points.

So, let us start with introducing necessary notation and appropriate mathematical description of so-called rigid motions and homogeneous transformations in **2**D and **3**D.

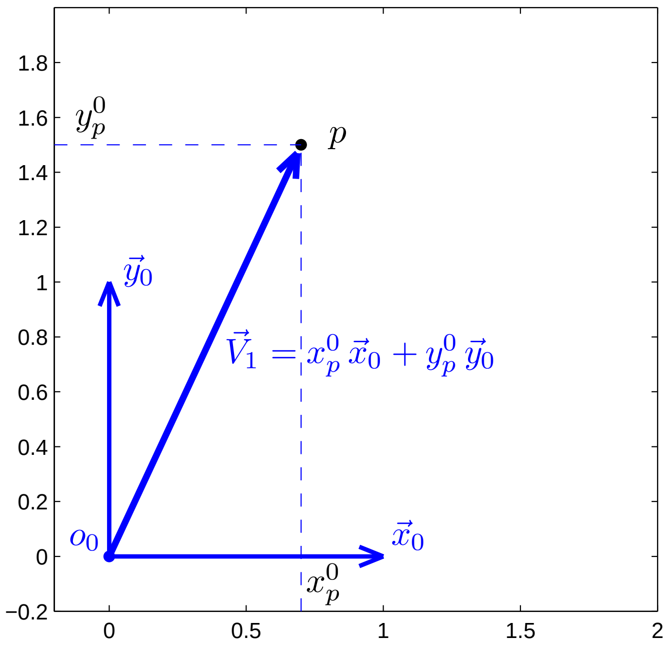
2.1 Rigid Motions & Homogeneous Transformations

We need to agree on how to describe and to denote frames (coordinate systems), points, and (free) vectors.

2.1.1 Frames, Points, and Vectors

Coordinates of points and vectors (in 2D)

Consider a point $\boldsymbol{p}$ in plane (in $\mathbb{R}^2$) and introduce a frame, that is another special point called the origin $\boldsymbol{o_0}$ and two mutually orthogonal axes $\boldsymbol{x_0}$ and $\boldsymbol{y_0}$ intersecting at $\boldsymbol{o_0}$. Orthogonal projection of the point into the axes define its coordinates.



The point $\boldsymbol{p}$ can be associated with the vector $\vec{\boldsymbol{V}}_1$ from $\boldsymbol{o_0}$ to $\boldsymbol{p}$, as shown above. Let $\boldsymbol{x_p^0}$ denote $\boldsymbol{x}$-coordinate and $\boldsymbol{y_p^0}$ denote $\boldsymbol{y}$-coordinate of the point $\boldsymbol{p}$ respectively in $(\boldsymbol{o_0}, \boldsymbol{x_0}, \boldsymbol{y_0})$-frame, called below for brevity the $\mathbf{0}$-frame. Note that if we denote by $\vec{\boldsymbol{x}}_0$ and $\vec{\boldsymbol{y}}_0$ the unite vectors defining directions of the axes, we have

$$\vec{\boldsymbol{V}}_1 = \overrightarrow{\boldsymbol{o_0 p}} = \boldsymbol{x_p^0}\,\vec{\boldsymbol{x}}_0 + \boldsymbol{y_p^0}\,\vec{\boldsymbol{y}}_0$$

that means that one should move from the point $\boldsymbol{o_0}$ exactly $\boldsymbol{x_p^0}$ units in the direction of $\boldsymbol{x_0}$ axis and $\boldsymbol{y_p^0}$ units in the direction of $\boldsymbol{y_0}$ axis to reach the point $\boldsymbol{p}$. Therefore, the two numbers $\boldsymbol{x_p^0}$ and $\boldsymbol{y_p^0}$ can be considered as coordinates of both, the point $\boldsymbol{p}$ and of the vector $\vec{\boldsymbol{V}}_1$ with respect to $\mathbf{0}$-frame. Note that vectors literally define displacement of the number of units defined by their magnitude in the specified direction.

Magnitude of the vector can be computed using the coordinates in any frame

$$\text{magnitude of } \vec{\boldsymbol{V}}_1 = |\vec{\boldsymbol{V}}_1| = \sqrt{\left(\boldsymbol{x_p^0}\right)^2 + \left(\boldsymbol{y_p^0}\right)^2}$$

but it is a specification of the vector that is frame-independent as well as the direction of the vector that can be computed as

$$\text{direction of } \vec{V}_1 = \frac{\vec{V}_1}{|\vec{V}_1|} = \frac{x_p^0}{|\vec{V}_1|}\,\vec{x}_0 + \frac{y_p^0}{|\vec{V}_1|}\,\vec{y}_0$$

Analogous definitions are valid in **3**D.

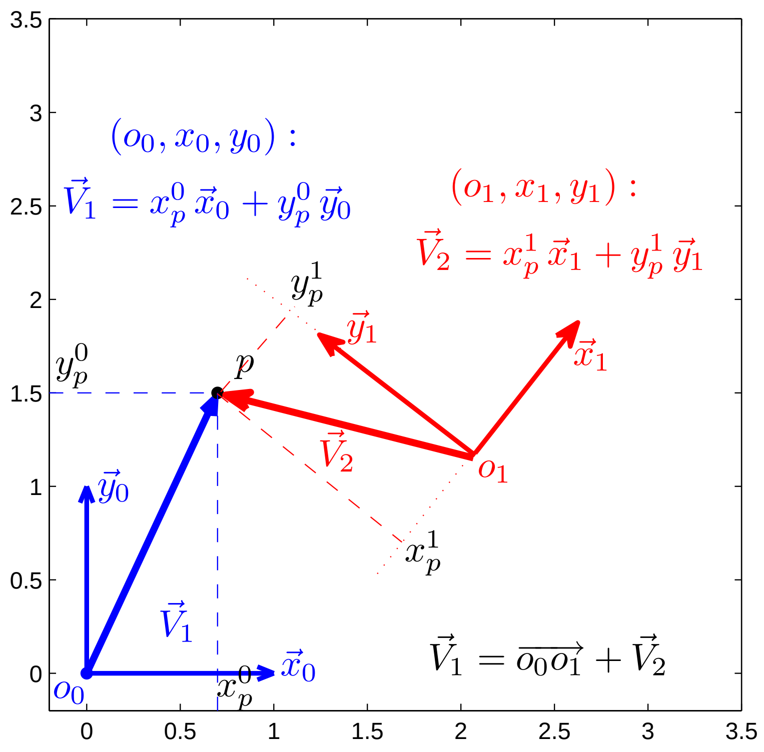
Vectors with magnitude equal to **1** are called unit vectors and are used to specify directions.

Let us summarize our notation:

- o_0 is the origin of the (o_0, x_0, y_0)-frame, to be called the **0**-frame
- $\vec{x}_0$ denotes the unit vector defining the direction of the x_0 axis
- $\vec{y}_0$ denotes the unit vector defining the direction of the y_0 axis
- x_p^0 denotes the x-coordinate of point p in the (o_0, x_0, y_0)-frame
- y_p^0 denotes the y-coordinate of point p in the (o_0, x_0, y_0)-frame
- $\vec{V}_1 = \overrightarrow{o_0 p}$ denotes the vector indicating the displacement from o_0 to p

Representations in different frames

Suppose now there are **0**-frame (o_0, x_0, y_0) and **1**-frame (o_1, x_1, y_1):



The point $\boldsymbol{p}$ can now be associated not only with vector $\vec{\boldsymbol{V}}_1 = \overrightarrow{\boldsymbol{o}_0\boldsymbol{p}}$ but also with vector $\vec{\boldsymbol{V}}_2 = \overrightarrow{\boldsymbol{o}_1\boldsymbol{p}}$ in a similar way. Now, we have

$$\vec{\boldsymbol{V}}_2 = \boldsymbol{x}_p^1\,\vec{\boldsymbol{x}}_1 + \boldsymbol{y}_p^1\,\vec{\boldsymbol{y}}_1$$

It is clear than that it is wise to use notation identifying frames. From now on, we will use the following notation for the two representations of $\boldsymbol{p}$

$$\boldsymbol{p}^0 = \boldsymbol{V}_1^0 = \begin{bmatrix} \boldsymbol{x}_p^0 \\ \boldsymbol{y}_p^0 \end{bmatrix} \qquad \text{and} \qquad \boldsymbol{p}^1 = \boldsymbol{V}_2^1 = \begin{bmatrix} \boldsymbol{x}_p^1 \\ \boldsymbol{y}_p^1 \end{bmatrix}$$

associated with the respective frames ($\mathbf{0}$ and $\mathbf{1}$).

Let us remind the reader the following.

- **A point** (material point) corresponds to a particular location in the space or within a rigid body.
- **A point** has different representations (coordinates) in different frames that can be associated with certain vectors.
- A **vector** (free vector) is completely defined by **magnitude** $|\vec{\boldsymbol{V}}|$ and **direction** $\frac{\vec{\boldsymbol{V}}}{|\vec{\boldsymbol{V}}|}$ and can be also represented by sets of coordinates combined into columns $\boldsymbol{V}^0$ or $\boldsymbol{V}^1$, e.g. in 2D: $\boldsymbol{x}_0^0 = \boldsymbol{x}_1^1 = \begin{bmatrix}\mathbf{1}\\\mathbf{0}\end{bmatrix}$, $\boldsymbol{y}_0^0 = \boldsymbol{y}_1^1 = \begin{bmatrix}\mathbf{0}\\\mathbf{1}\end{bmatrix}$.. It can be interpreted as a directed displacement.
- **Vectors** with the same direction and magnitude are identical (they are not attached to particular points that may be used to define them!)

Vectors $\vec{\boldsymbol{V}}_1 = \overrightarrow{\boldsymbol{o}_0\boldsymbol{p}}$ and $\vec{\boldsymbol{V}}_2 = \overrightarrow{\boldsymbol{o}_1\boldsymbol{p}}$ represent the same point $\boldsymbol{p}$ with respect to different frames ($\boldsymbol{V}_1^0$ and $\boldsymbol{V}_2^1$ defined above). It is clear that $\vec{\boldsymbol{V}}_1 = \overrightarrow{\boldsymbol{o}_0\boldsymbol{o}_1} + \vec{\boldsymbol{V}}_2$; however, to define a relation between $\boldsymbol{V}_1^0$ and $\boldsymbol{V}_2^1$, we need to take into account relative orientation of the two frames. Hence, it is clear that we need

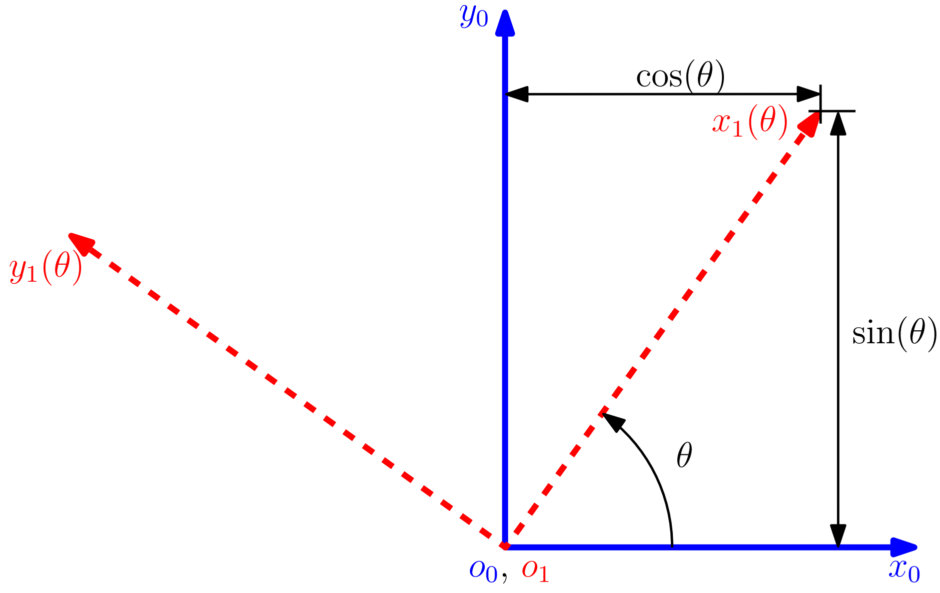
1. To find representations of the unite vectors defining one frame with respect to the other, i.e. to specify a rotation defining relative orientation of the two frames;
2. To find the relation between the origins $\boldsymbol{o}_0$ and $\boldsymbol{o}_1$ defining a relative translation of the frames, associated with the vector $\overrightarrow{\boldsymbol{o}_0\boldsymbol{o}_1}$.

Rotations are defined next.

2.1.2 Rotations in 2D and 3D

Rotations in 2D

Obviously, rotations in plane can be defined by a single angle. Suppose we have a fixed **0**-frame and a rotating **1**-frame with coinciding origins.



To find an appropriate way to parametrize rotations in **2**D, we can use simple trigonometry to find coordinates of the points defined by $\vec{\boldsymbol{x}}_1(\boldsymbol{\theta})$ and $\vec{\boldsymbol{y}}_1(\boldsymbol{\theta})$ giving directions of the moving axes $\boldsymbol{x}_1$ and $\boldsymbol{y}_1$ with respect to **0**-frame for an arbitrary value of the angle $\boldsymbol{\theta}$ defining relative rotation of the two frames.

It is clear from the picture that $\vec{\boldsymbol{x}}_1(\boldsymbol{\theta}) = \mathbf{cos}(\boldsymbol{\theta})\vec{\boldsymbol{x}}_0 + \mathbf{sin}(\boldsymbol{\theta})\vec{\boldsymbol{y}}_0$ and therefore in **0**-frame $\vec{\boldsymbol{x}}_1(\boldsymbol{\theta})$ and $\vec{\boldsymbol{y}}_1(\boldsymbol{\theta})$ represented as

$$\boldsymbol{x}_1^0(\boldsymbol{\theta}) = \begin{bmatrix} \mathbf{cos}(\boldsymbol{\theta}) \\ \mathbf{sin}(\boldsymbol{\theta}) \end{bmatrix}, \qquad \boldsymbol{y}_1^0(\boldsymbol{\theta}) = \boldsymbol{x}_1^0\left(\boldsymbol{\theta}+\frac{\pi}{2}\right) = \begin{bmatrix} \mathbf{cos}\left(\boldsymbol{\theta}+\frac{\pi}{2}\right) \\ \mathbf{sin}\left(\boldsymbol{\theta}+\frac{\pi}{2}\right) \end{bmatrix} = \begin{bmatrix} -\mathbf{sin}(\boldsymbol{\theta}) \\ \mathbf{cos}(\boldsymbol{\theta}) \end{bmatrix}$$

Now, for two representations of a point $\boldsymbol{p}$, rigidly attached to the moving **1**-frame, i.e. its coordinates are changing in **0**-frame and are fixed in **1**-frame

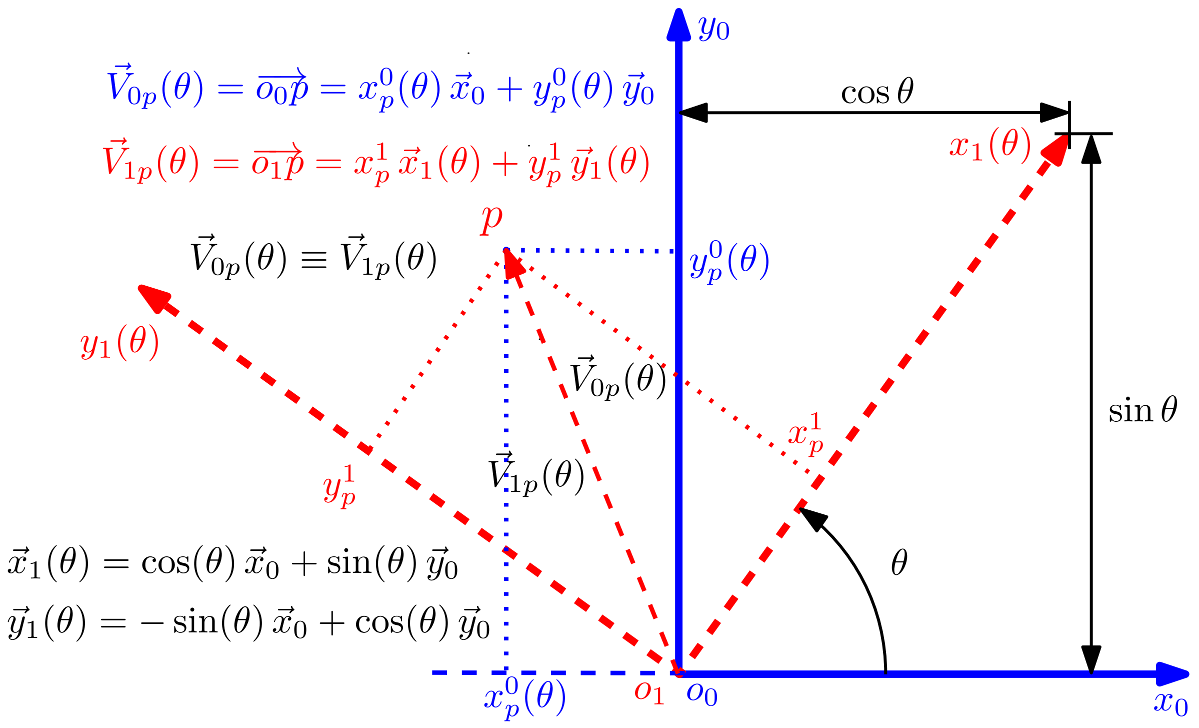
$$\boldsymbol{p}^0(\boldsymbol{\theta}) = \boldsymbol{V}_{0p}^0(\boldsymbol{\theta}) = \begin{bmatrix} \boldsymbol{x}_p^0(\boldsymbol{\theta}) \\ \boldsymbol{y}_p^0(\boldsymbol{\theta}) \end{bmatrix} \qquad \text{and} \qquad \boldsymbol{p}^1 = \boldsymbol{V}_{1p}^1 = \begin{bmatrix} \boldsymbol{x}_p^1 \\ \boldsymbol{y}_p^1 \end{bmatrix}$$

defined by vectors

$$\vec{\boldsymbol{V}}_{0p} = \overrightarrow{\boldsymbol{o}_0\boldsymbol{p}} = \boldsymbol{x}_p^0(\boldsymbol{\theta})\,\vec{\boldsymbol{x}}_0 + \boldsymbol{y}_p^0(\boldsymbol{\theta})\,\vec{\boldsymbol{y}}_0 \quad \text{and} \quad \vec{\boldsymbol{V}}_{1p} = \overrightarrow{\boldsymbol{o}_1\boldsymbol{p}} = \boldsymbol{x}_p^1\,\vec{\boldsymbol{x}}_1(\boldsymbol{\theta}) + \boldsymbol{y}_p^1\,\vec{\boldsymbol{y}}_1(\boldsymbol{\theta})$$

that must be identical since the origins coincide ($\overrightarrow{\boldsymbol{o}_0\boldsymbol{o}_1} = \vec{\boldsymbol{0}}$), we have, after substituting the expressions for $\vec{\boldsymbol{x}}_1(\boldsymbol{\theta})$ and $\vec{\boldsymbol{y}}_1(\boldsymbol{\theta})$, the following

$$\vec{\boldsymbol{V}}_{1p} = \boldsymbol{x}_p^1\left(\mathbf{cos}(\boldsymbol{\theta})\,\vec{\boldsymbol{x}}_0 + \mathbf{sin}(\boldsymbol{\theta})\,\vec{\boldsymbol{y}}_0\right) + \boldsymbol{y}_p^1\left(-\mathbf{sin}(\boldsymbol{\theta})\,\vec{\boldsymbol{x}}_0 + \mathbf{cos}(\boldsymbol{\theta})\,\vec{\boldsymbol{y}}_0\right) = \vec{\boldsymbol{V}}_{0p}$$


$\vec{V}_{0p}(\theta) = \overrightarrow{o_0p} = x_p^0(\theta)\,\vec{x}_0 + y_p^0(\theta)\,\vec{y}_0$
$\vec{V}_{1p}(\theta) = \overrightarrow{o_1p} = x_p^1\,\vec{x}_1(\theta) + y_p^1\,\vec{y}_1(\theta)$
$\vec{V}_{0p}(\theta) \equiv \vec{V}_{1p}(\theta)$
p
y_0
$\cos\theta$
$x_1(\theta)$
$y_p^0(\theta)$
$y_1(\theta)$
$\vec{V}_{0p}(\theta)$
x_p^1
$\sin\theta$
y_p^1
$\vec{V}_{1p}(\theta)$
θ
$\vec{x}_1(\theta) = \cos(\theta)\,\vec{x}_0 + \sin(\theta)\,\vec{y}_0$
$\vec{y}_1(\theta) = -\sin(\theta)\,\vec{x}_0 + \cos(\theta)\,\vec{y}_0$
$x_p^0(\theta)$
o_1
o_0
x_0

It follows that

$$\boldsymbol{x_p^0(\theta) = x_p^1\cos(\theta) - y_p^1\sin(\theta)} \quad \text{and} \quad \boldsymbol{y_p^0(\theta) = x_p^1\sin(\theta) + y_p^1\cos(\theta)}$$

and writing this in a matrix form one obtains

$$\begin{bmatrix} \boldsymbol{x_p^0(\theta)} \\ \boldsymbol{y_p^0(\theta)} \end{bmatrix} = \boldsymbol{R_1^0(\theta)} \begin{bmatrix} \boldsymbol{x_p^1} \\ \boldsymbol{y_p^1} \end{bmatrix} \quad \Leftrightarrow \quad \boldsymbol{p^0(\theta) = R_1^0(\theta)\,p^1}$$

where the matrix

$$\boldsymbol{R_1^0(\theta)} = \begin{bmatrix} \boldsymbol{\cos(\theta)} & \boldsymbol{-\sin(\theta)} \\ \boldsymbol{\sin(\theta)} & \boldsymbol{\cos(\theta)} \end{bmatrix}$$

is called a **rotation matrix**, which is a coordinate representation of a **rotation tensor $\boldsymbol{R(\theta)}$**. The latter can be interpreted as a transformation of vectors by rotation on angle $\boldsymbol{\theta}$ in counterclockwise direction. In particular, it defines **1**-frame as a **0**-frame rotated on angle $\boldsymbol{\theta}$. We have computed its representation in **0**-frame as an action on (linear transformation of) coordinates of vectors with respect to **1**-frame. This representation (the matrix) allows us to compute coordinates with respect to **0**-frame from coordinates with respect to **1**-frame.

Rotation matrices in 2D and 3D

The next properties of $\boldsymbol{R_1^0}$ (and other rotation matrices) are easy to verify

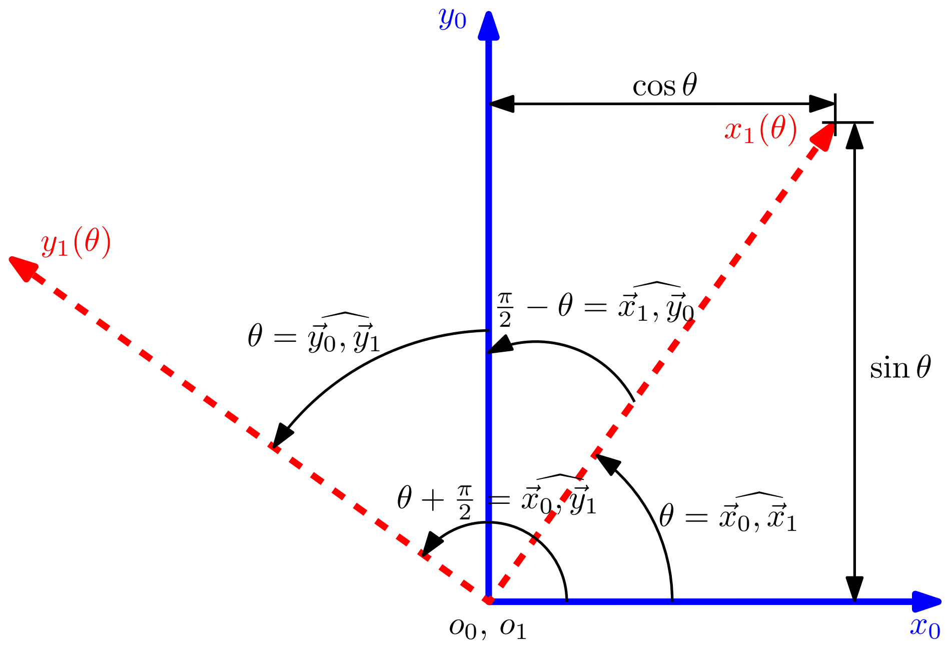
- $\boldsymbol{\det R_1^0(\theta) = \cos^2(\theta) + \sin^2(\theta) = 1}$

- $$\boldsymbol{R}_0^1(\boldsymbol{\theta}) = \left[\boldsymbol{R}_1^0(\boldsymbol{\theta})\right]^{-1} = \begin{bmatrix} \cos(\boldsymbol{\theta}) & \sin(\boldsymbol{\theta}) \\ -\sin(\boldsymbol{\theta}) & \cos(\boldsymbol{\theta}) \end{bmatrix} = \left[\boldsymbol{R}_1^0(\boldsymbol{\theta})\right]^T = \boldsymbol{R}_1^0(-\boldsymbol{\theta})$$

Let us consider now a way of defining the physical action of rotation by angle $\boldsymbol{\theta}$, i.e. the tensor $\boldsymbol{R}(\boldsymbol{\theta})$, in a coordinate-free way. To this end, we notice that the columns of the rotation matrix

$$\boldsymbol{R}_1^0(\boldsymbol{\theta}) = \begin{bmatrix} \cos(\boldsymbol{\theta}) & -\sin(\boldsymbol{\theta}) \\ \sin(\boldsymbol{\theta}) & \cos(\boldsymbol{\theta}) \end{bmatrix} = \begin{bmatrix} \cos(\boldsymbol{\theta}) & \cos(\boldsymbol{\theta}+\frac{\pi}{2}) \\ \cos(\frac{\pi}{2}-\boldsymbol{\theta}) & \cos(\boldsymbol{\theta}) \end{bmatrix} = \left[\boldsymbol{x}_1^0(\boldsymbol{\theta}) \,|\, \boldsymbol{y}_1^0(\boldsymbol{\theta})\right]$$

are defined by projections of the direction vectors of **1**-frame onto the axes of **0**-frame or by the cosines of the angles between the respective axes.


y_0
$\cos\theta$
$x_1(\theta)$
$y_1(\theta)$
$\frac{\pi}{2}-\theta = \widehat{\vec{x}_1,\vec{y}_0}$
$\theta = \widehat{\vec{y}_0,\vec{y}_1}$
$\sin\theta$
$\theta+\frac{\pi}{2} = \widehat{\vec{x}_0,\vec{y}_1}$
$\theta = \widehat{\vec{x}_0,\vec{x}_1}$
$o_0,\ o_1$
x_0

Vector projections can be defined using the notion of scalar product between two vectors, which can be computed either using a representation in an arbitrary frame or using the physical coordinate-free notions of magnitude of a vector and angle between two vectors:

$$\vec{\boldsymbol{V}}_1 \cdot \vec{\boldsymbol{V}}_2 = \left(\boldsymbol{V}_1^*\right)^T \boldsymbol{V}_2^* = |\vec{\boldsymbol{V}}_1|\,|\vec{\boldsymbol{V}}_2|\,\cos\left(\widehat{\vec{\boldsymbol{V}}_1, \vec{\boldsymbol{V}}_2}\right)$$

where $(\cdot)^*$ denotes a coordinate representation in an arbitrary frame ($* = \mathbf{0}$, $* = \mathbf{1}, \ldots$).

It is easy to see that

$$\boldsymbol{x}_1^0(\boldsymbol{\theta}) = \begin{bmatrix} \left(\boldsymbol{x}_1^0(\boldsymbol{\theta})\right)^T \boldsymbol{x}_0^0 \\ \left(\boldsymbol{x}_1^0(\boldsymbol{\theta})\right)^T \boldsymbol{y}_0^0 \end{bmatrix}, \qquad \boldsymbol{y}_1^0(\boldsymbol{\theta}) = \begin{bmatrix} \left(\boldsymbol{y}_1^0(\boldsymbol{\theta})\right)^T \boldsymbol{x}_0^0 \\ \left(\boldsymbol{y}_1^0(\boldsymbol{\theta})\right)^T \boldsymbol{y}_0^0 \end{bmatrix}$$

where, obviously,

$$x_0^0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad y_0^0 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad \left(\Leftarrow \quad \vec{x}_0 = 1\,\vec{x}_0 + 0\,\vec{y}_0, \quad \vec{y}_0 = 0\,\vec{x}_0 + 1\,\vec{y}_0 \right)$$

and, therefore, rotation of **1**-frame with respect to **0**-frame by angle θ can be defined as

$$R_1^0(\theta) = \left[x_1^0(\theta) \,|\, y_1^0(\theta) \right] = \left[\begin{array}{c|c} \vec{x}_1(\theta) \cdot \vec{x}_0 & \vec{y}_1(\theta) \cdot \vec{x}_0 \\ \vec{x}_1(\theta) \cdot \vec{y}_0 & \vec{y}_1(\theta) \cdot \vec{y}_0 \end{array} \right]$$

In a similar way, it can be shown that the matrix defining rotation of **1**-frame with respect to **0**-frame with the same origin in **3**-dimensions can be computed as $R_1^0 = \left[x_1^0 |\, y_1^0 |\, z_1^0 \right]$, i.e.

$$R_1^0 = \left[\begin{array}{c|c|c} (x_1^0)^T x_0^0 & (y_1^0)^T x_0^0 & (z_1^0)^T x_0^0 \\ (x_1^0)^T y_0^0 & (y_1^0)^T y_0^0 & (z_1^0)^T y_0^0 \\ (x_1^0)^T z_0^0 & (y_1^0)^T z_0^0 & (z_1^0)^T z_0^0 \end{array} \right] = \left[\begin{array}{c|c|c} \vec{x}_1 \cdot \vec{x}_0 & \vec{y}_1 \cdot \vec{x}_0 & \vec{z}_1 \cdot \vec{x}_0 \\ \vec{x}_1 \cdot \vec{y}_0 & \vec{y}_1 \cdot \vec{y}_0 & \vec{z}_1 \cdot \vec{y}_0 \\ \vec{x}_1 \cdot \vec{z}_0 & \vec{y}_1 \cdot \vec{z}_0 & \vec{z}_1 \cdot \vec{z}_0 \end{array} \right]$$

It can be verified that similar to the **2**D-case in **3**D for every θ

- Columns of R_1^0 are mutually orthogonal;
- $R_1^0 \left[R_1^0 \right]^T = I_3$ and $R_0^1 = \left[R_1^0 \right]^T$;
- $\det R_1^0 = 1$.

It is also known that if a rotation between two frames in **3**D is done around a fixed (stationary) axes through the origin, the constant direction of which is defined by a unit vector $\vec{k}$, on the angle θ in counterclockwise direction as observed from its arrow, than its matrix representation in the **0**-frame $(o_0 x_0 y_0 z_0)$ can be computed as follows:

$$R^0 = k^0 \,(k^0)^T + \cos(\theta) \left(I - k^0 \,(k^0)^T \right) + \sin(\theta)\, S^0(\vec{k})$$

where $I = x_0 x_0^T + y_0 y_0^T + z_0 z_0^T$ is the identity matrix, while with the **0**-frame-coordinate representation $(k^0)^T = \left[k_x^0,\, k_y^0,\, k_z^0 \right]$ we use the notation $S^0(\vec{k}) = \left[\begin{array}{c|c|c} 0 & -k_z^0 & k_y^0 \\ k_z^0 & 0 & -k_x^0 \\ -k_y^0 & k_x^0 & 0 \end{array} \right]$ to be introduced later when we talk about matrix representations of vector cross products. In coordinate-free form, using the standard notation for tensor and cross products between vectors, the formula above, which is known as Rodrigues' formula, can be written as

$$R(\theta) = \vec{k} \otimes \vec{k} + \cos(\theta)\, (I - \vec{k} \otimes \vec{k}) + \sin(\theta)\, \vec{k} \times I$$

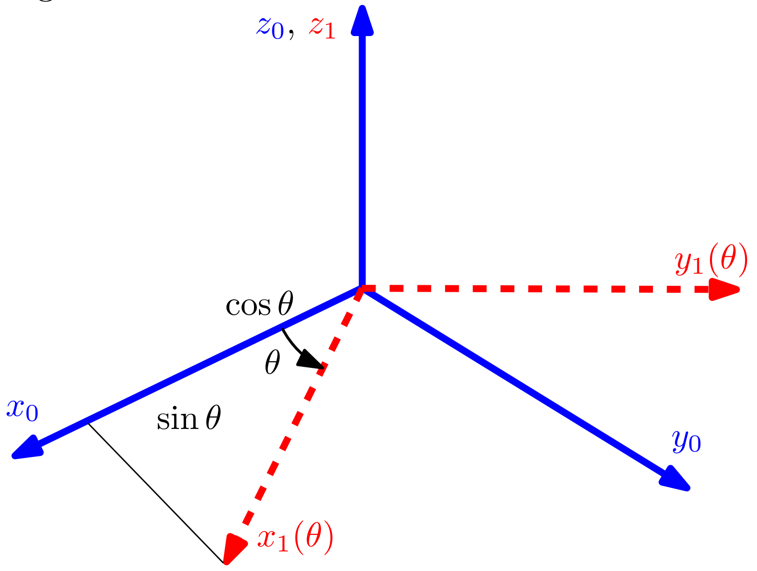
where $\vec{\boldsymbol{k}} \otimes \vec{\boldsymbol{k}}$, $\boldsymbol{I} = \vec{\boldsymbol{x}}_0 \otimes \vec{\boldsymbol{x}}_0 + \vec{\boldsymbol{y}}_0 \otimes \vec{\boldsymbol{y}}_0 + \vec{\boldsymbol{z}}_0 \otimes \vec{\boldsymbol{z}}_0$ and $\boldsymbol{S}(\vec{\boldsymbol{k}}) \equiv \vec{\boldsymbol{k}} \times \boldsymbol{I}$ denote actions on vectors that correspond

- to creating vector projections onto $\vec{\boldsymbol{k}}$: $\big(\vec{\boldsymbol{k}} \otimes \vec{\boldsymbol{k}}\big) \cdot \vec{\boldsymbol{v}} = \big(\vec{\boldsymbol{k}} \cdot \vec{\boldsymbol{v}}\big)\vec{\boldsymbol{k}}$,
- to keeping the vectors unchanged: $\boldsymbol{I} \cdot \vec{\boldsymbol{v}} = \vec{\boldsymbol{v}}$, and
- to making a cross product with $\vec{\boldsymbol{k}}$: $\big(\vec{\boldsymbol{k}} \times \boldsymbol{I}\big) \cdot \vec{\boldsymbol{v}} = \vec{\boldsymbol{k}} \times \vec{\boldsymbol{v}}$,

respectively. We will discuss later an alternative formula for computation of a coordinate representation of a rotation around a given direction.

Basic rotations in 3D

Let us consider now the most important for the development of our targeted solution of the forward kinematics problem examples of rotation in **3**D. Consider the following rotation from **0**-frame to **1**-frame



Here we have **1**-frame rotated by $\boldsymbol{\theta}$-angle around the $\boldsymbol{z_0}$-axis that coincides with the the $\boldsymbol{z_1}$-axis.

It is easy to see that

$$\begin{array}{rclrclrcl}
\vec{\boldsymbol{x}}_1(\boldsymbol{\theta}) \cdot \vec{\boldsymbol{x}}_0 &=& \cos\boldsymbol{\theta}, & \vec{\boldsymbol{y}}_1(\boldsymbol{\theta}) \cdot \vec{\boldsymbol{x}}_0 &=& -\sin\boldsymbol{\theta}, & \vec{\boldsymbol{z}}_1(\boldsymbol{\theta}) \cdot \vec{\boldsymbol{x}}_0 &=& \boldsymbol{0} \\
\vec{\boldsymbol{x}}_1(\boldsymbol{\theta}) \cdot \vec{\boldsymbol{y}}_0 &=& \sin\boldsymbol{\theta}, & \vec{\boldsymbol{y}}_1(\boldsymbol{\theta}) \cdot \vec{\boldsymbol{y}}_0 &=& \cos\boldsymbol{\theta}, & \vec{\boldsymbol{z}}_1(\boldsymbol{\theta}) \cdot \vec{\boldsymbol{y}}_0 &=& \boldsymbol{0} \\
\vec{\boldsymbol{x}}_1(\boldsymbol{\theta}) \cdot \vec{\boldsymbol{z}}_0 &=& \boldsymbol{0}, & \vec{\boldsymbol{y}}_1(\boldsymbol{\theta}) \cdot \vec{\boldsymbol{z}}_0 &=& \boldsymbol{0}, & \vec{\boldsymbol{z}}_1(\boldsymbol{\theta}) \cdot \vec{\boldsymbol{z}}_0 &=& \boldsymbol{1}
\end{array}$$

Therefore, we obtain

$$\boldsymbol{R}_1^0(\boldsymbol{\theta}) = \begin{bmatrix} \cos\boldsymbol{\theta} & -\sin\boldsymbol{\theta} & \boldsymbol{0} \\ \sin\boldsymbol{\theta} & \cos\boldsymbol{\theta} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{1} \end{bmatrix} \qquad \{=: \boldsymbol{R}_{z,\theta}\}$$

Same expression can be obtained from the Euler formula with $\vec{\boldsymbol{k}} = \vec{\boldsymbol{z}}_0 = \vec{\boldsymbol{z}}_1$, or in, say, **0**-frame coordinates with $(\boldsymbol{k}^0)^T = [\boldsymbol{0},\ \boldsymbol{0},\ \boldsymbol{1}]$.

This matrix, called a **basic rotation matrix**, clearly satisfies the following properties

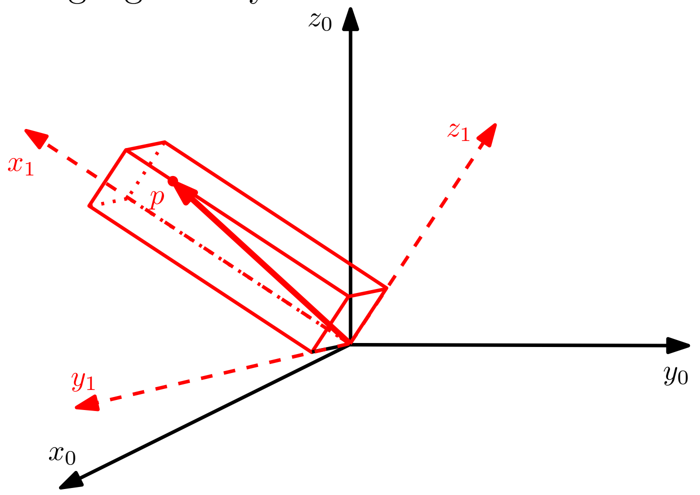
$$\boldsymbol{R}_{z,0} = \boldsymbol{I}_3, \qquad \boldsymbol{R}_{z,\theta}\boldsymbol{R}_{z,\phi} = \boldsymbol{R}_{z,\theta+\phi}, \qquad [\boldsymbol{R}_{z,\theta}]^{-1} = \boldsymbol{R}_{z,-\theta}$$

There are two other basic rotations in $\mathbf{3}$D (around $\vec{\boldsymbol{k}} = \vec{\boldsymbol{x}}_0$ and $\vec{\boldsymbol{k}} = \vec{\boldsymbol{y}}_0$):

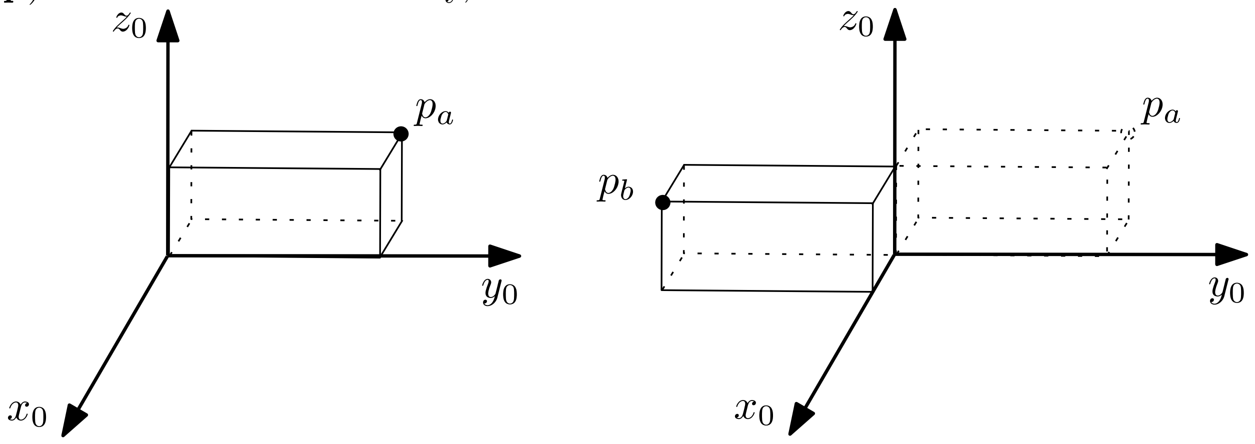
$$\boldsymbol{R}_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}, \qquad \boldsymbol{R}_{y,\theta} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

Rotation of a rigid body

We are now ready to present an application of the just introduced rotations in **3**-D. Let the **0**-frame define our world (fixed) frame, while the **1**-frame be attached to a rotating rigid body:



What will happen with points of the body (let say with a particular point $\boldsymbol{p}$) when we rotate the body, i.e. the **1**-frame?



How to trace the change of position (coordinates) of a particular point of the body with respect to the **0**-frame?

Suppose we are interested in coordinates of a particular point $\boldsymbol{p}$ of the body. Clearly, in the **1**-frame they are constant, but in the **0**-frame they are changing during the rotation!

Let the coordinates of $\boldsymbol{p}$ in the **1**-frame be $\boldsymbol{p}^1 == [\boldsymbol{x}_p^1, \boldsymbol{y}_p^1, \boldsymbol{z}_p^1]^T$, i.e.

$$\boldsymbol{p}^1 = \boldsymbol{x}_p^1\,\boldsymbol{x}_1^1 + \boldsymbol{y}_p^1\,\boldsymbol{y}_1^1 + \boldsymbol{z}_p^1\,\boldsymbol{z}_1^1 = \boldsymbol{x}_p^1 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \boldsymbol{y}_p^1 \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \boldsymbol{z}_p^1 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_p^1 \\ \boldsymbol{y}_p^1 \\ \boldsymbol{z}_p^1 \end{bmatrix}$$

and the coordinates $\boldsymbol{x}_p^1$, $\boldsymbol{y}_p^1$, $\boldsymbol{z}_p^1$ are not changing when the **1**-frame is rotated. We need to find the coordinates of $\boldsymbol{p}$ in the **0**-frame that are changing.

Clearly, with the rotation the basis of the **1**-frame is moving with respect to **0**-frame and we know how it does that: according to a rotation matrix $\boldsymbol{R}_1^0$ relating the two frames!

We have for the basis vectors of the **1**-frame:

$$\boldsymbol{x}_1^0 = \boldsymbol{R}_1^0\boldsymbol{x}_1^1 = \boldsymbol{R}_1^0 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{y}_1^0 = \boldsymbol{R}_1^0\boldsymbol{y}_1^1 = \boldsymbol{R}_1^0 \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \boldsymbol{z}_1^0 = \boldsymbol{R}_1^0\boldsymbol{z}_1^1 = \boldsymbol{R}_1^0 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Therefore the coordinates of the point $\boldsymbol{p}$ in **0**-frames are computed as follows

$$\boldsymbol{p}^0 = \boldsymbol{x}_p^1\boldsymbol{x}_1^0 + \boldsymbol{y}_p^1\boldsymbol{y}_1^0 + \boldsymbol{z}_p^1\boldsymbol{z}_1^0 = \boldsymbol{R}_1^0 \begin{bmatrix} \boldsymbol{x}_p^1 \\ 0 \\ 0 \end{bmatrix} + \boldsymbol{R}_1^0 \begin{bmatrix} 0 \\ \boldsymbol{y}_p^1 \\ 0 \end{bmatrix} + \boldsymbol{R}_1^0 \begin{bmatrix} 0 \\ 0 \\ \boldsymbol{z}_p^1 \end{bmatrix} = \underbrace{\boldsymbol{R}_1^0 \begin{bmatrix} \boldsymbol{x}_p^1 \\ \boldsymbol{y}_p^1 \\ \boldsymbol{z}_p^1 \end{bmatrix}}_{=\boldsymbol{R}_1^0\,\boldsymbol{p}^1}$$

Now, to handle in the future rotation of the tool as a result of rotation of all the links, we need to understand how to combine multiple rotations.

2.1.3 Composition of Rotations

We will see soon that when a body is involved in several (an ordered sequence of) rotations, it is important to distinguish between two situations:

- Rotations with Respect to the Current Frame (that is, except for the first one, every rotation is performed around one of the axis obtained after previous rotation)

- Rotations with Respect to the Fixed Frame (that is every rotation is performed around a not rotated axis)

However, before dealing with this, let us derive a simple (algebraic) composition rule for rotations.

Sequence of two rotations

Suppose that we have **3** different frames in 3$\boldsymbol{D}$:

$$(\boldsymbol{o}_0, \boldsymbol{x}_0, \boldsymbol{y}_0, \boldsymbol{z}_0), \quad (\boldsymbol{o}_1, \boldsymbol{x}_1, \boldsymbol{y}_1, \boldsymbol{z}_1), \quad (\boldsymbol{o}_2, \boldsymbol{x}_2, \boldsymbol{y}_2, \boldsymbol{z}_2).$$

with coinciding origins, i.e. $\boldsymbol{o}_0 = \boldsymbol{o}_1 = \boldsymbol{o}_2$ or $\overrightarrow{\boldsymbol{o}_0\boldsymbol{o}_1} = \overrightarrow{\boldsymbol{o}_1\boldsymbol{o}_2} = \overrightarrow{\boldsymbol{o}_2\boldsymbol{o}_0} = \vec{\boldsymbol{0}}$.

Any point $\boldsymbol{p}$ will have three coordinate representations:

$$\boldsymbol{p}^0 = [\boldsymbol{u}_0, \boldsymbol{v}_0, \boldsymbol{w}_0]^T, \quad \boldsymbol{p}^1 = [\boldsymbol{u}_1, \boldsymbol{v}_1, \boldsymbol{w}_1]^T, \quad \boldsymbol{p}^2 = [\boldsymbol{u}_2, \boldsymbol{v}_2, \boldsymbol{w}_2]^T$$

defined by three identical vectors ($\vec{\boldsymbol{V}}_0 = \vec{\boldsymbol{V}}_1 = \vec{\boldsymbol{V}}_2$):

$$\vec{\boldsymbol{V}}_0 = \boldsymbol{u}_0\, \vec{\boldsymbol{x}}_0 + \boldsymbol{v}_0\, \vec{\boldsymbol{y}}_0 + \boldsymbol{w}_0\, \vec{\boldsymbol{z}}_0, \quad \vec{\boldsymbol{V}}_1 = \boldsymbol{u}_1\, \vec{\boldsymbol{x}}_1 + \boldsymbol{v}_1\, \vec{\boldsymbol{y}}_1 + \boldsymbol{w}_1\, \vec{\boldsymbol{z}}_1,$$
$$\vec{\boldsymbol{V}}_2 = \boldsymbol{u}_2\, \vec{\boldsymbol{x}}_2 + \boldsymbol{v}_2\, \vec{\boldsymbol{y}}_2 + \boldsymbol{w}_2\, \vec{\boldsymbol{z}}_2$$

Orientation of each pair of frames can be defined by appropriate rotations that automatically define the transformations between the coordinate representations:

$$\boldsymbol{p}^0 = \boldsymbol{R}_1^0\, \boldsymbol{p}^1, \quad \boldsymbol{p}^1 = \boldsymbol{R}_2^1\, \boldsymbol{p}^2, \quad \boldsymbol{p}^0 = \boldsymbol{R}_2^0\, \boldsymbol{p}^2$$

Let us answer the question: How are the matrices $\boldsymbol{R}_1^0$, $\boldsymbol{R}_2^1$, and $\boldsymbol{R}_2^0$ related? One can compute $\boldsymbol{p}^0$ in two different ways

$$\boldsymbol{p}^0 = \boldsymbol{R}_1^0\, \boldsymbol{p}^1 = \boldsymbol{R}_1^0\, \boldsymbol{R}_2^1\, \boldsymbol{p}^2, \qquad \boldsymbol{p}^0 = \boldsymbol{R}_2^0\, \boldsymbol{p}^2$$

As a result, we conclude

$$\boxed{\boldsymbol{R}_1^0\, \boldsymbol{R}_2^1 \equiv \boldsymbol{R}_2^0}$$

which simply means that a rotation from **0**-frame to **2**-frame is equivalent to a composition of two rotations: from **0**-frame to **1**-frame and from **1**-frame to **2**-frame. To memorize this, notice that the lower index **1** and the upper index **1** symbolically cancel each other.

Combining rotations with respect to current axes

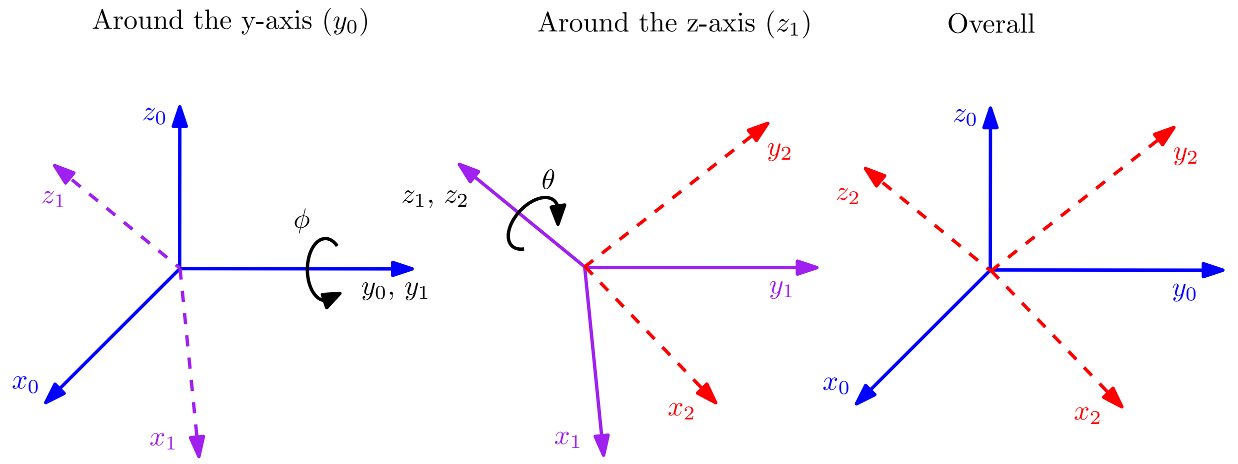
Let us use this rule to derive a composition of the following two rotations:

- First, **0**-frame is rotated by angle $\boldsymbol{\phi}$ around the current $\boldsymbol{y}$-axis, i.e. $\boldsymbol{y}_0$, to obtain **1**-frame,

- Second, **1**-frame is rotate by angle $\boldsymbol{\theta}$ around the current $\boldsymbol{z}$-axis, i.e. $\boldsymbol{z}_1$, to obtain the final **2**-frame.

defining the change of coordinates from **2**-frame to **0**-frame.

This is illustrated below



The rotations around the current $\boldsymbol{y}$-axis and $\boldsymbol{z}$-axis are basic rotations

$$\boldsymbol{R}_{y_0,\phi} = \begin{bmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{bmatrix} = \boldsymbol{R}_1^0, \quad \boldsymbol{R}_{z_1,\theta} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} = \boldsymbol{R}_2^1$$

Therefore, the overall rotation is $\boldsymbol{R}_2^0 = \boldsymbol{R}_1^0\,\boldsymbol{R}_2^1$, i.e.

$$\boldsymbol{R}_2^0 = \underbrace{\boldsymbol{R}_{y_0,\phi}}_{\text{first}} \quad \underbrace{\boldsymbol{R}_{z_1,\theta}}_{\text{second}} = \begin{bmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

and after the matrix multiplication

$$\boldsymbol{R}_2^0 = \boldsymbol{R}_{y_0,\phi}\boldsymbol{R}_{z_1,\theta} = \begin{bmatrix} c_\phi c_\theta & -c_\phi s_\theta & s_\phi \\ s_\theta & c_\theta & 0 \\ -s_\phi c_\theta & s_\phi s_\theta & c_\phi \end{bmatrix}, \qquad \{\Rightarrow\ \boldsymbol{p}^0 = \boldsymbol{R}_2^0\,\boldsymbol{p}^2\}$$

where the standard short notation is used for the trigonometric functions.

Order of rotations is important

Here, we should make an important observation: Rotations do not commute:

$$\boldsymbol{R}_{y,\phi}\boldsymbol{R}_{z,\theta} \neq \boldsymbol{R}_{z,\theta}\boldsymbol{R}_{y,\phi}$$

So, the order of rotations is important!

Indeed,

$$\underbrace{\boldsymbol{R}_{y_0,\phi}}_{\text{first}} \quad \underbrace{\boldsymbol{R}_{z_1,\theta}}_{\text{second}} = \boldsymbol{R}_{y,\phi}\ \boldsymbol{R}_{z,\theta} = \begin{bmatrix} c_\phi c_\theta & -c_\phi s_\theta & s_\phi \\ s_\theta & c_\theta & 0 \\ -s_\phi c_\theta & s_\phi s_\theta & c_\phi \end{bmatrix}$$

and

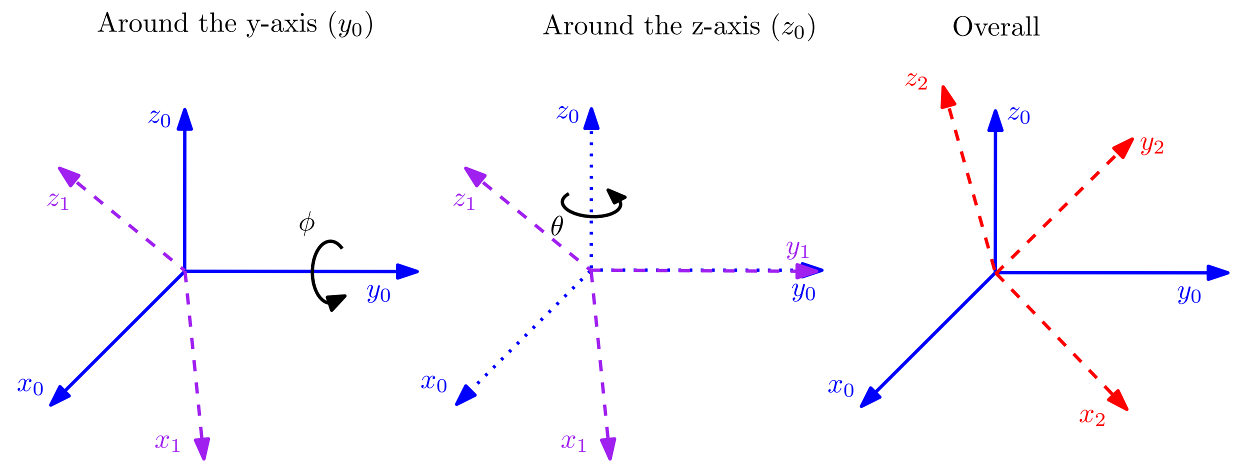
$$\underbrace{\boldsymbol{R}_{z_0,\theta}}_{\text{first}}\ \underbrace{\boldsymbol{R}_{y_1,\phi}}_{\text{second}} = \boldsymbol{R}_{z,\theta}\ \boldsymbol{R}_{y,\phi} = \begin{bmatrix} c_\phi c_\theta & -s_\theta & c_\theta s_\phi \\ s_\theta c_\phi & c_\theta & s_\phi s_\theta \\ -s_\phi & 0 & c_\phi \end{bmatrix}$$

Combining rotations with respect to fixed axes

Let us consider now consecutive rotations with respect to a fixed frame:

- First, $\mathbf{0}$-frame is rotated by angle ϕ around the current $\boldsymbol{y}$-axis, i.e. $\boldsymbol{y_0}$, to obtain $\mathbf{1}$-frame,
- Second, $\mathbf{1}$-frame is rotate by angle θ around the original $\boldsymbol{z}$-axis, i.e. $\boldsymbol{z_0}$, to obtain the final $\mathbf{2}$-frame.

They are illustrated below.



Obviously, the second rotation is not around one of the current coordinate axes. So, we either should use the Euler formula or a simple trick, described next, that allows to use coordinate representations of basic rotations in this case also. To use the trick, we should derive first a useful formula relating coordinate representations of a linear transformation, such as rotations, in two different frames.

Change of coordinates for representation of rotations

Consider two frames with a given relative rotation (and the same origin):

$$(\boldsymbol{o_0}, \boldsymbol{x_0}, \boldsymbol{y_0}, \boldsymbol{z_0}), \quad (\boldsymbol{o_1}, \boldsymbol{x_1}, \boldsymbol{y_1}, \boldsymbol{z_1}), \quad \boldsymbol{p}^0 = \boldsymbol{R}_1^0 \boldsymbol{p}^1$$

Given a linear transform $\boldsymbol{A}$, that acts on vectors:

$$\vec{\boldsymbol{a}} = \boldsymbol{A} \cdot \vec{\boldsymbol{b}}$$

for which we know a coordinate representation in the **0**-frame

$$\boldsymbol{a}^0 = \boldsymbol{A}^0\, \boldsymbol{b}^0$$

How to compute its representation $\boldsymbol{a}^1 = \boldsymbol{A}^1\, \boldsymbol{b}^1$ in the **1**-frame? Obviously, we just need to use the coordinate representations in the two frames:

- For a given $\boldsymbol{a}^0$, one can compute $\boldsymbol{a}^1 = \boldsymbol{R}_0^1\, \boldsymbol{a}^0 = \left[\boldsymbol{R}_1^0\right]^{-1} \boldsymbol{a}^0$
- For a given $\boldsymbol{b}^1$, one can compute $\boldsymbol{b}^0 = \boldsymbol{R}_1^0\, \boldsymbol{b}^1$

To define $\boldsymbol{A}^1$, the representation of $\boldsymbol{A}$ in the **1**-frame, one proceeds as follows

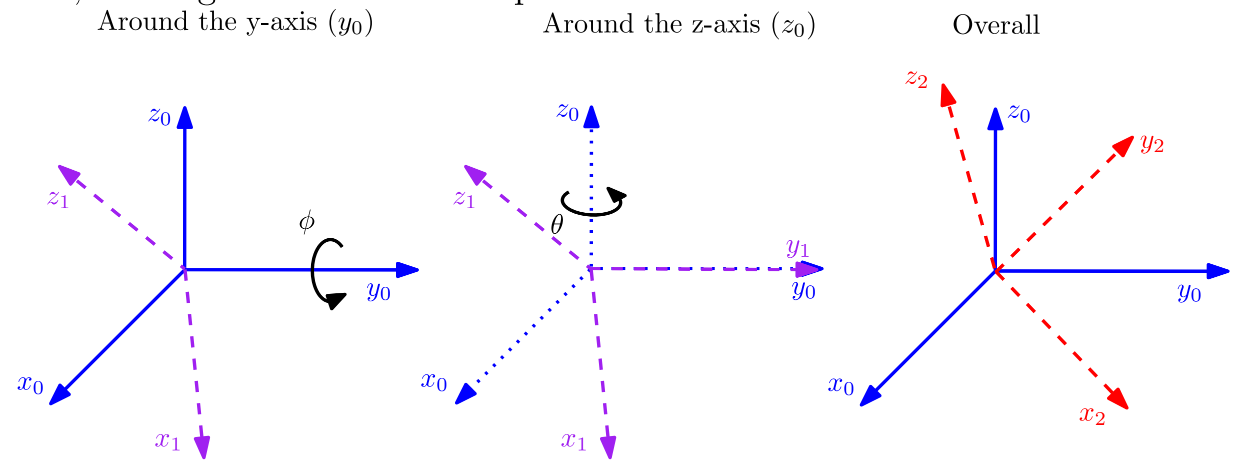
$$\boldsymbol{a}^1 = \left[\boldsymbol{R}_1^0\right]^{-1} \boldsymbol{a}^0 = \left[\boldsymbol{R}_1^0\right]^{-1} (\boldsymbol{A}^0\, \boldsymbol{b}^0) = \left[\boldsymbol{R}_1^0\right]^{-1} \boldsymbol{A}^0\, (\boldsymbol{R}_1^0\, \boldsymbol{b}^1) = \boldsymbol{A}^1\, \boldsymbol{b}^1$$

Since this relation is true for arbitrary $\boldsymbol{a}^1$ and $\boldsymbol{b}^1$, we conclude

$$\boxed{\boldsymbol{A}^1 = \left[\boldsymbol{R}_1^0\right]^{-1} \boldsymbol{A}^0\, \boldsymbol{R}_1^0} = \boldsymbol{R}_0^1\, \boldsymbol{A}^0\, \boldsymbol{R}_1^0 = \boldsymbol{R}_0^1\, \boldsymbol{A}^0 \left[\boldsymbol{R}_0^1\right]^{-1}$$

Combining rotations with respect to fixed axes (continued)

Now, let us go back to our composition of rotations:



We have two rotations

- the basic rotation $\boldsymbol{R}_1^0 = \boldsymbol{R}_{y_0,\phi}$: by angle $\boldsymbol{\phi}$ around $\boldsymbol{y_0}$-axis
- the rotation $\boldsymbol{R}_2^1$ defined as the rotation by angle $\boldsymbol{\theta}$ around $\boldsymbol{z_0}$-axis (not $\boldsymbol{z_1}$-axis)

The combined rotation will be

$$\boldsymbol{R}_2^0 = \boldsymbol{R}_1^0 \boldsymbol{R}_2^1 = \boldsymbol{R}_{y_0,\phi} \boldsymbol{R}_2^1 = \boldsymbol{R}_{y_0,\phi} \underbrace{\left[(\boldsymbol{R}_{y_0,\phi})^{-1}\, \boldsymbol{R}_{z_0,\theta} \boldsymbol{R}_{y_0,\phi}\right]}_{\boldsymbol{A}^1 = \left[\boldsymbol{R}_1^0\right]^{-1} \boldsymbol{A}^0\, \boldsymbol{R}_1^0}$$

where we use the just derived formula for transformation of the basic rotation representation $\boldsymbol{A}^0 = \boldsymbol{R}_{z_0,\theta}$ in another frame, the **1**-frame, which is related to the **0**-frame by another rotation matrix $\boldsymbol{R}_1^0 = \boldsymbol{R}_{y_0,\phi}$.

After cancellations of the product of a matrix and its inverse, we have

$$\boldsymbol{R}_2^0 = \underbrace{\boldsymbol{R}_{z_0,\theta}}_{\text{second}} \ \underbrace{\boldsymbol{R}_{y_0,\phi}}_{\text{first}} = \boldsymbol{R}_{z,\theta}\, \boldsymbol{R}_{y,\phi}$$

and comparing with composition of rotations with respect to current axes, we see that the order of multiplications of the coordinate representations is reversed! Actually, some authors consider this order as more natural since it coincides with the order of action on the coordinates of a vector.

Although the order of rotations is important as well as the order of matrix multiplications: $\boldsymbol{R}_{z,\theta}\, \boldsymbol{R}_{y,\phi} \neq \boldsymbol{R}_{y,\phi}\, \boldsymbol{R}_{z,\theta}$, it can be easily noticed that

$$\underbrace{\boldsymbol{R}_{z_0,\theta}}_{\text{second}} \ \underbrace{\boldsymbol{R}_{y_0,\phi}}_{\text{first}} = \boldsymbol{R}_{z,\theta}\, \boldsymbol{R}_{y,\phi} = \underbrace{\boldsymbol{R}_{z_0,\theta}}_{\text{first}} \ \underbrace{\boldsymbol{R}_{y_1,\phi}}_{\text{second}}$$

i.e. the same basic rotations can be performed in reversed order if we switch to the appropriate current axis instead of the fixed one.

Let us consider a few simple examples of combining rotations.

Example with two rotations

Find a representation of the rotation $\boldsymbol{R}_2^0$ defined by the following ordered sequence of basic rotations:

0 $\to$ **1** By $\boldsymbol{\theta}$ about the current axis $\boldsymbol{x}$, i.e. $\boldsymbol{x}_0$;

1 $\to$ **2** By $\boldsymbol{\phi}$ about the current axis $\boldsymbol{z}$, i.e. $\boldsymbol{z}_1$.

Since the rotations are performed around the current axes, we multiply the basic rotation matrices in the same order, to obtain

$$\boldsymbol{R}_2^0 = \boldsymbol{R}_{x,\theta}\, \boldsymbol{R}_{z,\phi}$$

The obtained transformation defines for any point of the **2**-frame with coordinates $\boldsymbol{p}^2 = \left[\boldsymbol{x}^2, \boldsymbol{y}^2, \boldsymbol{z}^2\right]^T$, its coordinates in the **0**-frame as

$$\boldsymbol{p}^0 = \boldsymbol{R}_2^0\, \boldsymbol{p}^2 = \boldsymbol{R}_{x,\theta}\, \boldsymbol{R}_{z,\phi}\, \boldsymbol{p}^2$$

Example with three rotations

Find a representation of the rotation $\boldsymbol{R}_3^0$ defined by the following ordered sequence of basic rotations:

0 → 1 By $\boldsymbol{\theta}$ about the current axis $\boldsymbol{x}$, i.e. $\boldsymbol{x_0}$;

1 → 2 By $\boldsymbol{\phi}$ about the current axis $\boldsymbol{z}$, i.e. $\boldsymbol{z_1}$;

2 → 3 By $\boldsymbol{\alpha}$ about the **fixed axis** $\boldsymbol{z}$, i.e. $\boldsymbol{z_0}$.

Following the rule discovered above: **"every new rotation with respect to the current axis implies multiplication by the corresponding matrix from the right (direct order), while every new rotation with respect to the fixed (initial) axis implies multiplication by the corresponding matrix from the left (reverse order)"** we obtain the answer

$$\boldsymbol{R}_3^0 = \boldsymbol{R}_{z,\alpha}\, \boldsymbol{R}_2^0 = \boldsymbol{R}_{z,\alpha}\, \boldsymbol{R}_{x,\theta}\, \boldsymbol{R}_{z,\phi}$$

Example with four rotations

Find a representation of the rotation $\boldsymbol{R}_4^0$ defined by the following ordered sequence of basic rotations:

0 → 1 By $\boldsymbol{\theta}$ about the current axis $\boldsymbol{x}$, i.e. $\boldsymbol{x_0}$;

1 → 2 By $\boldsymbol{\phi}$ about the current axis $\boldsymbol{z}$, i.e. $\boldsymbol{z_1}$;

2 → 3 By $\boldsymbol{\alpha}$ about the **fixed axis** $\boldsymbol{z}$, i.e. $\boldsymbol{z_0}$;

3 → 4 By $\boldsymbol{\beta}$ about the current axis $\boldsymbol{y}$, i.e. $\boldsymbol{y_3}$.

The result of three rotations, obtained above, should be multiplied by the last basic rotation matrix from the right

$$\boldsymbol{R}_4^0 = \boldsymbol{R}_3^0\, \boldsymbol{R}_{y,\beta} = \boldsymbol{R}_{z,\alpha}\, \boldsymbol{R}_{x,\theta}\, \boldsymbol{R}_{z,\phi}\, \boldsymbol{R}_{y,\beta}$$

Examples with five rotations

Find a representation of the rotation $\boldsymbol{R}_5^0$ defined by the following ordered sequence of basic rotations:

0 → 1 By $\boldsymbol{\theta}$ about the current axis $\boldsymbol{x}$, i.e. $\boldsymbol{x_0}$;

1 → 2 By $\boldsymbol{\phi}$ about the current axis $\boldsymbol{z}$, i.e. $\boldsymbol{z_1}$;

$\mathbf{2} \to \mathbf{3}$ By $\boldsymbol{\alpha}$ about the **fixed axis $\boldsymbol{z}$**, i.e. $\boldsymbol{z_0}$;

$\mathbf{3} \to \mathbf{4}$ By $\boldsymbol{\beta}$ about the current axis $\boldsymbol{y}$, i.e. $\boldsymbol{y_3}$;

$\mathbf{4} \to \mathbf{5}$ By $\boldsymbol{\delta}$ about the **fixed axis $\boldsymbol{x}$**, i.e. $\boldsymbol{x_0}$.

By now, the answer should be obvious:

$$\boldsymbol{R}_5^0 = \boldsymbol{R}_{x,\delta}\, \boldsymbol{R}_4^0 = \boldsymbol{R}_{x,\delta}\, \boldsymbol{R}_{z,\alpha}\, \boldsymbol{R}_{x,\theta}\, \boldsymbol{R}_{z,\phi}\, \boldsymbol{R}_{y,\beta}$$

But what should we do if we would like to compute $\tilde{\boldsymbol{R}}_{\tilde{5}}^0$ with, by some reasons, changed last rotation as follows:

$\mathbf{4} \to \tilde{\mathbf{5}}$ By $\boldsymbol{\delta}$ about the **axis $\boldsymbol{x}$ in 1-frame**, i.e. $\boldsymbol{x_1}$?

We can now think about the last rotation equivalently performed as a basic rotation around $\boldsymbol{x}$-axis but needed to be represented in another frame. So, we have

$$\tilde{\boldsymbol{R}}_{\tilde{5}}^0 = \boldsymbol{R}_4^0\, \boldsymbol{R}_5 = \boldsymbol{R}_{z,\alpha}\, \boldsymbol{R}_{x,\theta}\, \boldsymbol{R}_{z,\phi}\, \boldsymbol{R}_{y,\beta}\, \boldsymbol{R}_5$$

where

$$\boldsymbol{R}_5 = \left[\boldsymbol{R}_4^1\right]^{-1} \boldsymbol{R}_{x,\delta}\, \boldsymbol{R}_4^1, \qquad \text{with} \quad \boldsymbol{R}_4^1 = \boldsymbol{R}_{z,\phi} \left\{ [\boldsymbol{R}_2^0]^{-1}\, \boldsymbol{R}_{z,\alpha}\, \boldsymbol{R}_2^0 \right\} \boldsymbol{R}_{y,\beta}$$

and $\boldsymbol{R}_2^0 = \boldsymbol{R}_{x,\theta}\, \boldsymbol{R}_{z,\phi}$. The final answer can be obtained by a straightforward substitution.

Let us discuss now possible parameterizations of coordinate representations of rotations.

2.1.4 Parametrization of Rotation Matrices

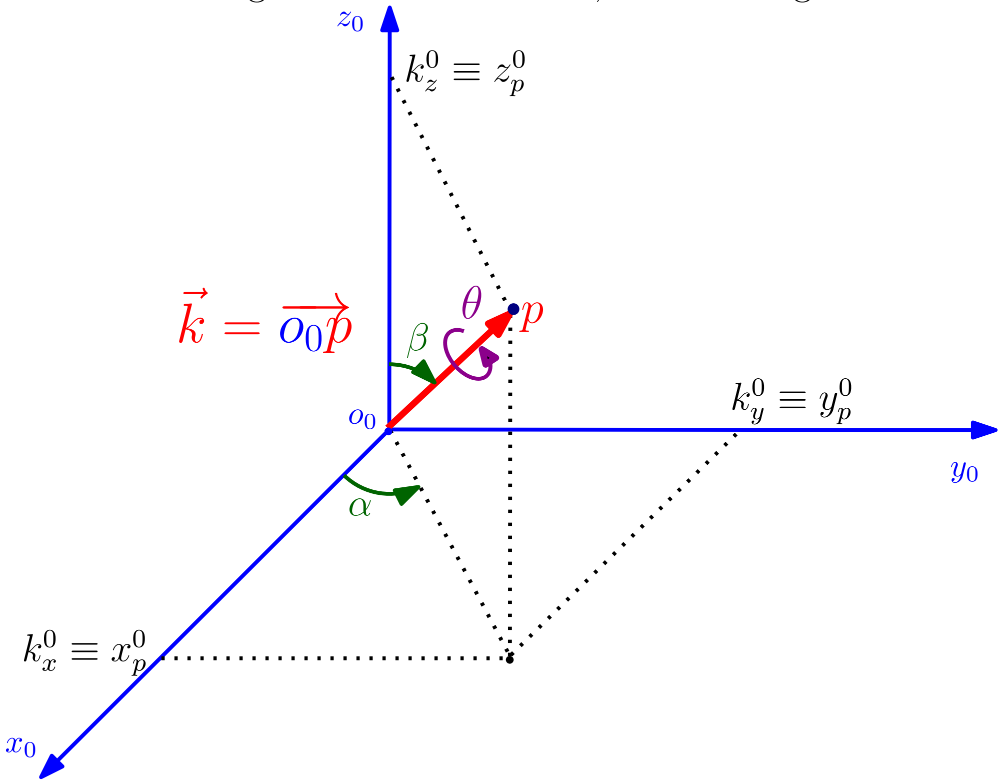
Any coordinate representation of a rotation matrix

- is of dimension $\mathbf{3} \times \mathbf{3}$, i.e. it is defined by $\mathbf{9}$-numbers;
- belongs to $\boldsymbol{\mathcal{SO}}(\mathbf{3})$, i.e. its $\mathbf{3}$ columns define vectors
  - of magnitude $\mathbf{1}$ ($\mathbf{3}$ equations)
  - that are orthogonal to each other ($\mathbf{3}$ equations)

Since there are $\mathbf{6}$ relations on $\mathbf{9}$ numbers, it should be sufficient to specify only $\mathbf{3}$ numbers to completely define a rotation matrix!

Angle-axis representation

A possible choice of such numbers follows from Leonard Euler's observation: Every rotation is defined by two numbers $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, defining the orientation of a unit vector $\vec{\boldsymbol{k}}$ along the axis of rotation, and the angle of rotation $\boldsymbol{\theta}$:



From this picture we see that

$$\boldsymbol{\alpha} = \mathbf{atan2}\left(\boldsymbol{k}_y^0, \boldsymbol{k}_x^0\right), \qquad \boldsymbol{\beta} = \mathbf{atan2}\left(\sqrt{(\boldsymbol{k}_x^0)^2 + (\boldsymbol{k}_y^0)^2}, \boldsymbol{k}_z^0\right)$$

where the Matlab-inspired function "**atan2**" stands for the four-quadrant inverse tangent, i.e. $\boldsymbol{\alpha} \in (-\boldsymbol{\pi}, \boldsymbol{\pi}]$ is such that $\mathbf{tan}(\boldsymbol{\alpha}) = \boldsymbol{k}_y^0 / \boldsymbol{k}_x^0$.

To derive a formula for the corresponding rotation matrix (an alternative formula have been presented earlier), we can perform rotation by angle $\boldsymbol{\theta}$ about an axis defined by $\vec{\boldsymbol{k}}$ as follows. Imagine a change of coordinates, defined by a matrix $\bar{\boldsymbol{R}}^0$ that makes the vector $\vec{\boldsymbol{k}}$ coincide with the frame's $\boldsymbol{z}$-axis. It follows from our derivations above, that we just need to use the basic rotation around $\boldsymbol{z}$ in combination with the change of coordinates formula. Note that with the angles $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ shown in the figure above, we can use rotations around $\boldsymbol{z}$-axis by $\boldsymbol{\alpha}$ followed by rotation around the current $\boldsymbol{y}$-axis by $\boldsymbol{\beta}$ to define the orientation of $\vec{\boldsymbol{k}}$. As a result one obtains the following product of basic rotation matrices

$$\boldsymbol{R}^0_{\vec{k},\boldsymbol{\theta}} = \bar{\boldsymbol{R}}^0\, \boldsymbol{R}^0_{\boldsymbol{z},\boldsymbol{\theta}} \left(\bar{\boldsymbol{R}}^0\right)^{-1}, \qquad \bar{\boldsymbol{R}}^0 = \boldsymbol{R}_{\boldsymbol{z},\boldsymbol{\alpha}}\, \boldsymbol{R}_{\boldsymbol{y},\boldsymbol{\beta}}$$

We should note that finding $\boldsymbol{\theta}$ and $\boldsymbol{k}^0$ (or $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$) from a give expression for $\boldsymbol{R}^0_{\vec{k},\theta}$ may be not trivial. However, it follows immediately from the Rodriges's formula for $\boldsymbol{R}^0_{\vec{k},\theta}$, discussed above,

$$\begin{bmatrix} (k^0_x)^2 & k^0_x\,k^0_y & k^0_x\,k^0_z \\ k^0_y\,k^0_x & (k^0_y)^2 & k^0_y\,k^0_z \\ k^0_z\,k^0_x & k^0_z\,k^0_y & (k^0_z)^2 \end{bmatrix} + c_\theta \begin{bmatrix} 1-(k^0_x)^2 & -k^0_x\,k^0_y & -k^0_x\,k^0_z \\ -k^0_y\,k^0_x & 1-(k^0_y)^2 & -k^0_y\,k^0_z \\ -k^0_z\,k^0_x & -k^0_z\,k^0_y & 1-(k^0_z)^2 \end{bmatrix} + s_\theta \begin{bmatrix} 0 & -k^0_z & k^0_y \\ k^0_z & 0 & -k^0_x \\ -k^0_y & k^0_x & 0 \end{bmatrix},$$

where $c_\theta = cos(\theta)$ and $s_\theta = \sin(\theta)$, which will not prove here, that

$$\cos(\theta) = \frac{1}{2}\left(\text{trace}\left\{\boldsymbol{R}^0_{\vec{k},\theta}\right\} - 1\right)$$

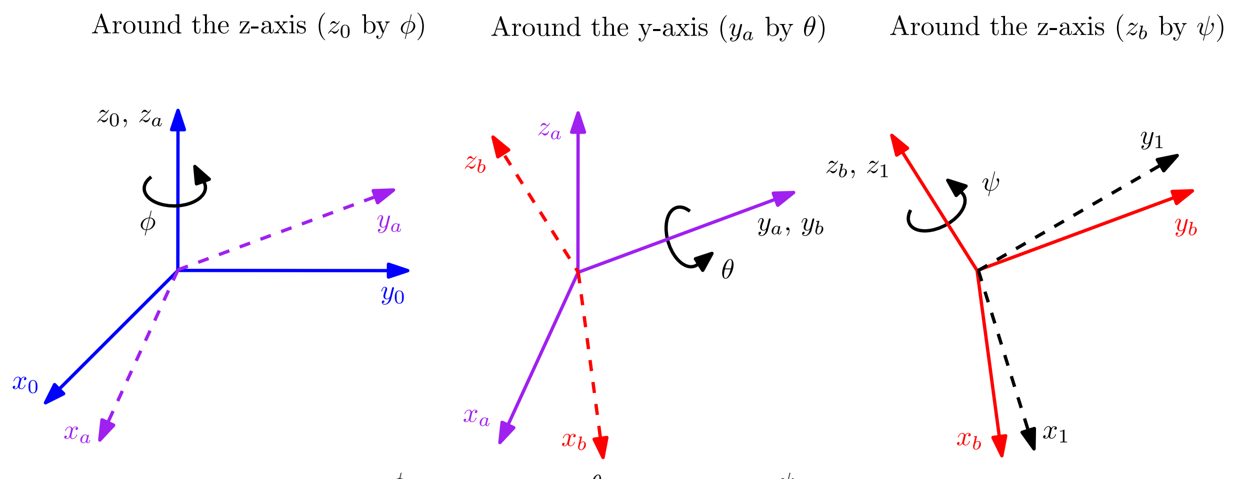
while the coordinates of $\vec{k}$ in 0-frame, k^0, can be computed from

$$\left[\begin{array}{c|c|c} 0 & -k^0_z & k^0_y \\ k^0_z & 0 & -k^0_x \\ -k^0_y & k^0_x & 0 \end{array}\right] = \frac{1}{2\,\sin(\theta)}\left(\boldsymbol{R}^0_{\vec{k},\theta} - \left(\boldsymbol{R}^0_{\vec{k},\theta}\right)^T\right)$$

Let us introduce now other standard parametrizations of rotations in 3D.

Euler angles

One of the most frequently used one is so called Euler Angles:



$$(x_0, y_0, z_0)\xrightarrow{\phi}(x_a, y_a, z_a)\xrightarrow{\theta}(x_b, y_b, z_b)\xrightarrow{\psi}(x_1, y_1, z_1)$$

Euler ZYZ-angles (ϕ, θ, and ψ) specify combination of 3 basic rotations about current axes (z_0, $y_1 = y_a$, and $z_2 = z_b$), so that

$$R_{ZYZ} := R_{z,\phi}\,R_{y,\theta}\,R_{z,\psi}$$

According to our previous derivations, we have

$$\boldsymbol{R}_{ZYZ} := \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \cdot \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

that results in

$$\boldsymbol{R}_{ZYZ} := \begin{bmatrix} c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\phi c_\theta s_\psi - s_\phi c_\psi & c_\phi s_\theta \\ s_\phi c_\theta c_\psi + c_\phi s_\psi & -s_\phi c_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta \\ -s_\theta c_\psi & s_\theta s_\psi & c_\theta \end{bmatrix}$$

Let us now solve the inverse problem, i.e. start with an arbitrary representation of a rotation matrix and compute the corresponding Euler angles:

Given the rotation matrix $\boldsymbol{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$,

how to find angles ϕ, θ, ψ?

It is enough to concentrate on the last row and the last column:

$$\boldsymbol{R} := \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\phi c_\theta s_\psi - s_\phi c_\psi & c_\phi s_\theta \\ s_\phi c_\theta c_\psi + c_\phi s_\psi & -s_\phi c_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta \\ -s_\theta c_\psi & s_\theta s_\psi & c_\theta \end{bmatrix}$$

We know that

$$\begin{aligned} {r_{13}}^2 + {r_{23}}^2 + {r_{33}}^2 &= 1 \\ {r_{31}}^2 + {r_{32}}^2 + {r_{33}}^2 &= 1 \end{aligned}$$

and it is useful to consider two cases: $|r_{33}| = 1$ and $|r_{33}| \neq 1$.

Case 1: $r_{33} = \pm 1 \quad \Rightarrow \quad r_{13} = r_{23} = r_{31} = r_{32} = 0$

$$r_{33} = 1 \quad \Rightarrow \quad \cos\theta = 1, \quad \sin\theta = 0$$

$$\Rightarrow \begin{bmatrix} r_{11} & r_{12} & 0 \\ r_{21} & r_{22} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos(\phi+\psi) & -\sin(\phi+\psi) & 0 \\ \sin(\phi+\psi) & \cos(\phi+\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\Rightarrow \boxed{\theta = 0, \quad \phi + \psi = \operatorname{atan2}(r_{21}, r_{11})}$$

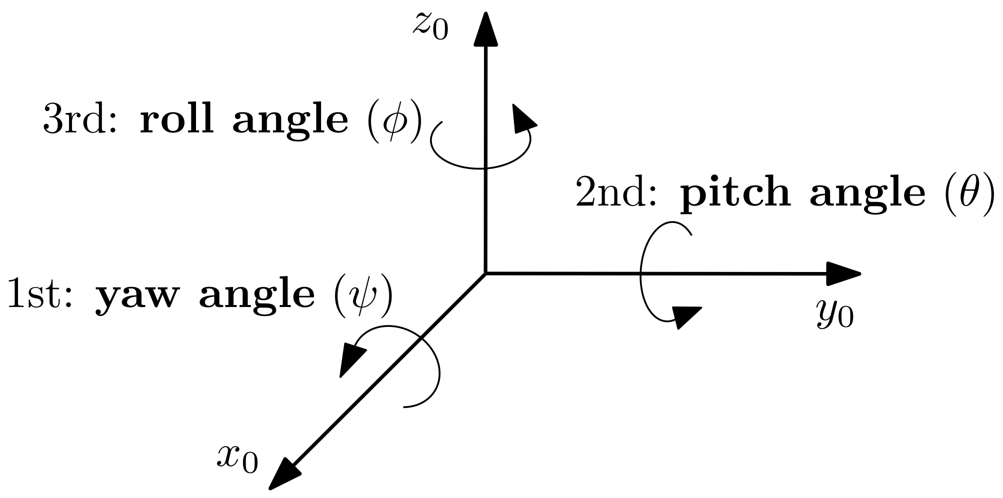
Case 2: ${r_{33}}^2 < 1 \Rightarrow \left\{ {r_{13}}^2 + {r_{23}}^2 \neq 0 \right\}, \left\{ {r_{31}}^2 + {r_{32}}^2 \neq 0 \right\}$

$$r_{33} = \cos\theta, \quad \sin\theta = \pm\sqrt{1 - {r_{33}}^2}$$

$$\phi = \text{atan}\mathbf{2}(-r_{23}, -r_{13}), \quad \psi = \text{atan}\mathbf{2}(r_{32}, -r_{31})$$

Roll, Pitch, and Yaw Angles

Let us introduce another representation, that is associated with decomposition into three rotations with respect to the fixed frame:


z_0
3rd: **roll angle** (ϕ)
2nd: **pitch angle** (θ)
1st: **yaw angle** (ψ)
y_0
x_0

Here, so called, roll, pitch, and yaw angles are angles of **3** rotations about the fixed axes $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{z}$ and so

$$\boldsymbol{R}_{xyz} := \boldsymbol{R}_{z,\phi} \cdot \boldsymbol{R}_{y,\theta} \cdot \boldsymbol{R}_{x,\psi}$$

that is

$$\boldsymbol{R}_{xyz} := \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\psi & -s_\psi \\ 0 & s_\psi & c_\psi \end{bmatrix}$$

or after performing multiplications

$$\boldsymbol{R}_{xyz} := \begin{bmatrix} c_\phi c_\theta & -s_\phi c_\psi + c_\phi s_\theta s_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ s_\phi c_\theta & c_\phi c_\psi + s_\phi s_\theta s_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{bmatrix}$$

It should be noticed that finding the angles from a given matrix representation can be done similarly to the case of Euler angles concentrating on the first row and the first column:

$$\boldsymbol{R}_{xyz} := \begin{bmatrix} c_\phi c_\theta & -s_\phi c_\psi + c_\phi s_\theta s_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ s_\phi c_\theta & c_\phi c_\psi + s_\phi s_\theta s_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{bmatrix}$$

Let us discuss now the case of representations of motions using frames that not necessary have coinciding origins.

2.1.5 Homogeneous Transformations

Let us remind that our goal is to have a simple to use mathematical tool (a machinery) for description of motions of rigid bodies representing links of a robotic manipulator. We will have frames attached (systematically assigned) to each link and it is obvious that in a typical situation they not only change relative orientations but also relative positions of the origins. Such changes are called rigid motions and they are introduced next.

Rigid Motions

Given two frames, a rigid motion is an ordered pair $(\boldsymbol{R}, \vec{\boldsymbol{d}})$, where a rotation tensor $\boldsymbol{R}$ represented in a particular frame by a rotation matrix from $\boldsymbol{\mathcal{SO}}(\mathbf{3})$ describes relative rotation, while $\vec{\boldsymbol{d}}$ is a vector describing translation that can be associated with the directed displacement between the origins of the two corresponding frames. The group of representations of all rigid motions is known as **Special Euclidean Group** denoted by $\boldsymbol{\mathcal{SE}}(\mathbf{3})$.

In general, as clear from the previous discussions, coordinates of a point $\boldsymbol{p}$ in $\mathbf{0}$-frame and $\mathbf{1}$-frame are related as follows

$$\boldsymbol{p}^0 = \boldsymbol{R}_1^0\, \boldsymbol{p}^1 + \boldsymbol{o}_1^0$$

where matrix $\boldsymbol{R}_1^0$ has been introduced above and $\vec{\boldsymbol{d}}_1 = \overrightarrow{\boldsymbol{o}_0\boldsymbol{o}_1}$ is defined by $\boldsymbol{d}_1^0 \equiv \boldsymbol{o}_1^0$.

Correspondingly, if there are three frames: $\mathbf{0}$-frame and $\mathbf{1}$-frame are related by the rigid motions $(\boldsymbol{R}_1, \vec{\boldsymbol{d}}_1)$ while $\mathbf{1}$-frame and $\mathbf{2}$-frame are related by the rigid motion $(\boldsymbol{R}_2, \vec{\boldsymbol{d}}_2)$, we have

$$\boldsymbol{p}^0 = \boldsymbol{R}_1^0\, \boldsymbol{p}^1 + \boldsymbol{d}_1^0, \qquad \boldsymbol{p}^1 = \boldsymbol{R}_2^1\, \boldsymbol{p}^2 + \boldsymbol{d}_2^1$$

so that the composite rigid motion between $\mathbf{0}$-frame and $\mathbf{2}$-frame is defined as

$$\boldsymbol{p}^0 = \boldsymbol{R}_1^0 \boldsymbol{R}_2^1 \boldsymbol{p}^2 + \boldsymbol{R}_1^0 \boldsymbol{d}_2^1 + \boldsymbol{d}_1^0$$

Matrix representation of rigid motions

Homogeneous transformation, which is to be introduced next, is just a more convenient way to rewrite the formula

$$\boldsymbol{p}^0 = \boldsymbol{R}_2^0\, \boldsymbol{p}^2 + \boldsymbol{d}_2^0, \qquad \boldsymbol{R}_2^0 = \boldsymbol{R}_1^0\, \boldsymbol{R}_2^1, \quad \boldsymbol{d}_2^0 = \boldsymbol{R}_1^0\, \boldsymbol{d}_2^1 + \boldsymbol{d}_1^0$$

as a product of matrices even in the case when all the origins are different.

The following identity can be easily verified

$$\begin{bmatrix} R_1^0 & d_1^0 \\ 0_{1\times 3} & 1 \end{bmatrix} \begin{bmatrix} R_2^1 & d_2^1 \\ 0_{1\times 3} & 1 \end{bmatrix} = \begin{bmatrix} R_1^0 R_2^1 & R_1^0 d_2^1 + d_1^0 \\ 0_{1\times 3} & 1 \end{bmatrix} \equiv \begin{bmatrix} R_2^0 & d_2^0 \\ 0_{1\times 3} & 1 \end{bmatrix}$$

To use it, for a rigid motion defined in 0-frame by $(R^0, d^0) \in \mathcal{SE}(3)$, we introduce the following 4×4-matrix

$$H^0 = \begin{bmatrix} R^0 & d^0 \\ 0_{1\times 3} & 1 \end{bmatrix}$$

which is called **homogeneous transformation** associated with (R^0, d^0).

Obviously, with such matrices we have composition of two rigid motions represented as a product

$$H_1^0 \, H_2^1 = H_2^0$$

Now, to directly use homogeneous transformations for computing coordinates of points, we need to extend the coordinates of a point p in 0-frame, p^0, and in 1-frame, p^1, by the auxiliary fourth coordinate taken as 1:

$$P^0 = \begin{bmatrix} p^0 \\ 1 \end{bmatrix}, \qquad P^1 = \begin{bmatrix} p^1 \\ 1 \end{bmatrix}$$

Then,

$$P^0 = \begin{bmatrix} p^0 \\ 1 \end{bmatrix} = \begin{bmatrix} R_1^0 p^1 + d^0 \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} R_1^0 & d^0 \\ 0_{1\times 3} & 1 \end{bmatrix}}_{H_1^0} \underbrace{\begin{bmatrix} p^1 \\ 1 \end{bmatrix}}_{P^1}$$

and the change of coordinates formula

$$p^0 = R_1^0 \, p^1 + d^0 \qquad \text{becomes} \qquad \boxed{P^0 = H_1^0 \, P^1}$$

or, in more details,

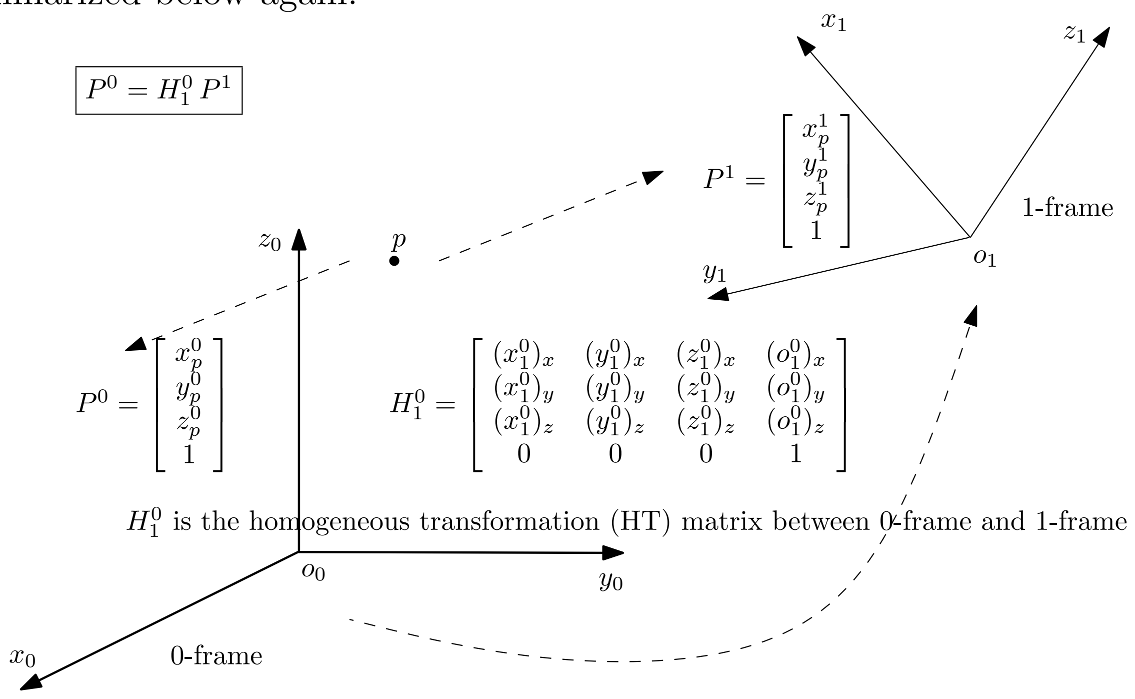
$$\underbrace{\begin{bmatrix} p^0 \\ 1 \end{bmatrix}}_{P^0} = \boxed{\begin{bmatrix} x_p^0 \\ y_p^0 \\ z_p^0 \\ 1 \end{bmatrix} = \begin{bmatrix} (x_1^0)_x & (y_1^0)_x & (z_1^0)_x & (o_1^0)_x \\ (x_1^0)_y & (y_1^0)_y & (z_1^0)_y & (o_1^0)_y \\ (x_1^0)_z & (y_1^0)_z & (z_1^0)_z & (o_1^0)_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_p^1 \\ y_p^1 \\ z_p^1 \\ 1 \end{bmatrix}} = \underbrace{\begin{bmatrix} R_1^0 & o_1^0 \\ 0_{1\times 3} & 1 \end{bmatrix}}_{H_1^0} \underbrace{\begin{bmatrix} p^1 \\ 1 \end{bmatrix}}_{P^1}$$

Note that the whole point of introducing an extra auxiliary (dummy) row above is the following: The formula for a composition of two rigid motions

$$\boldsymbol{p}^0 = \boldsymbol{R}_1^0\,\boldsymbol{R}_2^1\,\boldsymbol{p}^2 + \boldsymbol{R}_1^0\,\boldsymbol{d}_2^1 + \boldsymbol{d}_1^0 \qquad \text{becomes} \qquad \boldsymbol{P}^0 = \boldsymbol{H}_1^0\,\boldsymbol{H}_2^1\,\boldsymbol{P}^2$$

and now it can be performed (encoded) using multiplication of two $(\mathbf{4}\times\mathbf{4})$ matrices.

From now on, we will use homogeneous transformation (HT) matrices to represent relations among various frames and to describe rigid motions. The corresponding change of the coordinates of a point formula above is summarized below again.



$\boxed{P^0 = H_1^0\,P^1}$
x_1
z_1
$P^1 = \begin{bmatrix} x_p^1 \\ y_p^1 \\ z_p^1 \\ 1 \end{bmatrix}$
1-frame
y_1
o_1
z_0
p
$P^0 = \begin{bmatrix} x_p^0 \\ y_p^0 \\ z_p^0 \\ 1 \end{bmatrix}$
$H_1^0 = \begin{bmatrix} (x_1^0)_x & (y_1^0)_x & (z_1^0)_x & (o_1^0)_x \\ (x_1^0)_y & (y_1^0)_y & (z_1^0)_y & (o_1^0)_y \\ (x_1^0)_z & (y_1^0)_z & (z_1^0)_z & (o_1^0)_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$
H_1^0 is the homogeneous transformation (HT) matrix between 0-frame and 1-frame
o_0
y_0
x_0
0-frame

2.2 Forward Kinematics

With the mathematical description of rigid motions through homogeneous transformation (HT), we are ready to approach the problems of kinematics in a systematic way.

2.2.1 Kinematic Chains

Let us recall our basic assumptions and terminology:

- A robot manipulator is composed of a set of links (rigid bodies) connected together by joints;
- Each joint can be either

   - a **revolute joint** (a rotation of a link by an angle about an axis fixed on another link) or
   - a **prismatic joint** (a displacement of a link along a single axis fixed on another link)

  and cannot be

   - a more complex $\mathbf{2}$-DOF or $\mathbf{3}$-DOF joint, which assumed to be represented as a **combinations of the ones above** with introduction of fictitious mass-less and length-less links.

- Each joint connects two links. A robot manipulator with $\boldsymbol{n}$ joints will have $\boldsymbol{(n+1)}$ links;
- We number joints from $\mathbf{1}$ to $\boldsymbol{n}$, and links from $\mathbf{0}$ to $\boldsymbol{n}$. So that **joint $\boldsymbol{i}$ connects links $\boldsymbol{(i-1)}$ and $\boldsymbol{i}$**;
- The location of joint $\boldsymbol{i}$ is fixed with respect to the link $\boldsymbol{(i-1)}$.

For a robotic manipulator with $\boldsymbol{n}$ joints, we attach $\boldsymbol{n+1}$-frames. Each $\boldsymbol{i}$-frame is attached to the $\boldsymbol{i}$-th link. So, we should have the moving frames

$$(o_0, x_0, y_0, z_0), \quad (o_1, x_1, y_1, z_1), \; \ldots, \; (o_n, x_n, y_n, z_n)$$

Each point of $\boldsymbol{i}$-th link will be defined by fixed coordinates in the corresponding $\boldsymbol{i}$-frame. When manipulator is in motion, coordinates of all the points of various links in the fixed to the base of the robot $\mathbf{0}$-frame will be changing. They will be defined by the relative rigid motions of the links.

Relative orientations of the $\boldsymbol{n+1}$ frames is to be defined by $\boldsymbol{n}$ rotation matrices

$$R_1^0, \quad R_2^1, \quad \ldots, \quad R_{(n)}^{(n-1)}$$

that represent consecutive rotation between the current frames:

$$\Big\{\text{from } (x_0y_0z_0) \text{ to } (x_1y_1z_1)\Big\}, \quad \Big\{\text{from } (x_1y_1z_1) \text{ to } (x_2y_2z_2)\Big\}, \quad \ldots,$$
$$\Big\{\text{from } (x_{n-1}y_{n-1}z_{n-1}) \text{ to } (x_ny_nz_n)\Big\} \quad \text{respectively,}$$

while relative positions of the origins of the frames should be defined by

$$o_1^0, \qquad o_2^1, \qquad ,\ldots, \qquad o_n^{n-1}$$

giving coordinates of the origin of each frame with respect to the previous one. This information is to be used to define the HT between various couples of frames.

We will use the next basic assumptions and terminology:

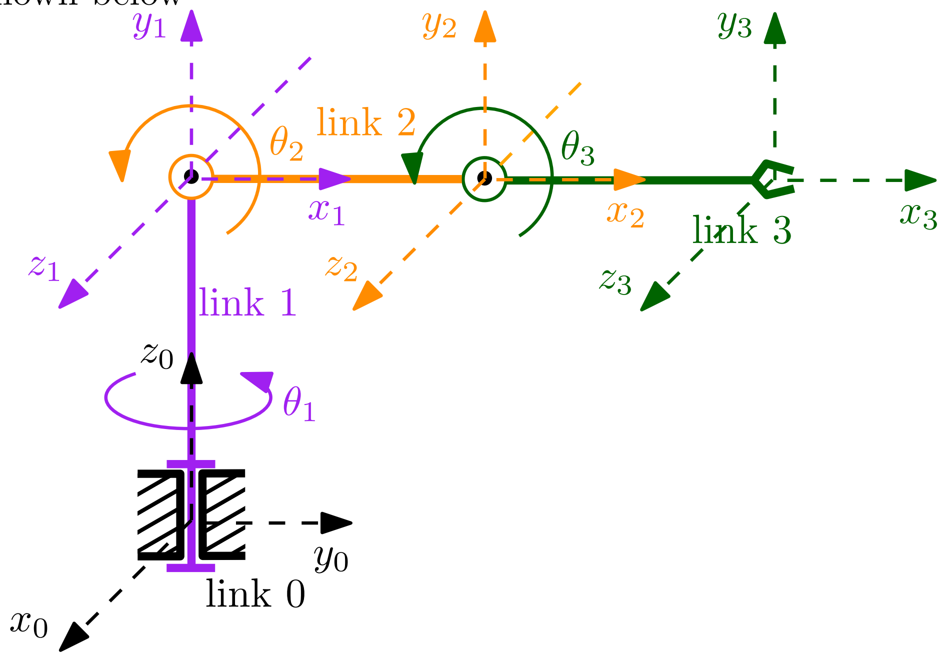
- The link $\mathbf{0}$ (the base of the robot) is fixed;
- With the $\boldsymbol{i}^{th}$ joint we associate a **joint variable**

$$\boldsymbol{q_i} = \begin{cases} \boldsymbol{\theta_i} & \text{if joint } \boldsymbol{i} \text{ is revolute} \\ \boldsymbol{d_i} & \text{if joint } \boldsymbol{i} \text{ is prismatic} \end{cases}$$

  defining the angle of rotation or the linear displacement respectively.

- For each **link $\boldsymbol{i}$** (with $\boldsymbol{i} = \mathbf{0}, \mathbf{1}, \ldots, \boldsymbol{n}$) we have a rigidly attached coordinate frame $(\boldsymbol{o_i x_i y_i z_i})$ described above;
- When joint $\boldsymbol{i}$ is actuated, the **link $\boldsymbol{i}$** and its frame experience a relative **rigid motion** defined by the corresponding HT;
- The frame $(\boldsymbol{o_0 x_0 y_0 z_0})$ attached to the base is referred to as the **inertial frame**.

An example of a set of coordinate frames attached to an elbow manipulator is shown below



We will use the next basic assumptions and terminology:

- Suppose $\boldsymbol{A_i} \equiv \boldsymbol{H}_i^{i-1}$ is a homogeneous transformation that gives
  - the relative **position** of the origin and
  - the relative **orientation** of the axes

  of the frame $(\boldsymbol{o_i x_i y_i z_i})$ with respect to the frame $(\boldsymbol{o_{i-1} x_{i-1} y_{i-1} z_{i-1}})$;

- The HT matrix $\boldsymbol{A_i} \equiv \boldsymbol{H}_i^{i-1} = \begin{bmatrix} \boldsymbol{R}_i^{i-1} & \boldsymbol{o}_i^{i-1} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$ is changing as the configuration of the robot is changing;

- $\boldsymbol{A}_i$ is the function of a scalar variable $\boldsymbol{q}_i$, which is the joint variable, i.e. $\boldsymbol{A}_i = \boldsymbol{A}_i(\boldsymbol{q}_i)$;

- Homogeneous transformation that expresses the position and orientation of $(\boldsymbol{o}_j \boldsymbol{x}_j \boldsymbol{y}_j \boldsymbol{z}_j)$ with respect to $(\boldsymbol{o}_i \boldsymbol{x}_i \boldsymbol{y}_i \boldsymbol{z}_i)$, i.e. $\boldsymbol{H}_j^i$, can be computed as follows

$$\boldsymbol{H}_j^i = \begin{cases} \boldsymbol{A}_{i+1}\boldsymbol{A}_{i+2}\dots\boldsymbol{A}_{j-1}\boldsymbol{A}_j, & \text{if } i < j \\ \boldsymbol{I}, & \text{if } i = j \end{cases}, \quad \boldsymbol{H}_j^i = \left(\boldsymbol{H}_i^j\right)^{-1}, \text{ if } i > j$$

and is called a **transformation matrix**.

To solve the forward kinematics problem, we need to compute the position and the orientation of the end-effector with respect to the inertia frame, i.e.

$$\boldsymbol{o}_n^0 \qquad \text{and} \qquad \boldsymbol{R}_n^0$$

Obviously, they can be extracted from the homogeneous transformation

$$\boldsymbol{H}_n^0 = \boldsymbol{A}_1(\boldsymbol{q}_1)\,\boldsymbol{A}_2(\boldsymbol{q}_2)\dots\boldsymbol{A}_{n-1}(\boldsymbol{q}_{n-1})\,\boldsymbol{A}_n(\boldsymbol{q}_n) = \begin{bmatrix} \boldsymbol{R}_n^0 & \boldsymbol{o}_n^0 \\ \boldsymbol{0} & \boldsymbol{1} \end{bmatrix}$$

Recall that for $j > i$ we have

$$\boldsymbol{H}_j^i = \boldsymbol{A}_{i+1}\boldsymbol{A}_{i+2}\dots\boldsymbol{A}_{j-1}\boldsymbol{A}_j = \begin{bmatrix} \boldsymbol{R}_j^i & \boldsymbol{o}_j^i \\ \boldsymbol{0} & \boldsymbol{1} \end{bmatrix}$$

where the following recursive definition can be verified

$$\boldsymbol{R}_j^i = \boldsymbol{R}_{i+1}^i \dots \boldsymbol{R}_j^{j-1}, \qquad \boldsymbol{o}_j^i = \boldsymbol{o}_{j-1}^i + \boldsymbol{R}_{j-1}^i \boldsymbol{o}_j^{j-1} \quad \text{for} \quad j > i+1$$

In order to solve the forward kinematics problem, one can start by assigning frames to all the links. This can be done in a variety of ways. However, arbitrary assignment will lead to the need of introducing the transformation between the frames depending on the joint variable in a complex way. This is not convenient and may lead to unnecessarily lengthy calculations. To avoid such complexity and to standardize the frame assignment, a convention introduced next is commonly used in robotics.

2.2.2 The Denavit-Hartenberg (DH) Convention

First of all, it should be noticed that a transformation between two frames can be defined by **6** parameters (**3** for the relative position of the origin and

3 for the relative orientation of the axes). However, the necessary number of parameters may be reduced using specific relation between the consecutive frames if they are assigned in a certain systematic way.

Second of all, it seems useful to have a procedure for frame assignment that ensures that:

- one of the parameters is the joint variable $\boldsymbol{q}$ (the angle $\boldsymbol{\theta}$ for every revolute joint and the position $\boldsymbol{d}$ for every prismatic joint),
- the other parameters are constant,
- the HT between each pair of frames is computed in a unified systematic way (for example as a product of basic rotations or translations),
- the number of parameters is as small as possible.

From these desirable requirements, it appears reasonable to have one axis of each frame, say $\boldsymbol{z}$, to be directed along the action of rotation or translation. In this case, two of the basic HT should be the rotation around $\boldsymbol{z}$ by $\boldsymbol{\theta}$ (to be denoted as $\mathrm{Rot}_{\boldsymbol{z},\boldsymbol{\theta}_i}$) and the translation along $\boldsymbol{z}$ by $\boldsymbol{d}$ (to be denoted as $\mathrm{Trans}_{\boldsymbol{z},\boldsymbol{d}_i}$). How many other basic HT we would need to include into a product representation and which one we should choose?

It can be shown that, using the structure of kinematic chains, it is possible to define transformations between the frames by just **4** parameters: $\boldsymbol{\theta}_i$, $\boldsymbol{d}_i$, $\boldsymbol{a}_i$, and $\boldsymbol{\alpha}_i$ and to represent each HT matrix $\boldsymbol{A}_i$ as a product of the following **4** special elementary HT matrices (with $\boldsymbol{q}_i = \boldsymbol{\theta}_i$ or $\boldsymbol{q}_i = \boldsymbol{d}_i$)

$$\boldsymbol{A}_i \equiv \boldsymbol{H}_i^{i-1}(\boldsymbol{q}_i) = \mathrm{Rot}(\boldsymbol{z}, \boldsymbol{\theta}_i)\,\mathrm{Trans}(\boldsymbol{z}, \boldsymbol{d}_i)\,\mathrm{Trans}(\boldsymbol{x}, \boldsymbol{a}_i)\,\mathrm{Rot}(\boldsymbol{x}, \boldsymbol{\alpha}_i)$$

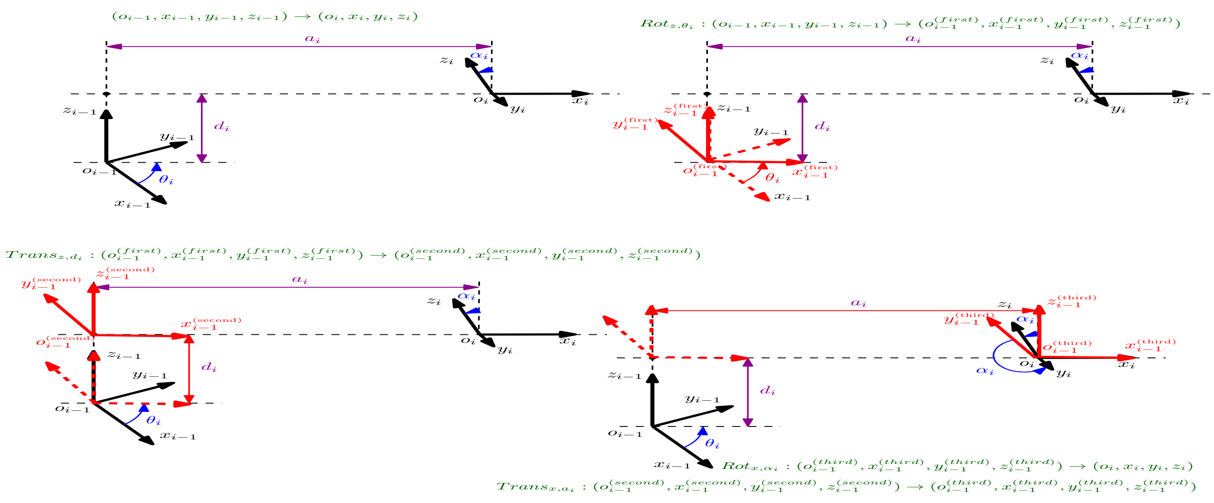
$$\boldsymbol{A}_i = \left[\begin{array}{ccc|c} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \left[\begin{array}{ccc|c} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

parameters of which are defined next.

The product is

$$\boldsymbol{A}_i = \left[\begin{array}{ccc|c} \cos(\boldsymbol{\theta}_i) & -\sin(\boldsymbol{\theta}_i)\cos(\boldsymbol{\alpha}_i) & \sin(\boldsymbol{\theta}_i)\sin(\boldsymbol{\alpha}_i) & \boldsymbol{a}_i\cos(\boldsymbol{\theta}_i) \\ \sin(\boldsymbol{\theta}_i) & \cos(\boldsymbol{\theta}_i)\cos(\boldsymbol{\alpha}_i) & -\cos(\boldsymbol{\theta}_i)\sin(\boldsymbol{\alpha}_i) & \boldsymbol{a}_i\sin(\boldsymbol{\theta}_i) \\ 0 & \sin(\boldsymbol{\alpha}_i) & \cos(\boldsymbol{\alpha}_i) & \boldsymbol{d}_i \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

Essentially, we are also to formulate **2** conditions on the pairwise relation between the assigned frames to remove the need to perform **2** actions when changing between coordinates expressed in them: translation along and rotation around the current $\boldsymbol{y}$-axis. The sequence of transformations from frame $\boldsymbol{i-1}$ to frame $\boldsymbol{i}$ is illustrated below.


$(o_{i-1}, x_{i-1}, y_{i-1}, z_{i-1}) \rightarrow (o_i, x_i, y_i, z_i)$
$Rot_{z,\theta_i} : (o_{i-1}, x_{i-1}, y_{i-1}, z_{i-1}) \rightarrow (o_{i-1}^{(first)}, x_{i-1}^{(first)}, y_{i-1}^{(first)}, z_{i-1}^{(first)})$
$Trans_{z,d_i} : (o_{i-1}^{(first)}, x_{i-1}^{(first)}, y_{i-1}^{(first)}, z_{i-1}^{(first)}) \rightarrow (o_{i-1}^{(second)}, x_{i-1}^{(second)}, y_{i-1}^{(second)}, z_{i-1}^{(second)})$
$Rot_{x,\alpha_i} : (o_{i-1}^{(third)}, x_{i-1}^{(third)}, y_{i-1}^{(third)}, z_{i-1}^{(third)}) \rightarrow (o_i, x_i, y_i, z_i)$
$Trans_{x,a_i} : (o_{i-1}^{(second)}, x_{i-1}^{(second)}, y_{i-1}^{(second)}, z_{i-1}^{(second)}) \rightarrow (o_{i-1}^{(third)}, x_{i-1}^{(third)}, y_{i-1}^{(third)}, z_{i-1}^{(third)})$
a_i
d_i
θ_i
α_i
o_{i-1}
x_{i-1}
y_{i-1}
z_{i-1}
o_i
x_i
y_i
z_i

DH parameters and DH conditions

The $\mathbf{4}$ parameters of the DH transformation **from frame $\boldsymbol{i-1}$ to frame $\boldsymbol{i}$**, for a particular assignment of the axes to be discussed shortly, are known as

- $\boldsymbol{\theta_i}$: the **link angle** (that is constant for prismatic joints), which is the constant angle between the unit vectors $\vec{\boldsymbol{x}}_{\boldsymbol{i-1}}$ and $\vec{\boldsymbol{x}}_{\boldsymbol{i}}$, i.e. it is the angle of the rotation around $\vec{\boldsymbol{z}}_{\boldsymbol{i-1}}$ (the first transformation: $\mathrm{Rot}_{\boldsymbol{z},\boldsymbol{\theta_i}}$);

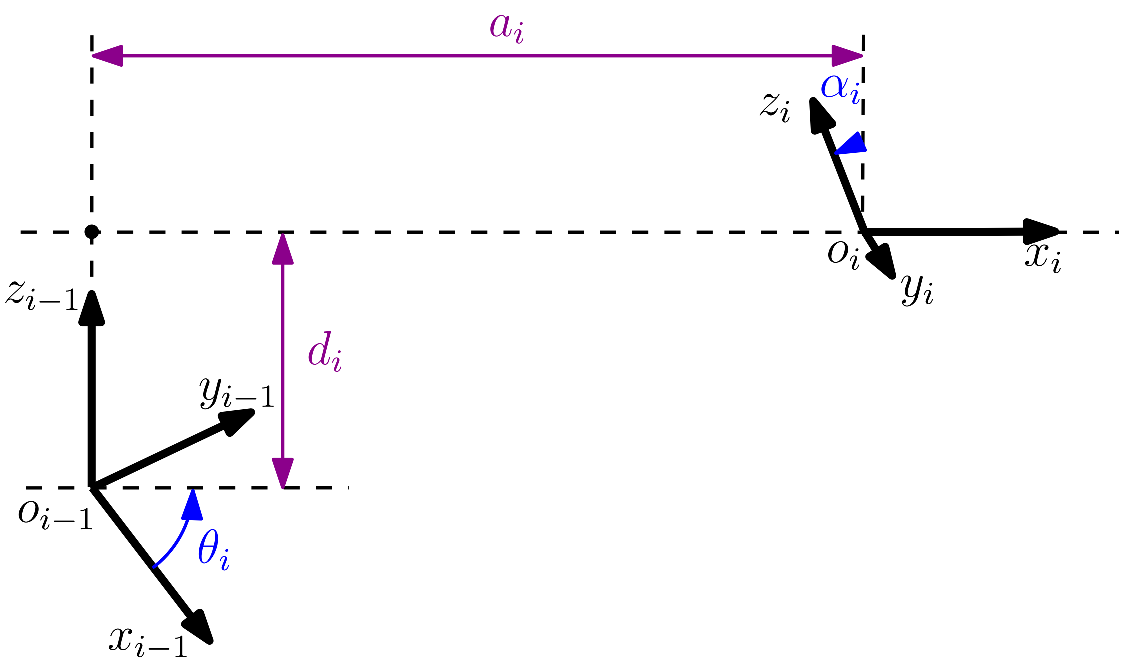
- $\boldsymbol{d_i}$: the **link offset** (that is constant for revolute joints), which is the constant distance from the previous origin $\boldsymbol{o_{i-1}}$ to the current axis $\boldsymbol{x_i}$, i.e. it is the displacement along $\vec{\boldsymbol{z}}_{\boldsymbol{i-1}}$ (the second: $\mathrm{Trans}_{\boldsymbol{z},\boldsymbol{d_i}}$);

- $\boldsymbol{a_i}$: the constant **link length**, which is the constant distance from the current origin $\boldsymbol{o_i}$ to the action axis of the previous frame $\boldsymbol{z_{i-1}}$, i.e. it is the displacement along $\vec{\boldsymbol{x}}_{\boldsymbol{i}}$ (the third: $\mathrm{Trans}_{\boldsymbol{x},\boldsymbol{a_i}}$);

- $\boldsymbol{\alpha_i}$: the constant **link twist**, which is the constant angle between the unit vectors defining the previous and current action axes, i.e. $\vec{\boldsymbol{z}}_{\boldsymbol{i-1}}$ and $\vec{\boldsymbol{z}}_{\boldsymbol{i}}$, i.e. it is the angle of the rotation around $\vec{\boldsymbol{x}}_{\boldsymbol{i}}$ (the last: $\mathrm{Rot}_{\boldsymbol{x},\boldsymbol{\alpha_i}}$).

See also https://www.youtube.com/watch?v=rA9tm0gTln8.

The parameters $\boldsymbol{a_i, \alpha_i, d_i, \theta_i}$ are identified on the next picture together with the two conditions on the way the frames $\boldsymbol{i-1}$ and $\boldsymbol{i}$ are assigned to allow reduction of the number of parameters by $\mathbf{2}$.



The two conditions are the following:

DH1: The axis $\boldsymbol{x_i}$ is perpendicular to the axis $\boldsymbol{z_{i-1}}$

DH2: The axis $\boldsymbol{x_i}$ intersects the axis $\boldsymbol{z_{i-1}}$

Interestingly enough, these two conditions, can be satisfied for every open kinematic chain.

Assigning Frames Following the DH Convention

Given a robot manipulator with

- $\boldsymbol{n}$ revolute and/or prismatic joints and, correspondingly,
- $\boldsymbol{(n+1)}$ links, numbered sequentially from $\boldsymbol{0}$ to $\boldsymbol{n+1}$, starting from the inertial base link and up to the last one holding the end-effector.

We describe below the commonly used **procedure** for systematically assigning coordinate frames to each link that ensures that the transformation between the frames can be written using the DH convention.
The algorithm of assigning $\boldsymbol{(n+1)}$ frames for $\boldsymbol{(n+1)}$ links

- treats separately first $\boldsymbol{n}$-frames and the last one (the end-effector frame),
- is recursive in first part, so that it is generic.

The $\boldsymbol{i^{th}}$ frame, i.e. the origin $\boldsymbol{o_i}$ and the three mutually orthogonal axes $\boldsymbol{x_i}$, $\boldsymbol{y_i}$, and $\boldsymbol{z_i}$, are to be attached to the $\boldsymbol{i^{th}}$ link and move together with it completely defining the motion of link $\boldsymbol{i}$ relative to link $\boldsymbol{i-1}$.

Assigning First n-Frames $(i = 0, 1, \dots, n-1)$ can be done in the next three steps.

Step 1 (Choice of z-axes):

- Choose **z_0-axis** along the **actuation line of the 1st-link**;
- Choose **z_1-axis** along the **actuation line of the 2nd-link**;
- ...
- Choose **z_{n-1}-axis** along the **actuation line of the n^{th}-link**;
- Leave z_n-axis not assigned for now.

After this step is done, it is left to assign:

- A **point** on each of z_i-axis that will be **the origin** o_i of the i^{th}-frame for $i = 1, \dots, n-1$ (Step 2),
- An **x_i-axis** for each frame (Step 2) so that the two DH-conditions hold, i.e. so that

  DH1: The axis x_i is perpendicular to the axis z_{i-1},

  DH2: The axis x_i intersects the axis z_{i-1}.
- An **y_i-axis** for each frame to have them right-oriented (Step 3)
- The last n^{th} (tool) frame in a convenient way.

Step 2 (Choice of x-axes and the origins): The axis x_0 and the origin of the base frame (o_0) are chosen arbitrary. For the other x-axes ($i = 1, 2, \dots, n-1$) this step is done recursively as follows.

- Suppose that we have chosen the $(i-1)^{th}$-frame and need to proceed with the i^{th}-frame.
- Note that for the i^{th}-frame, the z_i axis is already fixed.
- We need to ensure that:

  (1) **x_i intersects z_{i-1}**,

  (2) $x_i \perp z_{i-1}$,

  (3) $x_i \perp z_i$.

There are **3** possible cases:

(a) z_i and z_{i-1} are not coplanar, i.e. there is no plane that contain both axis

(b) $\boldsymbol{z}_i$ and $\boldsymbol{z}_{i-1}$ are parallel,

(c) $\boldsymbol{z}_i$ and $\boldsymbol{z}_{i-1}$ intersect,

that should be treated differently, as described below.

(a) **If z_i and z_{i-1} are not coplanar,** then there is the unique common perpendicular (direction of which is defined by $\vec{z}_i \times \vec{z}_{i-1}$) for both lines! Let as assign the $\boldsymbol{x}_i$-axis along this common perpendicular and the origin $\boldsymbol{o}_i$ for the $\boldsymbol{i}^{th}$-frame at the point of intersection of $\boldsymbol{z}_i$ and $\boldsymbol{x}_i$.

(b) **If z_i and z_{i-1} are parallel,** then there are many common perpendiculars for both lines! We can choose any one of them to define the origin $\boldsymbol{o}_i$ and the $\boldsymbol{x}_i$-axis for the $\boldsymbol{i}^{th}$-frame.

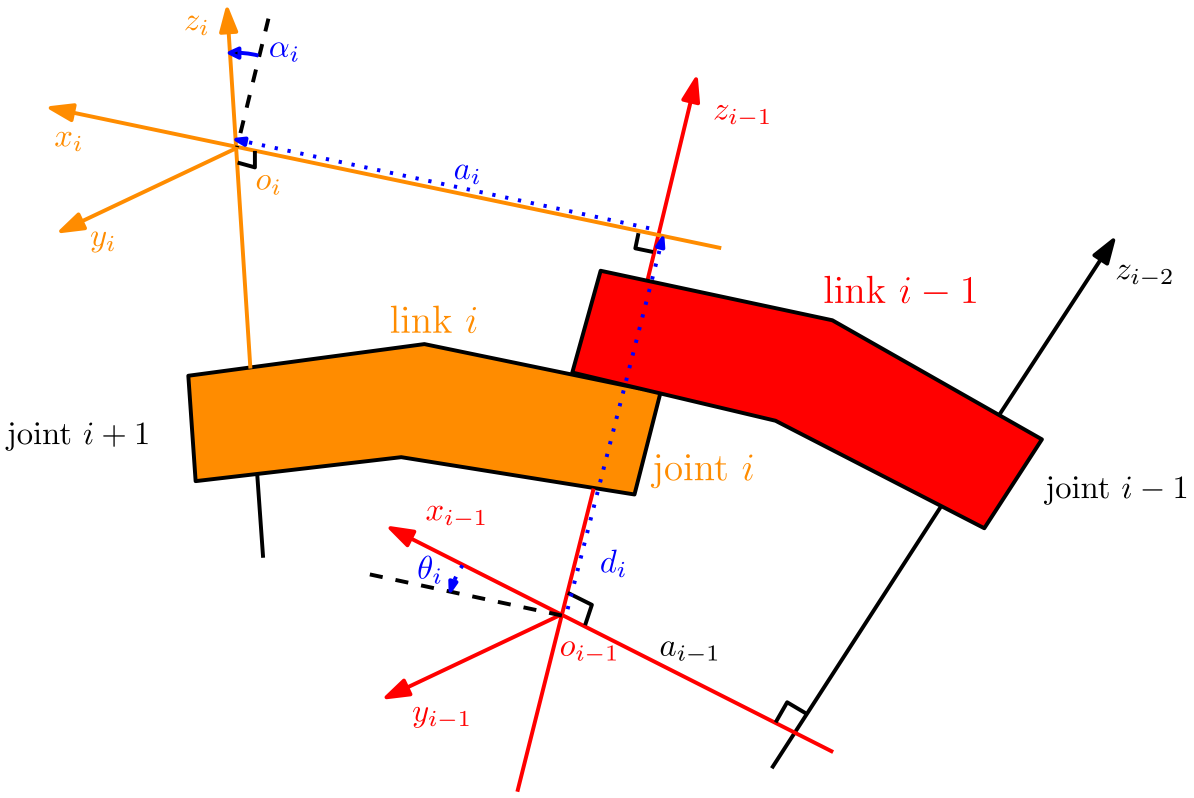
(c) **If z_i and z_{i-1} intersect,** then there is a vector orthogonal to the plane formed by $\boldsymbol{z}_i$ and $\boldsymbol{z}_{i-1}$! Let us take the point of intersection to be the new origin $\boldsymbol{o}_i$, while the $\boldsymbol{x}_i$-axis should follow this common perpendicular (defined by $\vec{z}_i \times \vec{z}_{i-1}$).

Note that, whenever possible, we just take either $\vec{x}_i = \vec{z}_i \times \vec{z}_{i-1}$ or $\vec{x}_i = \vec{z}_{i-1} \times \vec{z}_i$ to ensure (2) and (3) above.

Step 3 (Choice of $\boldsymbol{y}$-axes):

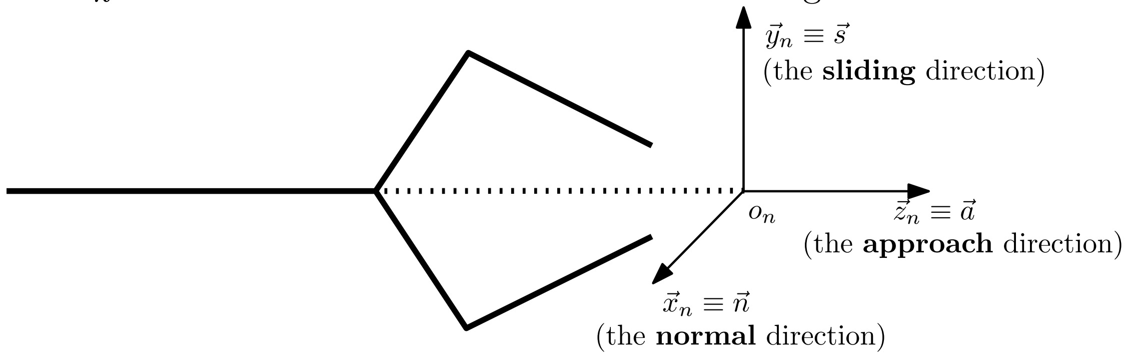
Suppose we have already chosen the axes $\boldsymbol{z}_i$ and $\boldsymbol{x}_i$ as well as the point $\boldsymbol{o}_i$ for the $\boldsymbol{i}^{th}$-frame. Than, we have the only possibility to obtain the right-handed coordinate system (i.e. the system with $\vec{x}_i \times \vec{y}_i = \vec{z}_i$, $\vec{y}_i \times \vec{z}_i = \vec{x}_i$, and $\vec{z}_i \times \vec{x}_i = \vec{y}_i$). We simply take $\boldsymbol{y}_i$ along $\vec{y}_i$ defined by the right-hand rule of the cross-product operation: $\vec{y}_i = \vec{z}_i \times \vec{x}_i$.

Note that as soon as all the frames are assigned, we can compute the DH parameters as described earlier and illustrated below.



Note, however, that we cannot use the procedure above for the last end-effector frame since we can not start assigning $\boldsymbol{z_n}$ along the actuation line of the not existing next link.

Assigning the Last Frame ($\boldsymbol{i = n}$) for the End-Effector. For most robots it is common to take the $\boldsymbol{z_n}$ axis coinciding with the $\boldsymbol{z_{n-1}}$ axis, while $\boldsymbol{y_n}$ and $\boldsymbol{x_n}$ axes can be taken as shown in the next figure


$\vec{y}_n \equiv \vec{s}$
(the **sliding** direction)
o_n
$\vec{z}_n \equiv \vec{a}$
(the **approach** direction)
$\vec{x}_n \equiv \vec{n}$
(the **normal** direction)

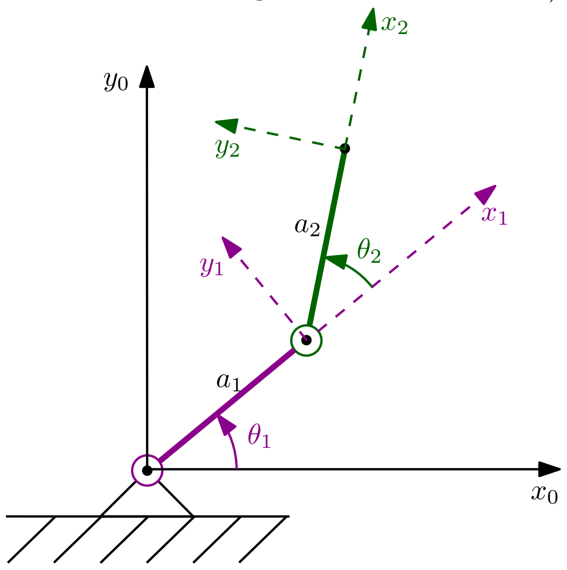
Then, the transformation between the $\boldsymbol{(n-1)}$-frame and the $\boldsymbol{n}$-frame corresponds to

- translation by $\boldsymbol{d_n}$ along $\boldsymbol{z_{n-1}}$-axis and
- rotation by $\boldsymbol{\theta_n}$ about $\boldsymbol{z_n}$-axis.

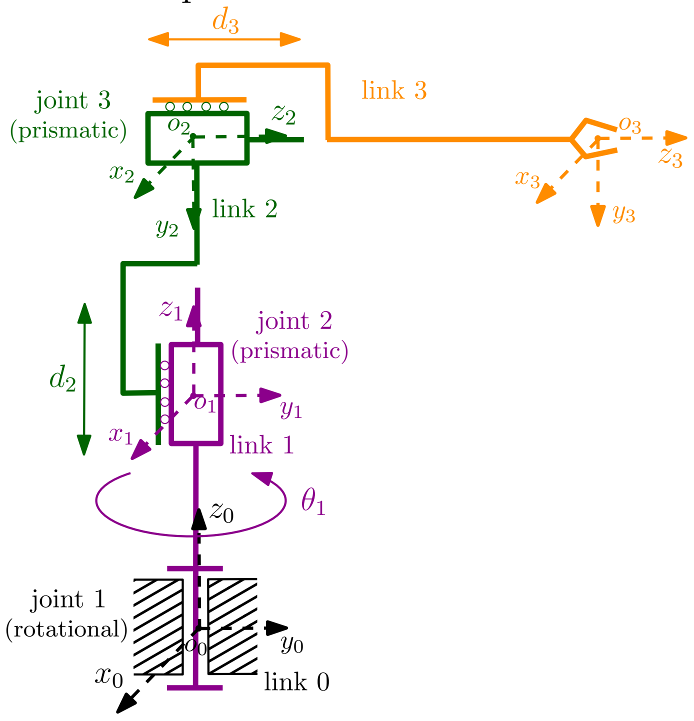
In general, whenever possible, one should make a choice that would lead to **simplifying the description of the transition between the frames**, i.e. if the actuation axis are parallel, their directions should be chosen identical, i.e. $\vec{\boldsymbol{z}}_i = \vec{\boldsymbol{z}}_{i-1}$, and if it is possible, without violating the DH conditions, one should assign $\vec{\boldsymbol{x}}_i = \vec{\boldsymbol{x}}_{i-1}$.

2.2.3 Examples of using DH convention

An example of the result of assigning the frames following our procedure for the planar two-link manipulator, considered in the introduction, is shown below (the axes $\boldsymbol{z_1}$ and $\boldsymbol{z_2}$ are orthogonal to the plane).



An example of the result of assigning the frames following our procedure for a **3**-link cylindrical manipulator is shown below.



From the picture above, one can extract the following set of

DH-parameters:

Link	$\boldsymbol{\theta_i}$	$\boldsymbol{d_i}$	$\boldsymbol{a_i}$	$\boldsymbol{\alpha_i}$
1	$\boldsymbol{\theta_1}$	$\boldsymbol{d_1}$	**0**	**0**
2	0	$\boldsymbol{d_2}$	**0**	$\boldsymbol{-\frac{\pi}{2}}$
3	0	$\boldsymbol{d_3}$	**0**	**0**

As soon as we know the DH parameters, we can proceed using the formulae above for computation of the HT matrices relating the frames. For the **3**-link cylindrical manipulator we obtain:

$$A_1 = \left[\begin{array}{ccc|c} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ \hline 0 & 0 & 0 & 1 \end{array}\right], \qquad A_3 = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ \hline 0 & 0 & 0 & 1 \end{array}\right],$$

$$A_2 = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & \cos(-\frac{\pi}{2}) & -\sin(-\frac{\pi}{2}) & 0 \\ 0 & \sin(-\frac{\pi}{2}) & \cos(-\frac{\pi}{2}) & d_2 \\ \hline 0 & 0 & 0 & 1 \end{array}\right],$$

and finally,

$$T_3^0 = A_1\, A_2\, A_3 .$$

In analogous way, one can assign frames for any manipulator an compute the transformations among the frames. As soon as this is done, we have a solution of the forward kinematics problem.

2.3 Inverse Kinematics

Let us consider now the inverse problem.

2.3.1 Problem Formulation

Given a 4×4 matrix of homogeneous transformation

$$H = \left[\begin{array}{c|c} R & o \\ \hline 0_{3\times 1} & 1 \end{array}\right] \in \mathcal{SE}(3)$$

obtained as a solution of the forward kinematics problem (i.e. $R = R_n^0$ and $o = o_n^0$), our goal is to find a solution (possibly one of many) of the equation

$$H_n^0(q_1, \ldots, q_n) = A_1(q_1)A_2(q_2)\cdots A_n(q_n) = H$$

So, we have **12** equations with respect to n variables: q_1, q_2, $\ldots$, q_n. However, these equations are not independent since **9** entries of the rotation matrix satisfy certain equations and can be defined by just **3** independent variables, such as Euler angles. Hence, if $n < 6$ either not every target position or not every orientation can be achieved; while if $n > 6$ there may be several solutions for particular target position and orientation.

Note that it is often advantageous to find a solution of

$$\boldsymbol{H}_n^0(q_1, \ldots, q_n) = \boldsymbol{H} = \left[\begin{array}{ccc|c} h_{11} & h_{12} & h_{13} & h_{14} \\ h_{21} & h_{22} & h_{23} & h_{24} \\ h_{31} & h_{32} & h_{33} & h_{34} \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

in an analytical form

$$q_k = f_k(h_{11}, h_{12}, \ldots, h_{34}), \quad k = 1, \ldots, n$$

since they should be often computed as fast as possible.

In fact, $\boldsymbol{H}$, as a function of time, defines the required for a particular task position of the end-effector, specified by $\boldsymbol{o}$, and the orientation of the frame attached to the end-effector, specified by $\boldsymbol{R}$. Therefore, the computed from the solution $q_k(t)$ become references to be enforced by actuators controlling the joint variables. Therefore, if there are several solutions, then an on-line numerical procedure may result in switching among different solutions and consequently in **possible jumps in reference signals** that lead to a challenging task for control design since no control action can result in discontinuous trajectories.

However one should keep in mind that, although highly desirable, finding an analytic solution is not always possible. An important situation when this can be done is considered next.

2.3.2 Kinematic Decoupling

Sometimes, it is possible to separate the joint variables that are responsible for defining the orientation of the end-effector, i.e. $\boldsymbol{R} \in \mathcal{SO}(3)$, and the ones that are responsible for defining the position of the end-effector, i.e. $\boldsymbol{o} \in \boldsymbol{R}^3$.

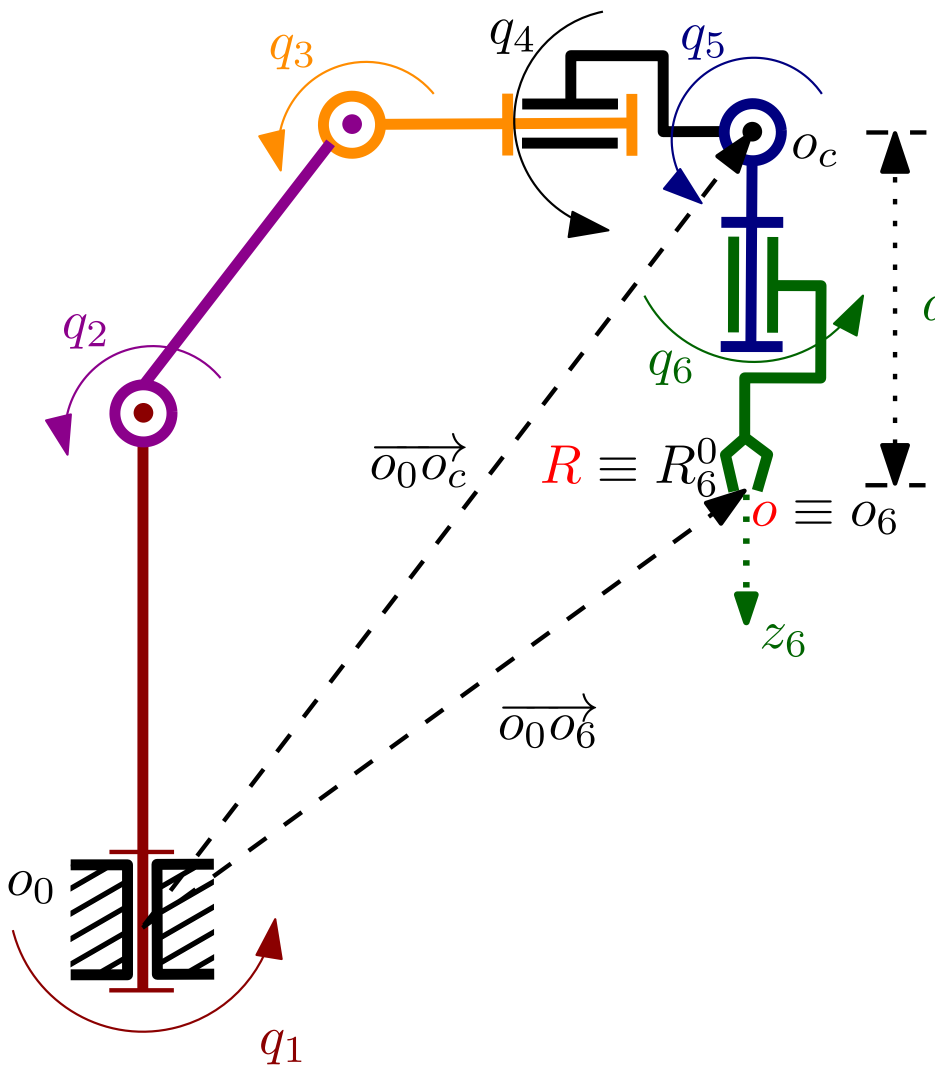
Suppose our manipulator has

- at least **6** joints (otherwise, either not all orientations or not all positions of the tool would be possible),
- the last **3** joint axes intersecting in one point (this is not a rear case).

then, the problem is decoupled into two sub-problems:

- inverse **position** kinematics and
- inverse **orientation** kinematics.

For an illustration, let us consider a 6-DOF articulated manipulator with last **3** axis of rotation intersecting at the single point $\boldsymbol{o}_c$ as shown in the scheme below



Here, (although it is not clear from the picture) the first **3** rotational joints correspond to an anthropomorphic arm with $\boldsymbol{a_1 = \alpha_2 = \alpha_3 = d_2 = d_3 = 0}$, $\boldsymbol{\alpha_1 = \pi/2}$ and the last **3** rotational joints model a spherical wrist with $\boldsymbol{a_4 = a_5 = a_6 = \alpha_6 = d_4 = d_5 = 0}$, $\boldsymbol{\alpha_4 = -\alpha_5 = -\pi/2.}$

Let us show that we can formulate the inverse position kinematics based on computation of $\boldsymbol{o_c^0}$ and $\boldsymbol{R_3^0}$ and inverse orientation kinematics based on computation of $\boldsymbol{R_6^3}$.

Since the last three joint axes intersect in one point, we have

$$\boldsymbol{o} = \boldsymbol{o_c^0} + \boldsymbol{d_6\, z_6^0} = \boldsymbol{o_c^0} + \boldsymbol{d_6\, R} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \boldsymbol{o_c^0} + \boldsymbol{d_6} \begin{bmatrix} \boldsymbol{r_{11}} & \boldsymbol{r_{12}} & \boldsymbol{r_{13}} \\ \boldsymbol{r_{21}} & \boldsymbol{r_{22}} & \boldsymbol{r_{23}} \\ \boldsymbol{r_{31}} & \boldsymbol{r_{32}} & \boldsymbol{r_{33}} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

where $\boldsymbol{o = o_6^0}$, $\boldsymbol{R = R_6^0}$, $\boldsymbol{o_c^0 = o_3^0}$, and $\boldsymbol{z_6^0 = R\, z_6^6 = \left[\, r_{13},\ r_{23},\ r_{33} \,\right]^T}$ is computed from $\boldsymbol{z_6^6 = \left[\, 0,\ 0,\ 1 \,\right]^T}$.

As a result, from given $\boldsymbol{o}$ and $\boldsymbol{R}$ one can compute $\boldsymbol{o}_c^0$

$$\boldsymbol{o}_c^0 = \begin{bmatrix} \boldsymbol{x_c} \\ \boldsymbol{y_c} \\ \boldsymbol{z_c} \end{bmatrix} = \boldsymbol{o} - \boldsymbol{d_6}\,\boldsymbol{R} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{o_x} - \boldsymbol{d_6 r_{13}} \\ \boldsymbol{o_y} - \boldsymbol{d_6 r_{23}} \\ \boldsymbol{o_x} - \boldsymbol{d_6 r_{33}} \end{bmatrix}$$

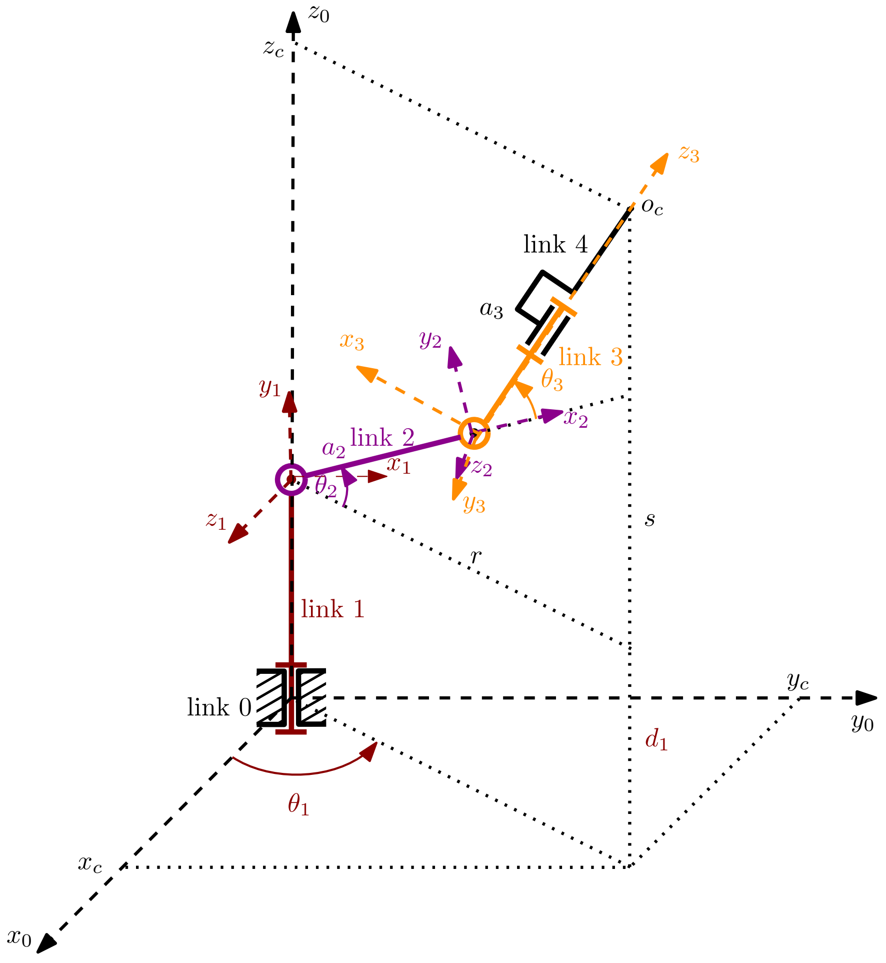
and formulate the **inverse position kinematics problem** as finding the three angles of the first three joints $\boldsymbol{\theta_1}$, $\boldsymbol{\theta_2}$, and $\boldsymbol{\theta_3}$ that produce the just computed values of $\boldsymbol{x_c}$, $\boldsymbol{y_c}$, and $\boldsymbol{z_c}$.

As soon as the inverse position kinematics problem is solved, we have a computed value for $\boldsymbol{R}_3^0$. Now, since

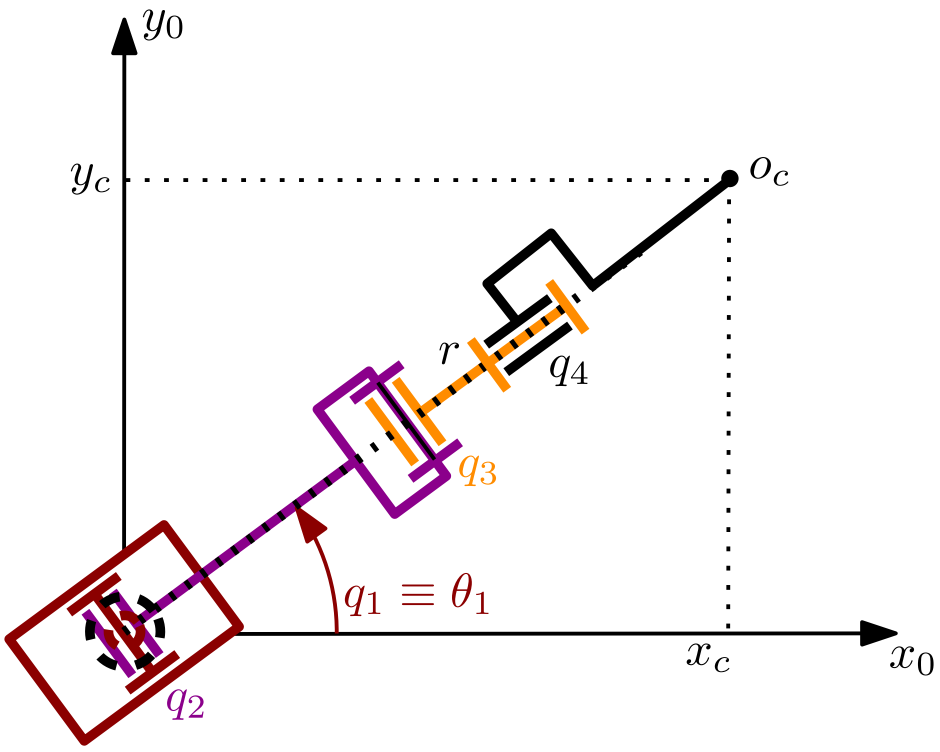
$$\boldsymbol{R} = \boldsymbol{R}_3^0 \cdot \boldsymbol{R}_6^3 \quad \Rightarrow \quad \boldsymbol{R}_6^3 = \left[\boldsymbol{R}_3^0\right]^{-1} \boldsymbol{R} = \left[\boldsymbol{R}_3^0\right]^{T} \boldsymbol{R}$$

we can formulate the **inverse orientation kinematics** problem as finding the three angles of the last three joints $\boldsymbol{\theta_4}$, $\boldsymbol{\theta_5}$, and $\boldsymbol{\theta_6}$ that result in just computed values of $\boldsymbol{R}_6^3$.

The following picture helps to find a solution of the first problem, i.e. of computing $\boldsymbol{\theta_1}$, $\boldsymbol{\theta_2}$, and $\boldsymbol{\theta_3}$ from $\boldsymbol{x_c}$, $\boldsymbol{y_c}$, and $\boldsymbol{z_c}$:



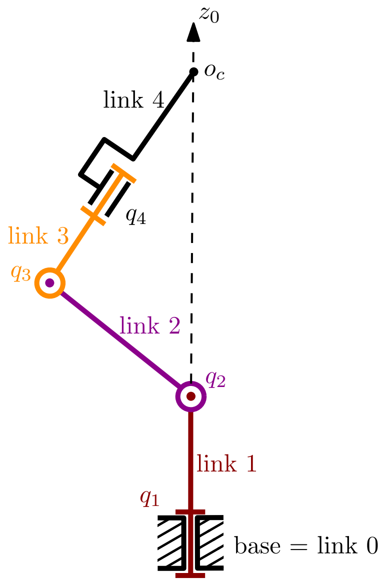
Now, looking at the projection onto the $(\boldsymbol{x_c}, \boldsymbol{y_c})$-plane, we see how to compute $\boldsymbol{\theta_1}$:



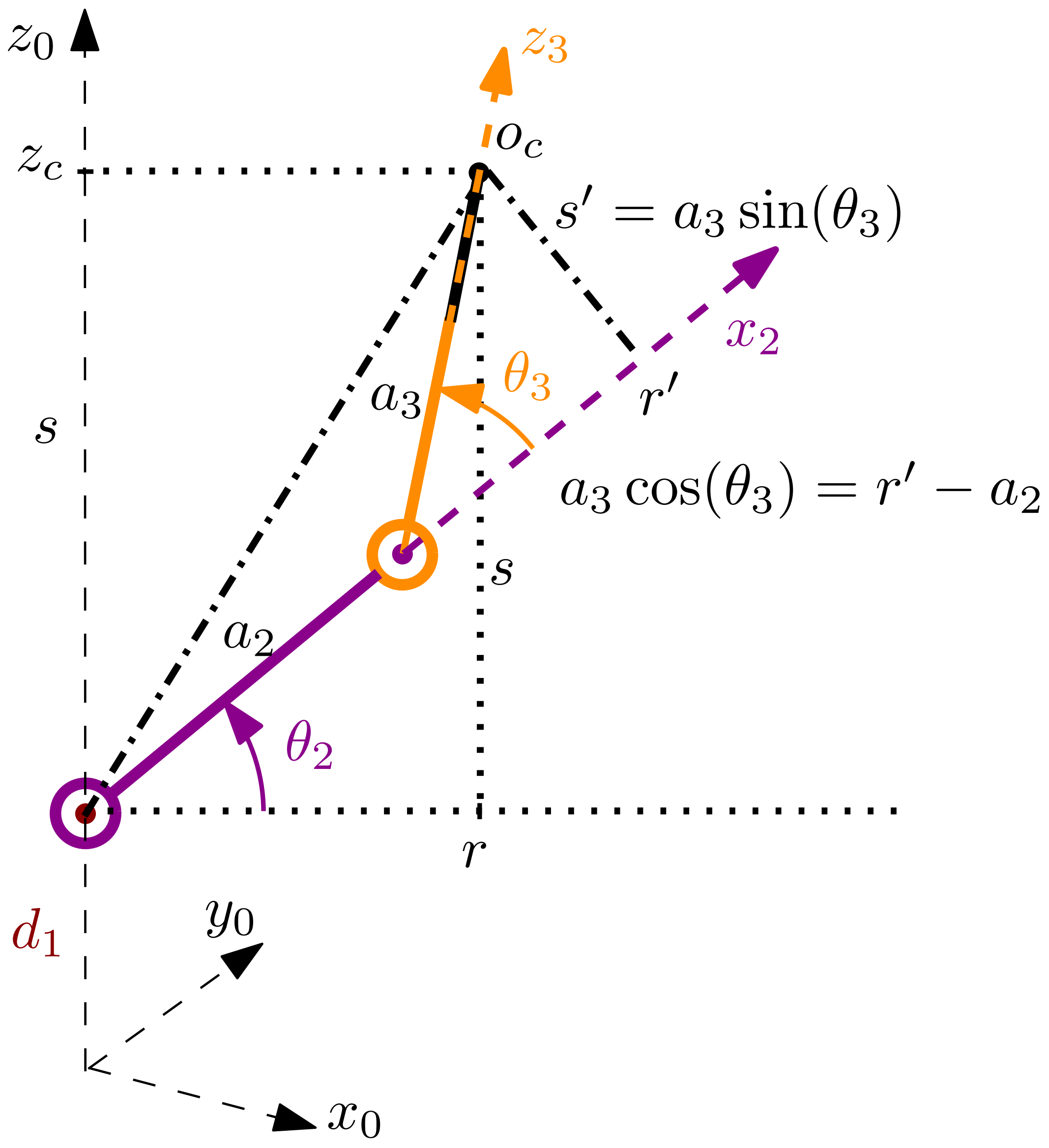
Obviously,

$$\boldsymbol{\theta_1} = \text{atan}\mathbf{2}(\boldsymbol{x_c}, \boldsymbol{y_c})$$

provided $(\boldsymbol{x_c}, \boldsymbol{y_c}) \neq (\mathbf{0}, \mathbf{0})$, i.e. provided the robot is not in a singular configuration like the one shown next



Now, in order to compute $\boldsymbol{\theta_2}$ and $\boldsymbol{\theta_3}$ we should look at the projection onto the appropriate vertical plane



This is the plane formed by the 2nd and the 3rd links. Now, it is not hard to see that

$$\cos\theta_3 = \frac{(r^2+s^2) - a_2^2 - a_3^2}{2a_2a_3}$$

where $s = z_c - d_1$ and $r = \sqrt{x_c^2 + y_c^2}$. Hence,

$$\cos\theta_3 = \frac{x_c^2 + y_c^2 + (z_c - d_1)^2 - a_2^2 - a_3^2}{2a_2a_3}$$

Other triangles help as to realize that as soon as θ_3 is found, we can easily compute θ_2:

$$\theta_2 = \text{atan}2(\sqrt{x_c^2 + y_c^2}, z_c - d_1) - \text{atan}2(a_2 + a_3\cos\theta_3, a_3\sin\theta_3).$$

This completes finding an analytical solution for the inverse position kinematics.

With the computed values of the first three angles at hand, we use the DH parameters of the links to compute the corresponding **3** HT:

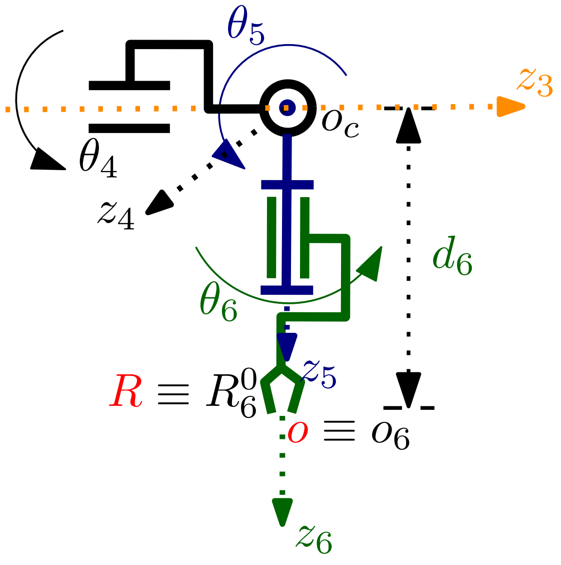
- Link 1: $\boldsymbol{\theta} = \boldsymbol{\theta}_1$, $\boldsymbol{d} = \boldsymbol{d}_1$, $\boldsymbol{a} = 0$, $\boldsymbol{\alpha} = \frac{\pi}{2}$ $\quad\Rightarrow\quad$ $\boldsymbol{H}_1^0$
- Link 2: $\boldsymbol{\theta} = \boldsymbol{\theta}_2$, $\boldsymbol{d} = 0$, $\boldsymbol{a} = \boldsymbol{a}_2$, $\boldsymbol{\alpha} = 0$ $\quad\Rightarrow\quad$ $\boldsymbol{H}_2^1$
- Link 3: $\boldsymbol{\theta} = \boldsymbol{\theta}_3 + \frac{\pi}{2}$, $\boldsymbol{d} = 0$, $\boldsymbol{a} = 0$, $\boldsymbol{\alpha} = \frac{\pi}{2}$ $\quad\Rightarrow\quad$ $\boldsymbol{H}_3^2$

Let us see what needs to be done to solve the inverse orientation kinematics problem.

We have computed $\boldsymbol{H}_3^0 = \boldsymbol{H}_1^0 \boldsymbol{H}_2^1 \boldsymbol{H}_3^2 \quad\Rightarrow\quad \boldsymbol{R}_3^0,\ \boldsymbol{o}_3^0 = \boldsymbol{o}_c^0$. Now, we can compute the rotation matrix

$$\boldsymbol{R}_6^3 = \left[\boldsymbol{R}_3^0\right]^{-1} \cdot \boldsymbol{R} = \begin{bmatrix} \boldsymbol{r}_{11} & \boldsymbol{r}_{12} & \boldsymbol{r}_{13} \\ \boldsymbol{r}_{21} & \boldsymbol{r}_{22} & \boldsymbol{r}_{23} \\ \boldsymbol{r}_{31} & \boldsymbol{r}_{32} & \boldsymbol{r}_{33} \end{bmatrix}$$

In order to find the required analytical expressions for $\boldsymbol{\theta}_4$, $\boldsymbol{\theta}_5$, and $\boldsymbol{\theta}_6$, we may notice that they can be associated with the Euler angles $(\boldsymbol{\phi}, \boldsymbol{\theta}, \boldsymbol{\psi})$:



Hence, we just need to compute the Euler angle representation for a given rotation matrix:

$$\boldsymbol{R}_6^3 = \boldsymbol{R}_{z,\phi} \cdot \boldsymbol{R}_{y,\theta} \cdot \boldsymbol{R}_{z,\psi}.$$

and the overall inverse kinematics problem for an articulated arm with a spherical wrist is solved.

This completes our brief overview of the forward and inverse position and orientation kinematics.

Next we talk about defining velocities and introduce a very important characteristics of a manipulator known as Jacobian.

Chapter 3

Velocity Kinematics and Jacobian

We are interested here to describe velocities of the various points of a robotic manipulator.

Let us recall that motions of rigid bodies, such as links of a manipulator, can be classified as:

- a (pure) **translation**, when all the points move along parallel straight lines passing the same distances during the same time;
- a **rotation around a fix axis**, when all the points move along circles centered at a particular not moving axis;
- a general **rigid motion**, which at every instant of time can be understood as a combination of an instantaneous translation and a rotation around an instantaneous axis.

Translations are easy to describe since all the velocities of the body are the same, while rotations can be described by rotation matrices, change in time of which defines so called angular velocity.

3.1 Angular Velocity for Characterizing Rotations

Let us begin with a discussion of rotations of rigid bodies around a fix axis and introduce the necessary mathematical apparatus.

3.1.1 Rotation around a Fixed Axis

When a rigid body rotates around a fixed axis the following is true.

- Every point of the body moves around a circle with a center at the axis of rotation.

- As the body rotates, every perpendicular to the axis line through any point of the body sweeps the same angle during the same time period. It is customary to use time evolution of such an angle, say $\boldsymbol{\theta}$, to characterize the rotation.

- If $\vec{\boldsymbol{k}}$ is a unit vector in the direction of the axis of rotation, then one can introduce the rigid body's axial vector of **angular velocity** given by
$$\vec{\boldsymbol{\omega}} = \left(\tfrac{d}{dt}\boldsymbol{\theta}\right)\,\vec{\boldsymbol{k}} \qquad \text{and} \qquad |\vec{\boldsymbol{\omega}}| = |\dot{\boldsymbol{\theta}}|$$

- The linear velocity of any point $\boldsymbol{p}$ of the body can be computed as
$$\vec{\boldsymbol{v}}_{\boldsymbol{p}} = \dot{\vec{\boldsymbol{r}}}_{\boldsymbol{p}} = \vec{\boldsymbol{v}}_{\boldsymbol{o}} + \vec{\boldsymbol{\omega}} \times \vec{\boldsymbol{r}}_{\boldsymbol{p}}$$
where $\vec{\boldsymbol{r}}_{\boldsymbol{p}} = \overrightarrow{\boldsymbol{op}}$ is the rotating vector to $\boldsymbol{p}$ from the origin $\boldsymbol{o}$ of an attached frame and $\vec{\boldsymbol{v}}_{\boldsymbol{o}}$ is the linear velocity of the origin, which is equal to zero if $\boldsymbol{o}$ lies on the axis. In the later case of $\vec{\boldsymbol{v}}_{\boldsymbol{o}} = \vec{\boldsymbol{0}}$, the formula for the speed is $|\vec{\boldsymbol{v}}_{\boldsymbol{p}}| = \dot{\boldsymbol{\theta}}\ \underbrace{\left(|\vec{\boldsymbol{r}}_{\boldsymbol{p}}|\ \mathbf{sin}(\widehat{\vec{\boldsymbol{k}}, \vec{\boldsymbol{r}}_{\boldsymbol{p}}})\right)}_{\text{distance from } \boldsymbol{p} \text{ to the axis}}$.

The aim of this section is to derive the formula above for $\vec{\boldsymbol{v}}_{\boldsymbol{p}}$ for the case of rotation of a rigid body not necessary along an axis with a fixed direction and to show how to compute the angular velocity for this formula.

3.1.2 Matrix Representation of Cross Products

We have mentioned above that description of velocities of points of a rotating rigid body can be done using the notion of a cross product between two vectors. Let us remind that the obtained (axial) vector can be computed with respect to a given (fixed) frame as follows.

Suppose coordinates of two vectors $\vec{\boldsymbol{a}}$ and $\vec{\boldsymbol{b}}$ in a $\mathbf{0}$-frame are

$$\boldsymbol{a}^0 = \begin{bmatrix} \boldsymbol{a}_x \\ \boldsymbol{a}_y \\ \boldsymbol{a}_z \end{bmatrix}, \qquad \boldsymbol{b}^0 = \begin{bmatrix} \boldsymbol{b}_x \\ \boldsymbol{b}_y \\ \boldsymbol{b}_z \end{bmatrix}$$

then the coordinates of the vector of their cross product

$$\vec{\boldsymbol{c}} = \vec{\boldsymbol{a}} \times \vec{\boldsymbol{b}}$$

in the $\mathbf{0}$-frame are

$$\boldsymbol{c}^0 = \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix} = \begin{bmatrix} a_y\, b_z - a_z\, b_y \\ a_z\, b_x - a_x\, b_z \\ a_x\, b_y - a_y\, b_x \end{bmatrix}$$

To simplify some derivations, for a given $\vec{a}$, it is useful to introduce a linear transformation (a tensor) $\boldsymbol{S}(\vec{a}) = \vec{a} \times \boldsymbol{I}$ acting on vectors by making a cross product with $\vec{a}$, i.e. for every $\vec{b}$

$$\boxed{\boldsymbol{S}(\vec{a}) \cdot \vec{b} = (\vec{a} \times \boldsymbol{I}) \cdot \vec{b} = \vec{a} \times \vec{b}}$$

in the $\mathbf{0}$-frame such transformation can be represented by the following matrix

$$\boldsymbol{S}^0(\vec{a}) = (\vec{a} \times \boldsymbol{I})^0 = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}$$

since it can be easily verified that

$$\boldsymbol{S}^0(\vec{a})\, \boldsymbol{b}^0 = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = \begin{bmatrix} a_y\, b_z - a_z\, b_y \\ a_z\, b_x - a_x\, b_z \\ a_x\, b_y - a_y\, b_x \end{bmatrix} = \boldsymbol{c}^0$$

As can be seen by inspection,

$$\boldsymbol{S}^0(\vec{a}) = -\left(\boldsymbol{S}^0(\vec{a})\right)^{T}$$

and it is not hard to verify, in a similar way as done for representation of rotations, that the matrices for $\mathbf{0}$-frame and $\mathbf{1}$-frame are related by

$$\boldsymbol{S}^1(\vec{a}) = (\boldsymbol{R}_1^0)^{T}\, \boldsymbol{S}^0(\vec{a})\, \boldsymbol{R}_1^0, \qquad \boldsymbol{S}^0(\vec{a}) = \boldsymbol{R}_1^0\, \boldsymbol{S}^1(\vec{a})\, (\boldsymbol{R}_1^0)^{T}$$

Note that an $\boldsymbol{n} \times \boldsymbol{n}$ matrix $\boldsymbol{S}^0$ is said to be **skew symmetric** and denoted $\boldsymbol{S}^0 \in \boldsymbol{so}(\boldsymbol{n})$ if

$$\boldsymbol{S}^0 + (\boldsymbol{S}^0)^{T} = 0$$

or, in terms of the elements

$$s_{ij}^0 + s_{ji}^0 = 0, \quad i = 1, \ldots, n,\ j = 1, \ldots, n$$

$$\Rightarrow \quad \left\{ s_{ii}^0 = 0 \quad \text{and} \quad s_{ij}^0 = -s_{ji}^0 \quad \forall\, i \neq j \right\}$$

Another important property of the skew symmetric matrices is the following identity

$$\vec{\boldsymbol{x}} \cdot \boldsymbol{S} \cdot \vec{\boldsymbol{x}} = (\boldsymbol{x}^0)^T \boldsymbol{S}^0 \boldsymbol{x} = 0 \quad \text{for any } \boldsymbol{S}^0 \in \boldsymbol{so}(3),\ \boldsymbol{x}^0 \in \mathbb{R}^3$$

As for every linear transformation, it can be also verified that

$$\boldsymbol{S}(\alpha\vec{\boldsymbol{a}} + \beta\vec{\boldsymbol{c}}) = \alpha\boldsymbol{S}(\vec{\boldsymbol{a}}) + \beta\boldsymbol{S}(\vec{\boldsymbol{c}})$$

for every scalars $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ and every vectors $\vec{\boldsymbol{a}}$ and $\vec{\boldsymbol{b}}$.

For application to rigid body motions, it is important to know what happens with cross products under rotation, say $\boldsymbol{R} \in \boldsymbol{\mathcal{SO}}(3)$. To this end, the next formula is not hard to check

$$\boldsymbol{R}\cdot\left(\boldsymbol{S}(\vec{\boldsymbol{a}})\cdot\vec{\boldsymbol{b}}\right) = \boxed{\boldsymbol{R}\cdot\left(\vec{\boldsymbol{a}}\times\vec{\boldsymbol{b}}\right) = (\boldsymbol{R}\cdot\vec{\boldsymbol{a}})\times\left(\boldsymbol{R}\cdot\vec{\boldsymbol{b}}\right)} = \boldsymbol{S}(\boldsymbol{R}\cdot\vec{\boldsymbol{a}})\cdot\left(\boldsymbol{R}\cdot\vec{\boldsymbol{b}}\right)$$

It just states that rotating a result of cross product of two vectors is the same as making cross product of the rotated vectors.

Another important property is the following

$$\boxed{\boldsymbol{R}\cdot\boldsymbol{S}(\vec{\boldsymbol{a}})\cdot\boldsymbol{R}^T = \boldsymbol{S}(\boldsymbol{R}\cdot\vec{\boldsymbol{a}})} \quad \text{for any } \boldsymbol{R}\in\boldsymbol{\mathcal{SO}}(3),\ \vec{\boldsymbol{a}}\in\mathbb{R}^3$$

It states that the result of the sequence of three transformations (rotation, making a cross product with $\vec{\boldsymbol{a}}$, performing inverse rotation) is the same as the result of making a cross product with rotated $\vec{\boldsymbol{a}}$.

To show this identity, take any $\vec{\boldsymbol{p}}$ and apply to it the transformation on the right and the formula above together with $\boldsymbol{R}\cdot\boldsymbol{R}^T = \boldsymbol{I}$:

$$\begin{aligned}\left[\boldsymbol{R}\cdot\boldsymbol{S}(\vec{\boldsymbol{a}})\cdot\boldsymbol{R}^T\right]\cdot\vec{\boldsymbol{p}} &= \boldsymbol{R}\cdot\left(\boldsymbol{S}(\vec{\boldsymbol{a}})\cdot\underbrace{[\boldsymbol{R}^T\cdot\vec{\boldsymbol{p}}]}_{\vec{\boldsymbol{b}}}\right)\\ &= \boldsymbol{S}(\boldsymbol{R}\cdot\vec{\boldsymbol{a}})\cdot\left(\boldsymbol{R}\cdot\underbrace{[\boldsymbol{R}^T\cdot\vec{\boldsymbol{p}}]}_{\vec{\boldsymbol{b}}}\right) = \left[\boldsymbol{S}(\boldsymbol{R}\cdot\vec{\boldsymbol{a}})\right]\cdot\vec{\boldsymbol{p}}\end{aligned}$$

To see how a cross product appears in the formula relating linear and angular velocities, we need to know how to compute derivatives of rotation tensors (or of rotation matrices as their coordinate representations).

3.1.3 Derivatives of Rotation Matrices

For an arbitrary rotation $\boldsymbol{R}(\boldsymbol{\theta}) \in \boldsymbol{\mathcal{SO}}(3)$ defined by an angle $\boldsymbol{\theta}$ and an axis, let us compute derivative of its coordinate representation in a fixed **0**-frame.

In other words, let us consider a matrix-valued differentiable function

$$\boldsymbol{\theta} \mapsto \boldsymbol{R}^0(\boldsymbol{\theta})$$

of a scalar variable $\boldsymbol{\theta}$ and compute its instantaneous rate of change $\frac{d}{d\theta}\boldsymbol{R}^0(\boldsymbol{\theta})$. Since for any $\boldsymbol{\theta}$ the matrix $\boldsymbol{R}^0(\boldsymbol{\theta})$ is a rotation matrix, we have

$$\boldsymbol{R}^0(\boldsymbol{\theta})\left(\boldsymbol{R}^0(\boldsymbol{\theta})\right)^T = \left(\boldsymbol{R}^0(\boldsymbol{\theta})\right)^T \boldsymbol{R}(\boldsymbol{\theta}) = \boldsymbol{I}$$

Differentiating this identity according to the product rule we obtain

$$\underbrace{\frac{d}{d\theta}\left[\boldsymbol{R}^0(\boldsymbol{\theta})\left(\boldsymbol{R}^0(\boldsymbol{\theta})\right)^T\right]}_{\boldsymbol{I}\,=\,\text{const}} = \frac{d}{d\theta}\left[\boldsymbol{R}^0(\boldsymbol{\theta})\right]\left(\boldsymbol{R}^0(\boldsymbol{\theta})\right)^T + \boldsymbol{R}^0(\boldsymbol{\theta})\frac{d}{d\theta}\left[\left(\boldsymbol{R}^0(\boldsymbol{\theta})\right)^T\right] = \boldsymbol{0}$$

Let us introduce the following linear transformation parametrized by $\boldsymbol{\theta}$

$$\boldsymbol{S}(\boldsymbol{\theta}) = \boldsymbol{R}'(\boldsymbol{\theta}) \cdot (\boldsymbol{R}(\boldsymbol{\theta}))^T, \qquad \text{where} \quad \boldsymbol{R}'(\boldsymbol{\theta}) = \frac{d}{d\theta}[\boldsymbol{R}(\boldsymbol{\theta})]$$

which is called **spin matrix** associated with a varying rotation matrix.

Then, using the following known identities for matrices $(\boldsymbol{A}\,\boldsymbol{B}^T)^T = \boldsymbol{B}\,\boldsymbol{A}^T$ and $\frac{d}{d\theta}\left[\left(\boldsymbol{R}^0(\boldsymbol{\theta})\right)^T\right] = \left[\frac{d}{d\theta}\left(\boldsymbol{R}^0(\boldsymbol{\theta})\right)\right]^T$, we can rewrite the result of differentiation above as $\boldsymbol{S}^0(\boldsymbol{\theta}) + \left(\boldsymbol{S}^0(\boldsymbol{\theta})\right)^T = \boldsymbol{0}$ and conclude

$$\boldsymbol{S}(\boldsymbol{\theta}) + \boldsymbol{S}^T(\boldsymbol{\theta}) = \boldsymbol{0}$$

which means that $\boldsymbol{S}(\boldsymbol{\theta})$ is skew symmetric; and, therefore, it defines a transformation equivalent to a cross product with a fixed vector. Now, multiplying the definition of $\boldsymbol{S}$ by $\boldsymbol{R}(\boldsymbol{\theta})$ on the right we obtain

$$\boldsymbol{S}(\boldsymbol{\theta}) \cdot \boldsymbol{R}(\boldsymbol{\theta}) = \boldsymbol{R}'(\boldsymbol{\theta}) \cdot \underbrace{(\boldsymbol{R}(\boldsymbol{\theta}))^T \cdot \boldsymbol{R}(\boldsymbol{\theta})}_{\boldsymbol{I}}$$

$$\Rightarrow \quad \boxed{\frac{d}{d\theta}[\boldsymbol{R}(\boldsymbol{\theta})] = \boldsymbol{S}(\boldsymbol{\theta}) \cdot \boldsymbol{R}(\boldsymbol{\theta})} \qquad \text{(Poisson equation for rotations)}$$

or, equivalently, in $\boldsymbol{0}$-frame: $\frac{d}{d\theta}\left[\boldsymbol{R}^0(\boldsymbol{\theta})\right] = \boldsymbol{S}^0(\boldsymbol{\theta})\,\boldsymbol{R}^0(\boldsymbol{\theta}),\ \boldsymbol{S}^0(\boldsymbol{\theta}) \in \boldsymbol{so}(3)$.

An example: computing $\boldsymbol{S}^0$ of a basic rotation.

Let $\boldsymbol{R}^0(\boldsymbol{\theta}) = \boldsymbol{R}_{z,\theta}$, that is consider the basic rotation around the axis $\boldsymbol{z}_0$ of the $\boldsymbol{0}$-frame, then

$$\boldsymbol{S}^0 = \frac{d}{d\theta}(\boldsymbol{R}_{z,\theta})\,\boldsymbol{R}_{z,\theta}^T = \frac{d}{d\theta}\left(\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}\right)\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}^T$$

$$= \begin{bmatrix} -\sin\theta & -\cos\theta & 0 \\ \cos\theta & -\sin\theta & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} = S^0(\vec{a})$$

where

$$a^0 = [a_x,\ a_y,\ a_z]^T = [0,\ 0,\ 1]^T \qquad \Rightarrow \qquad \vec{a} = \vec{z}_0$$

In a similar way it can be shown that

$$\tfrac{d}{d\theta}\left[R_{\vec{k},\theta}\right] \cdot \left(R_{\vec{k},\theta}\right)^T = S(\vec{k}) \qquad \left[\text{reminder: } \vec{a} \times B := S(\vec{a}) \cdot B\right],$$

which, with any differentiable $\theta(t)$ and a constant $\vec{k}$, implies

$$\tfrac{d}{dt}\left[R_{\vec{k},\theta(t)}\right] = \dot{\theta}\, S(\vec{k}) \cdot R_{\vec{k},\theta(t)} = \left(\dot{\theta}\,\vec{k}\right) \times R_{\vec{k},\theta(t)}$$

Let us see how the generalization of this formula for an arbitrary rotation derived above can be used for a formal introduction of the notion of angular velocity of a rigid body associated with an attached frame.

3.1.4 Linear and Angular Velocities of a Moving Frame

Angular Velocity

Suppose we have a rotating rigid body and the associated rotation $R(t) \in \mathcal{SO}(3)$ of the attached frame is a differentiable function of time.

Repeating the general arguments for computation of its derivative, we conclude that there exists a skew-symmetric time-dependent transformation S, known as tenor of spin, represented in coordinates by a spin matrix, defined by

$$S = \dot{R}(t) \cdot (R(t))^T$$

such that

$$\tfrac{d}{dt}R(t) = S \cdot R(t), \quad S \in so(3)$$

which is known as Poisson equation for rotations.

Since $S = -S^T$, there exist a vector $\vec{\omega}(t)$ such that

$$S = \vec{\omega}(t) \times I \qquad \text{i.e.} \qquad S^0 = S^0(\vec{\omega}(t)) = \begin{bmatrix} 0 & -\omega_z^0(t) & \omega_y^0(t) \\ \omega_z^0(t) & 0 & -\omega_x^0(t) \\ -\omega_y^0(t) & \omega_x^0(t) & 0 \end{bmatrix}$$

The (axial) vector $\vec{\omega}(t)$, defined by $\omega^0(t) = \left[\omega_x^0(t),\, \omega_y^0(t),\, \omega_z^0(t)\right]^T$, is called the **angular velocity** of a rotating frame with respect to the fixed (inertial) frame at time t.

As a useful example for a rotation parametrized using time-dependent Euler ZYZ-angles $(\phi(t), \theta(t), \psi(t))$

$$R_{ZYZ} := \begin{bmatrix} c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\phi c_\theta s_\psi - s_\phi c_\psi & c_\phi s_\theta \\ s_\phi c_\theta c_\psi + c_\phi s_\psi & -s_\phi c_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta \\ -s_\theta c_\psi & s_\theta s_\psi & c_\theta \end{bmatrix}$$

one can verify that

$$\tfrac{d}{dt}R_{ZYZ} \cdot R_{ZYZ}^T = S(\omega_{ZYZ}(t)), \quad \text{where} \quad \omega_{ZYZ}(t) = \begin{bmatrix} c_\phi s_\theta \dot\psi - s_\phi \dot\theta \\ s_\phi s_\theta \dot\psi + c_\phi \dot\theta \\ c_\theta \dot\psi + \dot\phi \end{bmatrix}$$

We should notice that, as shown above, whenever rotation is performed along a **fixed axis** in the direction of unit vector $\vec{k}$ by an angle $\theta(t)$, we have

$$\vec{\omega}(t) = \dot\theta\, \vec{k}.$$

However, in general, if both the angle, $\theta(t)$, and the direction of the axis of rotation, $\vec{k}(t)$, are time-varying, the following relation is true:

$$\vec{\omega} = \dot\theta\, \vec{k} + \sin(\theta)\, \dot{\vec{k}} + (1 - \cos(\theta))\, \vec{k} \times \dot{\vec{k}}$$

that, in particular, implies $\vec{\omega} = \frac{d}{dt}\left(\theta\, \vec{k}\right) + O\left(|\theta|^2\right)$.

Velocity of a point of a rotating rigid body

Consider a point p rigidly attached to a moving 1-frame, then its coordinates in a fixed (inertial) 0-frame (assuming that both origins are on the axis of rotation) are computed as follows

$$p^0(t) = R_1^0(t)\, p^1$$

Differentiating this expression we obtain

$$\begin{aligned} \tfrac{d}{dt}\left[p^0(t)\right] &= \tfrac{d}{dt}\left[R_1^0(t)\right] p^1 = S^0(t)\, R_1^0(t)\, p^1 \\ &= \omega^0(t) \times R_1^0(t)\, p^1 = \omega^0(t) \times p^0(t) \end{aligned}$$

Let us see now what happens with angular velocity if we have a combination of several (relative) rotations.

Relative Angular Velocities and Combining Relative Rotations

Suppose we have now a family of frames, such as in the case of frames attached to different links of a manipulator.

With every time-varying (differentiable) rotation matrix $\boldsymbol{R}_j^i(t)$ describing orientations of $\boldsymbol{j}$-frame moving with respect to $\boldsymbol{i}$-frame, we can associate the vector of **relative angular velocity** $\vec{\omega}_{i,j}$ that can be expressed in $\boldsymbol{k}$-frame as $\omega_{i,j}^k(t) = \boldsymbol{R}_i^k(t)\,\omega_{i,j}^i(t)$ according to the Poisson formula

$$\tfrac{d}{dt}\boldsymbol{R}_j^i(t) \equiv \dot{\boldsymbol{R}}_j^i = \boldsymbol{S}^i(\vec{\omega}_{i,j}(t))\,\boldsymbol{R}_j^i(t) \equiv \omega_{i,j}^i(t) \times \boldsymbol{R}_j^i(t)$$

Let us compute the relation among such coordinate representations.

Consider two moving frames (**1**-frame and **2**-frame) and a fixed **0**-frame, all with the common origin. The combination of rotations formula

$$\boldsymbol{R}_2^0(t) = \boldsymbol{R}_1^0(t)\,\boldsymbol{R}_2^1(t)$$

expresses rotation of **2**-frame with respect to **0**-frame in terms of rotation of **1**-frame with respect to **0**-frame and of **2**-frame with respect to **1**-frame.

Let us differentiate this expression

$$\begin{aligned}
\tfrac{d}{dt}\boldsymbol{R}_2^0(t) &= \tfrac{d}{dt}\left[\boldsymbol{R}_1^0(t)\right]\boldsymbol{R}_2^1(t) + \boldsymbol{R}_1^0(t)\tfrac{d}{dt}\left[\boldsymbol{R}_2^1(t)\right]\\
&= \boldsymbol{S}^0(\vec{\omega}_{0,1}(t))\,\underbrace{\boldsymbol{R}_1^0(t)\boldsymbol{R}_2^1(t)}_{=\boldsymbol{R}_2^0(t)} + \boldsymbol{R}_1^0(t)\left[\boldsymbol{S}^1(\vec{\omega}_{1,2}(t))\,\boldsymbol{R}_2^1(t)\right]\\
&= \boldsymbol{S}^0(\vec{\omega}_{0,1}(t))\boldsymbol{R}_2^0(t) + \boldsymbol{R}_1^0(t)\,\boldsymbol{S}^1(\vec{\omega}_{1,2}(t))\,\underbrace{\boldsymbol{R}_1^0(t)^T\boldsymbol{R}_1^0(t)}_{=\boldsymbol{I}}\,\boldsymbol{R}_2^1(t)\\
&= \boldsymbol{S}^0(\vec{\omega}_{0,1}(t))\boldsymbol{R}_2^0(t) + \underbrace{\boldsymbol{R}_1^0(t)\,\boldsymbol{S}^1(\vec{\omega}_{1,2}(t))\,\boldsymbol{R}_1^0(t)^T}_{=\boldsymbol{S}^0(\vec{\omega}_{1,2}(t))}\,\boldsymbol{R}_1^0(t)\boldsymbol{R}_2^1(t)\\
&= \boldsymbol{S}^0(\vec{\omega}_{0,1}(t))\boldsymbol{R}_2^0(t) + \boldsymbol{S}^0(\vec{\omega}_{1,2}(t))\,\underbrace{\boldsymbol{R}_1^0(t)\boldsymbol{R}_2^1(t)}_{=\boldsymbol{R}_2^0(t)}\\
&= \left[\boldsymbol{S}^0(\vec{\omega}_{0,1}(t)) + \boldsymbol{S}^0(\vec{\omega}_{1,2}(t))\right]\boldsymbol{R}_2^0(t)
\end{aligned}$$

Now, applying again the Poisson formula $\tfrac{d}{dt}\boldsymbol{R}_2^0(t) = \boldsymbol{S}^0(\vec{\omega}_{0,2}(t))\,\boldsymbol{R}_2^0(t)$

$$\boxed{\boldsymbol{S}^0(\vec{\omega}_{0,2}(t)) = \boldsymbol{S}^0(\vec{\omega}_{0,1}(t)) + \boldsymbol{S}^0(\vec{\omega}_{1,2}(t))}$$

This relation, together with the property: $\boldsymbol{S}(\boldsymbol{a}) + \boldsymbol{S}(\boldsymbol{c}) = \boldsymbol{S}(\boldsymbol{a}+\boldsymbol{c})$, imply that the sum of relative angular velocities gives the angular velocity

$$\boxed{\omega_{0,2}^0(t) = \omega_{0,1}^0(t) + \boldsymbol{R}_1^0(t)\omega_{1,2}^1(t) = \omega_{0,1}^0(t) + \omega_{1,2}^0(t)}$$

This formula can be generalized to the case of arbitrary number, say n, moving frames with the same origins as follows. We have

$$R_n^0(t) = R_1^0(t)R_2^1(t)\cdots R_n^{n-1}(t) \;\Rightarrow\; \tfrac{d}{dt}R_n^0(t) = S^0\big(\vec{\omega}_{0,n}(t)\big)R_n^0(t)$$

and, therefore, the angular velocity of the n-frame $\vec{\omega}_n$ can be computed adding (rotated) relative angular velocities

$$\begin{aligned}\omega_n^0 = \omega_{0,n}^0(t) &= \omega_{0,1}^0(t) + \omega_{1,2}^0(t) + \omega_{2,3}^0(t) + \cdots + \omega_{n-1,n}^0(t)\\ &= \omega_{0,1}^0 + R_1^0\omega_{1,2}^1 + R_2^0\omega_{2,3}^2 + \cdots + R_{n-1}^0\omega_{n-1,n}^{n-1}\end{aligned}$$

With a clear notions of angular velocity and relative angular velocity, we can proceed with describing velocities under rigid motions.

3.1.5 Linear Velocities under Rigid Motions

Consider a moving **1**-frame and a fixed (inertial) **0**-frame related by a time-varying homogeneous transform

$$H_1^0(t) = \begin{bmatrix} R_1^0(t) & o_1^0(t) \\ 0 & 1 \end{bmatrix}$$

i.e., coordinates of each point p fixed in the **1**-frame can be computed as

$$p^0(t) = R_1^0(t)p^1 + o_1^0(t)$$

Differentiating this equation and applying the Poisson formula we obtain

$$\begin{aligned}v_p^0(t) = \tfrac{d}{dt}p^0(t) &= \omega_1^0(t) \times \big(R_1^0(t)p^1\big) + \tfrac{d}{dt}\left[o_1^0(t)\right]\\ &= \omega_1^0(t) \times \underbrace{\big(p^0(t) - o_1^0(t)\big)}_{(\overrightarrow{o_1 p})^0} + \underbrace{\tfrac{d}{dt}\left[o_1^0(t)\right]}_{v_{o_1}^0(t)}\end{aligned}$$

where, as above, the **0**-frame representation of the angular velocity vector $\omega_1^0(t) = [\omega_x^0(t),\, \omega_y^0(t),\, \omega_z^0(t)]^T$ can be obtained from

$$\omega_1^0(t) \times I = \begin{bmatrix} 0 & -\omega_z^0(t) & \omega_y^0(t) \\ \omega_z^0(t) & 0 & -\omega_x^0(t) \\ -\omega_y^0(t) & \omega_x^0(t) & 0 \end{bmatrix} = \tfrac{d}{dt}\left(R_1^0(t)\right)\left(R_1^0(t)\right)^T$$

We are ready now to proceed with a description of velocities of a manipulator's end-effector as well as various other points.

3.2 Concept of the Manipulator Jacobian

Let us adapt the derivations above to the case of a kinematic chain.

3.2.1 Velocities of Points in the End-effector Frame

For a given $\boldsymbol{n}$-link manipulator with joint variables $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$, let us see how to compute the angular velocity of the end-effector frame and the linear velocity of its origin.

Suppose the forward kinematics problem is solved (see Chapter 2), i.e. let $\boldsymbol{H}_n^0(\boldsymbol{q})$ be the homogeneous transformation between the end-effector frame and the (inertial) base frame:

$$\boldsymbol{H}_n^0(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{R}_n^0(\boldsymbol{q}) & \boldsymbol{o}_n^0(\boldsymbol{q}) \\ \boldsymbol{0} & \boldsymbol{1} \end{bmatrix}, \quad \boldsymbol{q} = [\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n]^T$$

so that $\forall\, \boldsymbol{p}$ fixed in the end-effector frame with constant coordinates $\boldsymbol{p}^n$, its (changing during the motion) coordinates in the base frame are

$$\boldsymbol{p}^0 = \boldsymbol{R}_n^0\, \boldsymbol{p}^n + \boldsymbol{o}_n^0$$

Note that, since as the robot moves, the joint variables become functions of time

$$\boldsymbol{t} \;\to\; \boldsymbol{q}(\boldsymbol{t}) = [\boldsymbol{q}_1(\boldsymbol{t}), \ldots, \boldsymbol{q}_n(\boldsymbol{t})]^T$$

so that the formula above takes the following form

$$\boldsymbol{p}^0(\boldsymbol{t}) = \boldsymbol{R}_n^0(\boldsymbol{q}(\boldsymbol{t}))\, \boldsymbol{p}^n + \boldsymbol{o}_n^0(\boldsymbol{q}(\boldsymbol{t}))$$

From the discussion above, we know how to compute $\frac{d}{dt}\boldsymbol{p}^0(\boldsymbol{t})$:

$$\begin{aligned} \tfrac{d}{dt}\boldsymbol{p}^0(\boldsymbol{t}) &= \tfrac{d}{dt}\left[\boldsymbol{R}_n^0(\boldsymbol{q}(\boldsymbol{t}))\right] \boldsymbol{p}^n + \tfrac{d}{dt}\left[\boldsymbol{o}_n^0(\boldsymbol{q}(\boldsymbol{t}))\right] \\ &= \boldsymbol{S}^0(\vec{\boldsymbol{\omega}}_{0,n}(\boldsymbol{t}))\boldsymbol{R}_n^0(\boldsymbol{q}(\boldsymbol{t}))\, \boldsymbol{p}^n + \boldsymbol{v}_n^0(\boldsymbol{t}) \\ &= \boldsymbol{\omega}_{0,n}^0(\boldsymbol{t}) \times \boldsymbol{r}^0(\boldsymbol{t}) + \boldsymbol{v}_n^0(\boldsymbol{t}) \end{aligned}$$

with $\boldsymbol{r}^0(\boldsymbol{t}) = \boldsymbol{p}^0(\boldsymbol{t}) - \boldsymbol{o}_n^0(\boldsymbol{q}(\boldsymbol{t})) = \boldsymbol{R}_n^0(\boldsymbol{q}(\boldsymbol{t}))\, \boldsymbol{p}^n$ is the $\boldsymbol{0}$-frame representation of the vector $\vec{\boldsymbol{r}}$ from the origin of the $\boldsymbol{n}$-frame to the point $\boldsymbol{p}$, which is defined uniquely by the solution of the forward kinematics problem.

Therefore, to compute velocity of any point in the end-effector frame, it suffices to know representations in the $\boldsymbol{0}$-frame

- of the angular velocity of the end-effector frame $\boldsymbol{\omega}_{0,n}^0(\boldsymbol{t})$ and

- of the linear velocity of the the end-effector frame origin $v_n^0(t)$.

In principle, from a given solution of the forward kinematics problem $H_n^0(q(t)) = A_1(q_1(t)) \cdots A_n(q_n(t))$, which is a homogeneous transformation between the end-effector n-frame and the base 0-frame,

$$H_n^0(q(t)) = \begin{bmatrix} R_n^0(q(t)) & o_n^0(q(t)) \\ 0 & 1 \end{bmatrix}$$

one can extract $R_n^0(q_1, \ldots, q_n)$ and $o_n^0(q_1, \ldots, q_n)$ and use them

- to find the velocity $\vec{v}_n(t)$ of the the end-effector frame origin:

$$v_n^0(t) = \tfrac{d}{dt} o_n^0\,(q(t)) = \sum_{i=1}^{n} \left[\frac{\partial o_n^0(q_1, \ldots, q_n)}{\partial q_i} \right] \Bigg|_{q=q(t)} \tfrac{d}{dt} q_i(t)$$

- to find the angular velocity $\vec{\omega}_{0,n}(t)$ of the end-effector frame:

$$\omega_{0,n}^0 = \begin{bmatrix} \omega_x^0 \\ \omega_y^0 \\ \omega_z^0 \end{bmatrix} \Leftarrow \begin{bmatrix} 0 & -\omega_z^0 & \omega_y^0 \\ \omega_z^0 & 0 & -\omega_x^0 \\ -\omega_y^0 & \omega_x^0 & 0 \end{bmatrix} = \left(\sum_{i=1}^{n} \frac{\partial R_n^0(q)}{\partial q_i} \tfrac{d}{dt} q_i \right) \left(R_n^0(q) \right)^T$$
$$= \sum_{i=1}^{n} \left[\frac{\partial R_n^0(q)}{\partial q_i} \left(R_n^0(q) \right)^T \right]_{q=q(t)} \dot{q}_i(t)$$

3.2.2 Definition of the Jacobian

It is obvious from the expressions above that there are two matrix functions J_v and J_ω such that

$$v_n^0(t) = J_v(q(t))\, \tfrac{d}{dt} q(t) \qquad \text{and} \qquad \omega_{0,n}^0(t) = J_\omega(q(t))\, \tfrac{d}{dt} q(t)$$

Concatenation of them leads to a $6 \times n$-matrix function $J(\cdot)$ defined by

$$\xi(t) = \begin{bmatrix} v_n^0(t) \\ \\ \omega_{0,n}^0(t) \end{bmatrix} = J(q(t))\, \tfrac{d}{dt} q(t) = \begin{bmatrix} J_v(q(t)) \\ \\ J_\omega(q(t)) \end{bmatrix} \tfrac{d}{dt} q(t)$$

that is called the **manipulator Jacobian**.

Let us derive a simple formula for computation of the Jacobian.

3.2.3 Computing $J_\omega(q)$ Using DH Frames

Recall that angular velocity of a rigid body is defined by derivative of the representation of the corresponding rotation matrix, i.e.

$$R_n^0(t) = R_1^0(t) R_2^1(t) \cdots R_n^{n-1}(t) \;\Rightarrow\; \tfrac{d}{dt} R_n^0(t) = S^0(\vec{\omega}_{0,n}(t))\, R_n^0(t)$$

that imply, as shown above

$$\begin{aligned} \omega_{0,n}^0(t) &= \omega_{0,1}^0(t) + \omega_{1,2}^0(t) + \omega_{2,3}^0(t) + \cdots + \omega_{n-1,n}^0(t) \\ &= \omega_{0,1}^0 + R_1^0 \omega_{1,2}^1 + R_2^0 \omega_{2,3}^2 + \cdots + R_{n-1}^0 \omega_{n-1,n}^{n-1} \end{aligned}$$

To compute each term in the sum above, we should distinguish between two types of joints:

- If i^{th}-joint is revolute, then
  - the axis of rotation coincides with z_{i-1}, fixed in the $(i-1)$-frame;
  - the relative angular velocity in $(i-1)$-frame is

$$\omega_{i-1,i}^{i-1} = \dot{q}_i(t)\, z_{i-1}^{i-1} = \dot{q}_i(t) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

  and in the inertial frame is

$$\omega_{i-1,i}^0 = R_{i-1}^0\, \omega_{i-1,i}^0 = \dot{q}_i(t)\, z_{i-1}^0 = \dot{q}_i(t)\, R_{i-1}^0 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

  - for the future reference we mark this case by $\rho_i = 1$.
- If j^{th}-joint is prismatic, then
  - there is no relative rotation in the $(j-1)$-frame;
  - the relative angular velocity in $(j-1)$-frame is

$$\omega_{j-1,j}^{j-1} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

  - for the future reference we mark this case by $\rho_j = 0$.

Therefore, $\omega_{0,n}^0(t) = \omega_{0,1}^0(t) + \omega_{1,2}^0(t) + \omega_{2,3}^0(t) + \cdots + \omega_{n-1,n}^0(t)$

$$\begin{aligned} &= \omega_{0,1}^0 + R_1^0 \omega_{1,2}^1 + R_2^0 \omega_{2,3}^2 + \cdots + R_{n-1}^0 \omega_{n-1,n}^{n-1} \\ &= \rho_1 \, \dot{q}_1 \, z_0^0 + \rho_2 \, \dot{q}_2 \, z_1^0 + \cdots + \rho_n \, \dot{q}_n \, z_n^0 \end{aligned}$$

And, finally, we obtain

$$\omega_{0,n}^0(t) = \sum_{i=1}^{n} \rho_i \dot{q}_i z_{i-1}^0 = \underbrace{\left[\rho_1 z_0^0, \ \rho_2 z_1^0, \ \ldots, \ \rho_n z_{n-1}^0\right]}_{= J_\omega(q)} \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{bmatrix}$$

defining $J_\omega(q(t))$ since z_1^0 is a function of q_1, z_2^0 depends on q_1 and q_2, etc.

3.2.4 Computing $J_v(q)$ Using DH Frames

The coordinates of the linear velocity $v_n^0(t)$ of the origin of the end-effector frame can be computed differentiating the expression for $o_n^0(t)$ defined from forward kinematics equations.

In case when all the joint variables are constant, there is no motion of any frame origins and therefore $v_n^0(t) \equiv 0$. In fact, o_n^0 can be written as a function of the joint variables q_i, $i = 1, \ldots, n$:

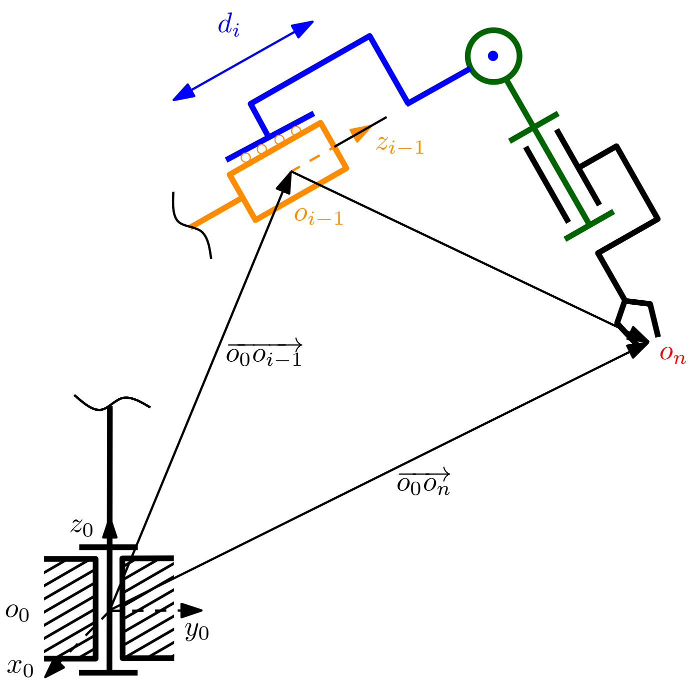
$$o_n^0 = O_n^0(q_1, q_2, \ldots, q_n)$$

Therefore, there are functions $J_{v_1}(q)$, ..., $J_{v_n}(q)$ such that

$$\begin{aligned} v_n^0(t) &= \tfrac{d}{dt} o_n^0(t) = J_{v_1}(q)\dot{q}_1 + J_{v_2}(q)\dot{q}_2 + \cdots + J_{v_n}(q)\dot{q}_n \\ &= \left[\frac{\partial}{\partial q_1} O_n^0\right] \dot{q}_1 + \left[\frac{\partial}{\partial q_2} O_n^0\right] \dot{q}_2 + \cdots + \left[\frac{\partial}{\partial q_n} O_n^0\right] \dot{q}_n \end{aligned}$$

To compute each of the partial derivatives above, we can assume that all the joint variables but one are fixed. As before, we should consider:

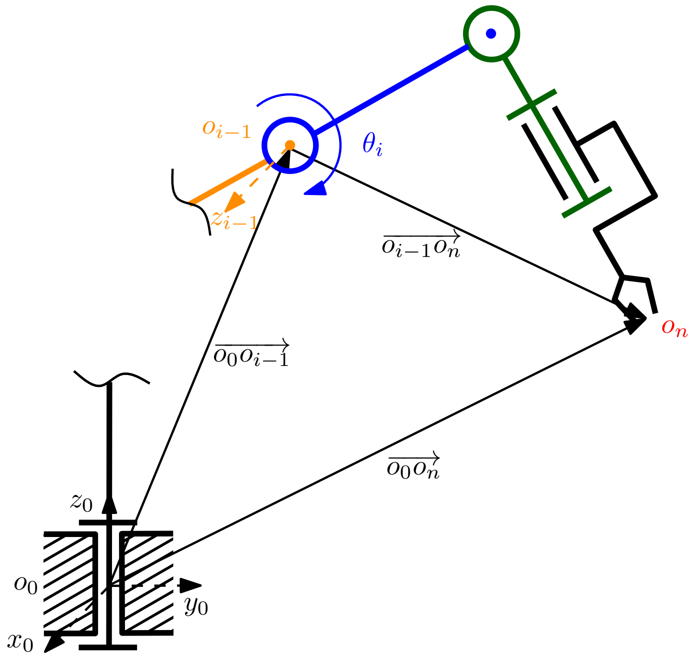
- The case of a prismatic i-th joint is illustrated below



With all other joints frozen, we have translation along the $\boldsymbol{z_{i-1}}$-axis with the speed $\frac{d}{dt}\boldsymbol{d_i}$, so that

$$\frac{d}{dt}\boldsymbol{o_n^0(t)} = \frac{d}{dt}\boldsymbol{d_i(t)R_{i-1}^0\, z_{i-1}^{i-1}} = \frac{d}{dt}\boldsymbol{d_i(t)\, z_{i-1}^0} = \boldsymbol{J_{v_i}\dot{d}_i}$$

- The case of a revolute $\boldsymbol{i}$-th joint is illustrated below


o_{i-1}
θ_i
z_{i-1}
$\overrightarrow{o_{i-1} o_n}$
o_n
$\overrightarrow{o_0 o_{i-1}}$
$\overrightarrow{o_0 o_n}$
z_0
o_0
y_0
x_0

With all other joints frozen, we have rotation around the $\boldsymbol{z_{i-1}}$-axis with the angular velocity $\boldsymbol{\omega_{i-1,i}^0} = \frac{d}{dt}\boldsymbol{\theta_i\, z_{i-1}^0}$, so that

$$\frac{d}{dt}\boldsymbol{o_n^0(t)} = \boldsymbol{\omega_{i-1,i}^0 \times r^0} = \left[\boldsymbol{\dot{\theta} z_{i-1}^0}\right] \times \left[\boldsymbol{o_n^0 - o_{i-1}^0}\right] = \boldsymbol{J_{v_i}\,\dot{q}_i}$$

Combining the two possible cases we conclude that for $i = 1, \dots, n$

$$J_{v_i} = \begin{cases} z_{i-1}^0, & \text{for prismatic joint } (\rho_i = 0) \\ z_{i-1}^0 \times \left[o_n^0 - o_{i-1}^0\right], & \text{for revolute joint } (\rho_i = 1) \end{cases}$$

or, in another notation,

$$J_{v_i} = \left((1-\rho_i) I - \rho_i \, S(o_n^0 - o_{i-1}^0)\right) z_{i-1}^0$$

and

$$J_v(q(t)) = [J_{v_1}, \dots, J_{v_n}]$$

In summary, we have the following recipe: Each 6×1 column of the $6 \times n$ Jacobian matrix

$$J = [J_1, \dots, J_n] \in \mathbb{R}^{6\times n}$$

corresponds to a particular joint $i = 1, \dots, n$ of the manipulator and takes the following form:

$$J_i = \begin{cases} \begin{bmatrix} z_{i-1}^0 \times \left(o_n^0 - o_{i-1}^0\right) \\ z_{i-1}^0 \end{bmatrix} & \text{if joint } i \text{ is revolute } (\rho_i = 1) \\ \begin{bmatrix} z_{i-1}^0 \\ 0 \\ 0 \\ 0 \end{bmatrix} & \text{if joint } i \text{ is prismatic } (\rho_i = 0) \end{cases}$$

where for $2 \le i \le n+1$: z_{i-1}^0 and o_{i-1}^0 can be extracted from the third and fourth columns of the homogeneous transformation matrices:

$$H_{i-1}^0 = \begin{bmatrix} R_{i-1}^0 & o_{i-1}^0 \\ 0_{1\times 3} & 1 \end{bmatrix} = \begin{bmatrix} x_{i-1}^0 & y_{i-1}^0 & z_{i-1}^0 & o_{i-1}^0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

computed by multiplication of HTs ($q_j = d_j$ or $q_j = \theta_j$) matrices:

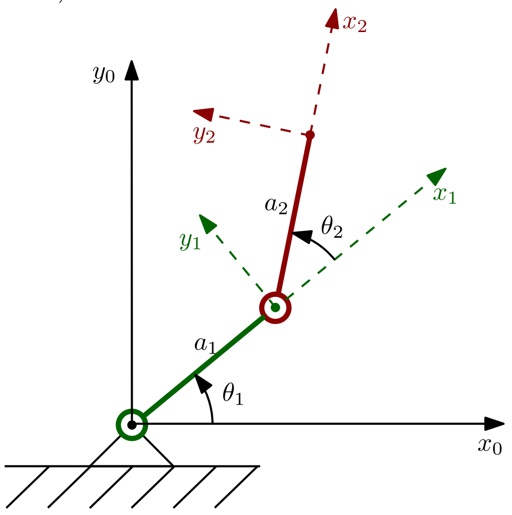
$$H_{i-1}^0(q_1, \dots, q_i) = H_1^0(q_1)\, H_2^1(q_2) \, \dots \, H_j^{j-1}(q_j) \, \dots \, H_i^{i-1}(q_i)$$

where each $H_j^{j-1}(q_j)$ can be defined from the DH parameters

$$H_j^{j-1}(q_j) = \text{Rot}_{z,\theta_j} \, \text{Trans}_{z,d_j} \, \text{Trans}_{x,a_j} \, \text{Rot}_{x,\alpha_j}$$
$$= \left[\begin{array}{ccc|c} \cos(\theta_j) & -\sin(\theta_j)\cos(\alpha_j) & \sin(\theta_j)\sin(\alpha_j) & a_j\cos(\theta_j) \\ \sin(\theta_j) & \cos(\theta_j)\cos(\alpha_j) & -\cos(\theta_j)\sin(\alpha_j) & a_j\sin(\theta_j) \\ 0 & \sin(\alpha_j) & \cos(\alpha_j) & d_j \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

Example of Computing Jacobian Using DH frames

Let us reconsider the planar two-link manipulator studied in Chapter 1. Following the DH convention, the three frames can be assigned as shown below



We would like to compute the $\mathbf{4}$ entries of the Jacobian matrix in the next expression

$$\begin{bmatrix} v_n^0(t) \\ \omega_{0,n}^0(t) \end{bmatrix} = \begin{bmatrix} J_{v_1}(q(t)) & J_{v_2}(q(t)) \\ J_{\omega_1}(q(t)) & J_{\omega_2}(q(t)) \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

Since both joints (so, $n = 2$) are revolute, we put $\rho_1 = \rho_2 = 1$ in the general formulae derived above to obtain the symbolic answer

$$\begin{bmatrix} v_n^0(t) \\ \omega_{0,n}^0(t) \end{bmatrix} = \begin{bmatrix} z_0^0 \times (o_2^0 - o_0^0) & z_1^0 \times (o_2^0 - o_1^0) \\ z_0^0 & z_1^0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$$

It is left to derive the expressions for the entries according to the particular choices of the origins (the three bold points in the picture above). This can be done either directly using simple geometry or following the forward kinematics HT computations with the DH convention routine, see Chapter 2. The result is

$$z_0^0 = z_1^0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \; o_0^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \; o_1^0 = \begin{bmatrix} a_1 c_1 \\ a_1 s_1 \\ 0 \end{bmatrix}, \; o_2^0 = \begin{bmatrix} a_1 c_1 + a_2 c_{12} \\ a_1 s_1 + a_2 s_{12} \\ 0 \end{bmatrix}$$

3.2.5 Analytic Jacobian

Sometimes, it is useful for a particular manipulation task to compute the rates of changes of the three angles (or other parameters of the rotation matrix) defining the orientation of the end-effector frame. As we have seen above, there are various way to parametrize a rotation matrix and one standard approach is to use the Euler angles. In fact, the time-derivatives of these angles are often more informative than the $\mathbf{3}$ components of the angular velocity of the frame defined by the matrix of spin.

In order to compute them, consider again the HT defining the solution of the forward kinematics problem

$$\boldsymbol{T}_n^0(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{R}_n^0(\boldsymbol{q}) & \boldsymbol{o}_n^0(\boldsymbol{q}) \\ \mathbf{0} & \mathbf{1} \end{bmatrix}, \quad \boldsymbol{q} = [q_1, \ldots, q_n]^T$$

There are several ways to represent (and to parametrize) the rotational matrix $\boldsymbol{R}_n^0(\boldsymbol{q})$. Let us consider the following one of them

$$\boldsymbol{R}_n^0 = \boldsymbol{R}(\phi, \theta, \psi) = \boldsymbol{R}_{z,\phi}\, \boldsymbol{R}_{y,\theta}\, \boldsymbol{R}_{z,\psi}$$

with (ϕ, θ, ψ) being the Euler angles ($\boldsymbol{ZYZ}$-parametrization).

Let $\boldsymbol{\alpha}(t) = [\phi(t),\, \theta(t),\, \psi(t)]^T$. Since

$$\boldsymbol{R}_n^0(\boldsymbol{q}) = \boldsymbol{R}(\boldsymbol{\alpha})$$

and the components of the angular velocity $\boldsymbol{\omega}_{0,n}^0(t)$ are defined by the $\mathbf{3}$ of-diagonal entries of the skew symmetric matrix

$$\boldsymbol{S}(\boldsymbol{\omega}_{0,n}^0(t)) = \tfrac{d}{dt}\boldsymbol{R}_n^0(\boldsymbol{q}(t))\, \left(\boldsymbol{R}_n^0(\boldsymbol{q}(t))\right)^T = \tfrac{d}{dt}\boldsymbol{R}(\boldsymbol{\alpha}(t))\, \left(\boldsymbol{R}(\boldsymbol{\alpha}(t))\right)^T$$

obviously, there exists $\boldsymbol{B}(\boldsymbol{\alpha})$ such that

$$\boxed{\boldsymbol{\omega}_{0,n}^0(t) = \boldsymbol{B}(\boldsymbol{\alpha}(t))\, \tfrac{d}{dt}\boldsymbol{\alpha}(t)}$$

In fact, it can be shown that

$$\boldsymbol{\omega}_{0,n}^0(t) = \begin{bmatrix} c_\phi s_\theta \dot{\psi} - s_\phi \dot{\theta} \\ s_\phi s_\theta \dot{\psi} + c_\phi \dot{\theta} \\ c_\theta \dot{\psi} + \dot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & -\sin(\phi) & \cos(\phi)\sin(\theta) \\ 0 & \cos(\phi) & \sin(\phi)\sin(\theta) \\ 1 & 0 & \cos(\theta) \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}$$

defining the components of $\boldsymbol{B}(\boldsymbol{\alpha})$ with $\det\{\boldsymbol{B}(\boldsymbol{\alpha})\} = -\sin(\theta)$.

Now, using the components of $\dot{\boldsymbol{\alpha}} = \left[\dot{\phi}(t),\, \dot{\theta}(t),\, \dot{\psi}(t)\right]^T$ instead of $\boldsymbol{\omega}_{0,n}^0(t)$

in the expression defining the normal Jacobian, we search for the expression

$$\begin{bmatrix} v_n^0(t) \\ \dot{\alpha}(t) \end{bmatrix} = J_a(q)\,\dot{q}$$

that defines the so-called **analytic Jacobian**.

It is not hard to find the relation between the manipulator Jacobian and the analytic Jacobian:

$$\begin{aligned} \begin{bmatrix} v_n^0(t) \\ \omega_{0,n}^0(t) \end{bmatrix} = J(q(t))\,\dot{q}(t) &= \begin{bmatrix} v_n^0(t) \\ B(\alpha(t))\dot{\alpha}(t) \end{bmatrix} \\ &= \begin{bmatrix} I & 0 \\ 0 & B(\alpha(t)) \end{bmatrix} \begin{bmatrix} v_n^0(t) \\ \dot{\alpha}(t) \end{bmatrix} \\ &= \begin{bmatrix} I & 0 \\ 0 & B(\alpha(t)) \end{bmatrix} J_a(q(t))\,\dot{q}(t) \end{aligned}$$

and to compute it whenever necessary (assuming $B(\alpha)$ is invertible).

3.2.6 Singularities

Some properties of the Jacobian are important for understanding limitations and possibilities to perform various motions. In particular, analyzing the Jacobian one can compute so-called singular configuration.

The manipulator Jacobian $J(q)$ is a $6 \times n$-matrix mapping

$$\begin{aligned} \xi &= \begin{bmatrix} v_n^0 \\ \omega_{0,n}^0 \end{bmatrix} = J(q)\,\tfrac{d}{dt}q = \begin{bmatrix} J_v(q) \\ J_\omega(q) \end{bmatrix} \tfrac{d}{dt}q \\ &= [J_1(q),\, J_2(q),\, \ldots,\, J_n(q)] \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{bmatrix} \end{aligned}$$

Note that

$$\xi \in \mathbb{R}^6 \quad \Rightarrow \quad \mathrm{rank}\,\big[J_1(q),\, J_2(q),\, \ldots,\, J_n(q)\big] \le 6, \ \forall\, q$$

Now, all the configurations $q^s = \big[q_1^s,\, \ldots,\, q_n^s\big]$ for which

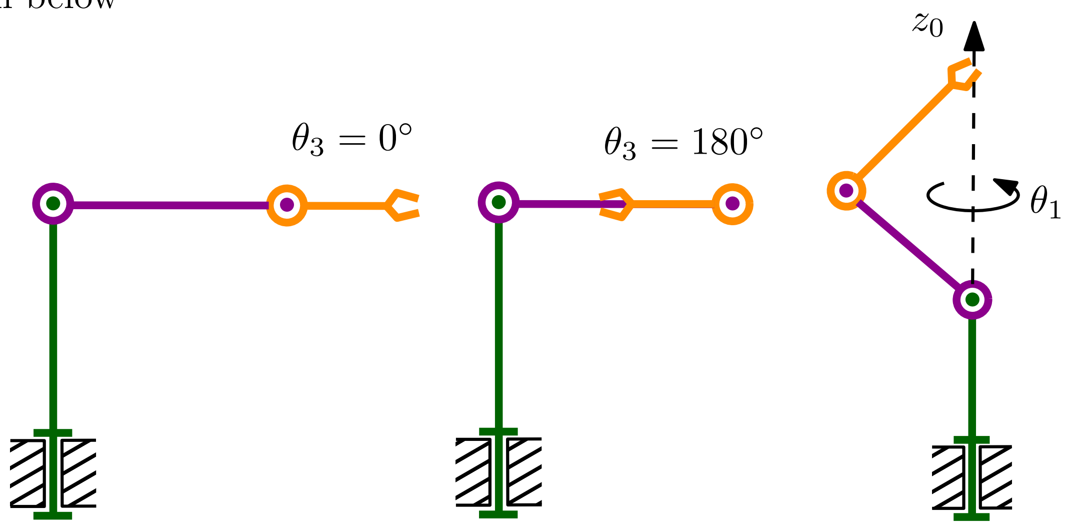
$$\mathrm{rank}\,[J_1(q_s),\, \ldots,\, J_n(q_s)] < \max_q \left\{\mathrm{rank}\,\big[J_1(q),\, \ldots,\, J_n(q)\big]\right\}$$

are called **singular**.

Identifying manipulator singularities are important due to the following

- Singularities represent configurations from which certain direction of motion of the end-effector may be unattainable;
- At singularity, bounded end-effector velocity $\boldsymbol{\xi}$ may correspond to unbounded joint velocities $\dot{\boldsymbol{q}}$.
- Singularities often correspond to points on the boundary of the manipulator workspace.

Two examples of singular configurations for the elbow manipulator are shown below



3.3 Inverse Velocity Kinematics

As discussed above, the manipulator Jacobian defines the velocity of the end-effector frame and its angular velocity from derivatives of the joint variables. Sometimes, it is useful to solve the inverse problem. For example, if we know the desired velocity characteristics for the tool, it is useful to know the corresponding target values of the directly controlled joint velocities.

3.3.1 Assigning Joint Velocities

Suppose we are given the desired vector of end-effector velocities $\boldsymbol{\xi} = \boldsymbol{\xi}_\star$; how to find the corresponding vector $\dot{\boldsymbol{q}}_\star$ of joint velocities?

The manipulator Jacobian $\boldsymbol{J}(\boldsymbol{q})$ is a $\boldsymbol{6} \times \boldsymbol{n}$-matrix mapping

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{v}_n^0 \\ \boldsymbol{\omega}_{0,n}^0 \end{bmatrix} = \boldsymbol{J}(\boldsymbol{q}) \, \tfrac{d}{dt}\boldsymbol{q} = [\boldsymbol{J}_1(\boldsymbol{q}), \boldsymbol{J}_2(\boldsymbol{q}), \ldots, \boldsymbol{J}_n(\boldsymbol{q})] \begin{bmatrix} \dot{\boldsymbol{q}}_1 \\ \vdots \\ \dot{\boldsymbol{q}}_n \end{bmatrix}$$

To solve the inverse velocity kinematics problem in the special case when the Jacobian is square and the configuration is not singular, we can invert it

$$\dot{q}_\star = J^{-1}(q_\star)\, \xi_\star$$

When $n > 6$ there are many solutions, but all of them can be found using the pseudo-inverse matrices as follows

$$\dot{q}_\star = J^+ \xi_\star + \left(I - J^+ J\right) b, \quad \forall\, b \in \mathbb{R}^n$$

where $\quad J^+(q_\star) = J^T \left(J J^T\right)^{-1} \quad$ or, for example, as

$$\dot{q}_\star = W^{-1} J_a^T \left[J_a W^{-1} J_a^T\right]^{-1} \dot{X}_\star + \left(I - W^{-1} J_a^T \left[J_a W^{-1} J_a^T\right]^{-1} J_a\right) b,$$

where J_a is the algebraic Jacobian, $b \in \mathbb{R}^n$ is taken arbitrary (or optimized to achieve an additional goal or a certain property) and W is a positive definite weighting matrix.

3.3.2 Inverse Kinematics based on Jacobian

Let us return to the case when the Jacobian is a square matrix, which is not singular in a vicinity of a planned motion.

With a given planned $\dot{X}_\star(t)$ one can integrate the equation above to obtain a numerical solution for the inverse position/orientation kinematics problem. For example, with

$$q_\star(t) = \int_0^t J_a^{-1}(q_\star(\tau))\, \dot{X}_\star(\tau)\, d\tau + q_\star(0)$$

However, especially if $X_\star(t) = \begin{bmatrix} o^0_{n\star}(t) \\ \alpha_\star(t) \end{bmatrix}$ is given and $\dot{X}_\star(t)$ is computed via differentiation, the difference between $X_\star(t)$ and $f(q_\star(t))$, where $f(\cdot)$ denotes the solution of the forward kinematics problem, will grow:

$$q_\star(t) = \int_0^t \left(J_a^{-1}(q_\star(\tau))\, \dot{X}_\star(\tau) + \text{off-set}\right) d\tau + q_\star(0) = \cdots + (\text{off-set})\, t$$

To fix the off-set-related and accumulation of numerical integration errors, one can solve the differential equation

$$\dot{q}_\star(t) = J_a^{-1}(q_\star)\left(\underbrace{\begin{bmatrix} v^0_{n\star}(t) \\ \dot{\alpha}_\star(t)\end{bmatrix}}_{\dot{X}_\star(t)} + K\underbrace{\left(\begin{bmatrix} o^0_{n\star}(t) \\ \alpha_\star(t)\end{bmatrix} - f(q_\star(t))\right)}_{e(t)}\right) \quad \Rightarrow \quad \dot{e} + Ke = 0$$

with $K > 0$ that leads to exponentially vanishing $e(t) = X_\star(t) - f(q_\star(t))$, provided $f(q) \equiv \begin{bmatrix} o^0_n(q) \\ \alpha(q)\end{bmatrix}$ denotes the appropriate solution of the forward kinematics problem and $J_a(q) \equiv \dfrac{\partial f(q)}{\partial q}$ is the related algebraic Jacobian.

3.3.3 Manipulability

Suppose that we have restricted possible joint velocities of a n-DOF robot as follows

$$\|\dot{q}\|^2 = \|\dot{q}_1\|^2 + \cdots + \|\dot{q}_n\|^2 \leq 1$$

Then, clearly, we have restricted possible body velocities by

$$\xi^T P(q_a)\xi = \xi^T\left(J(q_a)J^T(q_a)\right)^{-1}\xi = \left(J^+(q_a)\xi\right)^T J^+(q_a)\xi = \|\dot{q}\|^2 \leq 1$$

The set of values of ξ that satisfy the inequality above is called the **manipulability ellipsoid**. It illustrates how easy or difficult would be to move the end-effector in certain directions from the configuration q_a.

Part II

Path Planning and Dynamics

Chapter 4

Path and Trajectory Planning

Here we would like to discuss path and trajectory planning. By **"path"** we mean a geometric curve to follow. It can be a curve for an end-effector to follow in the $3D$ space or a curve defining the manipulator's postures by a infinite sequence of values for the generalized coordinates. When we equip such a curve with a particular time-evolution characteristics, such as velocity at every moment, we obtain a **"trajectory"**.

For various tasks, trajectories must be understood either as a set of functions describing time-evolution of the generalized coordinates or as a set of functions describing time-evolution of the end-effector. Relations between the two descriptions of a path can be obtained from a solution of the kinematics problem, see Chapter 2. While to establish relations between the two descriptions of a trajectory, one needs to use a solution of the velocity kinematics problem, see Chapter 3.

Let us proceed with a useful terminological overview.

4.1 Space Description

For a topology-based approaches the next spaces are typically introduced.

4.1.1 Configuration Space

Given a robot with n-links, the following terminology is used.

- A complete specification of location of all the links of the robot, by e.g. particular values of the generalized coordinates $\boldsymbol{q}$, is called its **configuration**.
- The set of all possible configurations is known as the **configuration space** $\boldsymbol{\mathcal{Q}} = \{\boldsymbol{q}\}$. It must be a subspace (or an embedded submanifold)

of the n-dimensional Euclidean space: $\mathcal{Q} \subset \mathbb{R}^n$.

For example, for 1-link revolute arm $\mathcal{Q}$ is the set of all possible orientations of the link, i.e., it is (topologically) a circle in plane:

$$\mathcal{Q} = \mathbb{S}^1 \quad \text{or} \quad \mathcal{Q} = SO(2)$$

As another example, consider a 2-link planar arm with revolute joints. The configuration space becomes a cross product of two circles, which is a two-dimensional torus:

$$\mathcal{Q} = \mathbb{S}^1 \times \mathbb{S}^1 = \mathbb{T}^2 \quad \leftarrow \quad \text{torus}$$

If one consider a rigid object moving on a plane, its location can be defined by coordinates of a single point and by an angle of its orientation. Hence, the configuration space is the cross product of a plane and a circle:

$$\mathcal{Q} = \{x,\, y,\, \theta\} = \mathbb{R}^2 \times \mathbb{S}^1$$

4.1.2 Workspace

Given a robot with n-links and its configuration space $\mathcal{Q} \subset \mathbb{R}^n$, the following is useful.

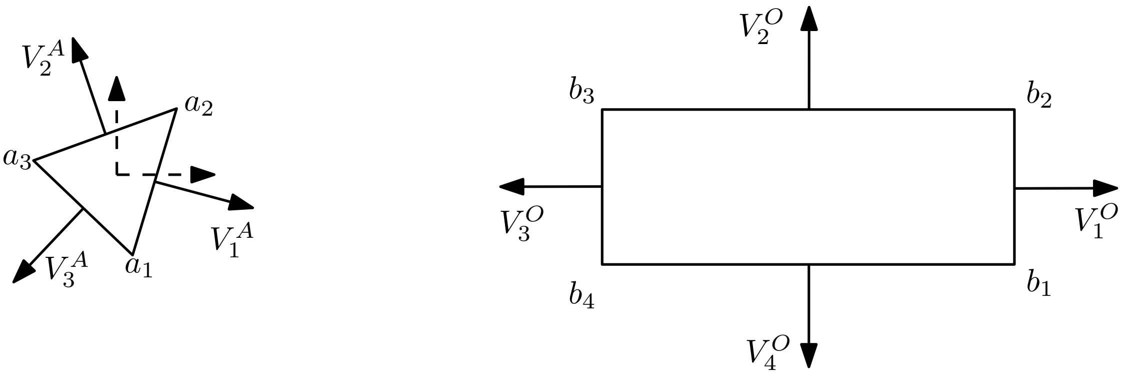
- Denote by $\mathcal{W} \subset \mathbb{R}^3$, the subset of the $3D$ space, where the robot moves. It is called **workspace** of the robot.
- The workspace $\mathcal{W}$ might contain various obstacles that occupy certain spaces $\mathcal{O}_i \subset \mathbb{R}^3$, $i = 1, 2, \ldots,$ the number of obstacles.
- Denote by $\mathcal{A}(q)$ a subset of workspace $\mathcal{W}$, which is occupied by the robot, described by a particular values of the generalized coordinates $q \in \mathcal{Q}$.
- Introduce a subset of configuration space that is occupied by obstacles

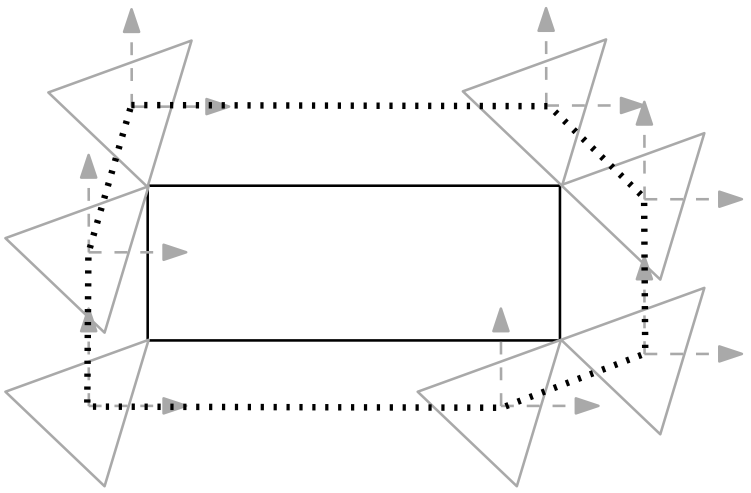
  $$\mathcal{QO} := \{q \in \mathcal{Q} : \mathcal{A}(q) \cap \mathcal{O}_i \neq \emptyset,\ \forall\, i\}$$

  in other words, $q \in \mathcal{QO}$ means that the robot is in collision with one of the obstacles.
- Then, the **collision-free configurations** are defined by

  $$\mathcal{Q}_{free} := \mathcal{Q} \setminus \mathcal{QO}$$

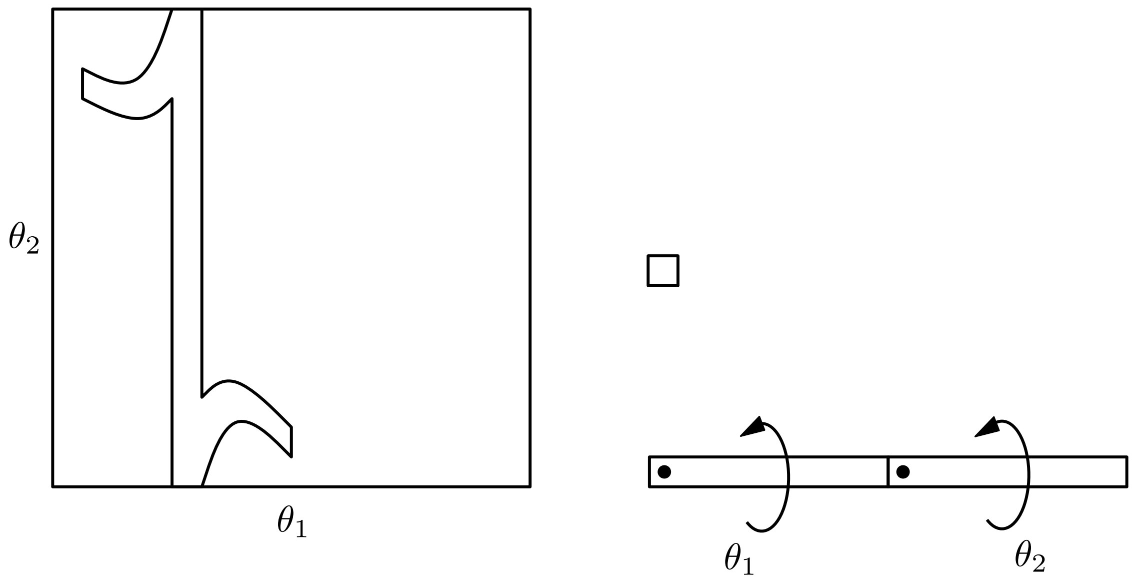
For example, consider a robot, with an end-effector shaped in the form of a triangle and moving in plane with a fixed orientation. Suppose the workspace, which is a plane, contains a rectangular obstacle. Than, obviously, $\mathcal{QO}$ is the closed set with the dashed boundary given in the figure below.





However, typically, computing $\mathcal{QO}$ (and even $\mathcal{A}(q)$) is not a trivial task.

As another example, consider a two-links planar $\boldsymbol{RR}$-type robot, the workspace of which contains a single square obstacle. The configuration space and the set $\mathcal{QO}$ occupied by the obstacle is shown below.



Note that from the figure above one can visualize the shaded region on the torus imagining that the edges of the square marked with $\boldsymbol{\theta_1}$ and $\boldsymbol{\theta_2}$ are glued together.

Whenever a collision free space $\boldsymbol{\mathcal{Q}_{free}}$ is computed, one would need to find an appropriate for a given task path that belongs to it.

4.2 Path Planning

Problem of Path Planning is typical formulated as the task to find a path in the configuration space $\boldsymbol{\mathcal{Q}}$ that

- connects an initial configuration $\boldsymbol{q_s}$ to a final configuration $\boldsymbol{q_f}$,
- does not collide any obstacle as the robot traverses the path.

Note that as soon as this is done, the pass of the end-effector is unambiguously defined by the solution of the forward kinematics problem.

So, formally, the task is to find a continuous function $\gamma(\cdot)$ such that

$$\gamma : [0,1] \to \boldsymbol{\mathcal{Q}_{free}} \quad \text{with } \gamma(0) = \boldsymbol{q_s}, \text{ and } \gamma(1) = \boldsymbol{q_f}.$$

However, typically, there are additional requirements such as

- Some intermediate (pass through) points $\boldsymbol{q_i}$ may be specified;
- The path should be sufficiently smooth;
- Some optimality is required (smallest path length, smallest curvature, etc.);
- Avoiding closeness to singular configurations.

There are several approaches to solve the path planning problem.

4.2.1 Path Planning: Potential Field Approach

The basic idea of this approach is the following.

- Treat a robot as a particle in $\mathbb{R}^n$ moving under the influence of an artificial potential field $\boldsymbol{U}(\cdot)$ to be defined.
- Function $\boldsymbol{U}(\cdot)$ should have
  - global minimum at $\boldsymbol{q_f}$ $\Rightarrow$ this point is attractive

  - maximum or to be $+\infty$ in the points of $\mathcal{QO}$ $\Rightarrow$ these points repel the robot

It is a good idea to construct such function $U(\cdot)$ in a simple form, where one can easily add or remove an obstacle and change q_f. The commonly used form for of $U(\cdot)$ is

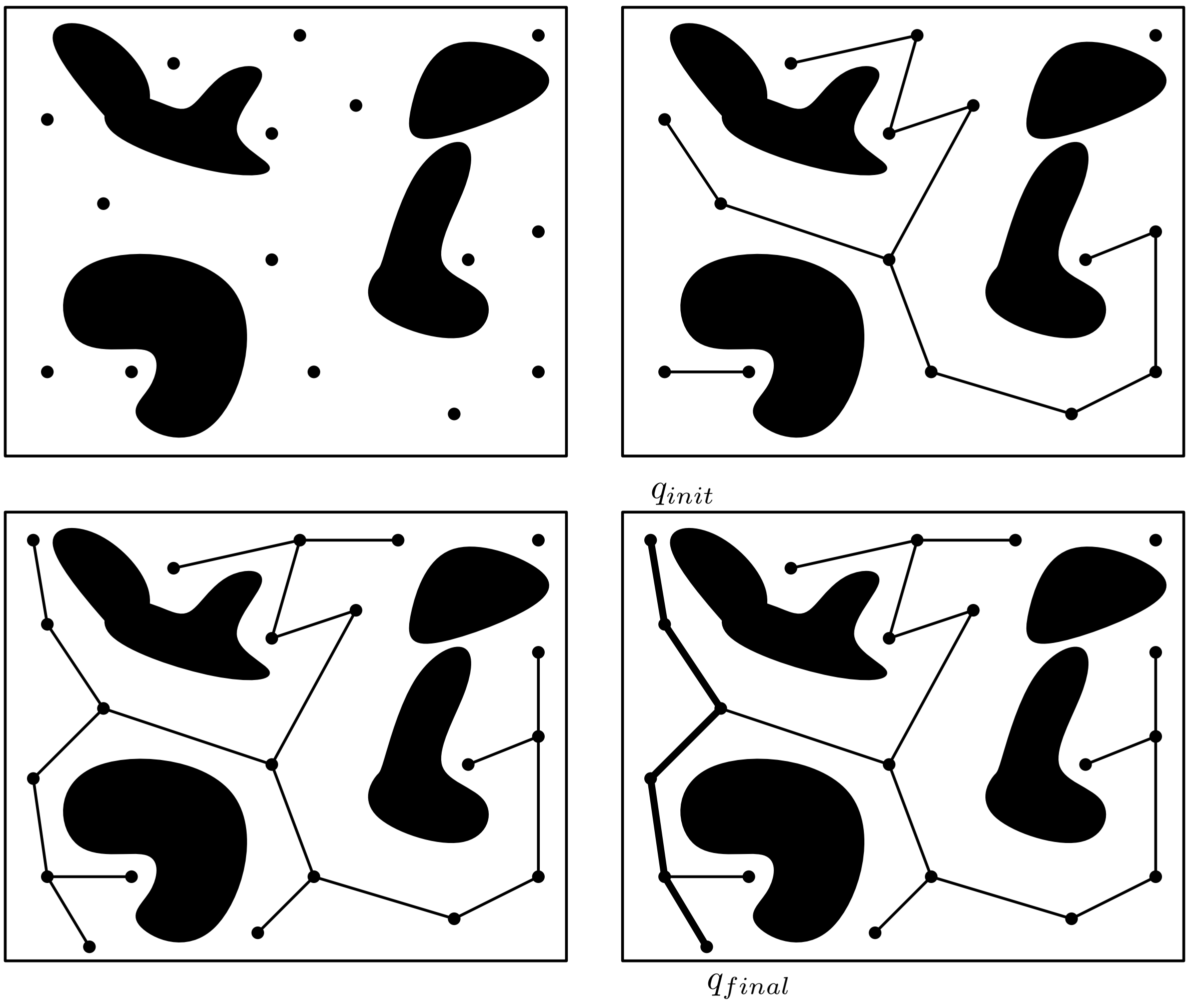
$$U(q) = U_{att}(q) + \left(U_{rep}^{(1)}(q) + U_{rep}^{(2)}(q) + \cdots + U_{rep}^{(N)}(q)\right)$$

4.2.2 Path Planning: Probabilistic Road Map Method

The basic idea of this approach is the following.

- Sample randomly the configuration space $\mathcal{Q}$, i.e. randomly choose a large set of points in $\mathcal{Q}$.
- Those samples that belong to $\mathcal{QO}$ are disregarded, while with the others one proceed as follows.
- Connect the points that are close pairwise in a particular way satisfying all the requirements and bridge all the isolated components (add additional points if necessary), e.g.
  - Choose the way to measure distance between points in $\mathcal{Q}$, i.e. define a continuous real-valued distance function $d(\cdot)$.
  - Choose $\varepsilon > 0$ and find k neighbors of distance no more than ε that can be connected to the current one without violating any requirements while following a simple pattern, such as connecting points along a straight line. This step may result in fragmentation of the workspace consisting of several disjoint components.
  - Make enhancement, that is, try to connect disjoint components, possibly introducing more samples.
- Try to compute a path by connecting the initial and final points to the closest samples and using a set of pairwise connection among the samples.

An example of following such a procedure in plane is shown below.



4.3 Trajectory Planning

4.3.1 From path to trajectory

In essence, trajectory is a path $\gamma : [0,1] \to \mathcal{Q}_{free}$ enhanced by an explicit parametrization of time:

$$[T_s, T_f] \ni t \ \rightsquigarrow \ \tau \in [0,1] : \ q(t) = \gamma(\tau) \in \mathcal{Q}_{free}$$

Specifying a trajectory means that, in addition to a geometrical description of a path, we make specifications on e.g.

- velocity $\frac{d}{dt}q(t)$ of a motion,
- acceleration $\frac{d^2}{dt^2}q(t)$ of a motion,
- jerk $\frac{d^3}{dt^3}q(t)$ of a motion,
- ...

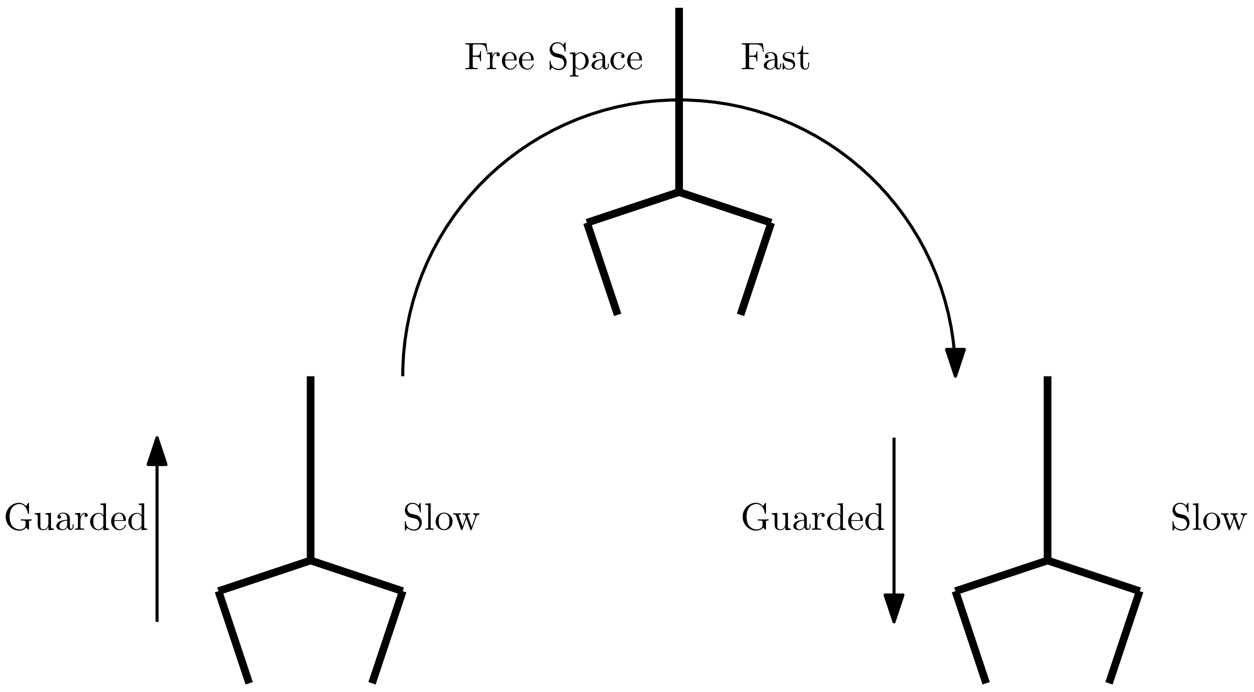
at each moment of time.

However, it is common to specify (or to redefine) a path in a form of a family of snap-shots

$$q_s, \quad q_1, \quad q_2, \quad q_3, \quad \ldots, \quad q_f$$

that can be obtained substituting various numbers from $[0, 1]$ into a computed path description $\gamma(\cdot)$.

Often, it is useful to group the points (make a segmentation) of snap-shots according to the task-defined accuracy. As an example, one can decompose a path into segments with fast and slow velocity profiles as illustrated below.



4.3.2 Trajectories for Point to Point Motions

Suppose now that a path is given in a form of a set of points with, possibly, additional specifications of velocities, accelerations, etc. at each of them, that are sufficiently distant from the obstacles. It is reasonable then to try finding a trajectory as a combination of pieces connecting pairs of close points.

Consider the i^{th} joint of a robot and suppose that the specification

$$\begin{array}{ll} \text{at time } t = t_0 \text{ is :} & q_i(t_0) = q_0, \ \frac{d}{dt}q(t_0) = v_0 \\ \text{at time } t = t_f \text{ is :} & q_i(t_f) = q_f, \ \frac{d}{dt}q(t_f) = v_f \end{array}$$

In addition, we might be given (or not) constraints on the accelerations

$$\frac{d^2}{dt^2}q(t_0) = \alpha_0, \qquad \frac{d^2}{dt^2}q(t_f) = \alpha_f$$

Let us generate a polynomial of a sufficiently large degree m

$$q(t) = a_0 + a_1 t + a_2 t^2 + \cdots + a_m t^m$$

that will allow us to satisfy the interpolation conditions defined by the specifications above.

Assuming the interpolation constraints

$$\begin{array}{rl} \text{at time } t = t_0 \text{ is :} & q_i(t_0) = q_0, \ \frac{d}{dt}q(t_0) = v_0 \\ \text{at time } t = t_f \text{ is :} & q_i(t_f) = q_f, \ \frac{d}{dt}q(t_f) = v_f \end{array}$$

for the 3^{rd}-order polynomial

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3, \quad \frac{d}{dt}q(t) = a_1 + 2a_2 t + 3a_3 t^2$$

one immediately obtains the following relations to satisfy

$$\begin{array}{rcl} q_0 & = & a_0 + a_1 t_0 + a_2 t_0^2 + a_3 t_0^3 \\ v_0 & = & a_1 + 2a_2 t_0 + 3a_3 t_0^2 \\ q_f & = & a_0 + a_1 t_f + a_2 t_f^2 + a_3 t_f^3 \\ v_f & = & a_1 + 2a_2 t_f + 3a_3 t_f^2 \end{array}$$

They can be written in the following matrix form

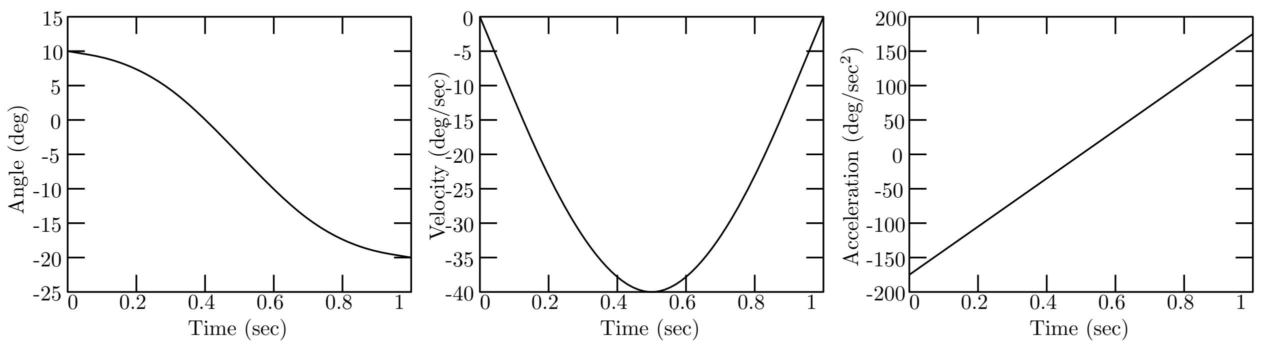
$$\begin{bmatrix} q_0 \\ v_0 \\ q_f \\ v_f \end{bmatrix} = \begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 \\ 0 & 1 & 2t_0 & 3t_0^2 \\ 1 & t_f & t_f^2 & t_f^3 \\ 0 & 1 & 2t_f & 3t_f^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

So, to be able to compute the required coefficients a_i, one needs to ensure that the determinant of the matrix above is not equal to zero.

Consider the following parameters for interpolation:

$$t = 0 \ \text{ and } \ t_f = 1, \quad q_0 = 10 \ \text{ and } \ q_f = -20, \quad v_0 = v_f = 0$$

the resulting trajectory is shown below



As can be seen the obtained trajectory is unacceptable due to very high accelerations. To avoid generating such trajectories one, obviously, needs to define additional interpolation conditions.

Consider the i^{th} joint of a robot and suppose that the specifications are

$$\begin{array}{ll} \text{at time } t = t_0 \text{ is :} & q_i(t_0) = q_0, \; \frac{d}{dt}q(t_0) = v_0 \\ \text{at time } t = t_f \text{ is :} & q_i(t_f) = q_f, \; \frac{d}{dt}q(t_f) = v_f \end{array}$$

and introduce additional constraints on the accelerations

$$\frac{d^2}{dt^2}q(t_0) = \alpha_0 \qquad \frac{d^2}{dt^2}q(t_f) = \alpha_f$$

Clearly, to find an interpolating polynomial, one needs to choose a polynomial of order ≥ 5 such as

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5$$

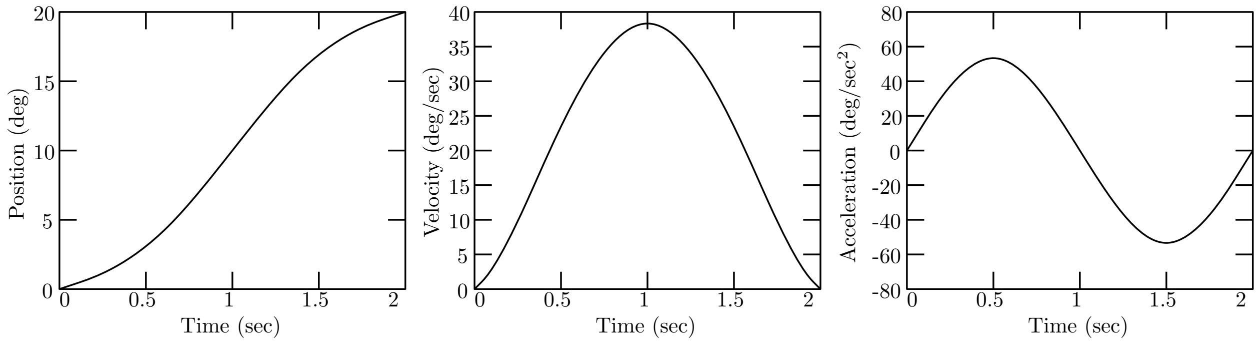
Consider, as above, the following parameters for interpolation:

$$t = 0 \;\text{ and }\; t_f = 1, \quad q_0 = 10 \;\text{ and }\; q_f = -20, \quad v_0 = v_f = 0$$

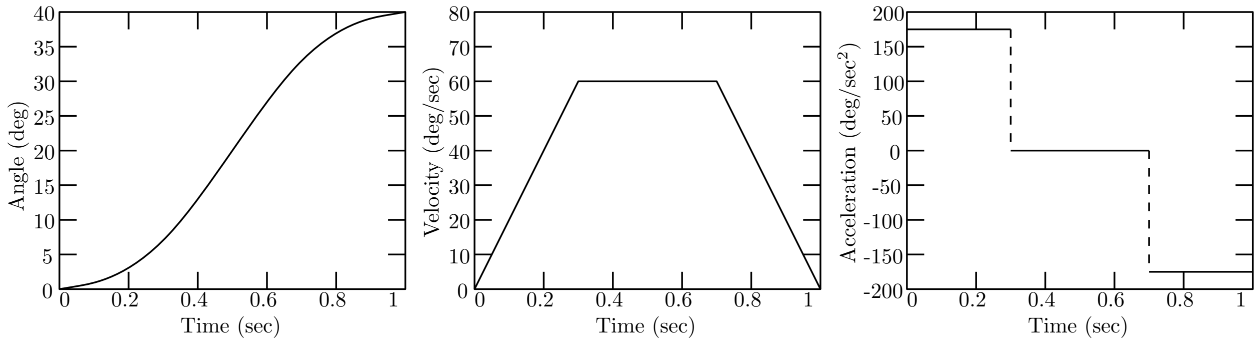
but this time let as require

$$\alpha_0 = 0 \;\text{ and }\; \alpha_f = 0.$$

The resulting trajectory is shown below



In the case when time optimality is desirable, trajectories are often generated using LSPB — Linear segments with parabolic blends, that are illustrated below.



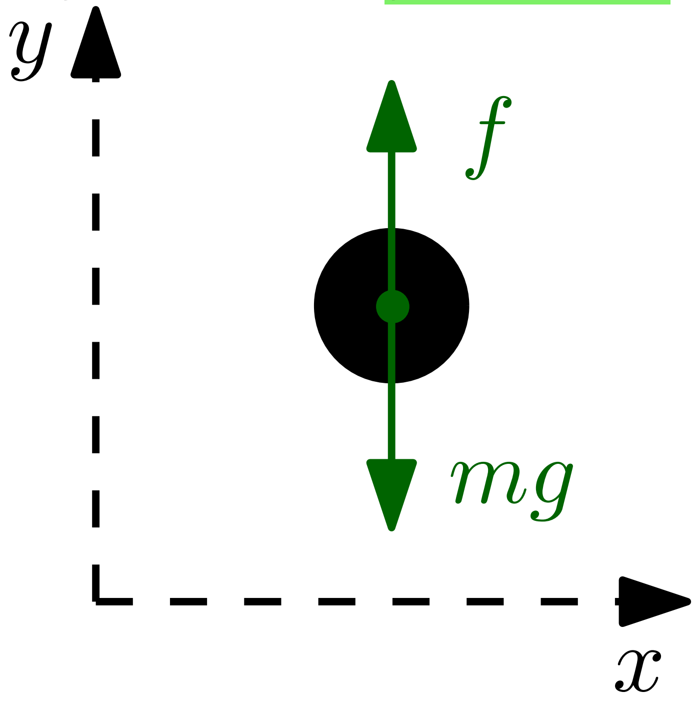
Chapter 5

Dynamics: Euler-Lagrange Equations

Let us now describe how to derive a mathematical equations describing dynamics. We will follow the algebraic approach, introduced by Lagrange. Although, it has several well-known shortcomings that include difficulty in introducing not-ideal physical interactions between various components of a robot, this seems to be the most straightforward way to derive equations of motion without using mechanical intuition.

5.1 Example: from Newton's Law to Euler-Lagrange Equations

Let us start with rewriting a classical Newton's equation of motion for a particle moving along a vertical line in a gravitational field, illustrated below.



The 2^{nd} Newton's law for the particle states that the article's acceleration is proportional to the sum of all the forces acting on it, i.e.

$$\boxed{m\frac{d^2}{dt^2}\,y = \sum F_i = f - mg}$$

where

- m is the mass, which is the (constant) inertial characteristic of the particle,
- f is the cumulative external (vertical) force, and
- $-mg$ is the gravitational force acting on the particle, which is proportional to the acceleration of the gravity that is equal to $-g$.

The above equation of motion of the particle, can be rewritten in a quite different way!

The left-hand side of the equation of motion can be represented as

$$m\frac{d^2}{dt^2}y = \frac{d}{dt}\left(m\frac{d}{dt}y\right) = \frac{d}{dt}\left(m\frac{\partial}{\partial \dot{y}}\left[\frac{1}{2}\,\dot{y}^2\right]\right) = \frac{d}{dt}\left(\frac{\partial}{\partial \dot{y}}\mathcal{K}\right)$$

where

$$\mathcal{K} = \frac{1}{2}m\dot{y}^2$$

is so-called **kinetic energy**.

The gravitational force, appearing in the equation of motion, can be represented as

$$mg = \frac{\partial}{\partial y}\left[mgy\right] = \frac{\partial}{\partial y}\mathcal{P}$$

where

$$\mathcal{P} = mgy$$

is so-called **potential energy** of the gravity.

Then, the second Newton's law can be rewritten as the so-called **Euler-Lagrange equation**

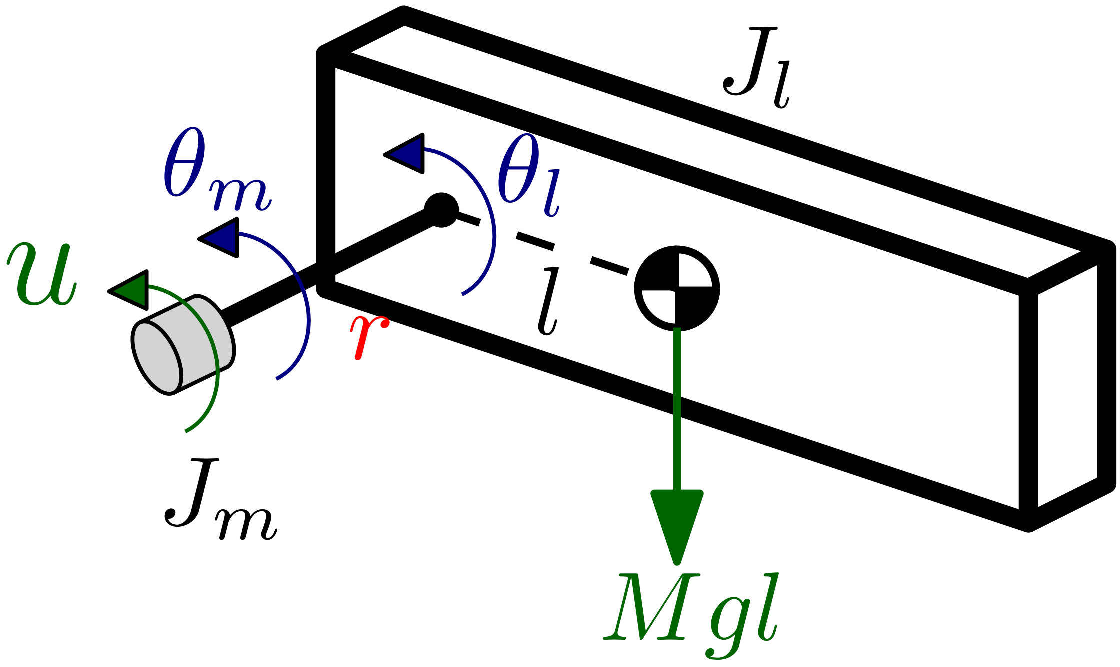
$$\boxed{\frac{d}{dt}\left(\frac{\partial}{\partial \dot{y}}\mathcal{L}\right) - \frac{\partial}{\partial y}\mathcal{L} = f} \quad \text{with} \quad \mathcal{L} = \mathcal{K} - \mathcal{P}$$

where the function $\mathcal{L}(y, \dot{y})$ is called the **Lagrangian** and is computed as the difference between the kinetic and the potential energies.

Equations of motion to describe dynamics of many mechanical systems can be derived using such formalism instead of direct application of the Newton's second law. Evidently, one just needs to know how to compute the energies.

5.2 Example: Dynamics of a Single Link Robot

We will discuss later how to compute the energies for a multi-link robot in a systematic way. Let us, however, illustrate derivation of the equation of motion for a one-link rotational robot schematically shown below.



Here, we have a rigid link of mass $\boldsymbol{M}$, absolute location of which is defined by the angle $\boldsymbol{\theta_l}$, that is coupled through a gearbox with ration $\boldsymbol{r}$, to a rotor of a DC motor, location of which is defined by the angle $\boldsymbol{\theta_m}$. The torque produced by the motor is denoted by $\boldsymbol{u}$ and it is amplified by the gearbox with the gain $\boldsymbol{r}$. Note that $\boldsymbol{\theta_m = r\theta_l}$ since (in the considered ideal case) the angle is reduced by the gear box with the gain $\boldsymbol{1/r}$.

The kinetic energy of our robot is a sum of kinetic energies of the link and of the rotor. Each of them is proportional to the square of the corresponding angular velocity, i.e.

- Kinetic energy:

$$\mathcal{K} = \frac{1}{2} J_m \dot{\theta}_m^2 + \frac{1}{2} J_l \dot{\theta}_l^2 = \frac{1}{2}\left(r^2 J_m + J_l\right)\dot{\theta}_l^2$$

where $\boldsymbol{J_m}$ and $\boldsymbol{J_l}$ are moments of inertia defining inertial characteristics of the two rigid bodies (the rotor of the motor and the link, respectively).

The potential energy (up to a constant initial value) is proportional to the height of the center of mass of the link with the coefficient of proportionality equal to $\boldsymbol{Mg}$, i.e.

- Potential energy:

$$\mathcal{P} = M g\, l\Big(1 - \cos\theta_l\Big)$$

Therefore,

- the Lagrangian is

$$\mathcal{L} = \mathcal{K} - \mathcal{P} = \frac{1}{2}\left(r^2 J_m + J_l\right)\dot{\theta}_l^2 - M g\, l\Big(1 - \cos\theta_l\Big)$$

and one can easily compute the equation for the dynamics to be

$$\boxed{\tfrac{d}{dt}\left(\tfrac{\partial}{\partial\dot{\theta}_l}\mathcal{L}\right) - \tfrac{\partial}{\partial\theta_l}\mathcal{L} = \tau_l = r u}$$

where $f = \tau_l = r u$ since the torque of the motor is amplified by the gearbox.

Finally, one obtains

$$\boxed{\left(r^2 J_m + J_l\right)\ddot{\theta}_l + M g l \sin\theta_l = r u}$$

The same equation can be obtained using the Lagrangian in terms of the motor variable

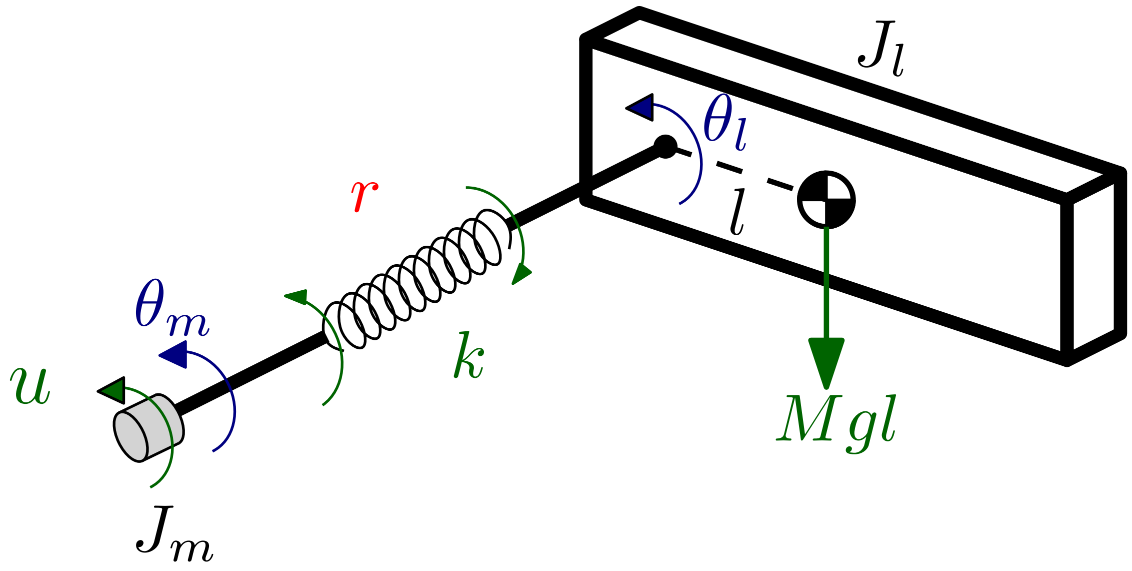
$$\mathcal{L} = \frac{1}{2}\left(J_m + \frac{J_l}{r^2}\right)\dot{\theta}_m^2 - M g\, l\Big(1 - \cos\frac{\theta_m}{r}\Big)$$

and the Euler-Lagrange equation

$$\tfrac{d}{dt}\left(\tfrac{\partial}{\partial\dot{\theta}_m}\mathcal{L}\right) - \tfrac{\partial}{\partial\theta_m}\mathcal{L} = \tau_m = u$$

Note that in the right-hand side here one must use the torque applied to the motor, τ_m, and not the torque applied to the link, τ_l, as in the previous case. This is related to the notion of the generalized force. We will see later that the choice of a generalized coordinate is not important (it could be, e.g., θ_l or θ_m) but one must be careful with obtaining a correct expression for the generalized force. In fact, $\theta_m = r\theta_l$ implies $\tau_l = r\,\tau_m$. This is related to the principle of virtual work, to be formulated later, that in the considered simple case can be written as $\tau_l\,\delta\theta_l = \tau_m\,\delta\theta_m$.

It is also useful to derive equations of motions for the same single link example but dropping the assumption of a perfect gearbox, that is with possibility to have $\theta_m \neq r\theta_l$. To proceed, one needs to propose a mechanical model for the connection of the two rigid bodies: the rotor of the motor and the link. One possibility is to use an ideal spring model illustrated below.



In this case, the energies become functions of two independent generalized coordinates and their derivatives. Including into the expression of the potential energy a quadratic term in $(\theta_m - r\theta_l)$ with a gain equal to the coefficient of stiffness of the spring, one obtains the Lagrangian

$$\mathcal{L} = \frac{1}{2}J_m\dot{\theta}_m^2 + \frac{1}{2}J_l\dot{\theta}_l^2 - Mgl\left(1 - \cos\theta_l\right) - \frac{1}{2}k\left(\frac{\theta_m}{r} - \theta_l\right)^2$$

and use it to obtain two differential equations

$$\frac{d}{dt}\left(\frac{\partial}{\partial\dot{\theta}_l}\mathcal{L}\right) - \frac{\partial}{\partial\theta_l}\mathcal{L} = \tau_l\,, \qquad \frac{d}{dt}\left(\frac{\partial}{\partial\dot{\theta}_m}\mathcal{L}\right) - \frac{\partial}{\partial\theta_m}\mathcal{L} = \tau_m$$

Realizing that the generalized forces are $\tau_l = 0$ and $\tau_m = u$, we have

$$J_l\,\ddot{\theta}_l + Mgl\sin\theta_l = k\left(\tfrac{\theta_m}{r} - \theta_l\right), \qquad J_m\,\ddot{\theta}_m + \tfrac{k}{r}\left(\tfrac{\theta_m}{r} - \theta_l\right) = u$$

It is useful to recover the previous equation of dynamics obtained under assumption of the identity $\theta_m - r\theta_l \equiv 0$, which is called a holonomic constraint (to be introduced in the next section). This can be done if we also assume that the stiffness is infinite, i.e $k = \infty$ and rewrite the differential equations substituting the elastic force with an unknown reaction force keeping the perfect relation between the two angles

$$F = k\left(\tfrac{\theta_m}{r} - \theta_l\right), \quad J_l\,\ddot{\theta}_l + Mgl\sin\theta_l = F, \quad J_m\,\ddot{\theta}_m + \frac{1}{r}F = u$$

The last two differential equations together with the holonomic constraint (the identity imposed on the variables) are called Euler-Lagrange equations of the first kind or constrained Euler-Lagrange equations.

Now, a single differential equation describing dynamics can be obtained substituting either $\boldsymbol{\theta_m = r\theta_l}$ or $\boldsymbol{\theta_l = \theta_m / r}$ and eliminating the unknown reaction force $\boldsymbol{F}$. However, in addition to $\left(\boldsymbol{r^2 J_m + J_l}\right)\ddot{\boldsymbol{\theta}}_l + \boldsymbol{Mgl}\sin\boldsymbol{\theta_l} = \boldsymbol{ru}$ we also obtain the expression for the force $\boldsymbol{F} = \boldsymbol{Mgl}\sin\boldsymbol{\theta_l} + \boldsymbol{J_l}\frac{\boldsymbol{ru - Mgl}\sin\boldsymbol{\theta_l}}{\boldsymbol{r^2 J_m + J_l}}$ that can be used to verify systems' integrity.

We will derive below Euler-Lagrange equations for a robotic manipulator. After that, we will show how to compute potential and kinetic energies using the variables of the DH convention defined in Chapter 2. However, in order to proceed, we need to introduce a few useful notions such as holonomic constraints, degrees of freedom, virtual work, generalized coordinates, and generalized forces.

5.3 Holonomic Constraints and Virtual Work

The Euler-Lagrange formalism leads to a description of dynamics for a mechanical system in the form of a system of nonlinear interconnected differential equations. The number of these equations coincides with the so-called number of degrees of freedom (DOF), provided the mechanical parts of the system are subject to so-called holonomic constraints. Both notions are to be defined next.

5.3.1 Degrees of Freedom and Holonomic Constraints

The **number of degrees of freedom** (DOF) is the number of scalar variables that are necessary and sufficient to describe location of all the components of a mechanical system. For example, location of a particle (material point) in $\boldsymbol{3D}$ can be described by its $\boldsymbol{3}$ Cartesian coordinates and therefore, a single particle has $\boldsymbol{3}$ DOF.

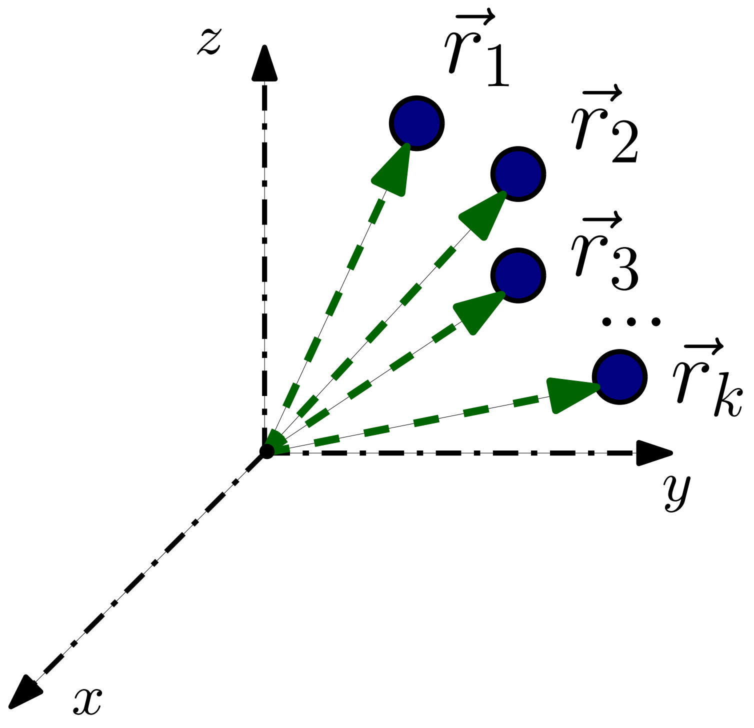
If, by some reasons, we know that a particle is restricted to stay on a particular plane, then its location can be described by $\boldsymbol{2}$ Cartesian (or polar) coordinates introduced on the plane. Hence, it has $\boldsymbol{2}$ DOF. Here, it is useful to realize, that the restriction of the particle location can be described by a single (linear) scalar algebraic equation on its $\boldsymbol{3}$ coordinates (recall the equation of a plane in a Euclidean space). Such a linear or nonlinear identity satisfied by coordinates is called a holonomic constraint. Formally, a **holonomic constraint** is an algebraic relation among the scalar quantities specifying locations of various components of a mechanical system. One can formally

find the number of DOF of a system with constraints as a difference between the number of DOF for a system without constraints and the number of constraints, provided the constraints are independent (no one of them is implied by others) and the number of constraints on each component is not bigger than the number of DOF of this component.

For an illustration, consider a system of $\mathbf{2}$ particles in a vertical plane connected by a weightless rod. To use the definition directly, one can compute the number of DOF as follows: To describe location of the system we need to specify $\mathbf{2}$ coordinates of one particle and $\mathbf{1}$ angle that the rod makes with a fixed line on the plane. Hence, the number of DOF for the system is $\mathbf{3 = 2 + 1}$. To compute this number using constraints, one say that there are $\mathbf{2}$ particles in plane with $\mathbf{2}$ DOFs each and $\mathbf{1}$ holonomic constraint (the distance between them), so that we have $\mathbf{3 = 2 \cdot 2 - 1}$. Alternatively, one can see $\mathbf{2}$ particles in space, i.e. with $\mathbf{3}$ DOFs each, and $\mathbf{3}$ holonomic constraints ($\mathbf{1}$ fixed distance and $\mathbf{2}$ restrictions to the plane), so that $\mathbf{3 = 3 \cdot 2 - (1 + 2)}$.

A more interesting example is a free rigid body in space. We know from our discussion in Chapter 2 that, due to rigidity, locations of all points can be described by $\mathbf{6}$ variables: $\mathbf{3}$ describe location of a single point, called a pole, while $\mathbf{3}$ other describe the orientation of the body. From the other hand, note that, formally, a rigid body consists of an infinite number of particles, distances among which are pairwise constant. So, taking $\mathbf{3}$ particles belonging to the body that do not belong to a single line (to ensure that two constraints do not imply the other) we compute that the number of DOF is not less than $\mathbf{6 = 3 \cdot 3 - 3}$, where we counted $\mathbf{3}$ particles with $\mathbf{3}$ DOFs each and $\mathbf{3}$ fixed distances. Now, one can notice that adding another particle would add $\mathbf{3}$ DOFs together with at least $\mathbf{3}$ fixed distances to the other particles. So, the number of DOF seems really to be $\mathbf{6}$.

Let us now, for simplicity, forget the fact that we work with rigid bodies and formalize the notions just discussed for a mechanical system formed by a finite set of particles. To this end, consider an unconstrained system of $\boldsymbol{k}$ particles in $\mathbf{3}\boldsymbol{D}$. The number of DOF is $\mathbf{3}\,\boldsymbol{k}$ if they are not restricted and is less, if the particles are constrained.



Each of l (independent) constraints imposed on k particles with coordinates $r_1, r_2, \dots, r_k \in \mathbb{R}^3$, written with respect to an inertial frame[1], is called **holonomic**, if it can be written as a scalar identity

$$g_i(r_1, r_2, \dots, r_k) = 0, \qquad i = 1, 2, \dots, l$$

For example, for the system of 2 particles joined by a massless rigid wire of length L we have

$$r_1, r_2 \in \mathbb{R}^3 : \ g(r_1, r_2) = \|r_1 - r_2\|^2 - L^2 = (r_1 - r_2)^T (r_1 - r_2) - L^2 = 0$$

It is important to realize that

Presence of each holonomic constraint implies presence of (internal) forces (called **constraint forces**), that make the particles to obey the constraint.

Differentiating each constraint function g_i with respect to time, one obtains another constraint, which is another identity that must be satisfied by the state variables, i.e. by the positions and velocities

$$\tfrac{d}{dt} g_i(r_1, r_2, \dots, r_k) = \frac{\partial g_i}{\partial r_1} \tfrac{d}{dt} r_1 + \cdots + \frac{\partial g_i}{\partial r_k} \tfrac{d}{dt} r_k = 0$$

or

$$\frac{\partial g_i}{\partial r_1} dr_1 + \cdots + \frac{\partial g_i}{\partial r_k} dr_k = 0$$

[1]Here and below we skip the superscript indicating this particular frame since it is the only frame.

that is a constraint in the form

$$\omega_1(r_1,\dots r_k)dr_1 + \cdots + \omega_k(r_1,\dots r_k)dr_k = 0$$

If a constraint of this kind appears not as a consequence of a holonomic constraint, that is if it cannot be integrated to a form of a holonomic constraint, than it is called **non-holonomic**. Non-holonomic constraints can be often understood as a consequence of ideal sliding that restricts direction of possible velocities but not possible positions (e.g., a car cannot move in the direction orthogonal to the wheels but can certainly be driven in the corresponding parallel location).

5.3.2 Generalized Coordinates, Virtual Displacements

Let us consider the case when only holonomic constraints are present and, for the sake of simplicity, assume as above that we have just a finite number of particles, rigidly connected pairwise by weightless wires.

If a system of k particles, positions of which are specified as components of $3D$ vectors $\vec{r}_i$, $i = 1,\dots,k$, is subject to a set of $l \geq 1$ holonomic constraints, then

- it may be possible to express the coordinates of the particles (locally) as functions of n variables, $q_1,\dots,q_n$, with $n \leq 3k$

$$r_1 = r_1(q_1,\dots,q_n),\ r_2 = r_2(q_1,\dots,q_n),\ \dots,\ r_k = r_k(q_1,\dots,q_n)$$

Suppose this is done. Then,

- the smallest set of such variables $q_1,\dots,q_n$ is called the set of **generalized coordinates**, while
- the number of such variables, n, is called the **number of degrees of freedom**.

It should be noted that if the system consists of an **infinite** number of particles, then it might have **finite** number of DOF. For example, the number of DOF for an unconstrained rigid body, moving in $3D$ is 6.

Given a system of k-particles under l holonomic constraint

$$g_i(r_1,\ r_2,\ \dots,\ r_k) = 0, \qquad i = 1,\ 2,\ \dots,\ l$$

Suppose, r_i for $i = 1,\dots,k$ satisfy the constraints, by definition the set of infinitesimal displacements

$$\delta r_1,\ \delta r_2,\ \dots,\ \delta r_k$$

that are consistent with the constraints, i.e.

$$g_i(r_1+\delta r_1,\, r_2+\delta r_2,\, \ldots,\, r_k+\delta r_k) = 0, \qquad i = 1,\, 2,\, \ldots,\, l$$

are called **virtual displacements**.

Expanding the left-hand side of the last expression into the Taylor series, using the previous expression and dropping higher-order terms, one immediately obtains

$$\frac{\partial g_i}{\partial r_1}\,\delta r_1 + \cdots + \frac{\partial g_i}{\partial r_k}\,\delta r_k = 0, \quad i = 1,\, 2,\, \ldots,\, l$$

This is a system of l linear equation with respect to k vector variables δr_i that must be satisfied. The fact that the constraints are independent mathematically means that the rectangular matrix constructed from the partial derivatives has the maximal rank. It follows that only

$$n = 3\,k - l \quad \text{in } 3D \qquad \text{and} \qquad n = 2\,k - l \quad \text{in } 2D$$

virtual displacement are independent. The respective n independent linear combinations of the virtual displacement can be taken arbitrary and define the freedom of motions that are consistent with all the holonomic constraints. This is why n is called the number of DOF.

As an example, let us compute virtual displacements of a rigid planar bar. Such infinitesimal motions do not destroy the virtual constraint on the end-points of the bar

$$g = (r_1 - r_2)^{T}\,(r_1 - r_2) - L^2 = 0$$

if r_1 and r_2 are perturbed

$$r_1 \;\to\; (r_1 + \delta r_1)\,, \qquad r_2 \;\to\; (r_2 + \delta r_2)$$

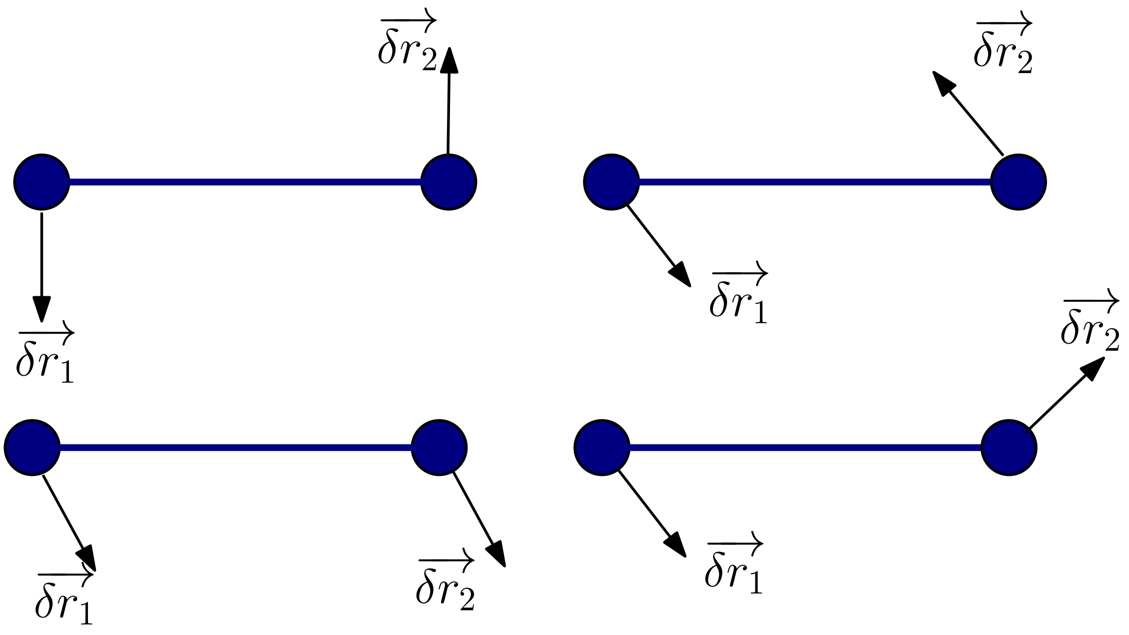
that is

$$\Big((r_1 + \delta r_1) - (r_2 + \delta r_2)\Big)^{T}\Big((r_1 + \delta r_1) - (r_2 + \delta r_2)\Big) = L^2$$

It follows (dropping the second order terms) that

$$(r_1 - r_2)^{T}\,(\delta r_1 - \delta r_2) = 0$$

This means that the vector $\vec{r}_1 - \vec{r}_2$, defining relative displacement of the end-point of the bar, must be orthogonal to the vector $\delta\vec{r}_1 - \delta\vec{r}_2$, which is defined by the differences of the possible virtual displacement of the end points. Some examples, including the pure translations of the bar and the pure rotation of the bar are shown in the figure below.


$\overrightarrow{\delta r_2}$
$\overrightarrow{\delta r_1}$
$\overrightarrow{\delta r_2}$
$\overrightarrow{\delta r_1}$
$\overrightarrow{\delta r_2}$
$\overrightarrow{\delta r_1}$
$\overrightarrow{\delta r_2}$
$\overrightarrow{\delta r_1}$

Note that all other possible rigid motions are clearly combinations of translations and rotations. Let us assume now that the bar is restricted to the plane. Instead of introducing two other holonomic constraints, let us assume that the vectors defining the end-point are in $\boldsymbol{2D}$, i.e.

$$\boldsymbol{r_1} = \begin{bmatrix} \boldsymbol{x_1} \\ \boldsymbol{y_1} \end{bmatrix}, \qquad \boldsymbol{r_2} = \begin{bmatrix} \boldsymbol{x_2} \\ \boldsymbol{y_2} \end{bmatrix}$$

Let us introduce $\boldsymbol{n = 3 = 4 - 1 = 2\,k - l}$ generalized coordinates as the coordinates of the left end-points and the angle defining orientation of the bar on the plane

$$\boldsymbol{q_1 = x_1}, \qquad \boldsymbol{q_2 = y_1}, \qquad \boldsymbol{q_3 = \mathrm{atan2}\,(y_2 - y_1, x_2 - x_1)}$$

Expressing the vectors in terms of the generalized coordinates, one obtains

$$\boldsymbol{r_1} = \begin{bmatrix} \boldsymbol{q_1} \\ \boldsymbol{q_2} \end{bmatrix}, \qquad \boldsymbol{r_2} = \begin{bmatrix} \boldsymbol{q_1 + L\cos(q_3)} \\ \boldsymbol{q_2 + L\sin(q_3)} \end{bmatrix}$$

while for the (not independent) virtual displacements we have

$$\boldsymbol{\delta r_1} = \begin{bmatrix} \boldsymbol{\delta q_1} \\ \boldsymbol{\delta q_2} \end{bmatrix}, \qquad \boldsymbol{\delta r_2} = \begin{bmatrix} \boldsymbol{\delta q_1 - L\sin(q_3)\,\delta q_3} \\ \boldsymbol{\delta q_2 + L\cos(q_3)\,\delta q_3} \end{bmatrix}$$

Re-computing the restriction on the virtual displacements one obtains the identity

$$(\boldsymbol{r}_1 - \boldsymbol{r}_2)^T (\delta \boldsymbol{r}_1 - \delta \boldsymbol{r}_2) = \begin{bmatrix} L\cos(q_3) \\ L\sin(q_3) \end{bmatrix}^T \begin{bmatrix} -L\sin(q_3)\,\delta q_3 \\ L\cos(q_3)\,\delta q_3 \end{bmatrix} \equiv 0$$

that tells us that all three generalized coordinates are indeed independent and not restricted by the holonomic constraint in any way. Note however, that it is typically easy to compute the number of DOF but may in general be difficult to introduce the generalized coordinates. Whenever they are introduced, the holonomic constraints must be automatically satisfied.

Using the concept of virtual displacements one can introduce a very important principle of mechanics.

5.3.3 Principle of Virtual Work and Constraint Forces

Consider a system of k-particles; suppose that

- The system has $l \geq 1$ holonomic constraints, that is each i-th particle is exposed to a (lumped) **constraint force** $\boldsymbol{f}_i^c$, $i = 1, \ldots, k$;
- There are, possibly, (lumped) external forces $\boldsymbol{f}_i^e$, $i = 1, \ldots, k$, applied to the particles as well;
- Each particle is in an equilibrium (i.e. at rest) $\Rightarrow$ $\boldsymbol{f}_i^c + \boldsymbol{f}_i^e = 0$, $\forall i$.

As a result, trivially, the **virtual work**, which is defined as the product of the total force and the virtual displacement,

$$\delta W^i = \left(\boldsymbol{f}_i^c + \boldsymbol{f}_i^e\right)^T \delta \boldsymbol{r}_i,$$

done by all the forces acting on each i^{th} particle, is equal to zero.

Then, the total virtual work done by all the forces applied to the whole system of particle is zero as well:

$$\delta W = \sum_{i=1}^{k} \delta W^i = \sum_{i=1}^{k} \left(\boldsymbol{f}_i^c + \boldsymbol{f}_i^e\right)^T \delta \boldsymbol{r}_i = 0$$

Now, the constraint forces are defined as forces that keep the constraints satisfied, while the virtual displacements are assumed to be consistent with them. Hence, one can assume that the work done by the constraint forces along the virtual displacement must be equal to zero, i.e.

$$\sum_{i=1}^{k} \left(\boldsymbol{f}_i^c\right)^T \delta \boldsymbol{r}_i = 0$$

Whenever this key assumption is true, the holonomic constraints are called **ideal**; and we will assume that this is the case.

Combining the last two equations, one concludes that for a system of particles, which are at rest, the work done by all the external forces along the virtual displacement is equal to zero as well

$$\boxed{\sum_{i=1}^{k} \left(\boldsymbol{f}_i^e\right)^T \delta \boldsymbol{r}_i = \boldsymbol{0}}$$

This equation is called the **principle of virtual work**.

Next, we will introduce a generalization of this principle to the case of a system of moving particles and use it to derive the equations of motions that do not contain the internal forces implied by the constraints.

As a result, we will be able to directly predict and compute the motions as solutions of a system of differential equations without computation of the constraint forces, provided we have all the external forces specified as functions of time or other internal variables.

5.4 D'Alembert Principle and Dynamics

Let us derive the Euler-Lagrange equations of motion for a finite family of particles using generalized coordinates and a fundamental general principle introduced next.

5.4.1 D'Alembert Principle

Consider a system of $\boldsymbol{k}$-particles; suppose that

- The system has $\boldsymbol{l} \geq \boldsymbol{1}$ **ideal holonomic constraints**, that is each $\boldsymbol{i}$-th particle is exposed to a (lumped) **constraint force** $\boldsymbol{f}_i^c$, $\boldsymbol{i} = \boldsymbol{1}, \ldots, \boldsymbol{k}$;
- There are, possibly, (lumped) external forces $\boldsymbol{f}_i^e$, $\boldsymbol{i} = \boldsymbol{1}, \ldots, \boldsymbol{k}$, applied to the particles as well;
- **The system is moving.**

The D'Alembert Principle states that for a system of moving particles the work done by all the external forces along the virtual displacements is equal to zero provided we count the inertia forces, i.e.

$$\boxed{\sum_{i=1}^{k} \left(\boldsymbol{f}_i^e - \frac{d}{dt}\left[\boldsymbol{m}_i \dot{\boldsymbol{r}}_i\right]\right)^T \delta \boldsymbol{r}_i = \boldsymbol{0}}$$

where $\boldsymbol{m}_i$ is the mass of the $\boldsymbol{i}^{th}$ particle.

Obviously, the expression above reduces to the principle of virtual work whenever the particles are not moving so that $\dot{\boldsymbol{r}}_i = \boldsymbol{0}\ \forall \boldsymbol{i}$.

To derive equations of motions we will proceed as follows:

- Rewrite $\sum_{i=1}^{k} \left(\boldsymbol{f}_i^e\right)^T \delta \boldsymbol{r}_i$ as a function of generalized coordinates $\boldsymbol{q}_i$ and the (independent) **virtual generalized displacements** $\delta \boldsymbol{q}_i$, that are different from the virtual displacements $\delta \boldsymbol{r}_i$. This will lead to a definition of a generalized force.
- Rewrite $\sum_{i=1}^{k} \frac{d}{dt}\left[\boldsymbol{m}_i \dot{\boldsymbol{r}}_i\right]^T \delta \boldsymbol{r}_i$ as a function of generalized coordinates $\boldsymbol{q}_i$ and the virtual generalized displacements $\delta \boldsymbol{q}_i$ as well.
- Substitute everything into the equation of the D'Alembert principle and use the fact that $\delta \boldsymbol{q}_i$ are independent and arbitrary by combining the terms and equating to zeros the coefficients in front of each $\delta \boldsymbol{q}_i$.

5.4.2 Generalized Forces

Using the chain rule, the not-independent virtual displacements δr_i, $i = 1, \ldots, k$ in $3D$, which are subject to l holonomic constraints, are computed in terms of the independent scalar generalized displacements δq_j, $j = 1, \ldots, n$, as follows

$$\delta r_i = \sum_{j=1}^{n} \delta q_j \, \frac{\partial r_i}{\partial q_j}, \quad i = 1, \ldots, k$$

where the partial derivatives are taken component-wise and, as before, $n = 3\,k - l$. Then,

$$\begin{aligned} \sum_{i=1}^{k} (f_i^e)^T \delta r_i &= \sum_{i=1}^{k} (f_i^e)^T \left(\delta q_j \sum_{j=1}^{n} \frac{\partial r_i}{\partial q_j} \right) \\ &= \sum_{j=1}^{n} \delta q_j \left(\sum_{i=1}^{k} (f_i^e)^T \frac{\partial r_i}{\partial q_j} \right) = \sum_{j=1}^{n} \delta q_j \, \psi_j \end{aligned}$$

The function ψ_j is called j^{th} generalized force:

$$\psi_j = \sum_{i=1}^{k} \left((f_i^e)^T \frac{\partial r_i}{\partial q_j} \right) = \sum_{i=1}^{k} \left((f_i^e)^T \frac{\partial \dot r_i}{\partial \dot q_j} \right)$$

where for the last equality we have used the identity $\boxed{\dfrac{\partial r_i}{\partial q_j} = \dfrac{\partial \dot r_i}{\partial \dot q_j}}$ that follows from the chain rule of differentiation

$$\dot r_i = \frac{\partial r_i}{\partial q_j} \dot q_j + \sum_{k \neq j} \frac{\partial r_i}{\partial q_k} \dot q_k \quad \Rightarrow \quad \frac{\partial}{\partial \dot q_j} (\dot r_i) = \frac{\partial}{\partial \dot q_j} \left(\frac{\partial r_i}{\partial q_j} \dot q_j + \ldots \right)$$

The next two cases are somewhat generic:

- When q_j is defined as a position variable, e.g. when it is one of the coordinates of one of the particles, ψ_j can be associated with a projection of a force acting along its direction of change.
- When q_j is defined as an angular variable, e.g. when it is an angle (partially or fully) defining orientation of a mass-less link connecting two particles, ψ_j can be associated with a moment acting along the corresponding rotation.

As an example, let us return to a planar bar but, for the sake of simplicity, assume that it is formed by two particles connected by a rigid mass-less rod.

We have

$$\dot{\boldsymbol{r}}_1 = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}, \qquad \dot{\boldsymbol{r}}_2 = \begin{bmatrix} \dot{q}_1 - L\sin(q_3)\,\dot{q}_3 \\ \dot{q}_2 + L\cos(q_3)\,\dot{q}_3 \end{bmatrix}$$

Suppose now that there are two (lumped) external forces acting on the particles

$$\boldsymbol{f}_1^e = \begin{bmatrix} F_{1x} \\ F_{1y} \end{bmatrix}, \qquad \boldsymbol{f}_2^e = \begin{bmatrix} F_{2x} \\ F_{2y} \end{bmatrix}$$

Using the formula above one can easily compute the generalized forces

$$\psi_1 = F_{1x} + F_{2x}, \quad \psi_2 = F_{1y} + F_{2y}, \quad \psi_3 = F_{2x}(-L\sin q_3) + F_{2y}(L\cos q_3)$$

So, the first generalized force is the projection of the sum of forces applied to the system onto the x-axis, the second generalized force is the projection of the sum of forces applied to the system onto the y-axis, while the third generalized force is the moment of all the forces applied to the system computed with respect to the left end-point of the bar.

In a similar way, the notion of a **generalized force**, which is typically a force or a moment, can be introduced in the case when the mechanical system consists of several **rigid bodies**. The difference is that the fact that there are infinitely many points give rise to the notion of a total (external) torque applied to a rigid body in parallel with a total (external) force. As a result, the generalized force ψ_j acting on the virtual displacement δq_j are formed not only by forces $\boldsymbol{f}_i^e$ but also by torques $\boldsymbol{\tau}_i^e$ applied to the system of bodies (they can be understood as action of two forces with identical magnitudes, opposite directions that are applied at different points). The generalized forces can be computed using the next formula

$$\psi_j = \sum_{i=1}^{k} \left(\left(\boldsymbol{f}_i^e\right)^T \frac{\partial \boldsymbol{v}_i^e}{\partial \dot{q}_j} + \left(\boldsymbol{\tau}_i^e\right)^T \frac{\partial \boldsymbol{\omega}_i^e}{\partial \dot{q}_j} \right)$$

where $\boldsymbol{v}_i^e = \dot{\boldsymbol{r}}_i^e = -\dot{\boldsymbol{r}}_i$ represents coordinates of the vector opposite to the linear velocity of the point of application of the force and $\boldsymbol{\omega}_i^e = -\boldsymbol{\omega}_i$ represents the coordinates of the vector opposite to the angular velocity of the i-th rigid body respectively. Obviously, this formula can be re-written using appropriate Jacobian matrices as

$$\psi_j = \sum_{i=1}^{k} \left(\left(\frac{\partial \boldsymbol{v}_i^e}{\partial \dot{q}_j} \right)^T \boldsymbol{f}_i^e + \left(\frac{\partial \boldsymbol{\omega}_i^e}{\partial \dot{q}_j} \right)^T \boldsymbol{\tau}_i^e \right) = \sum_{i=1}^{k} \left(J_{v_i^e}^T \boldsymbol{f}_i^e + J_{\omega_i^e}^T \boldsymbol{\tau}_i^e \right)$$

Let us return to the case of a finite number of particles and proceed according to the stated above plan.

5.4.3 Virtual Work of the Inertial Forces

The expression for the virtual work of the inertial forces can be rewritten as

$$\sum_{i=1}^{k} \tfrac{d}{dt}\left[m_i \dot{r}_i\right]^T \delta r_i = \sum_{i=1}^{k} m_i \ddot{r}_i^T \left(\delta q_j \sum_{j=1}^{n} \frac{\partial r_i}{\partial q_j}\right) = \sum_{i=1}^{k}\sum_{j=1}^{n} m_i\, \delta q_j\, \ddot{r}_i^T \frac{\partial r_i}{\partial q_j}$$

Let us now use the following identity, obtained using the product rule for differentiation

$$\frac{d}{dt}\left[\dot{r}_i^T \frac{\partial r_i}{\partial q_j}\right] = \ddot{r}_i^T \frac{\partial r_i}{\partial q_j} + \dot{r}_i^T \frac{d}{dt}\left[\frac{\partial r_i}{\partial q_j}\right]$$

It follows that

$$\sum_{j=1}^{n}\sum_{i=1}^{k} m_i\, \delta q_j\, \ddot{r}_i^T \frac{\partial r_i}{\partial q_j} = \sum_{j=1}^{n} \delta q_j \left[\sum_{i=1}^{k} m_i \left(\frac{d}{dt}\left[\dot{r}_i^T \frac{\partial r_i}{\partial q_j}\right] - \dot{r}_i^T \frac{d}{dt}\left[\frac{\partial r_i}{\partial q_j}\right]\right)\right]$$

Now, we will use the following identities

$$v_i = \dot{r}_i = \sum_{j=1}^{n} \frac{\partial r_i}{\partial q_j}\dot{q}_j = \sum_{l=1}^{n} \frac{\partial r_i}{\partial q_l}\dot{q}_l \quad \Rightarrow \quad \frac{\partial v_i}{\partial \dot{q}_j} = \frac{\partial r_i}{\partial q_j}$$

and

$$\frac{d}{dt}\left[\frac{\partial r_i}{\partial q_j}\right] = \sum_{l=1}^{n} \frac{\partial^2 r_i}{\partial q_j \partial q_l}\dot{q}_l = \frac{\partial}{\partial q_j}\left[\sum_{l=1}^{n} \frac{\partial r_i}{\partial q_l}\dot{q}_l\right] = \frac{\partial v_i}{\partial q_j}$$

Substituting them into the last expression for the virtual work of the inertial forces, we obtain

$$\begin{aligned} \sum_{i=1}^{k} \tfrac{d}{dt}\left[m_i \dot{r}_i\right]^T \delta r_i &= \sum_{j=1}^{n} \delta q_j \left[\sum_{i=1}^{k} \frac{d}{dt}\left[m_i v_i^T \frac{\partial v_i}{\partial q_j}\right] - m_i v_i^T \frac{\partial v_i}{\partial q_j}\right] \\ &= \sum_{j=1}^{n} \delta q_j \left[\frac{d}{dt}\frac{\partial \mathcal{K}}{\partial \dot{q}_j} - \frac{\partial \mathcal{K}}{\partial q_j}\right] \end{aligned}$$

where

$$\mathcal{K} = \mathcal{K}(q, \dot{q}) = \frac{1}{2}\sum_{i=1}^{k} m_i\, v_i^T v_i = \frac{1}{2}\sum_{i=1}^{k} m_i \left\|v_i\right\|^2 = \frac{1}{2}\sum_{i=1}^{k} m_i \left\|\dot{r}_i\right\|^2$$

is called the **kinetic energy** of the system of particles. It can be treated as a function of generalized coordinates and generalized velocities and can be computed from the amplitudes of the velocities (speeds) of all the particles.

5.4.4 Equations of Motion for Systems of Particles

To summarize, the expression for the d'Alembert principle

$$\sum_{i=1}^{k}\left(f_i^e - \frac{d}{dt}\left[m_i\dot{r}_i\right]\right)^T \delta r_i = 0$$

can be rewritten using the following two expressions derived above

$$\sum_{i=1}^{k}\frac{d}{dt}\left[m_i\dot{r}_i\right]^T \delta r_i = \sum_{j=1}^{n}\left[\frac{d}{dt}\frac{\partial\mathcal{K}}{\partial\dot{q}_j} - \frac{\partial\mathcal{K}}{\partial q_j}\right]\delta q_j, \quad \sum_{i=1}^{k}\left(f_i^e\right)^T \delta r_i = \sum_{j=1}^{n}\psi_j\delta q_j$$

as follows

$$\sum_{j=1}^{n}\left\{\frac{d}{dt}\frac{\partial\mathcal{K}}{\partial\dot{q}_j} - \frac{\partial\mathcal{K}}{\partial q_j} - \psi_j\right\}\delta q_j = 0$$

Using the fact that the generalized virtual displacement δq_j are independent and arbitrary, one concludes that the coefficients in front of δq_j must be equal to zeros and therefore

$$\frac{d}{dt}\frac{\partial\mathcal{K}}{\partial\dot{q}_j} - \frac{\partial\mathcal{K}}{\partial q_j} = \psi_j, \qquad j = 1, \dots n$$

It is customary to separate from the generalized forces ψ_j the ones that can be computed as partial derivatives of a scalar function $\mathcal{P}(q)$ that is called **potential energy** and often is caused by gravity. With such a separation, the equation above can be rewritten as

$$\boxed{\frac{d}{dt}\frac{\partial\mathcal{L}}{\partial\dot{q}_j} - \frac{\partial\mathcal{L}}{\partial q_j} = \tau_j}, \qquad \psi_j = -\frac{\partial\mathcal{P}}{\partial q_j} + \tau_j, \qquad \boxed{\mathcal{L} = \mathcal{K}(q,\dot{q}) - \mathcal{P}(q)},$$

where $\mathcal{L}(q,\dot{q})$ is called the **Lagrangian** and is computed as the difference between the kinetic energy and the potential energy of the system.

Let us continue consideration of our planar bar example. Suppose a system of two particles connected by a mass-less link is moving in a vertical plane as above but now it is under influence of not only external forces but also gravitational ones.

We already know that the planar two-component forces applied to the articles result in

$$\psi_1 = F_{1x}+F_{2x}, \quad \psi_2 = F_{1y}+F_{2y}, \quad \psi_3 = -F_{2x}L\sin(q_3)+F_{2y}L\cos(q_3)$$

Differentiating the expressions for the coordinates of the position vectors

$\vec{r}_1$ and $\vec{r}_2$ with respect to time we obtain as before

$$v_1 = \dot{r}_1 = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}, \qquad v_2 = \dot{r}_2 = \begin{bmatrix} \dot{q}_1 - L\sin(q_3)\dot{q}_3 \\ \dot{q}_2 + L\cos(q_3)\dot{q}_3 \end{bmatrix}$$

Hence, the kinetic energy is

$$\mathcal{K} = \frac{1}{2}\Big(m_1\|v_1\|^2 + m_2\|v_2\|^2\Big) = (m_1+m_2)\left(\dot{q}_1^2 + \dot{q}_2^2\right) + \frac{m_2L^2}{2}\dot{q}_3^2 + m_2L(\dot{q}_2\cos(q_3) - \dot{q}_1\sin(q_3))\dot{q}_3$$

To compute the potential energy, we just add potential energies of the particles that (up to a non-important constant offset) are proportional to the y-coordinates of the articles. So,

$$\mathcal{P} = (m_1\, r_1 + m_2\, r_2)\begin{bmatrix} 0 \\ g \end{bmatrix} = g\Big((m_1+m_2)q_2 + m_2L\sin(q_3)\Big)$$

Now, $\mathcal{L} = \mathcal{K} - \mathcal{P}$, the partial derivatives are

$$\frac{\partial \mathcal{L}}{\partial q_1} = 0, \quad \frac{\partial \mathcal{L}}{\partial q_2} = -(m_1+m_2)g, \quad \frac{\partial \mathcal{L}}{\partial q_3} = -m_2L\Big((g + \dot{q}_3\dot{q}_1)\cos(q_3) + \dot{q}_3\dot{q}_2\sin(q_3)\Big)$$

$$\frac{\partial \mathcal{L}}{\partial \dot{q}_1} = 2(m_1+m_2)\dot{q}_1 - m_2L\sin(q_3)\dot{q}_3, \quad \frac{\partial \mathcal{L}}{\partial \dot{q}_2} = 2(m_1+m_2)\dot{q}_2 + m_2L\cos(q_3)\dot{q}_3,$$
$$\frac{\partial \mathcal{L}}{\partial \dot{q}_3} = m_2L^2\dot{q}_3 + m_2L(\dot{q}_2\cos(q_3) - \dot{q}_1\sin(q_3))$$

and substituting them into the equations of motion we obtain the final answer

$$\begin{aligned}
&2(m_1+m_2)\ddot{q}_1 - m_2L\sin(q_3)\ddot{q}_3 - m_2L\cos(q_3)\dot{q}_3^2 = F_{1x} + F_{2x},\\
&2(m_1+m_2)\ddot{q}_2 + m_2L\cos(q_3)\ddot{q}_3 - m_2L\sin(q_3)\dot{q}_3^2 = (F_{1y} - gm_1) + (F_{2y} - gm_2),\\
&-m_2\sin(q_3)\ddot{q}_1 + m_2\cos(q_3)\ddot{q}_2 + m_2L\ddot{q}_3 = (F_{2y} - gm_2)\cos(q_3) - F_{2x}\sin(q_3)
\end{aligned}$$

Equations of motion for other systems of particles with holonomic constraints can be derived in a similar straightforward way.

Moreover, the same expression,

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}_j} - \frac{\partial \mathcal{L}}{\partial q_j} = \tau_j, \quad j = 1,\ldots,n, \qquad \mathcal{L} = \mathcal{K}(q,\dot{q}) - \mathcal{P}(q),$$

known as **Euler-Lagrange equations**, is valid for a system of rigid bodies with n DOF as well. However, the expressions for the energies are more involved than in the case of a finite number of particles. Let us derive them.

5.5 Lagrangian for an n-Link Robot

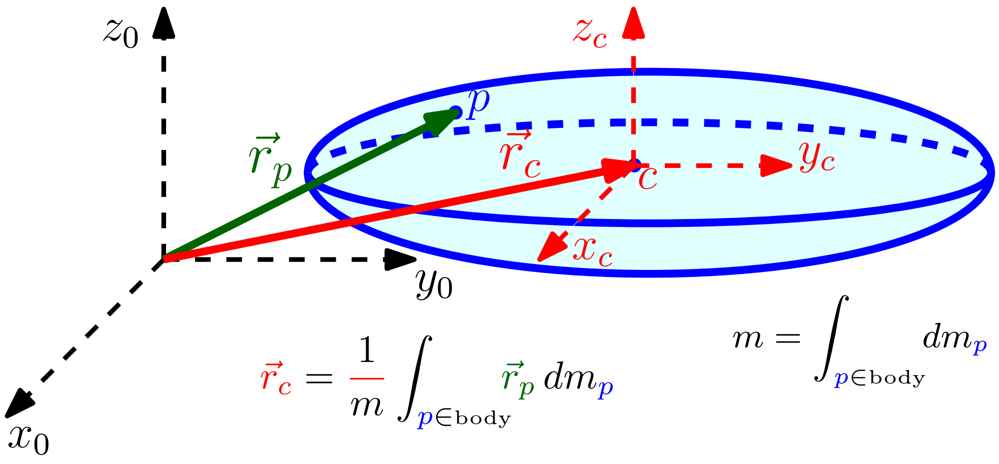
An $\boldsymbol{n}$-link robot is a mechanical system with $\boldsymbol{n}$ DOF, where the variables of the kinematic description with the DH convention of Chapter 2 can be taken as the generalized coordinates.

The **Lagrangian** for a robot can be computed adding the kinetic energies of the links and subtracting their potential energies.

5.5.1 Kinetic Energy for a Rigid Body

We start with a rigid body, or a single link that is not restricted by any holonomic constraints.

A kinetic energy of a rigid body is a quadratic form of linear velocities of its single chosen point, called the pole, and of either components of the angular velocity or other quantities specifying the rate of rotation (such as, e.g., derivatives of the Euler angles).



$$\vec{r}_c = \frac{1}{m}\int_{p\in\text{body}} \vec{r}_p\, dm_p \qquad m = \int_{p\in\text{body}} dm_p$$

In the particular case when the pole is taken at the center of mass, the kinetic energy can be computed as a sum of two terms responsible for the energy of rotation and the energy of translation, more precisely,

$$\begin{aligned}
\mathcal{K} &= \frac{1}{2}\int_{p\in\text{body}} \underbrace{\left|\vec{v} + \vec{\omega}\times(\vec{r}_p - \vec{r})\right|}_{v_p^T v_p} dm_p \\
&= \frac{1}{2}\int_{p\in\text{body}} \Big(v + \omega\times(r_p - r)\Big)^T \Big(v + \omega\times(r_p - r)\Big)\, dm_p \\
&= \frac{1}{2}\int_{p\in\text{body}} \Big(v - S(\vec{r}_p - \vec{r})\,\omega\Big)^T \Big(v - S(\vec{r}_p - \vec{r})\,\omega)\Big)\, dm_p \\
&= \frac{1}{2}\int_{p\in\text{body}} v^T v\, dm_p + \frac{1}{2}\int_{p\in\text{body}} \omega^T S^T(\vec{r}_p - \vec{r})\, S(\vec{r}_p - \vec{r})\,\omega\, dm_p \\
&= \frac{1}{2} m\, v^T v + \frac{1}{2}\omega^T \mathcal{I}\omega
\end{aligned}$$

where $\boldsymbol{v} = \dot{\boldsymbol{r}}$ represents the velocity in $\boldsymbol{3D}$ of the center of mass, defined by $\boldsymbol{r} = \dfrac{1}{m}\displaystyle\int_{p\in\text{body}} \boldsymbol{r}_p\, dm_p$, and $\boldsymbol{\omega}$ represents the body's angular velocity, $m = \int_{p\in\text{body}} dm_p$ is the total mass, while the 3×3 symmetric positive definite matrix $\mathcal{I}$ is so-called matrix of inertia.

An unrestricted rigid body has 6 DOF. So, to find a useful expression for $\mathcal{K}$ one needs:

- To choose 6 generalized coordinates $q_1, \ldots, q_6$,
- To compute the coordinates of the vectors specifying the velocity of the center of mass $\boldsymbol{v} = \dot{\boldsymbol{r}}$ and of the angular velocity $\boldsymbol{\omega}$ as functions of $q_1, \ldots, q_6$ and their derivatives,
- To compute from CAD-drawings or to identify otherwise the mass m and the 6 independent components of $\mathcal{I}$.

One should compute vector $\boldsymbol{r}$ and find its time derivative. To compute the angular velocity, see Chapter 3, one needs to find the matrix of rotation $R(t)$, see Chapter 2, compute the matrix of spin

$$S(t) = \tfrac{d}{dt}R(t)\dot{R}^T(t)$$

and find the components of $\boldsymbol{\omega}$ from the components of $S(t) = \boldsymbol{\omega}(t) \times I$.

The time varying components of the matrix of inertia $\mathcal{I}(t)$ in the inertial frame can be computed from components of the constant matrix of inertia in the body frame $\mathcal{I}_0$:

$$\mathcal{I}(t) = \int_{p\in\text{body}} S^T(\vec{r}_p - \vec{r})\, S(\vec{r}_p - \vec{r})\, dm_p = R(t)\, \mathcal{I}_0\, R^T(t)$$

due to the change of coordinates formula for representations of the tensor of inertia, i.e. $A^1 = R_1^0\, A^0\, (R_1^0)^T$, see Chapter 2.

The 3×3 matrix of inertia

$$\mathcal{I}_0 = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}, \qquad \mathcal{I}_0 = \mathcal{I}_0^T$$

is constant in the body frame and since

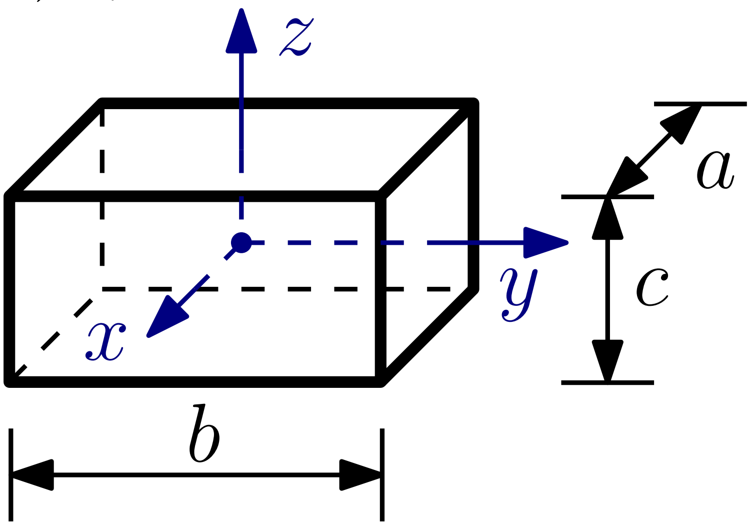
$$\mathcal{I}_0 = \int_{p\in\text{body}} \underbrace{\begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}}_{S^T([x,y,z]^T)} \underbrace{\begin{bmatrix} 0 & z & -y \\ -z & 0 & x \\ y & -x & 0 \end{bmatrix}}_{S([x,y,z]^T)} dm_p$$

its components can be computed as follows

$$\begin{aligned}
I_{xx} &= \int\!\!\int\!\!\int (y^2+z^2)\rho(x,y,z)\,dxdydz\\
I_{yy} &= \int\!\!\int\!\!\int (x^2+z^2)\rho(x,y,z)\,dxdydz\\
I_{zz} &= \int\!\!\int\!\!\int (y^2+x^2)\rho(x,y,z)\,dxdydz\\
I_{xy}=I_{yx} &= -\int\!\!\int\!\!\int xy\rho(x,y,z)\,dxdydz\\
I_{xz}=I_{zx} &= -\int\!\!\int\!\!\int xz\rho(x,y,z)\,dxdydz\\
I_{yz}=I_{zy} &= -\int\!\!\int\!\!\int yz\rho(x,y,z)\,dxdydz
\end{aligned}$$

where $\rho(x,y,z)$ represents the mass density function, while the integration is done over the whole volume occupied by the body assuming that the origin coincides with the center of mass. Note that $m = \int\!\!\int\!\!\int \rho(x,y,z)\,dxdydz$.

Let us consider a simple example of a rectangular brick with uniform mass density $\rho(x,y,z) = \rho =$ const and dimensions $a \times b \times c$ shown below



One can verify that the center of mass coincides with the origin put in the geometric center by checking that

$$x_c = \frac{1}{m}\int_{-c/2}^{c/2}\int_{-b/2}^{b/2}\int_{-a/2}^{a/2} \rho\, x\, dxdydz = 0, \qquad y_c = \frac{1}{m}\int_{-c/2}^{c/2}\int_{-b/2}^{b/2}\int_{-a/2}^{a/2} \rho\, y\, dxdydz = 0,$$

$$z_c = \frac{1}{m}\int_{-c/2}^{c/2}\int_{-b/2}^{b/2}\int_{-a/2}^{a/2} \rho\, z\, dxdydz = 0$$

Clearly,

$$m = \int_{-c/2}^{c/2}\int_{-b/2}^{b/2}\int_{-a/2}^{a/2} \rho\, dxdydz = \rho abc,$$

$$\begin{aligned} I_{xx} &= \int_{-c/2}^{c/2}\int_{-b/2}^{b/2}\int_{-a/2}^{a/2} (y^2+z^2)\rho\, dxdydz \\ &= \rho\frac{abc}{12}(b^2+c^2) = \frac{m}{12}(b^2+c^2) \end{aligned}$$

and in a similar fashion

$$I_{yy} = \frac{m}{12}(a^2+c^2), \quad I_{zz} = \frac{m}{12}(a^2+b^2), \quad I_{xy} = I_{xz} = I_{yz} = 0.$$

Let us move now to a more useful case for us and consider an (open) kinematic chain of rigid bodies.

5.5.2 Kinetic Energy for an n-Link Robot

We will use the formula

$$\mathcal{K} = \frac{1}{2}\sum_{i=1}^{n}\left(m_i\, v_i^T v_i + \omega_i^T\, \mathcal{I}_i\, \omega_i\right).$$

where $\mathcal{I}_i = R(t)\, \mathcal{I}_{0i}\, R^T(t)$ and m_i are matrices of inertia and masses of the links.

So, we need to express

- $v_i = \dot{r}_i$ as a function of generalized coordinates q and velocities $\dot{q}$ and
- ω_i as function of generalized coordinates q and velocities $\dot{q}$.

for $i = 1, \ldots, n$.

The necessary relations are given by the Jacobian matrices

$$v_i = J_{v_i}(q)\,\dot{q}, \qquad \omega_i = J_{\omega_i}(q)\,\dot{q}$$

that can be computed as shown in Chapter 3.

Therefore, the final form of kinetic energy is

$$\mathcal{K} = \frac{1}{2}\dot{q}^T\left[\sum_{i=1}^{n}\left(m_i J_{v_i}^T(q) J_{v_i}(q) + J_{\omega_i}^T(q) R_i^0(q)\mathcal{I}_{0i}\left(R_i^0(q)\right)^T J_{\omega_i}(q)\right)\right]\dot{q}$$

5.5.3 Potential Energy for an n-Link Robot

The potential energy is a sum of potential energies of all the links.

$$\mathcal{P} = \sum_{i=1}^{n} \mathcal{P}_i$$

Potential energy of i^{th}-link is

$$\mathcal{P}_i = -m_i g^T r_{ci}^0$$

where r_{ci}^0 represents the position of the link's center of mass and $g \equiv g^0$ denotes the direction of the gravity in 0-frame multiplied by the gravity constant, which is around 9.8 meters per second.

So, the total potential energy of the robot

$$\mathcal{P} = \sum_{i=1}^{n} \mathcal{P}_i = -\sum_{i=1}^{n} m_i \, g^T \, r_{ci}^0(q)$$

where g represents the vector acceleration of gravity in the 0frame and $r_{ci}^0(q)$ can be computed in the similar way as the solution of the forward kinematics problem given in Chapter 2 from the constant locations of the centers of mass r_{ci}^i of the links with respect to the attached frame: $r_{ci}^0(q) = o_i^0 + R_i^0(q)\, r_{ci}^i$.

5.6 Equations for Dynamics an n-Link Robot

We are ready to discuss the equations of motion written in terms of the generalized coordinates that can be taken as the variables of the DH-convention introduced in Chapter 2.

5.6.1 General form of equations of motion

As has been discussed above, under certain assumptions, the equations of motions of a mechanical system (of particles or rigid bodies) with purely holonomic constraints and external forces or/and torques can be derived as follows.

Procedure 1 **(Computational Procedure)** *One can proceed according to the following steps.*

1. *Identify the smallest required number of independent variables, known as the number of degrees of freedom (DOF)* n*;*

2. *Introduce a set of such variables, called generalized coordinates* $(q_1, \ldots, q_n)$*;*

3. *Compute kinetic energy* $\mathcal{K}(q, \dot q)$ *and potential energy* $\mathcal{P}(q)$ *as function of these variables and their derivatives, known as generalized velocities;*

4. *Compute the Lagrangian* $\mathcal{L} = \mathcal{K} - \mathcal{P}$ *and derive the system of* n *second order differential equations, known as Euler-Lagrange equations:*

$$\frac{d}{dt}\left[\frac{\mathcal{L}(q,\dot q)}{\partial \dot q_j}\right] - \frac{\mathcal{L}(q,\dot q)}{\partial q_j} = \tau_j \ , \quad j = 1, \ldots, n$$

where generalized forces τ_j *are found using the principle of virtual work*

$$\begin{aligned}\tau_j &= \sum_{i=1}^{k_f} \left(f_i^e\right)^T \frac{\partial v_i^e}{\partial \dot q_j} + \sum_{i=1}^{k_m} \left(\tau_i^e\right)^T \frac{\partial \omega_i^e}{\partial \dot q_j} \\ &= \sum_{i=1}^{k_f} \left(J^j_{v_i^e}(q)\right)^T f_i^e + \sum_{i=1}^{k_m} \left(J^j_{\omega_i^e}(q)\right)^T \tau_i^e\end{aligned}$$

from external forces $\left\{\vec f_i^e\right\}_{i=1}^{k_f}$ *that produce work along directions defined by* $\{\vec v_i^e\}_{i=1}^{k_f}$ *or/and torques* $\left\{\vec\tau_i^e\right\}_{i=1}^{k_m}$ *that produce work rotating rigid bodies along directions defined by angular velocities* $\{\vec\omega_i^e\}_{i=1}^{k_m}$*, which are not accounted in the potential energy.*

In the case when, except for the control torques u and gravity, the only other external force $\vec f^e$ is applied to the tool at the origin of the last frame and the only external torque $\vec\tau^e$ is applied to the tool, the matrices in the expressions for the generalized forces become related to the manipulator Jacobian, introduced in Chapter 3 resulting in

$$\tau = -\left(J_v(q)\right)^T f^e - \left(J_\omega(q)\right)^T \tau^e + \frac{\partial q^u}{\partial q} u = -\left(J(q)\right)^T \begin{bmatrix} f^e \\ \tau^e \end{bmatrix} + B(q)\, u$$

where q^u define the variables along which the control forces work and correspondingly $B(q)$ is the actuation-dependent matrix of partial derivatives, which is the identity matrix in the most common case when the control torques work directly on the introduced generalized coordinates: i.e. when $q^u = q$. Note that "$-$" appears above due to "the change of the view" from forces and torques produced by a manipulator at a possible contact to those applied to the tool from outside.

The procedure described above should be combined with computation of the energies described above as well as with appropriate expressions for linear and angular velocities derived in Chapter 3 using DH-convention from Chapter 2.

Let us summarize what we have shown above in relation to computations of energies.

- In general, the kinetic energy is

$$\mathcal{K} \;=\; \frac{1}{2}\dot{q}^T D(q)\dot{q} = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}(q)\dot{q}_i\dot{q}_j$$

  where

$$D(q) = \sum_{i=1}^{n}\left[m_i J_{v_i}^T(q) J_{v_i}(q) + J_{\omega_i}^T(q) R_i^0(q)\mathcal{I}_{0i}\big(R_i^0(q)\big)^T J_{\omega_i}(q)\right]$$

- The potential forces can be separated from the other generalized forces: $\psi_j = -\dfrac{\partial \mathcal{P}(q)}{\partial q_j} + \tau_j$ and the potential energy is

$$\mathcal{P}(q) = -\sum_{i=1}^{n} m_i\, g^T\, r_{ci}^0(q),$$

  where the centers of masses $r_{ci}^0(q)$ and the vector of gravitational acceleration g are expressed in the inertial frame.

- We can introduce a scalar function, called Lagrangian, $\mathcal{L} = \mathcal{K}(q,\dot{q}) - \mathcal{P}(q)$ and write the equation of motion in the compact form

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}_j} - \frac{\partial \mathcal{L}}{\partial q_j} = \tau_j, \qquad j = 1,\ldots,n$$

5.6.2 Structure of the Equations of Motion

The equations of motion have a particular structure implied by

$$\frac{d}{dt}\frac{\partial \mathcal{K}}{\partial \dot{q}_k} - \frac{\partial(\mathcal{K}-\mathcal{P})}{\partial q_k} = \tau_k, \quad k = 1,\ldots,n$$

and the form of the Lagrangian.

Indeed, the first term can be expressed as

$$\frac{\partial \mathcal{K}}{\partial \dot{q}_k} = \frac{\partial}{\partial \dot{q}_k}\left[\frac{1}{2}\dot{q}^T D(q)\dot{q}\right] = \sum_{j=1}^{n} d_{kj}\dot{q}_j \;\Rightarrow\; \frac{d}{dt}\frac{\partial \mathcal{K}}{\partial \dot{q}_k} = \frac{d}{dt}\left[\sum_{j=1}^{n} d_{kj}(q)\dot{q}_j\right]$$

and rewritten as follows

$$\begin{aligned}
\frac{d}{dt}\left[\sum_{j=1}^{n} d_{kj}\dot{q}_j\right] &= \sum_{j=1}^{n} d_{kj}(q)\ddot{q}_j + \sum_{j=1}^{n} \tfrac{d}{dt}\left[d_{kj}(q)\right]\dot{q}_j \\
&= \sum_{j=1}^{n} d_{kj}(q)\ddot{q}_j + \sum_{j=1}^{n}\left(\sum_{i=1}^{n}\frac{\partial d_{kj}(q)}{\partial q_i}\dot{q}_i\right)\dot{q}_j \\
&= \sum_{j=1}^{n} d_{kj}(q)\ddot{q}_j + \sum_{j=1}^{n}\sum_{i=1}^{n}\left(\frac{\partial d_{kj}(q)}{\partial q_i}\right)\dot{q}_i\dot{q}_j \\
&= \sum_{j=1}^{n} d_{kj}(q)\ddot{q}_j + \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\left(\frac{\partial d_{kj}(q)}{\partial q_i} + \frac{\partial d_{ki}(q)}{\partial q_j}\right)\dot{q}_i\dot{q}_j
\end{aligned}$$

The second term of the equations of motion is equal to

$$\begin{aligned}
\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial q_k} &= \frac{\partial}{\partial q_k}\left[\frac{1}{2}\dot{q}^T D(q)\dot{q} - \mathcal{P}\right] = \frac{1}{2}\dot{q}^T\left[\frac{\partial}{\partial q_k}D(q)\right]\dot{q} - \frac{\partial}{\partial q_k}\mathcal{P} \\
&= \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\frac{\partial d_{ij}(q)}{\partial q_k}\dot{q}_i\dot{q}_j - \frac{\partial}{\partial q_j}\mathcal{P}(q)
\end{aligned}$$

To sum up, the equations of motion can be rewritten as follows

$$\sum_{j=1}^{n} d_{kj}\ddot{q}_j + \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\left(\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j}\right)\dot{q}_i\dot{q}_j - \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\frac{\partial d_{ij}}{\partial q_k}\dot{q}_i\dot{q}_j + \frac{\partial\mathcal{P}}{\partial q_k} = \tau_k$$

and, combining the terms in the middle, as

$$\boxed{\sum_{j=1}^{n} d_{kj}(q)\ddot{q}_j + \sum_{j=1}^{n}\sum_{i=1}^{n} c_{ijk}(q)\dot{q}_i\dot{q}_j + g_k(q) = \tau_k}$$

with

$$c_{ijk}(q) = \frac{1}{2}\left(\frac{\partial d_{kj}(q)}{\partial q_i} + \frac{\partial d_{ki}(q)}{\partial q_j} - \frac{\partial d_{ij}(q)}{\partial q_j}\right), \qquad g_k(q) = \frac{\partial}{\partial q_k}\mathcal{P}(q)$$

Finally, introducing the so-called matrix of centrifugal and Coriolis forces $C(q,\dot{q})$ defined by its components

$$C_{kj}(q,\dot{q}) = \sum_{i=1}^{n} c_{ijk}(q)\dot{q}_i = \frac{1}{2}\sum_{i=1}^{n}\left(\frac{\partial d_{kj}(q)}{\partial q_i} + \frac{\partial d_{ki}(q)}{\partial q_j} - \frac{\partial d_{ij}(q)}{\partial q_j}\right)\dot{q}_i$$

one can obtain the equations in the following compact matrix form

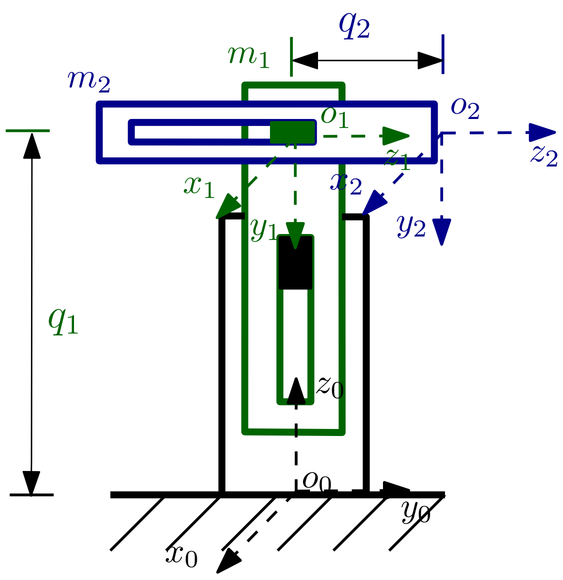
$$\boldsymbol{D}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}$$

Let us see examples of deriving the equations of motions for particular examples of robotic manipulators.

5.7 Examples for Equations of Motion

5.7.1 Two-Link Cartesian Manipulator

Consider the following **2**-link robot of ***PP***-type



For this system we need

- to solve forward kinematics problem;
- to compute manipulator Jacobian;
- to compute kinetic and potential energies and the Euler-Lagrange equations.

Forward Kinematics and Jacobian

DH parameters for computing homogeneous transformations

$$\boldsymbol{T}(\boldsymbol{q_i}) = \mathrm{Rot}_{\boldsymbol{z},\boldsymbol{\theta}} \cdot \mathrm{Trans}_{\boldsymbol{z},\boldsymbol{d}} \cdot \mathrm{Trans}_{\boldsymbol{x},\boldsymbol{a}} \cdot \mathrm{Rot}_{\boldsymbol{x},\boldsymbol{\alpha}}$$

are

$$\begin{aligned} T_1^0: &\quad \theta = 0, \quad d = q_1, \quad a = 0, \quad \alpha = -\frac{\pi}{2} \\ T_2^1: &\quad \theta = 0, \quad d = q_2, \quad a = 0, \quad \alpha = 0 \end{aligned}$$

The kinetic energy of the system is

$$\mathcal{K} = \tfrac{1}{2}\left[m_1 v_{c1}^T v_{c1} + \omega_1^T \mathcal{I}_1 \omega_1\right] + \tfrac{1}{2}\left[m_2 v_{c2}^T v_{c2} + \omega_2^T \mathcal{I}_2 \omega_2\right]$$

where the velocities of the centers of masses coincide with velocities of all the other points of the respective links

$$\begin{aligned} v_{c1} &= J_{v1}\,\dot q = \left[J_{v1}^{(1)}\ J_{v1}^{(2)}\right]\begin{bmatrix}\dot q_1\\ \dot q_2\end{bmatrix} = J_{v1}^{(1)}\dot q_1 + J_{v1}^{(2)}\dot q_2 \\ v_{c2} &= J_{v2}\,\dot q = \left[J_{v2}^{(1)}\ J_{v2}^{(2)}\right]\begin{bmatrix}\dot q_1\\ \dot q_2\end{bmatrix} = J_{v2}^{(1)}\dot q_1 + J_{v2}^{(2)}\dot q_2 \end{aligned}$$

and the angular velocities of the links

$$\begin{aligned} \omega_1 &= J_{\omega 1}\,\dot q = \left[J_{\omega_1}^{(1)}\ J_{\omega_1}^{(2)}\right]\begin{bmatrix}\dot q_1\\ \dot q_2\end{bmatrix} = J_{\omega_1}^{(1)}\dot q_1 + J_{\omega_1}^{(2)}\dot q_2 \\ \omega_2 &= J_{\omega 2}\,\dot q = \left[J_{\omega_2}^{(1)}\ J_{\omega_2}^{(2)}\right]\begin{bmatrix}\dot q_1\\ \dot q_2\end{bmatrix} = J_{\omega_2}^{(1)}\dot q_1 + J_{\omega_2}^{(2)}\dot q_2 \end{aligned}$$

can be computed using the velocity Jacobian.

To compute the Jacobian we can use the expressions derived above for the case of DH-frames, i.e, non-zero vectors are obtained for $i \le k$ according to the following expressions

$$\begin{aligned} J_{vk}^{(i)} &= \begin{cases} z_{i-1}^0, & \text{for } \underline{\text{prismatic}} \text{ joint} \\ z_{i-1}^0 \times \left[o_{ck}^0 - o_{i-1}^0\right], & \text{for revolute joint} \end{cases} \\ J_{\omega k}^{(i)} &= \begin{cases} 0, & \text{for } \underline{\text{prismatic}} \text{ joint} \\ z_{i-1}^0, & \text{for revolute joint} \end{cases} \end{aligned}$$

So, we have

$$J_{v1} = \left[z_0^0 \ \ 0_{3\times 1}\right] = \left[\begin{bmatrix}0\\0\\1\end{bmatrix} \ \ \begin{bmatrix}0\\0\\0\end{bmatrix}\right], \quad J_{v2} = \left[z_0^0 \ \ z_1^0\right] = \left[\begin{bmatrix}0\\0\\1\end{bmatrix} \ \ \begin{bmatrix}0\\1\\0\end{bmatrix}\right]$$

$$J_{\omega 1} = 0_{3\times 1}, \qquad J_{\omega 2} = 0_{3\times 1}$$

To sum up:

- Angular velocities ω_1 and ω_2 of both links are **zeros**.

- Linear velocities of centers of masses are

$$\begin{aligned} v_{c1} &= \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \dot{q}_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \dot{q}_2 = \begin{bmatrix} 0 \\ 0 \\ \dot{q}_1 \end{bmatrix} \\ v_{c2} &= \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \dot{q}_1 + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \dot{q}_2 = \begin{bmatrix} 0 \\ \dot{q}_2 \\ \dot{q}_1 \end{bmatrix} \end{aligned}$$

- The kinetic energy is

$$\mathcal{K} = \tfrac{1}{2} m_1 v_{c1}^T v_{c1} + \tfrac{1}{2} m_2 v_{c2}^T v_{c2} = \tfrac{1}{2} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}^T \begin{bmatrix} m_1 + m_2 & 0 \\ 0 & m_2 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

Potential Energy (PE) for Two-Link Cartesian Manipulator

Here are immediate observations:

- PE is independent of the second link position,
- It depends on the height of center of mass of the whole robot.

It follows that, up to an additive constant off-set, $\mathcal{P} = g\ (m_1 + m_2)\ q_1$.

Euler-Lagrange Equations for 2-Link Cartesian Manipulator

Given the kinetic $\mathcal{K}$ and potential $\mathcal{P}$ energies, the dynamics are

$$\frac{d}{dt}\left[\frac{\partial \mathcal{L}}{\partial \dot{q}}\right] - \frac{\partial \mathcal{L}}{\partial q} = \tau$$

where the difference between the kinetic and potential energies is

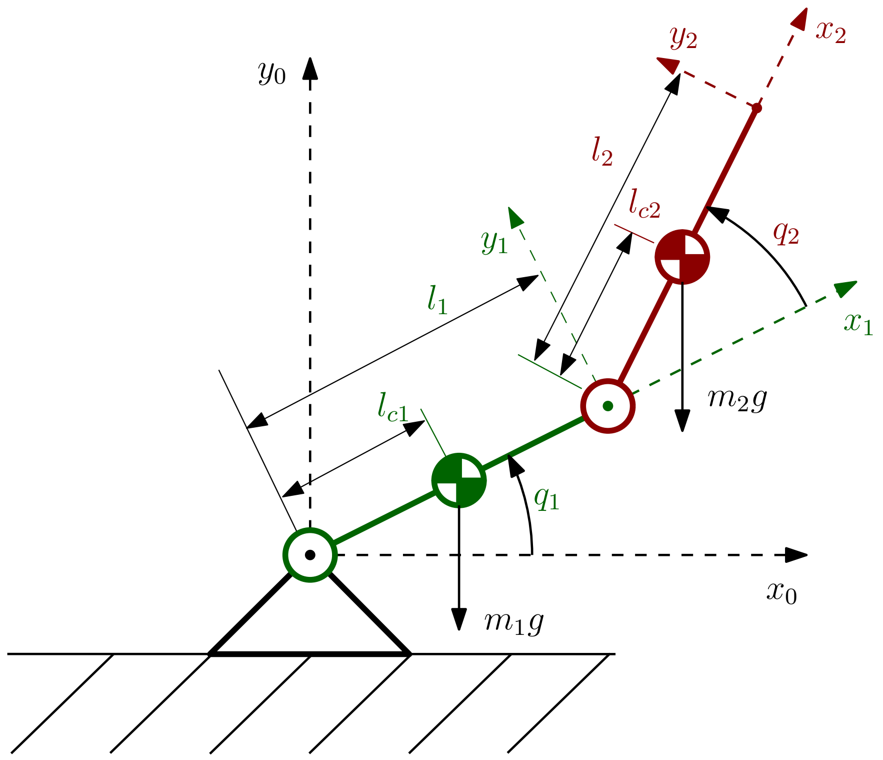
$$\mathcal{L} = \tfrac{1}{2} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}^T \begin{bmatrix} m_1 + m_2 & 0 \\ 0 & m_2 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} - g\,(m_1 + m_2)\, q_1$$

Differentiating we obtain the Euler-Lagrange equations

$$\begin{aligned} (m_1 + m_2)\ddot{q}_1 + g(m_1 + m_2) &= \tau_1 \\ m_2 \ddot{q}_2 &= \tau_2 \end{aligned}$$

5.7.2 Planar Elbow Manipulator

Consider the planar **2**-link manipulator of ***RR*** type (two-link planar articulated robot) shown below together with an assignment of DH frames.



This manipulator has been already studied in Chapter 1, where we have stated the equations of motion without derivations.

For this system we would like

- to compute forward kinematics and manipulator Jacobian,
- to compute the energies and the Euler-Lagrange equations.

Forward Kinematics and Jacobian

DH parameters for computing homogeneous transformations

$$\boldsymbol{T}(\boldsymbol{q_i}) = \mathrm{Rot}_{\boldsymbol{z},\boldsymbol{\theta}} \cdot \mathrm{Trans}_{\boldsymbol{z},\boldsymbol{d}} \cdot \mathrm{Trans}_{\boldsymbol{x},\boldsymbol{a}} \cdot \mathrm{Rot}_{\boldsymbol{x},\boldsymbol{\alpha}}$$

are

$$\begin{array}{lllll} \boldsymbol{T}_1^0 : & \boldsymbol{\theta} = \boldsymbol{q}_1, & \boldsymbol{d} = 0, & \boldsymbol{a} = \boldsymbol{l}_1, & \boldsymbol{\alpha} = 0 \\ \boldsymbol{T}_2^1 : & \boldsymbol{\theta} = \boldsymbol{q}_2, & \boldsymbol{d} = 0, & \boldsymbol{a} = \boldsymbol{l}_2, & \boldsymbol{\alpha} = 0 \end{array}$$

The kinetic energy of the system is

$$\boldsymbol{\mathcal{K}} = \tfrac{1}{2}\left[\boldsymbol{m}_1 \boldsymbol{v}_{c1}^T \boldsymbol{v}_{c1} + \boldsymbol{\omega}_1^T \boldsymbol{\mathcal{I}}_1 \boldsymbol{\omega}_1\right] + \tfrac{1}{2}\left[\boldsymbol{m}_2 \boldsymbol{v}_{c2}^T \boldsymbol{v}_{c2} + \boldsymbol{\omega}_2^T \boldsymbol{\mathcal{I}}_2 \boldsymbol{\omega}_2\right]$$

where the velocities of the centers of mass can be computed in a similar way as the velocities of the end-points of the links

$$\begin{aligned} v_{c1} &= J_{v1}\,\dot q = \left[J_{v1}^{(1)}\; J_{v1}^{(2)}\right]\begin{bmatrix}\dot q_1\\ \dot q_2\end{bmatrix} = J_{v1}^{(1)}\dot q_1 + J_{v1}^{(2)}\dot q_2\\ v_{c2} &= J_{v2}\,\dot q = \left[J_{v2}^{(1)}\; J_{v2}^{(2)}\right]\begin{bmatrix}\dot q_1\\ \dot q_2\end{bmatrix} = J_{v2}^{(1)}\dot q_1 + J_{v2}^{(2)}\dot q_2 \end{aligned}$$

and the angular velocities of the links

$$\begin{aligned} \omega_1 &= J_{\omega 1}\,\dot q = \left[J_{\omega_1}^{(1)}\; J_{\omega_1}^{(2)}\right]\begin{bmatrix}\dot q_1\\ \dot q_2\end{bmatrix} = J_{\omega_1}^{(1)}\dot q_1 + J_{\omega_1}^{(2)}\dot q_2\\ \omega_2 &= J_{\omega 2}\,\dot q = \left[J_{\omega_2}^{(1)}\; J_{\omega_2}^{(2)}\right]\begin{bmatrix}\dot q_1\\ \dot q_2\end{bmatrix} = J_{\omega_2}^{(1)}\dot q_1 + J_{\omega_2}^{(2)}\dot q_2 \end{aligned}$$

can be computed using the velocity Jacobian.

To compute the Jacobian we can use the expressions derived above for the case of DH-frames, i.e, non-zero vectors are obtained for $i \le k$ according to the following expressions

$$\begin{aligned} J_{vk}^{(i)} &= \begin{cases} z_{i-1}^0, & \text{for prismatic joint}\\ z_{i-1}^0 \times \left[o_{ck}^0 - o_{i-1}^0\right], & \text{for \underline{revolute} joint}\end{cases}\\ J_{\omega k}^{(i)} &= \begin{cases} 0, & \text{for prismatic joint}\\ z_{i-1}^0, & \text{for \underline{revolute} joint}\end{cases} \end{aligned}$$

So, we have for the linear velocity Jacobian

$$\begin{aligned} J_{v1}^{(1)} &= z_0^0 \times (o_{c_1} - o_0) = \begin{bmatrix}0\\0\\1\end{bmatrix} \times \begin{bmatrix} l_{c_1}\cos q_1\\ l_{c_1}\sin q_1\\ 0\end{bmatrix} = \begin{bmatrix} -l_{c_1}\sin q_1\\ l_{c_1}\cos q_1\\ 0\end{bmatrix},\\ J_{v2}^{(1)} &= z_0^0 \times (o_{c2} - o_0)\\ &= \begin{bmatrix}0\\0\\1\end{bmatrix} \times \left(\begin{bmatrix} l_1\cos q_1\\ l_1\sin q_1\\ 0\end{bmatrix} + \begin{bmatrix} l_{c_2}\cos(q_1+q_2)\\ l_{c_2}\sin(q_1+q_2)\\ 0\end{bmatrix}\right) = \begin{bmatrix} -l_1\sin q_1 - l_{c_2}\sin(q_1+q_2)\\ l_1\cos q_1 + l_{c_2}\cos(q_1+q_2)\\ 0\end{bmatrix}\\ J_{v1}^{(2)} &= \begin{bmatrix}0\\0\\0\end{bmatrix},\\ J_{v2}^{(2)} &= z_1^0 \times (o_{c2} - o_1) = \begin{bmatrix}0\\0\\1\end{bmatrix} \times \begin{bmatrix} l_{c_2}\cos(q_1+q_2)\\ l_{c_2}\sin(q_1+q_2)\\ 0\end{bmatrix} = \begin{bmatrix} -l_{c_2}\sin(q_1+q_2)\\ l_{c_2}\cos(q_1+q_2)\\ 0\end{bmatrix} \end{aligned}$$

For the angular velocity Jacobian we have

$$J_{\omega_1}^{(1)} = z_0^0 = \begin{bmatrix}0\\0\\1\end{bmatrix},\quad J_{\omega_2}^{(1)} = z_0^0 = \begin{bmatrix}0\\0\\1\end{bmatrix},\quad J_{\omega_1}^{(2)} = \begin{bmatrix}0\\0\\0\end{bmatrix},\quad J_{\omega_2}^{(2)} = z_1^0 = \begin{bmatrix}0\\0\\1\end{bmatrix}$$

Finally, the kinetic energy $\mathcal{K}$ is

$$\begin{aligned}
\mathcal{K} &= \tfrac{1}{2}\left[m_1 v_{c1}^T v_{c1} + \omega_1^T \mathcal{I}_1 \omega_1\right] + \tfrac{1}{2}\left[m_2 v_{c2}^T v_{c2} + \omega_2^T \mathcal{I}_2 \omega_2\right] \\
&= \tfrac{1}{2}\left[m_1 \left(J_{v_1}^{(1)} \dot q_1\right)^2 + I_1 \left(J_{\omega_1}^{(1)} \dot q_1\right)^2\right] \\
&\quad + \tfrac{1}{2}\left[m_2 \left(J_{v_2}^{(1)} \dot q_1 + J_{v_2}^{(2)} \dot q_2\right)^2 + I_2 \left(J_{\omega_2}^{(1)} \dot q_1 + J_{\omega_2}^{(2)} \dot q_2\right)^2\right] \\
&= \tfrac{1}{2}\begin{bmatrix} \dot q_1 \\ \dot q_2 \end{bmatrix}^T \begin{bmatrix} d_{11} & d_{12} \\ d_{12} & d_{22} \end{bmatrix} \begin{bmatrix} \dot q_1 \\ \dot q_2 \end{bmatrix}
\end{aligned}$$

where

$$\begin{aligned}
d_{11} &= m_1 l_{c_1}^2 + m_2\left(l_1^2 + l_{c_2}^2 + 2 l_1 l_{c_2} \cos q_2\right) + I_1 + I_2 \\
d_{12} &= m_2\left(l_{c_2}^2 + l_1 l_{c_2} \cos q_2\right) + I_2 \\
d_{22} &= m_2 l_{c_2}^2 + I_2
\end{aligned}$$

Potential Energy (PE) for Two-Link Elbow Manipulator

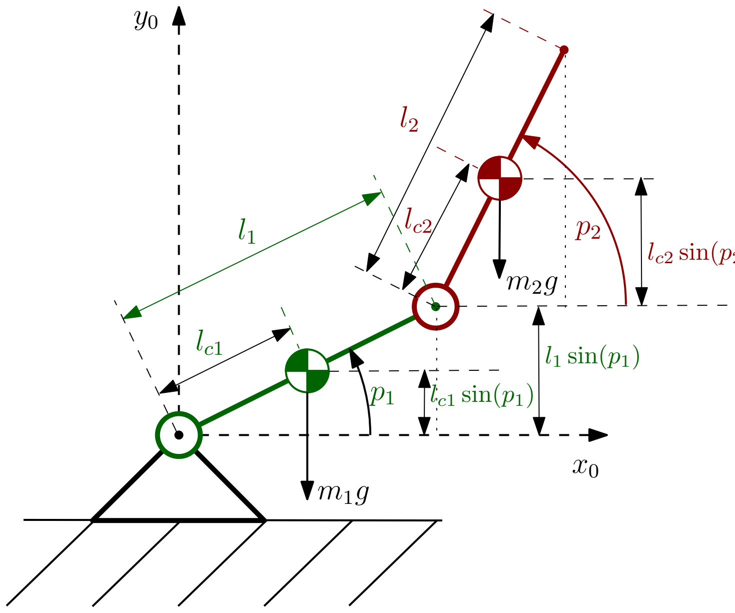
We proceed as follows

- PE of the 1st link is $\mathcal{P}_1 = m_1 g y_{c_1} = m_1 g l_{c_1} \sin q_1$;
- PE of the 2nd link is
  $\mathcal{P}_2 = m_1 g y_{c_2} = m_2 g \left(l_1 \sin q_1 + l_{c_2} \sin(q_1 + q_2)\right)$;
- So, total PE is $\mathcal{P}_1 + \mathcal{P}_2$.

The result of applying the Euler-Lagrange formulae to $\mathcal{L} = \mathcal{K} - \mathcal{P}$ has been already presented in Chapter 1 above.

5.7.3 Double Link Pendulum with Absolute Angles

Let us now show how to use Euler-Lagrange equations without the machinery of using DH parameters. Consider the same planar RR-type manipulator but suppose we would like to use the absolute angles p_1 and p_2, see the figure below, as the generalized coordinates.



If we would like to deriving the equations of motion directly in these coordinates, we should proceed as follows

- Compute the kinetic energy $\mathcal{K}(p_1, p_2, \dot{p}_1, \dot{p}_2)$;
- Compute the potential energy $\mathcal{P}(p_1, p_2)$;
- Use them to compute the Lagrangian and to obtain the Euler-Lagrange equations.

Note that a possible reason for such a choice of the generalized coordinates would be the fact that the generalized forces work exactly on δp_1 and δp_2 respectively. This would be the case if both motors driving the links are fixed to the base so that, in particular, the second link is actuated through a power-train such as a belt. In this case, in the right-hand sides of the Euler-Lagrange equations we should have the torques τ_1 and τ_2 produced by the motors (possibly magnified if gearboxes are present).

Note that an alternative way would be to use the DH-parameters q_1 and q_2 but to recompute the generalized forces for the Euler-Lagrange equations, written in terms of q_1 and q_2 using the principle of virtual work

$$\tau_1^{(p)} = \tau_1^{(q)} \frac{\partial \omega_1}{\partial \dot{q}_1} + \tau_2^{(q)} \frac{\partial \omega_1}{\partial \dot{q}_2}, \qquad \tau_2^{(p)} = \tau_1^{(q)} \frac{\partial \omega_2}{\partial \dot{q}_1} + \tau_2^{(q)} \frac{\partial \omega_2}{\partial \dot{q}_2}$$

where, as computed above, $\omega_1 = \dot{q}_1$ and $\omega_2 = \dot{q}_1 + \dot{q}_2$ so that the right-hand sides of the equations, written in terms of q_1 and q_2 should be $\tau_1^{(q)} = \tau_1^{(p)}$ and $\tau_2^{(q)} = \tau_2^{(p)} - \tau_1^{(p)}$.

Let us, however, avoid using these arguments and write the equations in terms of p_1 and p_2 directly with, correspondingly, the right-hand sides taken directly as $\tau_1 = \tau_1^{(p)}$ and $\tau_2 = \tau_2^{(p)}$.

Let us start with kinetic energy

$$\mathcal{K} = \tfrac{1}{2}\left[m_1 v_{c_1}^T v_{c1} + \omega_1^T \mathcal{I}_1 \omega_1\right] + \tfrac{1}{2}\left[m_2 v_{c_2}^T v_{c2} + \omega_2^T \mathcal{I}_2 \omega_2\right]$$

In order to compute v_{c1} and v_{c2}, we use simple geometry to get the coordinates of the center of mass of the first link.

The coordinates of the center of mass of the first link are

$$x_{c_1} = l_{c_1}\cos(p_1), \qquad y_{c_1} = l_{c_1}\sin(p_1)$$

Therefore,

$$\dot{x}_{c_1} = -l_{c_1}\sin(p_1)\,\dot{p}_1, \qquad \dot{y}_{c_1} = l_{c_1}\cos(p_1)\,\dot{p}_1$$

while it is obvious that the angular velocity of the first link is

$$\omega_1 = \dot{p}_1\, z_0^0,$$

since we have the simple case of rotation in plane.

The x-coordinate of the center of mass of the second link is

$$x_{c_2} = l_1\cos(p_1) + l_{c_2}\cos(p_2)$$

and its derivative is

$$\dot{x}_{c_2} = -l_1\sin(p_1)\,\dot{p}_1 - l_{c_2}\sin(p_2)\,\dot{p}_2$$

The y-coordinate of the center of mass of the second link is

$$y_{c_2} = l_1\sin(p_1) + l_{c_2}\sin(p_2)$$

and its derivative is

$$\dot{y}_{c_2} = l_1\cos(p_1)\,\dot{p}_1 + l_{c_2}\cos(p_2)\,\dot{p}_2$$

while it is again obvious that the angular velocity of the second link is

$$\omega_2 = \dot{p}_2\, z_1^0.$$

It follows that

$$\begin{aligned}
\mathcal{K} &= \tfrac{1}{2}\left[m_1 v_{c_1}^T\, v_{c1} + \omega_1^T \mathcal{I}_1 \omega_1\right] + \tfrac{1}{2}\left[m_2 v_{c_2}^T\, v_{c2} + \omega_2^T \mathcal{I}_2 \omega_2\right] \\
&= \tfrac{1}{2}\begin{bmatrix}\dot p_1\\ \dot p_2\end{bmatrix}^T D(p_1,p_2)\begin{bmatrix}\dot p_1\\ \dot p_2\end{bmatrix} \\
&= \tfrac{1}{2}\begin{bmatrix}\dot p_1\\ \dot p_2\end{bmatrix}^T \begin{bmatrix} m_1 l_{c_1}^2 + m_2 l_1^2 + I_1 & m_2 l_1 l_{c_2}\cos(p_2 - p_1) \\ m_2 l_1 l_{c_2}\cos(p_2-p_1) & m_2 l_{c_2}^2 + I_2\end{bmatrix}\begin{bmatrix}\dot p_1\\ \dot p_2\end{bmatrix}
\end{aligned}$$

Now, the potential energy can be easily computed

$$\begin{aligned}
\mathcal{P} &= m_1\, g\, y_{c_1} + m_2\, g\, y_{c_2} \\
&= m_1\, g\, l_{c_1}\, \sin(p_1) + m_2\, g\, \left[l_1\, \sin(p_1) + l_{c_2}\, \sin(p_2)\right]
\end{aligned}$$

Finally, the equations of motion can be computed from

$$\frac{d}{dt}\left[\frac{\partial \mathcal{K}}{\partial \dot p_1}\right] - \frac{\partial(\mathcal{K}-\mathcal{P})}{\partial p_1} = \tau_1\,, \qquad \frac{d}{dt}\left[\frac{\partial \mathcal{K}}{\partial \dot p_2}\right] - \frac{\partial(\mathcal{K}-\mathcal{P})}{\partial p_2} = \tau_2$$

Here and above we have considered the case of so-called open kinematic chains. Following DH-convention routine typically allows saving much time for developing equation for kinematics and dynamics. However, this is not possible if the number of links is smaller than the number of DOF due to presence of a kinematic loop.

5.7.4 Euler-Lagrange equations in excessive coordinates

Sometimes, it is to compute Lagrangian as a function of more variables the the number of degrees of freedom. Suppose we have $k \geq 2$ variables

$$\theta_1, \qquad \theta_2, \qquad \ldots, \qquad \theta_k, \qquad \rightarrow \qquad \theta = [\theta_1 \;\; \theta_2 \;\; \ldots \;\; \theta_k]^T$$

and $1 \leq l < k$ relations, called holonomic constraints, among them

$$g_i(\theta_1, \theta_2, \ldots, \theta_k) = 0, \qquad i = 1, 2, \ldots, l \qquad \rightarrow \qquad g(\theta) = 0_{l\times 1}$$

and these relations are not easy to solve for $n = k - l$ independent variables, called generalized coordinates. It is possible to modify the equations describing dynamics, derived above for a system of particles, to include the additional l constraint forces. First, one computes the energies as functions of all the not independent variables. After that, applying the principle of virtual work, and introducing l constraint forces:

$$f_1^c, \qquad f_2^c, \qquad \ldots, \qquad f_l^c, \qquad \rightarrow \qquad F^c = \left[f_1^c \;\; f_2^c \;\; \ldots \;\; f_l^c\right]^T$$

one obtains $\boldsymbol{k}$ differential equations of second order

$$\frac{d}{dt}\left[\frac{\partial \mathcal{K}}{\partial \dot{\theta}_i}\right] - \frac{\partial(\mathcal{K}-\mathcal{P})}{\partial \theta_i} = \tau_i - \left(J^c(\theta)\right)^T F^c, \qquad i = 1, \ldots, k$$

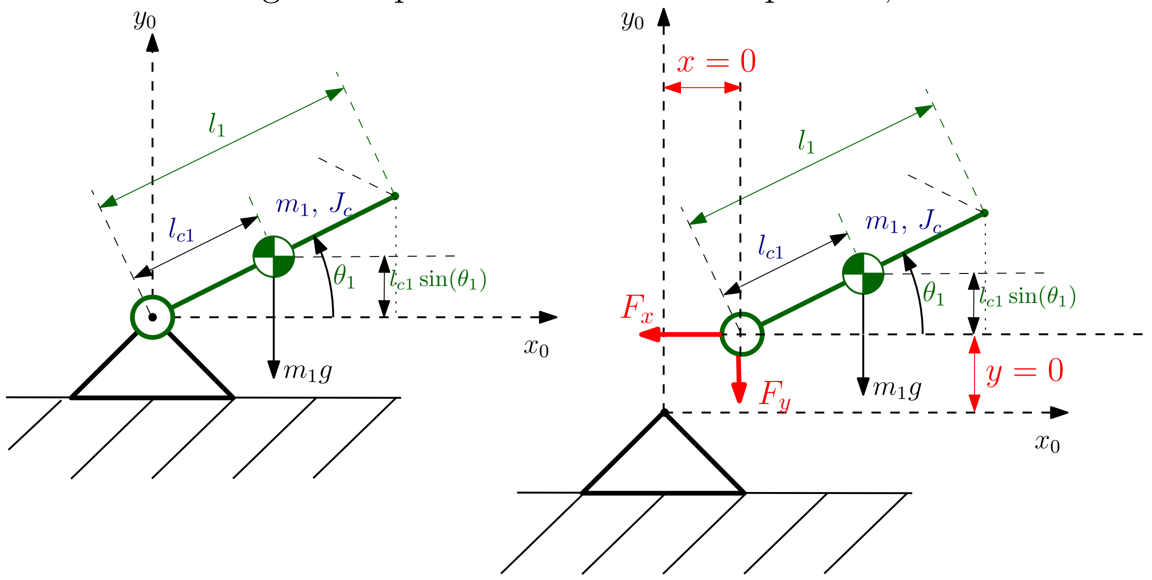
where τ_i denote generalized external forces acting on θ_i and the Jacobian of constraints is obtained by differentiations of the constraint identities:

$$J^c(\theta) = \frac{\partial g(\theta)}{\partial \theta} \qquad \Leftarrow \qquad \frac{d}{dt}\, g(\theta) = J^c(\theta)\,\dot{\theta}$$

The equations above are sometimes called Euler-Lagrange equations of first kind or Euler-Lagrange equations with constraint forces in excessive coordinates.

Often the constraint forces can be eliminated from the equations as in the case illustrated in the next simple example.

Consider a single-link planar rotational manipulator, illustrated below.



The dynamics can be described either using the single generalized coordinate θ_1 or using the three variable: x, y, and θ_1, which are subject to two holonomic constraints:

$$g(\theta) = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Computing coordinates of the center of mass and their derivatives

$$\begin{bmatrix} x_{c1} \\ y_{c1} \end{bmatrix} = \begin{bmatrix} x + l_{c1}\cos\theta_1 \\ y + l_{c1}\sin\theta_1 \end{bmatrix}, \qquad \begin{bmatrix} \dot{x}_{c1} \\ \dot{y}_{c1} \end{bmatrix} = \begin{bmatrix} \dot{x} - l_{c1}\dot{\theta}_1\sin\theta_1 \\ \dot{y} + l_{c1}\dot{\theta}_1\cos\theta_1 \end{bmatrix}$$

one can obtain the expressions for the energies

$$\mathcal{K} = \tfrac{1}{2}\, m_1 \left(\dot{x}_{c1}^2 + \dot{y}_{c1}^2\right) + \tfrac{1}{2}\, J_c\, \dot{\theta}_1^2, \qquad \mathcal{P} = m_1\, g\, l_{c1}\sin\theta_1$$

and of the Euler-Lagrange equations with constraint forces:

$$
\begin{aligned}
\frac{d}{dt}\left[\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial \dot{\theta}_1}\right]-\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial \theta_1} &= \tau_1 = \tau_1 + \frac{\partial\,[x\;\;y]}{\partial \theta}\begin{bmatrix} F_x \\ F_y \end{bmatrix}\\
\frac{d}{dt}\left[\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial \dot{x}}\right]-\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial x} &= F_x = \frac{\partial\,[x\;\;y]}{\partial x}\begin{bmatrix} F_x \\ F_y \end{bmatrix}\\
\frac{d}{dt}\left[\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial \dot{y}}\right]-\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial y} &= F_y = \frac{\partial\,[x\;\;y]}{\partial y}\begin{bmatrix} F_x \\ F_y \end{bmatrix}
\end{aligned}
$$

which can be rewritten as

$$
\begin{aligned}
m_1 l_{c1}^2 \ddot{\theta}_1 + m_1 l_{c1}(\ddot{y}\cos\theta_1 - \ddot{x}\sin\theta_1) + m_1 g l_{c1}\cos\theta_1 &= \tau_1\\
m_1\left(\ddot{x} - l_{c1}\ddot{\theta}_1\sin\theta_1 - l_{c1}\dot{\theta}_1^2\cos\theta_1\right) &= F_x\\
m_1\left(\ddot{y} + l_{c1}\ddot{\theta}_1\cos\theta_1 - l_{c1}\dot{\theta}_1^2\sin\theta_1\right) &= F_y
\end{aligned}
$$

Now, substituting the relations that follow from the constraints $x \equiv 0$ and $y \equiv 0$, i.e. $\ddot{x} = 0$ and $\ddot{y} = 0$, we recover not only the equation that can be derived using the generalized coordinate θ_1 without constraints and additional variables:

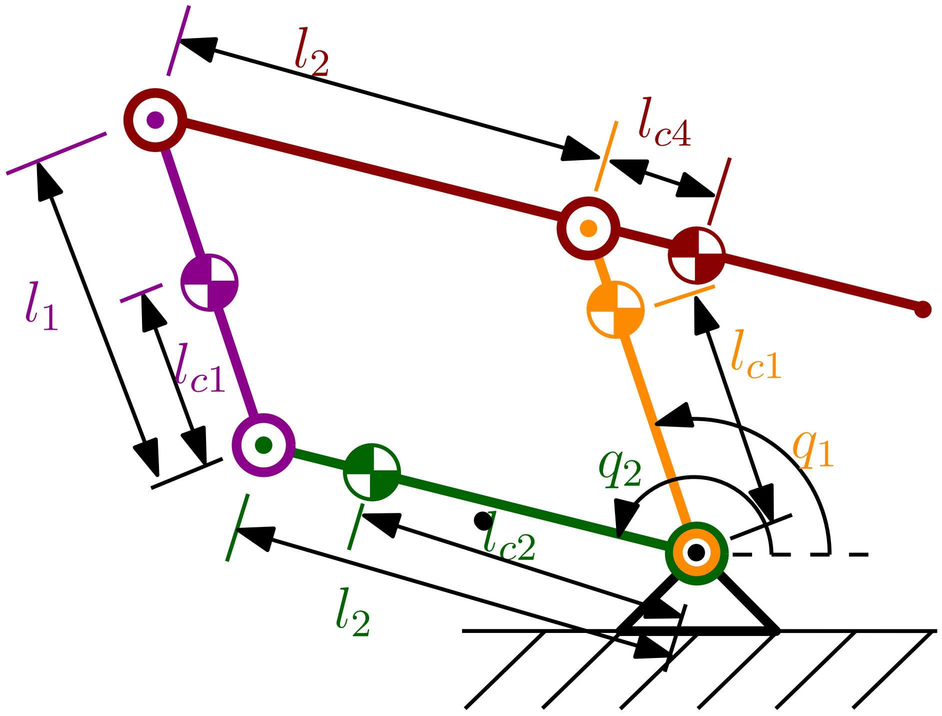
$$
m_1\, l_{c1}^2\, \ddot{\theta}_1 + m_1\, g\, l_{c1}\, \cos\theta_1 = \tau_1
$$

but also the expressions for the reaction (constraint) forces:

$$
\begin{aligned}
F_x &= \left(\frac{\tau_1}{l_{c1}} - m_1 g\cos\theta_1\right)\sin\theta_1 - m_1 l_{c1}\dot{\theta}_1^2\cos\theta_1\\
F_y &= \left(\frac{\tau_1}{l_{c1}} - m_1 g\cos\theta_1\right)\cos\theta_1 - m_1 l_{c1}\dot{\theta}_1^2\sin\theta_1
\end{aligned}
$$

5.7.5 Mechanism of a Closed Kinematic Chain

Let us consider now a more challenging example of a mechanism shown below.



Since we have a closed kinematic chain, using DH-convention is not possible. Hence, we need to introduce generalized coordinates and proceed with all the derivations without the DH-machinery.

We need, besides choosing the coordinates,

- to compute the kinetic energy $\mathcal{K}$;
- to compute the potential energy $\mathcal{P}$;
- to substitute them into the Euler-Lagrange equations.

First of all, it is obvious that two angles completely define location of all the components. Hence, the number of DOF is **2**. A possible choice of the two generalized coordinates is taking the two absolute angles q_1 and q_2 defining orientations of the first two links, see the figure above.

Let us start with the kinetic energy

$$\begin{aligned}\mathcal{K} &= \tfrac{1}{2}\left[m_1 v_{c_1}^T v_{c_1} + \omega_1^T \mathcal{I}_1 \omega_1\right] + \tfrac{1}{2}\left[m_2 v_{c_2}^T v_{c_2} + \omega_2^T \mathcal{I}_2 \omega_2\right] \\ &\qquad + \tfrac{1}{2}\left[m_3 v_{c_3}^T v_{c_3} + \omega_3^T \mathcal{I}_3 \omega_3\right] + \tfrac{1}{2}\left[m_4 v_{c_4}^T v_{c_4} + \omega_4^T \mathcal{I}_4 \omega_4\right]\end{aligned}$$

Begin with the coordinates of the centers of mass of the two links, location of which are directly specified by q_1 and q_2

$$\begin{aligned} x_{c_1} &= l_{c_1}\cos q_1, & y_{c_1} &= l_{c_1}\sin q_1 \\ x_{c_2} &= l_{c_2}\cos q_2, & y_{c_2} &= l_{c_2}\sin q_2 \end{aligned}$$

It can be seen that for the other two links, essentially using the symmetry, we have

$$\begin{aligned} x_{c_3} &= l_2 \cos q_2 + l_{c_3} \cos q_1 \\ y_{c_3} &= l_2 \sin q_2 + l_{c_3} \sin q_1 \end{aligned}$$

and

$$\begin{aligned} x_{c_4} &= l_1 \cos q_1 - l_{c_4} \cos q_2 \\ y_{c_4} &= l_1 \sin q_1 - l_{c_4} \sin q_2 \end{aligned}$$

Now, as above and using the symmetry

$$\begin{aligned} \vec{\omega}_1 = \vec{\omega}_3 = \dot{q}_1 \vec{k} \\ \vec{\omega}_2 = \vec{\omega}_4 = \dot{q}_2 \vec{k} \end{aligned}$$

where $\vec{k}$ is orthogonal to the plane and directed towards us.

It follows that

$$\begin{aligned} \mathcal{K} &= \tfrac{1}{2}\left[m_1 v_{c_1}^T v_{c_1} + \omega_1^T \mathcal{I}_1 \omega_1\right] + \tfrac{1}{2}\left[m_2 v_{c_2}^T v_{c_2} + \omega_2^T \mathcal{I}_2 \omega_2\right] \\ &\qquad + \tfrac{1}{2}\left[m_3 v_{c_3}^T v_{c_3} + \omega_3^T \mathcal{I}_3 \omega_3\right] + \tfrac{1}{2}\left[m_4 v_{c_4}^T v_{c_4} + \omega_4^T \mathcal{I}_4 \omega_4\right] \\ &= \tfrac{1}{2} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}^T \begin{bmatrix} d_{11}(q_1, q_2) & d_{12}(q_1, q_2) \\ d_{12}(q_1, q_2) & d_{22}(q_1, q_2) \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \end{aligned}$$

where

$$\begin{aligned} d_{11} &= m_1 l_{c_1}^2 + m_3 l_{c_3}^2 + m_4 l_4^2 + I_1 + I_3 \\ d_{12} &= \left[m_3 l_2 l_{c_3} - m_4 l_1 l_{c_4}\right] \cos(q_2 - q_1) \\ d_{22} &= m_2 l_{c_2}^2 + m_3 l_2^2 + m_4 l_{c_4}^2 + I_2 + I_4 \end{aligned}$$

Computing the potential energy is not hard

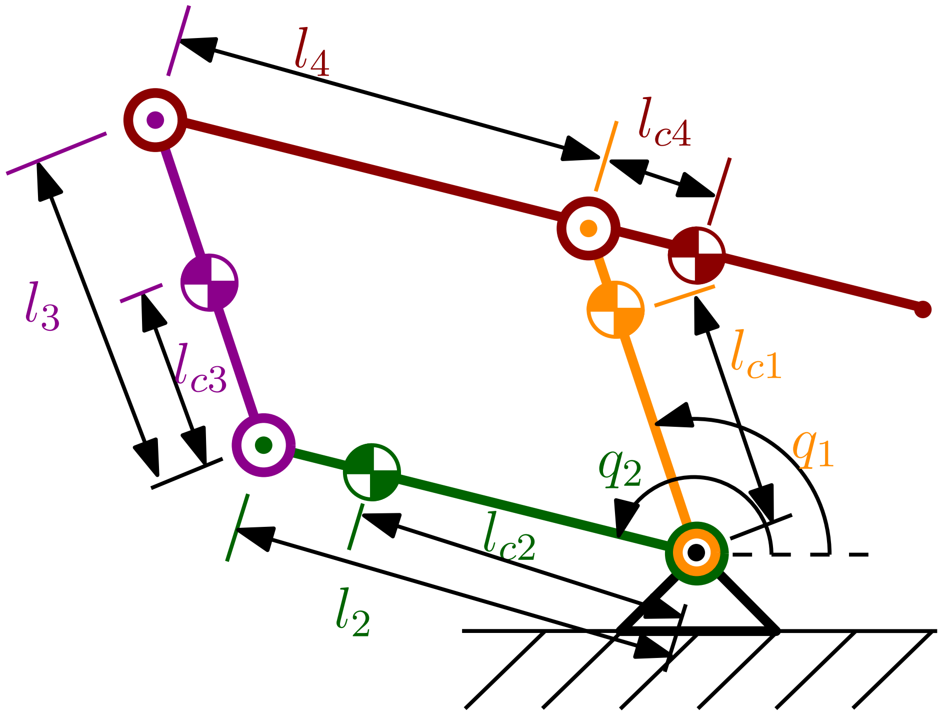
$$\begin{aligned} \mathcal{P} &= \textstyle\sum_{i=1}^4 m_i \, g \, y_{c_i} \\ &= \left[m_1 l_{c_1} + m_3 l_{c_3} + m_4 l_1\right] \, g \, \sin q_1 \\ &\qquad + \left[m_2 l_{c_2} + m_3 l_2 - m_4 l_{c_4}\right] \, g \, \sin q_2 \end{aligned}$$

Finally, Euler-Lagrange formalism can be used to derive the equations of motion. Note that in the special case when $m_3 l_2 l_{c_3} - m_4 l_1 l_{c_4} = 0$, the matrix of inertia is diagonal and constant. As a result, the obtained equations

$$d_{11} \, \ddot{q}_1 + \tfrac{\partial}{\partial q_1} \mathcal{P} = \tau_1 \qquad\qquad d_{22} \, \ddot{q}_2 + \tfrac{\partial}{\partial q_2} \mathcal{P} = \tau_2$$

are decoupled and simple.

It is of interest to consider the case without symmetry, i.e. assuming $l_3 \neq l_1$ and/or $l_4 \neq l_2$, illustrated below

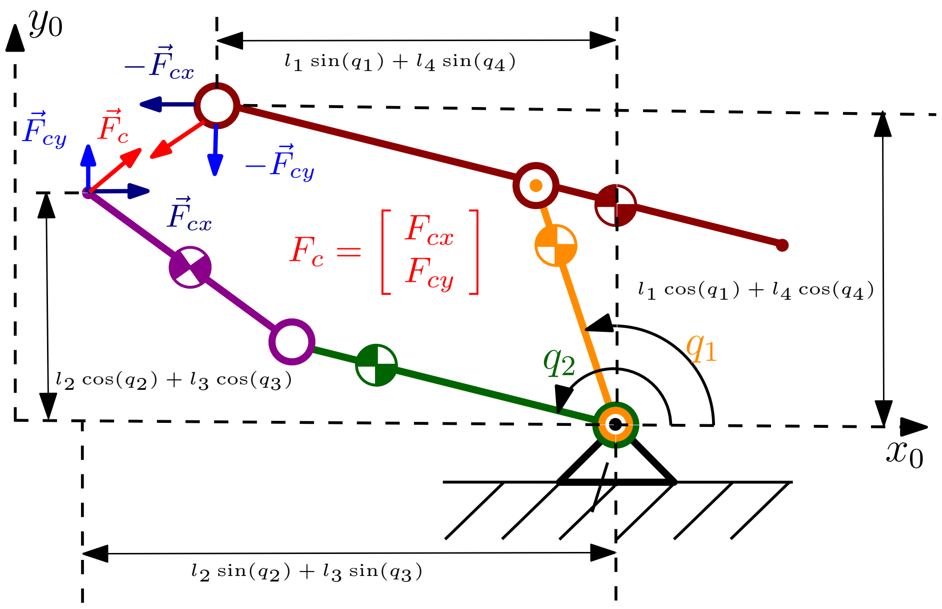


Now, we have to modify the expressions for the centers of mass for the third and forth links since we will have almost always $\boldsymbol{q_3} \neq \boldsymbol{q_1}$ and $\boldsymbol{q_4} \neq \boldsymbol{q_2}$:

$$\begin{bmatrix} x_{c3} \\ y_{c3} \end{bmatrix} = \begin{bmatrix} l_2 \cos q_2 + l_{c3} \cos q_3 \\ l_2 \sin q_2 + l_{c3} \sin q_3 \end{bmatrix}, \qquad \begin{bmatrix} x_{c4} \\ y_{c4} \end{bmatrix} = \begin{bmatrix} l_1 \cos q_1 - l_{c4} \cos q_4 \\ l_1 \sin q_1 - l_{c4} \sin q_4 \end{bmatrix}$$

and can compute the Lagrangian in a similar way as above but as a function of all eight variables: $\boldsymbol{q_1}$, $\boldsymbol{q_2}$, $\boldsymbol{q_3}$, $\boldsymbol{q_4}$, $\boldsymbol{\dot q_1}$, $\boldsymbol{\dot q_2}$, $\boldsymbol{\dot q_3}$, and $\boldsymbol{\dot q_4}$, which are not independent.

There exist two independent holonomic constraints $\boldsymbol{N_c(q)} \in \mathbb{R}^2$, related to presence of the closed kinematic loop. They can be found by computing $\boldsymbol{x}$ and $\boldsymbol{y}$ coordinates of a point following two branches o the loop. For example, “braking” the loop as illustrated below



y_0
$-\vec{F}_{cx}$
$l_1 \sin(q_1) + l_4 \sin(q_4)$
$\vec{F}_{cy}$
$\vec{F}_c$
$-\vec{F}_{cy}$
$\vec{F}_{cx}$
$F_c = \begin{bmatrix} F_{cx} \\ F_{cy} \end{bmatrix}$
$l_1 \cos(q_1) + l_4 \cos(q_4)$
q_1
q_2
$l_2 \cos(q_2) + l_3 \cos(q_3)$
x_0
$l_2 \sin(q_2) + l_3 \sin(q_3)$

one obtains

$$\begin{aligned}
N_c(q) &= \begin{bmatrix} l_1 \cos q_1 + l_4 \cos q_4 - l_2 \cos q_2 - l_3 \cos q_3 \\ l_1 \sin q_1 + l_4 \sin q_4 - l_2 \sin q_2 - l_3 \sin q_3 \end{bmatrix} = 0 \\
\tfrac{d}{dt} N_c(q) &= J_{c,1}(q) \begin{bmatrix} \dot q_1 \\ \dot q_2 \end{bmatrix} + J_{c,2}(q) \begin{bmatrix} \dot q_3 \\ \dot q_4 \end{bmatrix} = 0 \\
\tfrac{d^2}{dt^2} N_c(q) &= \begin{bmatrix} \dot J_{c,1}(q,\dot q) & \dot J_{c,2}(q,\dot q) \end{bmatrix} \dot q + \begin{bmatrix} J_{c,1}(q) & J_{c,2}(q) \end{bmatrix} \ddot q = 0
\end{aligned}$$

The right-hand side of the system of four second-order Euler-Lagrange equations must be modified by adding constraint forces, which appear pre-multiplied by transposed Jacobian of the constraint, according to the virtual work principle:

$$\frac{d}{dt}\left[\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial \dot q}\right] - \frac{\partial(\mathcal{K}-\mathcal{P})}{\partial q} = \tau + J_c^T F_c$$

One can compute constraint forces F_c and reduce the dynamics to just $4-2=2$ second-order differential equations for two generalized coordinates, e.g. $\{q_1, q_2\} \in \mathbb{R}^2$.

Grouping the variables $q_r = [q_1, q_2]^T$ and $q_c = [q_3, q_4]^T$, one can observe that the dynamics is structured as

$$\begin{bmatrix} D_1(q) \\ D_2(q) \end{bmatrix} \begin{bmatrix} \ddot q_r \\ \ddot q_c \end{bmatrix} + \begin{bmatrix} C_1(q,\dot q) \\ C_2(q,\dot q) \end{bmatrix} \begin{bmatrix} \dot q_r \\ \dot q_c \end{bmatrix} + \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} = \begin{bmatrix} \tau_r \\ 0 \end{bmatrix} + \begin{bmatrix} J_{c,1}^T F_c \\ J_{c,2}^T F_c \end{bmatrix}$$

From the holonomic constraint we can substitute

$$\begin{aligned}
& q_c = f(q_r), \quad f : N_c(q_r, q_c) = 0 \quad \to q_c \quad \text{(nontrivial when no symmetry!)} \\
& \dot q_c = -J_{c,2}^{-1}(q) J_{c,1}(q) \dot q_r \\
& \ddot q_c = -J_{c,2}^{-1}(q) \left[\dot J_{c,1}(q, \dot q_r) \dot q_r + J_{c,1}(q) \ddot q_r + \dot J_{c,2}(q, \dot q_r) \dot q_c \right]
\end{aligned}$$

The constraint force is computed from the second equation

$$F_c = \begin{bmatrix} F_{cx} \\ F_{cy} \end{bmatrix} = \left(J_{c,2}^T\right)^{-1} \left(D_2(q) \begin{bmatrix} \ddot q_r \\ \ddot q_c \end{bmatrix} + C_2(q, \dot q_r) \begin{bmatrix} \dot q_r \\ \dot q_c \end{bmatrix} + G_2 \right)$$

and we can substitute it into the first equation. After collecting similar terms the reduced dynamics takes the form

$$D_r(q) \begin{bmatrix} \ddot q_1 \\ \ddot q_2 \end{bmatrix} + C_r(q, \dot q) \begin{bmatrix} \dot q_1 \\ \dot q_2 \end{bmatrix} + G_r = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

In the symmetric case ($l_1 = l_3$ and $l_2 = l_4$), one can proceed in the same way and would obtain equations identical to the ones derived above.

However, a bonus for these more complicated derivation is the possibility to compute the internal constraint forces that in practice should be kept within certain limits in order to avoid braking the system.

Let us discuss now some structural properties of the equations of motion that can be used to check results of numerical simulations and to derive dynamic-model-based feedback control laws.

5.8 Properties of Equations of Motion

Some properties of the system can be derived directly from the structure of the Euler-Lagrange equations. Let us present some of such properties that will be used later for analysis and for design of feedback control laws.

5.8.1 Passivity Relation

Given a mechanical system

$$\frac{d}{dt}\left[\frac{\partial \mathcal{L}}{\partial \dot{q}}\right] - \frac{\partial \mathcal{L}}{\partial q} = \tau \quad \Leftrightarrow \quad D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau$$

with

$$\mathcal{L} = \tfrac{1}{2}\,\dot{q}^T D(q)\dot{q} - P(q)$$

the total mechanical energy is given by

$$\mathcal{H} = \tfrac{1}{2}\,\dot{q}^T D(q)\dot{q} + P(q).$$

Let us compute the time-derivative of the energy along the solutions under an arbitrary vector of generalized forces τ using the resolved Euler-Lagrange equation

$$D(q)\ddot{q} = \tau - C(q,\dot{q})\dot{q} - g(q)$$

Differentiating $\mathcal{H}$ along a solution of the system, we have

$$\begin{aligned}
\tfrac{d}{dt}\mathcal{H} &= \tfrac{d}{dt}\left[\tfrac{1}{2}\dot{q}^T D(q)\dot{q} + P(q)\right] \\
&= \tfrac{1}{2}\ddot{q}^T D(q)\dot{q} + \tfrac{1}{2}\dot{q}^T D(q)\ddot{q} + \tfrac{1}{2}\dot{q}^T \tfrac{d}{dt}\left[D(q)\right]\dot{q} + \dot{q}^T \frac{\partial \mathcal{P}}{\partial q} \\
&= \dot{q}^T D(q)\ddot{q} + \tfrac{1}{2}\dot{q}^T \tfrac{d}{dt}\left[D(q)\right]\dot{q} + \dot{q}^T \frac{\partial \mathcal{P}}{\partial q} \\
&= \dot{q}^T\left[\tau - C(q,\dot{q})\dot{q} - g(q)\right] + \tfrac{1}{2}\dot{q}^T \tfrac{d}{dt}\left[D(q)\right]\dot{q} + \dot{q}^T \frac{\partial \mathcal{P}}{\partial q}
\end{aligned}$$

$$
\begin{aligned}
&= \dot{q}^T\tau + \dot{q}^T\left(\tfrac{1}{2}\tfrac{d}{dt}\left[D(q)\right] - C(q,\dot{q})\right)\dot{q} + \dot{q}^T \underbrace{\left(\frac{\partial \mathcal{P}}{\partial q} - g(q)\right)}_{=0} \\
&= \dot{q}^T\tau + \underbrace{\dot{q}^T\left(\tfrac{1}{2}\tfrac{d}{dt}\left[D(q)\right] - C(q,\dot{q})\right)\dot{q}}_{=0} \\
&= \dot{q}^T\tau
\end{aligned}
$$

In the derivations above we have used two identities.

The first one, $\dfrac{\partial \mathcal{P}}{\partial q} = g(q)$, follows from the definition, while the second one, $\dot{q}^T\left(\frac{1}{2}\frac{d}{dt}\left[D(q)\right] - C(q,\dot{q})\right)\dot{q} = 0$, is to be derived below.

Meanwhile, the obtained differential relation

$$\tfrac{d}{dt}\mathcal{H} \;=\; \dot{q}^T\tau,$$

which in the case of $\tau = 0$ corresponds to the energy conservation law, can be integrated

$$\int_0^T \tfrac{d}{dt}\mathcal{H}(q,\dot{q})dt \;=\; \mathcal{H}(q(T),\dot{q}(T)) - \mathcal{H}(q(0),\dot{q}(0)) = \int_0^T \dot{q}(t)^T\tau(t)dt$$

and, in the case when the total energy is non-negative, we conclude

$$\int_0^T \dot{q}(t)\tau(t)dt \geq -\mathcal{H}(q(0),\dot{q}(0))$$

This is called passivity (dissipativity) relation in integral form.

Note that the total energy can be assumed positive inside every compact set of the state space since the potential energy is defined up to an arbitrary constant off-set.

Let us show now that $\dot{q}^T\left(\frac{1}{2}\frac{d}{dt}\left[D(q)\right] - C(q,\dot{q})\right)\dot{q} = 0$. In fact, we shall show a stronger property

$$v^T\left(\tfrac{1}{2}\tfrac{d}{dt}\left[D(q)\right] - C(q,\dot{q})\right)v = 0 \qquad \forall v \in \mathbb{R}^n$$

that is equivalent to the skew-symmetry of the matrix $\frac{d}{dt}\left[D(q)\right] - 2C(q,\dot{q})$.

Skew Symmetry of $\dot{D}(q) - 2\,C(q,\dot{q})$

To check that

$$N = \tfrac{d}{dt}\left[D(q)\right] - 2C(q,\dot{q}) \quad \Rightarrow \quad N^T = -N$$

we compute the $(k, j)^{th}$ element of this matrix

$$\begin{aligned} n_{kj} = \tfrac{d}{dt} d_{kj} - 2c_{kj} &= \sum_{i=1}^{n} \frac{\partial d_{kj}}{\partial q_i} \dot{q}_i - \sum_{i=1}^{n} \left[\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k} \right] \dot{q}_i \\ &= \sum_{i=1}^{n} \left\{ \frac{\partial d_{kj}}{\partial q_i} - \left[\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k} \right] \right\} \dot{q}_i \\ &= \sum_{i=1}^{n} \left\{ \frac{\partial d_{ij}}{\partial q_k} - \frac{\partial d_{ki}}{\partial q_j} \right\} \dot{q}_i = \sum_{i=1}^{n} \left\{ \frac{\partial d_{ji}}{\partial q_k} - \frac{\partial d_{ki}}{\partial q_j} \right\} \dot{q}_i \end{aligned}$$

where for the last equality we have used the symmetry of D: $d_{ij} = d_{ji}$ $\forall i, j$ follows from $D = D^T$.

It is now obvious that

$$n_{kj} = -n_{jk}$$

and so the skew-symmetry property is established.

5.8.2 Bounds on Matrix of Inertia

The kinetic energy of any mechanical system

$$\mathcal{K}(q, \dot{q}) = \tfrac{1}{2} \dot{q}^T D(q) \dot{q}$$

is non-negative, since it is derived as a sum of positive terms, each of which is computed as a product of a positive constant and the square of the magnitude of a velocity vector.

It is clear that $D(q)$ is symmetric and non-negative matrix at an arbitrary configuration defined by values of the generalized coordinates. Moreover, there exist two non-negative scalar functions $m(q)$ and $M(q)$ such that

$$0 \le m(q)\, I \le D(q) \le M(q)\, I$$

for every q.

However, in general, it is not true that $D(q)$ is positive definite, i.e. it can happen that there exists q^* such that $m(q^*) = 0$ and $\det D(q^*) = 0$.

As an example, consider the point-mass spherical pendulum of length l, orientation of which is defined by the two angles θ and ϕ of the spherical coordinate system. The kinetic energy can be computed as the product of the mass m and the magnitude of the velocity expressed in the generalized coordinates. The result is

$$\mathcal{K} = \tfrac{1}{2} m l^2 \left[\sin^2(\theta)\dot{\phi}^2 + \dot{\theta}^2 \right] = \frac{1}{2} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \end{bmatrix}^T \begin{bmatrix} m l^2 \sin^2\theta & 0 \\ 0 & m l^2 \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \end{bmatrix}$$

and we have a singularity at $\theta = 0$.

5.8.3 Linearity of Dynamics in Parameters

Consider the equations of motion of a mechanical system derived above

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau$$

and suppose that we have measured (or estimated somehow)

- Control torque $\tau = \tau(t)$, $t \in [0, T]$

- Positions, velocities and accelerations

$$q = q(t), \quad \dot{q} = \dot{q}(t), \quad \ddot{q} = \ddot{q}(t), \quad t \in [0, T]$$

on some nontrivial interval of time, i.e. with $T > 0$.

An important question is the following:

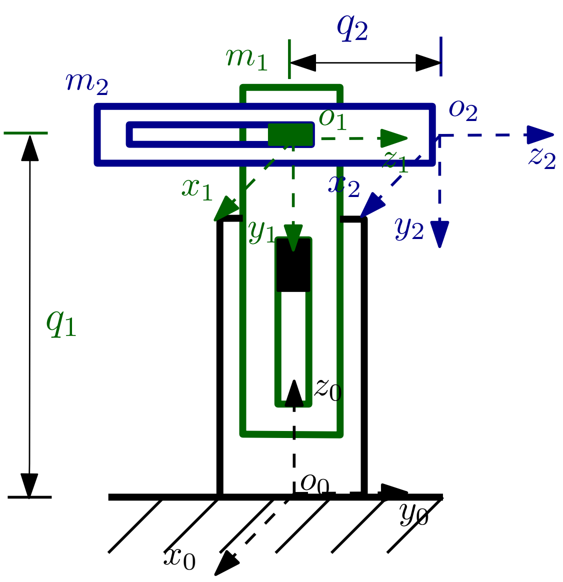
Can we reconstruct the physical parameters of the system (masses, coefficients of inertia matrices, dimensions) ?

and if the answer is yes, how this can be done?

Let us attempt to answer these questions for an example.

Example: Two Link Cartesian Manipulator

Let us reconsider the two-link planar manipulator of PP-type shown below



The equations of motion are

$$(m_1+m_2)\ddot{q}_1+g(m_1+m_2)=\tau_1, \qquad m_2\ddot{q}_2=\tau_2$$

and involve only **2** parameters.

They can be rewritten as follows

$$\begin{bmatrix} \ddot{q}_1(t)+g & 0 \\ 0 & \ddot{q}_2(t) \end{bmatrix} \begin{bmatrix} m_1+m_2 \\ m_2 \end{bmatrix} = \begin{bmatrix} \tau_1(t) \\ \tau_2(t) \end{bmatrix}$$

The equation for parameters

$$\underbrace{Y(\ddot{q}_1,\ddot{q}_2)}_{REGRESSOR} \qquad \underbrace{\begin{bmatrix} p_1 \\ p_2 \end{bmatrix}}_{UNKNOWN\ CONSTANTS} \qquad = \begin{bmatrix} \tau_1(t) \\ \tau_2(t) \end{bmatrix}$$

with $Y(\ddot{q}_1,\ddot{q}_2)=\begin{bmatrix} \ddot{q}_1(t)+g & 0 \\ 0 & \ddot{q}_2(t) \end{bmatrix}$, $p_1=m_1+m_2$, and $p_2=m_2$, can be solved for every moment of time when $\ddot{q}_2\neq 0$ and $\ddot{q}_1\neq -g$.

It is clear that

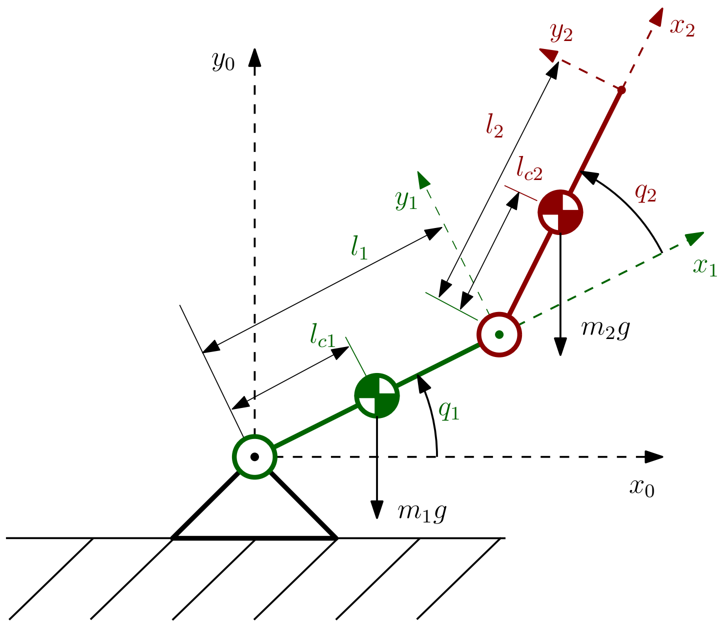
$$m_2=\tau_2/\ddot{q}_2, \qquad m_1=\tau_1/(\ddot{q}_1-g)-\tau_2/\ddot{q}_2$$

In general, it is possible to write the dynamics in the following form

$$D(q)\ddot{q}+C(q,\dot{q})\dot{q}+G(q)\equiv Y(q,\dot{q},\ddot{q})\,p=\tau$$

where the so-called regressor $Y(q,\dot{q},\ddot{q})$ is formed by known functions and p contains various combinations of physical parameters (masses, moments of inertia, and various lengths). Note, however, that finding the appropriate vector of parameters p with the lowest dimension and the corresponding regressor can be a nontrivial computational task for a many-DOF system.

For example, for a planar elbow manipulator



we have

$$
\begin{aligned}
\boldsymbol{D}(\boldsymbol{q}) &= \begin{bmatrix} p_1 + p_2 + 2p_3\cos q_2 & p_2 + p_3 \cos q_2 \\ p_2 + p_3\cos q_2 & p_3 \end{bmatrix} \\
\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= \begin{bmatrix} -p_3 \sin(q_2)\dot{q}_2 & -p_3\sin(q_2)(\dot{q}_1 + \dot{q}_2) \\ p_3 \sin(q_2)\dot{q}_1 & 0 \end{bmatrix} \\
\boldsymbol{G}(\boldsymbol{q}) &= \begin{bmatrix} p_4 \cos q_1 + p_5\cos(q_1 + q_2) \\ p_5 \cos(q_1 + q_2) \end{bmatrix}
\end{aligned}
$$

where a minimal set of $\boldsymbol{l} = \mathbf{5}$ parameters can be taken as

$$
\boldsymbol{p} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{bmatrix} = \begin{bmatrix} m_1 l_{c_1}^2 + m_2 l_1^2 + I_1 \\ m_2 l_{c_2}^2 + I_2 \\ m_2 l_1 l_{c_2} \\ (m_1 l_{c_1} + m_2 l_1) g \\ m_2 l_{c_2} g \end{bmatrix}
$$

leading to the regressor

$$
\boldsymbol{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) \equiv \boldsymbol{D}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) =
$$

$$
\begin{bmatrix} \ddot{q}_1 & \ddot{q}_1 + \ddot{q}_2 & (2\ddot{q}_1 + \ddot{q}_2)\cos q_2 - (2\dot{q}_1\dot{q}_2 + \dot{q}_2^2)\sin q_2 & \cos q_1 & \cos(q_1 + q_2) \\ 0 & \ddot{q}_1 & \ddot{q}_1 \cos q_2 + \ddot{q}_2 + \dot{q}_1^2 \sin q_2 & 0 & \cos(q_1 + q_2) \end{bmatrix}
$$

It is also often a nontrivial or impossible (as in this case) task to obtain the estimates for the values of the real physical parameters from the values of the components of $\boldsymbol{p}$.

Part III

Robotics Control

Chapter 6

Multivariable Control of Manipulators

Here we discuss a few mathematics-based approaches for derivation of feedback control laws. Let us start with a small improvement of our dynamic model. We will assume that our actuators are DC-motors that produce generalized forces acting through gear-boxes directly on each of the generalized coordinates. Let us merge a simplified model of the actuators with the mechanical model described above.

6.1 Dynamics with Geared DC-Motors

Given a mechanical system with n DOF, dynamics of which is described by the following Euler-Lagrange equations

$$D(q)\ddot{q}+C(q,\dot{q})\dot{q}+g(q) = \tau, \qquad q = [q_1, \ldots, q_n]^T, \quad \tau = [\tau_1, \ldots, \tau_n]^T$$

and suppose that each DOF is controlled by a geared DC-motor, dynamics of which can be described as a separated 1-DOF mechanical system with viscous friction and a generalized force, which is proportional to a control signal V_k,

$$J_{m_k}\ddot{\theta}_k + B_k\dot{\theta}_k = K_{m_k}V_k + F_k, \qquad F_k = -\frac{1}{r_k}\tau_k, \qquad k = 1, \ldots, n$$

where $F_k = -\frac{1}{r_k}\tau_k$ represents the constraint force corresponding to the reaction force that is applied to the mechanical part τ_k, which is reduced in amplitude due to the gear and has an opposite sign, while

- θ_k is the angle of the rotor of the k^{th} motor;
- r_k is the gear ration;

- J_{m_k}, B_k, K_{m_k} are parameters of the k^{th} DC-motor:
  - J_{m_k} is the inertia of the rotor,
  - B_k is the combined coefficient of the viscous friction and of the induced back electromotive force, and
  - K_{m_k} is the so-called motor constant, which is the gain between the applied voltage and the produced torque.

Moreover, suppose that the generalized coordinates for the n-DOF mechanical system and for the motors are related by the gear ratios

$$\theta_{m_k} = r_k\, q_k, \quad k = 1, \ldots, n$$

Then, the actuator equations can be rewritten as follows

$$r_k^2 J_{m_k} \ddot{q}_k + r_k^2 B_k \dot{q}_k = r_k K_{m_k} V_k - \tau_k, \quad k = 1, \ldots, n$$

We will give a few more details on derivation of these equation in Chapter 7.

The equations of the combined dynamical system

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau, \qquad q = \left[q_1, \ldots, q_n\right]^T, \quad \tau = [\tau_1, \ldots, \tau_n]^T$$

$$r_k^2 J_{m_k} \ddot{q}_k + r_k^2 B_k \dot{q}_k = r_k K_{m_k} V_k - \tau_k, \qquad k = 1, \ldots, n$$

can be rewritten as one, if we exclude τ, which is the vector of internal forces, keeping valid our n holonomic constraints $\theta_{m_k} = r_k\, q_k$ for $k = 1, \ldots, n$!

Finally, we obtain

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + B\dot{q} + g(q) = u$$

where

- $M(q) = D(q) + J$ with $J = \operatorname{diag}\left\{r_1^2 J_{m_1}, \ldots, r_n^2 J_{m_n}\right\}$, so $M(q)$ is positive definite;
- $B = \left[r_1^2 B_1, r_2^2 B_2, \ldots, r_n^2 B_n\right]^T$
- $u = [u_1, u_2, \ldots, u_n]^T$ with $u_k = r_k K_{m_k} V_k$, $k = 1, \ldots, n$

Let us consider possible strategies for designing stabilizing feedback control laws.

6.2 PD-Controller Design

Given a mechanical system

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{B}\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u}$$

we are interested

- to design a controller to stabilize a particular configuration of the robot: $\boldsymbol{q} = \boldsymbol{q_d} = \text{const}$,
- to analyze the closed-loop system stability.

6.2.1 The Case of Absent Gravity

Suppose first that $\boldsymbol{B} = \boldsymbol{0}$ and $\boldsymbol{g}(\boldsymbol{p}) = \boldsymbol{0}$. We will show later how to remove this assumption.

Let the control law be taken as the classical proportional-differential (PD) feedback

$$\boldsymbol{u} = -\boldsymbol{K_p}\left(\boldsymbol{q} - \boldsymbol{q_d}\right) - \boldsymbol{K_d}\dot{\boldsymbol{q}}$$

with $\boldsymbol{K_p}$ and $\boldsymbol{K_d}$ taken as either diagonal matrices with positive elements or arbitrary symmetric positive definite matrices.

To analyze the behavior of the closed-loop system

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \boldsymbol{u} = -\boldsymbol{K_p}\left(\boldsymbol{q} - \boldsymbol{q_d}\right) - \boldsymbol{K_d}\dot{\boldsymbol{q}}$$

consider the scalar function

$$\boldsymbol{V}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \tfrac{1}{2}\,\dot{\boldsymbol{q}}^T\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}} + \tfrac{1}{2}\left(\boldsymbol{q} - \boldsymbol{q_d}\right)^T\boldsymbol{K_p}\left(\boldsymbol{q} - \boldsymbol{q_d}\right)$$

Its time-derivative along solutions of the closed-loop system is

$$\begin{aligned}
\tfrac{d}{dt}\boldsymbol{V} &= \dot{\boldsymbol{q}}^T\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{q}}^T\tfrac{d}{dt}\tfrac{1}{2}\left[\boldsymbol{M}(\boldsymbol{q})\right]\dot{\boldsymbol{q}} + \dot{\boldsymbol{q}}^T\boldsymbol{K_p}\left(\boldsymbol{q} - \boldsymbol{q_d}\right)\\
&= \dot{\boldsymbol{q}}^T\left[\boldsymbol{u} - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}\right] + \dot{\boldsymbol{q}}^T\tfrac{d}{dt}\tfrac{1}{2}\left[\boldsymbol{M}(\boldsymbol{q})\right]\dot{\boldsymbol{q}} + \dot{\boldsymbol{q}}^T\boldsymbol{K_p}\left(\boldsymbol{q} - \boldsymbol{q_d}\right)\\
&= \dot{\boldsymbol{q}}^T\left[\boldsymbol{u} + \boldsymbol{K_p}\left(\boldsymbol{q} - \boldsymbol{q_d}\right)\right] + \underbrace{\dot{\boldsymbol{q}}^T\left\{\tfrac{d}{dt}\tfrac{1}{2}\left[\boldsymbol{D}(\boldsymbol{q}) + \boldsymbol{J}\right] - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\right\}\dot{\boldsymbol{q}}}_{=\boldsymbol{0}}\\
&= \dot{\boldsymbol{q}}^T\left[\boldsymbol{u} + \boldsymbol{K_p}\left(\boldsymbol{q} - \boldsymbol{q_d}\right)\right]\\
&= \dot{\boldsymbol{q}}^T\left[-\boldsymbol{K_p}\left(\boldsymbol{q} - \boldsymbol{q_d}\right) - \boldsymbol{K_d}\dot{\boldsymbol{q}} + \boldsymbol{K_p}\left(\boldsymbol{q} - \boldsymbol{q_d}\right)\right] = -\dot{\boldsymbol{q}}^T\boldsymbol{K_d}\dot{\boldsymbol{q}}
\end{aligned}$$

So, finally, since $\boldsymbol{K_d} > \boldsymbol{0}$ or $\boldsymbol{K_d} = \text{diag}\left\{\boldsymbol{K_{d1}}, \boldsymbol{K_{d2}}, \ldots, \boldsymbol{K_{dn}}\right\} > \boldsymbol{0}$, we conclude

$$\tfrac{d}{dt}\boldsymbol{V} = -\dot{\boldsymbol{q}}^T\boldsymbol{K_d}\dot{\boldsymbol{q}} \le \boldsymbol{0}.$$

It follows that

- $V(q(t), \dot{q}(t))$ is monotonically not increasing!

and it is not hard to see that

- V is positive definite (hence, bounded from below by 0), i.e. $V \geq 0$ and $V(q, \dot{q}) = 0 \Leftrightarrow \{q = q_d, \dot{q} = 0\}$

Therefore, according to elementary results from real analysis,

$$\boxed{\exists \lim_{t\to+\infty} V(q(t), \dot{q}(t)) = V_\infty} \quad \text{and} \quad \boxed{\exists \lim_{t\to+\infty} \dot{q}(t) = \dot{q}_\infty(t) = 0}$$

If we substitute this limit trajectory into the dynamics, we obtain

$$M(q_\infty)\underbrace{\ddot{q}_\infty}_{=0} + C(q_\infty, \dot{q}_\infty)\underbrace{\dot{q}_\infty}_{=0} = -K_p(q_\infty - q_d) - K_d\underbrace{\dot{q}_\infty}_{=0}$$

that is

$$0 = -K_p(q_\infty - q_d)$$

Now,

$$K_p > 0 \quad \text{or} \quad K_p = \operatorname{diag}\{K_{p1}, K_{p2}, \ldots, K_{pn}\} > 0 \ \Rightarrow \ \boxed{q_\infty = q_d}$$

Therefore, according to either the Barbashin-Krasovskiĭ theorem or the invariance principle $\boxed{\exists \lim_{t\to+\infty} q(t) = q_d}$, and, moreover, $(q, \dot{q}) = (q_d, 0)$ is a globally asymptotically stable (GAS) equilibrium of the closed-loop system.

6.2.2 PD-Controller with Gravity Compensation

Given a mechanical system

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + B\dot{q} + g(q) = u,$$

how to modify the previously PD controller

$$u = -K_p(q - q_d) - K_d\dot{q}$$

to work in the case of $B \neq 0$ and $g(p) \neq 0$?

Obviously, the controller

$$u = -K_p(q - q_d) - K_d\dot{q} + g(q)$$

is stabilizing, if K_p and K_d are diagonal matrices with positive elements or just positive definite matrices such that

$$K_p > 0, \qquad K_d + B > 0.$$

We will leave to the reader investigation of the so-called $\boldsymbol{PD+}$ controller

$$\boldsymbol{u} = -\boldsymbol{K}_p\left(\boldsymbol{q} - \boldsymbol{q}_d\right) - \boldsymbol{K}_d\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}_d)$$

that can be shown to ensure GAS as well with the help of the following auxiliary function

$$\boldsymbol{V}(\boldsymbol{e}, \dot{\boldsymbol{q}}) = \tfrac{1}{2}\,\dot{\boldsymbol{q}}^T\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}} + \tfrac{1}{2}\,\boldsymbol{e}^T\boldsymbol{K}_p\boldsymbol{e} + \boldsymbol{P}(\boldsymbol{e} + \boldsymbol{q}_d) - \boldsymbol{P}(\boldsymbol{q}_d) + \boldsymbol{e}^T\boldsymbol{g}(\boldsymbol{q}_d),$$

where $\boldsymbol{e} = \boldsymbol{q} - \boldsymbol{q}_d$, provided the gravitational force is globally Lipschitz, i.e. $\exists\boldsymbol{\alpha} > \boldsymbol{0}$ s.t.

$$\Big(\boldsymbol{g}(\boldsymbol{q}) - \boldsymbol{g}(\boldsymbol{q}_d)\Big)^T\Big(\boldsymbol{g}(\boldsymbol{q}) - \boldsymbol{g}(\boldsymbol{q}_d)\Big) \leq \boldsymbol{\alpha}\left(\boldsymbol{q} - \boldsymbol{q}_d\right)^T\left(\boldsymbol{q} - \boldsymbol{q}_d\right), \qquad \forall\boldsymbol{q}$$

and

$$\boldsymbol{K}_p - \boldsymbol{\alpha}\boldsymbol{I} > 0, \qquad \boldsymbol{K}_d + \boldsymbol{B} > 0.$$

where $\boldsymbol{I}$ is the identity matrix.

6.3 Feedback Linearization

Let us consider now one possible control design strategy that can be applied in the case when $\boldsymbol{q}_d(\boldsymbol{t}) \neq$ const. This is known as **tracking** problem as opposed to the **point-to-point** or **regulation** problem solved above.

6.3.1 Feedback Linearization Under No Uncertainty

Given a mechanical system

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{B}\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u},$$

how to define a control transformation

$$\boldsymbol{u} = \boldsymbol{\alpha}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{\beta}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{v}$$

such that the closed-loop system is linear and stabilizable? In other words, such that, possibly after a transformation of state variables

$$[\boldsymbol{q}, \dot{\boldsymbol{q}}] \longleftrightarrow \boldsymbol{x},$$

the original dynamics is equivalent to

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\,\boldsymbol{x} + \boldsymbol{B}\,\boldsymbol{v}$$

where the pair (A, B) is stabilizable?

In order to cancel many nonlinear terms we can simply start with

$$u = M(q)v + C(q,\dot{q})\dot{q} + B\dot{q} + g(q)$$

Then, obviously, we obtain

$$M(q)\ddot{q} = M(q)v \quad \Longleftrightarrow \quad \ddot{q} = v$$

using that $M(q) > 0$.

Now,

$$\ddot{q} = v \quad \Longleftrightarrow \quad \boxed{\dot{x} = \underbrace{\begin{bmatrix} 0_n & I_n \\ 0_n & 0_n \end{bmatrix}}_{=A} x + \underbrace{\begin{bmatrix} 0_n \\ I_n \end{bmatrix}}_{=B} v, \quad x = [q^T, \dot{q}^T]^T}$$

A stabilizing feedback control law for the obtained linear system can be designed easier. A possible choice is again a PD feedback but this time it is applied after a nonlinear feedback transformation.

So, for a given mechanical system

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + B\dot{q} + g(q) = u$$

and a desired twice continuously differentiable trajectory $q_d = q_d(t)$, let us introduce the controller

$$u \;=\; M(q)v + C(q,\dot{q})\dot{q} + B\dot{q} + g(q)$$

where

$$v \;=\; \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t))$$

with $K_p > 0$ and $K_d > 0$.

Then, the closed-loop system is

$$\ddot{q} = v = \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t))$$

and it can be rewritten in the error-variables as

$$\boxed{\ddot{e} + K_d\,\dot{e} + K_p\,e = 0, \qquad e = q - q_d(t)}$$

It is obvious that $e \to 0$ as $t \to \infty$, while this is usually not true in the case when the simple PD-feedback, suggested above for the special case of constant reference trajectory, is applied directly for the tracking problem.

6.3.2 Robust and Adaptive Motion Control

We would like to propose now some modifications of the feedback linearization controller presented above. To this end, we notice that, in reality, for a given mechanical system

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = u$$

with a desired smooth trajectory $q_d = q_d(t)$, the controller

$$\begin{aligned} u &= \boxed{M(q)v + C(q,\dot{q})\dot{q} + g(q)} \\ v &= \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t)) \end{aligned}$$

cannot be implemented since the values of the parameters, which appear in the expressions for the elements of $M(q)$, $C(q,\dot{q})$, and $g(q)$, are never known precisely!

Of course, approximations might be used

$$\begin{aligned} u &= \boxed{[M + \triangle M]\, v + [C + \triangle C]\, \dot{q} + [g + \triangle g]} \\ v &= \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t)) \end{aligned}$$

If performance of the approximate realization of the feedback linearization-based design is not acceptable, one can use one the following two standard approaches to improve it.

- **Robust Control**:
  Design K_p, K_d, and, if necessary, an additional input w in the controller

  $$\begin{aligned} u &= \boxed{[M + \triangle M]\, v + [C + \triangle C]\, \dot{q} + [g + \triangle g]} \\ v &= \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t)) + w \end{aligned}$$

  such that the error signal

  $$e(t) = q(t) - q_d(t) \approx 0 \qquad \forall \{\triangle M, \triangle C, \triangle g\} \in \mathcal{W}$$

- **Adaptive Control**: Improve the estimates for $M(q)$, $C(q,\dot{q})$, and $g(q)$ on-line until the error signal vanishes. Here we might aim at finding the law of change for the estimates achieving $\triangle M,\ \triangle C,\ \triangle g \to 0$.

6.3.3 Robust Control Based on Feedback Linearization

Given a trajectory $\boldsymbol{q} = \boldsymbol{q}_d(\boldsymbol{t})$, consider our system

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u}$$

with approximate feedback linearization control law based on available approximations (nominal values) of the system parameters

$$\boldsymbol{u} \;=\; [\boldsymbol{M} + \triangle\boldsymbol{M}]\,\boldsymbol{v} + [\boldsymbol{C} + \triangle\boldsymbol{C}]\,\dot{\boldsymbol{q}} + [\boldsymbol{g} + \triangle\boldsymbol{g}]$$

Substituting, we obtain

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = [\boldsymbol{M} + \triangle\boldsymbol{M}]\,\boldsymbol{v} + [\boldsymbol{C} + \triangle\boldsymbol{C}]\,\dot{\boldsymbol{q}} + [\boldsymbol{g} + \triangle\boldsymbol{g}]$$

$$\Leftrightarrow \quad \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} = [\boldsymbol{M} + \triangle\boldsymbol{M}]\,\boldsymbol{v} + \triangle\boldsymbol{C}\dot{\boldsymbol{q}} + \triangle\boldsymbol{g}$$

$$\Leftrightarrow \quad \ddot{\boldsymbol{q}} = \boldsymbol{M}^{-1}[\boldsymbol{M} + \triangle\boldsymbol{M}]\,\boldsymbol{v} + \boldsymbol{M}^{-1}[\triangle\boldsymbol{C}\dot{\boldsymbol{q}} + \triangle\boldsymbol{g}] \pm \boldsymbol{M}^{-1}\boldsymbol{M}\boldsymbol{v}$$

Finally, the closed-loop system can be rewritten as

$$\ddot{\boldsymbol{q}} = \boldsymbol{v} + \boldsymbol{\eta}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{v})$$

where

$$\boldsymbol{\eta}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{v}) = \boldsymbol{M}^{-1}(\boldsymbol{q})\,[\triangle\boldsymbol{M}(\boldsymbol{q})]\,\boldsymbol{v} + \boldsymbol{M}^{-1}(\boldsymbol{q})\,[\triangle\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \triangle\boldsymbol{g}(\boldsymbol{q})]$$

Since

$$\triangle\boldsymbol{M} = \boldsymbol{0}, \quad \triangle\boldsymbol{C} = \boldsymbol{0}, \quad \triangle\boldsymbol{g} = \boldsymbol{0} \qquad \Rightarrow \qquad \boldsymbol{\eta} = \boldsymbol{0}$$

it makes sence to modify our previous design

$$\boldsymbol{v} \;=\; \ddot{\boldsymbol{q}}_d(\boldsymbol{t}) - \boldsymbol{K}_p(\boldsymbol{q} - \boldsymbol{q}_d(\boldsymbol{t})) - \boldsymbol{K}_d(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_d(\boldsymbol{t})) + \boldsymbol{w}$$

to keep the same performance in the unrealistic case of perfect knowledge of the parameters with $\boldsymbol{w} = \boldsymbol{0}$.

With this choice, our closed-loop system

$$\ddot{\boldsymbol{q}} = \boldsymbol{v} + \boldsymbol{\eta}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{v})$$

becomes

$$(\ddot{\boldsymbol{q}} - \ddot{\boldsymbol{q}}_d) + \boldsymbol{K}_d\,(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_d) + \boldsymbol{K}_p\,(\boldsymbol{q} - \boldsymbol{q}_d) = \boldsymbol{w} + \boldsymbol{\eta}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{v})$$

and can be rewritten as

$$\frac{d}{dt}e = \underbrace{\begin{bmatrix} 0_n & I_n \\ -K_p & -K_d \end{bmatrix}}_{A} e + \underbrace{\begin{bmatrix} 0_n \\ I_n \end{bmatrix}}_{B} (w + \eta)\,, \quad e = \begin{bmatrix} q - q_d \\ \dot q - \dot q_d \end{bmatrix}$$

or

$$\frac{d}{dt}e = A\,e + B\,(w + \eta(q, \dot q, v))\,, \quad e = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} \equiv \begin{bmatrix} q - q_d(t) \\ \dot q - \dot q_d(t) \end{bmatrix}$$

We should now design w for the system

$$\frac{d}{dt}e = Ae + B\,[w + \bar\eta(t, e, w)]$$

where $\eta(q, \dot q, v)$ is rewritten as

$$\bar\eta(t, e, w) = \eta\Big(\underbrace{e_1 + q_d(t)}_{q},\ \underbrace{e_2 + \dot q_d(t)}_{\dot q},\ \underbrace{\ddot q_d(t) - \left[\,K_p \quad K_d\,\right] e + w}_{v}\Big)$$

Under certain assumptions on the mechanical system, it can be verified that there are positive constants α,$\gamma_{1,2,3}$ such that the following estimate on $\bar\eta(t, e, w)$ is true

$$\|\bar\eta(t, e, w)\| \le \alpha\,\|w\| + \gamma_1\,\|e\| + \gamma_2\,\|e\|^2 + \gamma_3, \quad \alpha < 1$$

Now, the matrix A is stable; hence $\forall\, Q > 0$, $\exists\, P = P^T > 0$ such that

$$A^T P + PA = -Q$$

With such P, consider a Lyapunov function candidate

$$V(e) = e^T P e$$

and compute its time-derivative along the solutions

$$\begin{aligned} \frac{d}{dt}V &= \frac{d}{dt}e^T P e + e^T P \frac{d}{dt}e \\ &= e^T(A^T P + PA)e + 2e^T PB\,[w + \bar\eta(t, e, w)] \\ &= -e^T Q e + 2e^T PB\,[w + \bar\eta] \\ &\le 0 \quad \leftarrow \ \text{How to achieve this by choosing } w\text{?} \end{aligned}$$

Let us look at the second term in

$$\frac{d}{dt}V \ = \ -e^T Q e + 2e^T PB\,[w + \bar\eta]$$

when w has the form

$$w = -\rho(\cdot)\frac{z}{\sqrt{z^T z}}, \qquad z = B^T P e, \qquad \rho \text{ is a function to choose}$$

We have

$$e^T P B\left[w + \bar{\eta}\right] = z^T\left(-\rho\frac{z}{\sqrt{z^T z}} + \bar{\eta}\right) \|z\|\left(-\rho + \|\bar{\eta}\|\right)$$

To sum up, we search for a scalar function $\rho(\cdot)$ such that

$$(-\rho + \|\bar{\eta}\|) \leq 0 \quad \Leftrightarrow \quad \|\bar{\eta}\| \leq \rho$$

and we have

$$\|\bar{\eta}(t, e, w)\| \leq \alpha\,\|w\| + \gamma_1\,\|e\| + \gamma_2\,\|e\|^2 + \gamma_3, \quad \alpha < 1$$

with

$$w = -\rho(\cdot)\frac{z}{\sqrt{z^T z}}$$

Therefore, we must have

$$\alpha\left\|\rho(\cdot)\frac{z}{\sqrt{z^T z}}\right\| + \gamma_1\,\|e\| + \gamma_2\,\|e\|^2 + \gamma_3 \leq \rho(\cdot)$$

$$\Rightarrow \alpha\rho(\cdot) + \gamma_1\,\|e\| + \gamma_2\,\|e\|^2 + \gamma_3 \leq \rho(\cdot)$$

$$\Rightarrow \rho(e) \geq \frac{1}{1-\alpha}\left[\gamma_1\,\|e\| + \gamma_2\,\|e\|^2 + \gamma_3\right]$$

Finally, for a given trajectory $q = q_d(t)$, we design stabilizing feedback controller for

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) = u$$

in the following form

$$\begin{aligned}
u &= \left[M(q) + \triangle M(q)\right] v + \left[C(q, \dot{q}) + \triangle C(q, \dot{q})\right]\dot{q} + \left[g(q) + \triangle g(q)\right] \\
v &= \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t)) + w \\
w &= \begin{cases} -\rho(e)\frac{z}{\sqrt{z^T z}}, & \text{if } z = B^T P e \neq 0 \\ 0, & \text{if } z = B^T P e = 0 \end{cases}
\end{aligned}$$

where $\rho(\cdot)$ is any function that satisfies the inequality

$$\rho \geq \frac{1}{1-\alpha}\left[\gamma_1\,\|e\| + \gamma_2\,\|e\|^2 + \gamma_3\right]$$

where constants α, γ_1-γ_2 are from the inequality

$$\|\eta\| \le \alpha\,\|w\| + \gamma_1\,\|e\| + \gamma_2\,\|e\|^2 + \gamma_3, \quad \alpha < 1$$

and

$$\eta = M^{-1}(q)\left[\triangle M(q) v + \triangle C(q,\dot q)\dot q + \triangle g(q)\right], \quad e = \begin{bmatrix} q - q_d \\ \dot q - \dot q_d \end{bmatrix}$$

Implemented controller above, however, often leads to very fast switching of the control signal known as **chattering** phenomenon. Let us discuss what it means and how to avoid it. If the control variable is scalar then the expression

$$w = \begin{cases} -\rho(e)\frac{z}{\sqrt{z^T z}}, & \text{if } z = B^T P e \neq 0 \\ 0, & \text{if } z = B^T P e = 0 \end{cases}$$

becomes

$$w = \begin{cases} -\rho(e)\,\mathrm{sign}(z), & \text{if } z = B^T P e \neq 0 \\ 0, & \text{if } z = B^T P e = 0 \end{cases}$$

This means that whenever $z(t)$ is changing its sign, the control signal has to be changed instantaneously with a non-zero increment! Since such change may be impossible to implement by an actuator, and due to various delays and other implementation imperfections, high frequency oscillations in the control signal with large amplitude, known as chattering, appear.

One of possible choices to avoid the discontinuity in the control signal is to use the following continuous control law

$$w = \begin{cases} -\rho(e)\,\mathrm{sign}(B^T P e), & \text{if } |B^T P e| > \delta \\ -\frac{1}{\delta}\,\rho(e)\,B^T P e, & \text{if } |B^T P e| \le \delta \end{cases}$$

Let us analyze the implications of such a modification.

The closed-loop system can be rewritten as

$$\tfrac{d}{dt}e = A\,e + B\left[w(e) + \eta\right], \quad w \in \mathbb{R}^1, \quad \eta \in \mathbb{R}^1$$

where we have assumed that

$$|\eta| \le \rho(e)$$

with some known in advance function $\rho(e)$.

As above, we have

$$A = \begin{bmatrix} 0_n & I_n \\ -K_p & -K_d \end{bmatrix}, \quad B = \begin{bmatrix} 0_n \\ I_n \end{bmatrix}, \quad e = \begin{bmatrix} q - q_d \\ \dot q - \dot q_d \end{bmatrix}$$

Since matrix $\boldsymbol{A}$ is stable (Hurwitz), $\forall\, \boldsymbol{Q} > \boldsymbol{0}$, $\exists\, \boldsymbol{P} = \boldsymbol{P}^T > \boldsymbol{0}$ such that

$$\boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} = -\boldsymbol{Q}$$

As before, we consider the quadratic Lyapunov function candidate

$$\boldsymbol{V}(\boldsymbol{e}) = \boldsymbol{e}^T\boldsymbol{P}\boldsymbol{e}$$

and compute its time derivative along the systems' trajectories

$$\begin{array}{rcl}\frac{d}{dt}\boldsymbol{V} &=& \frac{d}{dt}\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{e} + \boldsymbol{e}^T\boldsymbol{P}\frac{d}{dt}\boldsymbol{e}\\ &=& \boldsymbol{e}^T(\boldsymbol{A}^T\boldsymbol{P}+\boldsymbol{P}\boldsymbol{A})\boldsymbol{e} + 2\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}\,[\boldsymbol{w}+\boldsymbol{\eta}]\\ &=& -\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} + 2\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}\,[\boldsymbol{w}+\boldsymbol{\eta}]\end{array}$$

With the modified control law, for those $\boldsymbol{e}$ that satisfy $|\boldsymbol{B}^T\boldsymbol{P}\boldsymbol{e}| > \boldsymbol{\delta}$ we have as before

$$\begin{array}{rcl}\frac{d}{dt}\boldsymbol{V} &=& -\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} + 2\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}\,[\boldsymbol{w}+\boldsymbol{\eta}]\\ &=& -\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} + 2\boldsymbol{z}\,[\boldsymbol{w}+\boldsymbol{\eta}]\\ &<& 2\boldsymbol{z}\,[-\boldsymbol{\rho}(\boldsymbol{e})\,\mathrm{sign}(\boldsymbol{z}) + \boldsymbol{\eta}]\\ &=& -2|\boldsymbol{z}|\,[\boldsymbol{\rho}(\boldsymbol{e}) - \boldsymbol{\eta}\,\mathrm{sign}(\boldsymbol{z})]\\ &\le& -2|\boldsymbol{z}|\,\boxed{[\boldsymbol{\rho}(\boldsymbol{e}) - |\boldsymbol{\eta}|]} \le \boldsymbol{0}\end{array}$$

while for those $\boldsymbol{e}$ that satisfy $|\boldsymbol{z}| = |\boldsymbol{B}^T\boldsymbol{P}\boldsymbol{e}| \le \boldsymbol{\delta}$ we have

$$\begin{array}{rcl}\frac{d}{dt}\boldsymbol{V} &=& -\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} + 2\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{B}\,[\boldsymbol{w}+\boldsymbol{\eta}]\\ &=& -\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} + 2\boldsymbol{z}\left[-\frac{1}{\boldsymbol{\delta}}\boldsymbol{\rho}(\boldsymbol{e})\,\boldsymbol{z} + \boldsymbol{\eta}\right]\\ &=& -\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} + 2\left[-\frac{1}{\boldsymbol{\delta}}\boldsymbol{\rho}(\boldsymbol{e})\,|\boldsymbol{z}|^2 + \boldsymbol{z}\,\boldsymbol{\eta}\right]\\ &\le& -\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} + 2\left[-\frac{1}{\boldsymbol{\delta}}\boldsymbol{\rho}(\boldsymbol{e})\,|\boldsymbol{z}|^2 + |\boldsymbol{z}|\,|\boldsymbol{\eta}|\right]\\ &\le& -\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} + 2\left[-\frac{1}{\boldsymbol{\delta}}\boldsymbol{\rho}(\boldsymbol{e})\,|\boldsymbol{z}|^2 + \boldsymbol{\rho}(\boldsymbol{e})|\boldsymbol{z}|\right]\\ &\le& -\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} + 2\boldsymbol{\rho}(\boldsymbol{e})\qquad \underbrace{\left[-\frac{1}{\boldsymbol{\delta}}\,|\boldsymbol{z}|^2 + |\boldsymbol{z}|\right]}_{\boxed{\text{What is the maximum of this expression?}}}\\ &\le& -\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} + 2\boldsymbol{\rho}(\boldsymbol{e})\qquad \underbrace{\left[-\frac{1}{\boldsymbol{\delta}}\left|\frac{\boldsymbol{\delta}}{2}\right|^2 + \left|\frac{\boldsymbol{\delta}}{2}\right|\right]}_{\boxed{\text{the maximum is at } \boldsymbol{z} = \frac{\boldsymbol{\delta}}{2}}}\end{array}$$

$$= \quad -e^T Q e + 2\rho(e)\,\frac{\delta}{4} = -e^T Q e + \rho(e)\,\frac{\delta}{2}$$

Hence, for those e that satisfies $|z| = |B^T P e| \leq \delta$ we have

$$\tfrac{d}{dt} V \leq -e^T Q e + \rho(e)\,\frac{\delta}{2} \leq 0$$

provided

$$\Big[\|Q\|\Big] \|e\|^2 \geq e^T Q e > \frac{\delta}{2}\rho(e)$$

with

$$\frac{\delta}{2}\rho(e) \geq \frac{\delta/2}{1-\alpha}\Big[\gamma_1 \|e\| + \gamma_2 \|e\|^2 + \gamma_3\Big]$$

Let us summarize the results:

- To avoid chattering, the next modification is suggested

$$w = \begin{cases} -\rho(e)\,\mathrm{sign}(B^T P e), & \text{if } |B^T P e| > \delta \\ -\dfrac{1}{\delta}\,\rho(e)\, B^T P e, & \text{if } |B^T P e| \leq \delta \end{cases}$$

- With such modification, we cannot prove asymptotic stability of the closed-loop system :(
- But we can show that the value of the Lyapunov function is decreasing everywhere, except some vicinity of the origin of the error dynamics.
- The size of this set can be modified, and depends on δ. Moreover, it can be made arbitrary small by reducing δ.
- Such property is called **ultimate boundedness**.

Example:

Consider a smooth target reference signal $q_d(t)$ and the system

$$M\,\ddot{q} = u, \quad M = 1 + \varepsilon$$

Suppose our nominal (estimated) value for M is 1 and so it has the error

$$\triangle M = -\varepsilon$$

where the value of ε is not known but it is reasonable to assume that we know an interval of values to which it must belong.

The system is linear, so that the transforms $u \to v \to w$ are

$$\begin{aligned} u &= (M + \triangle M)v = v \\ v &= \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t)) + w \end{aligned}$$

If for instance, $K_p = K_d = 1$, then the transformed system is

$$(1+\varepsilon)\ddot{q} = \ddot{q}_d(t) - (q - q_d(t)) - (\dot{q} - \dot{q}_d(t)) + w$$

or

$$(\ddot{q} - \ddot{q}_d(t)) + (\dot{q} - \dot{q}_d(t)) + (q - q_d(t)) = w + \eta(\cdot)$$

with

$$\eta = \frac{\varepsilon}{1+\varepsilon}\left[\ddot{q}_d(t) - (q - q_d(t)) - (\dot{q} - \dot{q}_d(t)) + w\right]$$

To proceed with robust design we need

- To rewrite the system

  $$(\ddot{q} - \ddot{q}_d(t)) + (\dot{q} - \dot{q}_d(t)) + (q - q_d(t)) = w + \eta(\cdot)$$

  in a state-space form

  $$\tfrac{d}{dt}e = Ae + B\left[w + \eta(\cdot)\right], \quad e = \begin{bmatrix} q - q_d \\ \dot{q} - \dot{q}_d \end{bmatrix}$$

- To solve the Lyapunov equation

  $$A^T P + PA = -Q, \qquad \text{with, e.g.,} \quad Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

- To find a range of ε such that the function

  $$\eta = \frac{\varepsilon}{1+\varepsilon}\left[\ddot{q}_d(t) - (q - q_d(t)) - (\dot{q} - \dot{q}_d(t)) + w\right]$$

  satisfies the bound

  $$\|\eta\| \le \alpha\,\|w\| + \gamma_1\,\|e\| + \gamma_2\,\|e\|^2 + \gamma_3, \quad \alpha < 1$$

Clearly, we can take

$$\gamma_1 = K_p + K_d = 2, \quad \gamma_2 = 0, \quad \alpha = \sup_{\varepsilon} \frac{\varepsilon}{1+\varepsilon}, \quad \gamma_3 = \alpha \sup_{t \ge 0} |\ddot{q}_d(t)|$$

6.3.4 Adaptive Control Based on Feedback Linearization

Let us consider now an alternative approach to the robust design, known as adaptive design. Here, instead of introducing another control input, we derive the so-called adaptation law that defines how to change on-line our guesses for the systems' parameters.

Given a mechanical system

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u}$$

and the desired trajectory $\boldsymbol{q} = \boldsymbol{q}_d(\boldsymbol{t})$, introduce the controller

$$\begin{array}{rcl} \boldsymbol{u} & = & \hat{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{v} + \hat{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \hat{\boldsymbol{g}}(\boldsymbol{q}) \\ \boldsymbol{v} & = & \ddot{\boldsymbol{q}}_d(\boldsymbol{t}) - \boldsymbol{K}_p(\boldsymbol{q} - \boldsymbol{q}_d(\boldsymbol{t})) - \boldsymbol{K}_d(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_d(\boldsymbol{t})) \end{array}$$

where in opposite to the robust design:

- The elements of $\hat{\boldsymbol{M}}(\cdot)$, $\hat{\boldsymbol{C}}(\cdot)$, $\hat{\boldsymbol{g}}(\cdot)$ are not fixed but changing with time in an appropriate way and they are variables to tune, while
- There is no additional control input, i.e. $\boldsymbol{w} = \boldsymbol{0}$!

Let us find the dynamical equations for updating values of $\hat{\boldsymbol{M}}(\cdot)$, $\hat{\boldsymbol{C}}(\cdot)$, $\hat{\boldsymbol{g}}(\cdot)$ provided that we measure $\boldsymbol{q}(\boldsymbol{t})$, $\dot{\boldsymbol{q}}(\boldsymbol{t})$, and $\ddot{\boldsymbol{q}}(\boldsymbol{t})$.

Let us rewrite the system

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u}, \quad \boldsymbol{u} = \hat{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{v} + \hat{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \hat{\boldsymbol{g}}(\boldsymbol{q})$$

$$\boldsymbol{v} = \ddot{\boldsymbol{q}}_d(\boldsymbol{t}) - \boldsymbol{K}_p(\boldsymbol{q} - \boldsymbol{q}_d(\boldsymbol{t})) - \boldsymbol{K}_d(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_d(\boldsymbol{t}))$$

as follows

$$\hat{\boldsymbol{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \Big[\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) - \hat{\boldsymbol{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}}\Big] = \boldsymbol{u}$$

$$\boldsymbol{u} = \hat{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{v} + \hat{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \hat{\boldsymbol{g}}(\boldsymbol{q})$$

or

$$\begin{array}{rcl} \hat{\boldsymbol{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} & = & \hat{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{v} + \Big[\Big(\hat{\boldsymbol{M}}(\boldsymbol{q}) - \boldsymbol{M}(\boldsymbol{q})\Big)\ddot{\boldsymbol{q}} + \Big(\hat{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}}) - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\Big)\dot{\boldsymbol{q}} \\ & & \quad + \hat{\boldsymbol{g}}(\boldsymbol{q}) - \boldsymbol{g}(\boldsymbol{q})\Big] \\ & = & \hat{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{v} + \boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})\Big[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\Big] \end{array}$$

where

- $\boldsymbol{\theta}$ is the vector of true (unknown) parameters of the model,

- $\hat{\boldsymbol{\theta}}$ is the vector of current estimates of the parameters,

- $\boldsymbol{Y}(\cdot)$ is the vector function, values of which we can compute measuring $\boldsymbol{q}(t)$, $\dot{\boldsymbol{q}}(t)$, $\ddot{\boldsymbol{q}}(t)$, $\boldsymbol{q}_d(t)$ and its derivatives, i.e.

$$\begin{aligned} \boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})\,\boldsymbol{\theta} &= \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) \\ \boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})\,\hat{\boldsymbol{\theta}} &= \hat{\boldsymbol{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \hat{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \hat{\boldsymbol{g}}(\boldsymbol{q}) \end{aligned}$$

Our closed-loop system is

$$\ddot{\boldsymbol{q}} = \boldsymbol{v} + \hat{\boldsymbol{M}}(\boldsymbol{q})^{-1}\boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})\left[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right]$$

and with the PD part of the feedback linearization design we have

$$(\ddot{\boldsymbol{q}} - \ddot{\boldsymbol{q}}_d) + \boldsymbol{K}_d(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_d) + \boldsymbol{K}_p(\boldsymbol{q} - \boldsymbol{q}_d) = \hat{\boldsymbol{M}}^{-1}(\boldsymbol{q})\,\boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})\left[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right]$$

where there are variables that we can change! How to do this update?

To answer this question, let us rewrite the system in the state-space form

$$\tfrac{d}{dt}\boldsymbol{e} = \underbrace{\begin{bmatrix} 0 & \boldsymbol{I} \\ -\boldsymbol{K}_p & -\boldsymbol{K}_d \end{bmatrix}}_{=\boldsymbol{A}}\boldsymbol{e} + \underbrace{\begin{bmatrix} 0 \\ \boldsymbol{I} \end{bmatrix}}_{=\boldsymbol{B}}\boldsymbol{\Phi}\left[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right], \quad \boldsymbol{e} = \begin{bmatrix} \boldsymbol{q} - \boldsymbol{q}_d(t) \\ \dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_d(t) \end{bmatrix}$$

where

$$\boldsymbol{\Phi} = \boldsymbol{\Phi}(\hat{\boldsymbol{\theta}},\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}}) = \hat{\boldsymbol{M}}^{-1}(\boldsymbol{q})\,\boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})$$

Now we need to choose the update law for $\hat{\boldsymbol{\theta}}$ in

$$\tfrac{d}{dt}\boldsymbol{e} = \boldsymbol{A}\boldsymbol{e} + \boldsymbol{B}\boldsymbol{\Phi}\left[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right]$$

to ensure that $\boldsymbol{e} \to \boldsymbol{0}$. Note that since $\boldsymbol{\theta}$ is unknown, we can not just take $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}$ but since it is constant, we might be able to introduce an integral action to attenuate its impact.

Let us take the solution of the Lyapunov equation

$$\boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} = -\boldsymbol{Q} < 0, \quad \boldsymbol{P} > 0$$

and consider the Lyapunov function candidate

$$\boldsymbol{V} = \boldsymbol{e}^T\boldsymbol{P}\boldsymbol{e} + \tfrac{1}{2}\left[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right]^T\boldsymbol{\Gamma}\left[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right], \quad \boldsymbol{\Gamma} > 0$$

Then

$$\begin{aligned}
\tfrac{d}{dt}V &= \tfrac{d}{dt}\left\{e^T P e\right\} + \tfrac{d}{dt}\left\{\tfrac{1}{2}\left[\hat{\theta}-\theta\right]^T \Gamma \left[\hat{\theta}-\theta\right]\right\} \\
&= \left[Ae + B\Phi\left[\hat{\theta}-\theta\right]\right]^T Pe + e^T P\left[Ae + B\Phi\left[\hat{\theta}-\theta\right]\right] \\
&\quad + \tfrac{1}{2}\left[\tfrac{d}{dt}\hat{\theta} - 0\right]^T \Gamma\left[\hat{\theta}-\theta\right] + \tfrac{1}{2}\left[\hat{\theta}-\theta\right]^T \Gamma\left[\tfrac{d}{dt}\hat{\theta} - 0\right] \\
&= e^T\left[A^T P + PA\right]e + \left[\hat{\theta}-\theta\right]^T\left\{\Phi^T B^T P e + \Gamma \tfrac{d}{dt}\hat{\theta}\right\} \\
&= -e^T Q e + \left[\hat{\theta}-\theta\right]^T \underbrace{\left\{\Phi^T B^T P e + \Gamma \tfrac{d}{dt}\hat{\theta}\right\}}_{=0} \le 0
\end{aligned}$$

With the proposed update law for $\hat{\theta}$ the closed-loop system is

$$\tfrac{d}{dt}e = Ae + B\Phi\left[\theta - \hat{\theta}\right], \quad \tfrac{d}{dt}\hat{\theta} = -\Gamma^{-1}\Phi^T B^T P e$$

The inequality

$$\tfrac{d}{dt}V = \tfrac{d}{dt}\left[e^T P e + \tfrac{1}{2}\left[\hat{\theta}-\theta\right]^T \Gamma\left[\hat{\theta}-\theta\right]\right] = -e^T Q e$$

implies that (thanks to Barbalat lemma)

$e(t) \to 0$ as $t \to +\infty$ and $(\hat{\theta}(t) - \theta)$ is bounded

However, some difficulty may occur trying to implement the adaptive control law above. Note that it is necessary

- To measure (or accurately estimate) $\ddot{q}$ for computing the regressor;
- To ensure that the matrix $\hat{M}(q)$ at each time moment is invertible.

6.4 Passivity Based Motion Control

Let us discuss now how to redesign the feedback controller above avoiding the need of computing $\ddot{q}$. We will use the passivity property.

6.4.1 Passivity Based Control Under No Uncertainty

Given a trajectory $q = q_d(t)$, consider the system

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = u$$

with the controller

$$\boldsymbol{u} = \boldsymbol{M}(\boldsymbol{q})\boldsymbol{a} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{b} + \boldsymbol{g}(\boldsymbol{q}) - \boldsymbol{K}\boldsymbol{r}, \quad \boldsymbol{K} = \operatorname{diag}\{\boldsymbol{k}_1,\ldots,\boldsymbol{k}_n\}$$

where the variables $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{r}$ are to be chosen.

Note that for feedback linearization we have used the controller above with

$$\boldsymbol{a} = \ddot{\boldsymbol{q}}_d - \boldsymbol{K}_d(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_d) - \boldsymbol{K}_p(\boldsymbol{q} - \boldsymbol{q}_d), \qquad \boldsymbol{b} = \dot{\boldsymbol{q}}, \qquad \boldsymbol{r} = \boldsymbol{0}$$

to obtain a linear closed-loop system. Here, we will suggest another choice of $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{r}$ that will lead to a nonlinear closed-loop system of special structure, which would allow us to suggest an energy-like Lyapunov function candidate for analysis.

If we substitute the controller, we obtain

$$\boldsymbol{M}(\boldsymbol{q})\left[\ddot{\boldsymbol{q}} - \boldsymbol{a}\right] + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\left[\dot{\boldsymbol{q}} - \boldsymbol{b}\right] + \boldsymbol{K}\boldsymbol{r} = \boldsymbol{0}$$

Let us impose the following relations among $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{r}$

$$\tfrac{d}{dt}\boldsymbol{r} = \ddot{\boldsymbol{q}} - \boldsymbol{a}, \qquad \boldsymbol{r} = \dot{\boldsymbol{q}} - \boldsymbol{b} \qquad \Rightarrow \qquad \tfrac{d}{dt}\boldsymbol{b} = \boldsymbol{a}$$

and take

$$\boldsymbol{b} = \dot{\boldsymbol{q}}_d(t) - \boldsymbol{\Lambda}\left(\boldsymbol{q} - \boldsymbol{q}_d(t)\right)$$

with $\boldsymbol{\Lambda} = \operatorname{diag}\{\boldsymbol{\lambda}_1,\ldots,\boldsymbol{\lambda}_n\} > \boldsymbol{0}$, so that

$$\boldsymbol{r} = \dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_d(t) + \boldsymbol{\Lambda}\left(\boldsymbol{q} - \boldsymbol{q}_d(t)\right),$$

$$\boldsymbol{a} = \ddot{\boldsymbol{q}}_d(t) - \boldsymbol{\Lambda}\left(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_d(t)\right).$$

Clearly, $\quad \boldsymbol{q} = \boldsymbol{q}_d(t) \quad \Rightarrow \quad \boldsymbol{b} = \dot{\boldsymbol{q}}_d(t) \quad$ and $\quad \boldsymbol{a} = \ddot{\boldsymbol{q}}_d(t)$
while $\quad \boldsymbol{r} \equiv \boldsymbol{0} \quad \Rightarrow \quad \boldsymbol{q} - \boldsymbol{q}_d(t) \to \boldsymbol{0}$ (exponentially) as $\boldsymbol{t} \to \boldsymbol{\infty}$.

Then, the closed-loop system can be written as

$$\boldsymbol{M}(\boldsymbol{q})\tfrac{d}{dt}\boldsymbol{r} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{r} + \boldsymbol{K}\boldsymbol{r} = \boldsymbol{0}$$

and we would like to show that $\boldsymbol{r} \to \boldsymbol{0}$.

Consider the Lyapunov function candidate

$$\boldsymbol{V}(\tilde{\boldsymbol{q}},\boldsymbol{r}) = \tfrac{1}{2}\boldsymbol{r}^T\boldsymbol{M}(\tilde{\boldsymbol{q}} + \boldsymbol{q}_d)\boldsymbol{r} + \tilde{\boldsymbol{q}}^T\boldsymbol{K}\boldsymbol{\Lambda}\tilde{\boldsymbol{q}}$$

where $\tilde{\boldsymbol{q}} = \boldsymbol{q} - \boldsymbol{q}_d$.

Computing its time derivative along the solutions we obtain

$$\begin{aligned}
\frac{d}{dt}V &= \frac{d}{dt}\left[\frac{1}{2}r^T M(q) r\right] + \frac{d}{dt}\left[\tilde{q}^T K\Lambda\tilde{q}\right] \\
&= r^T M(q)\frac{d}{dt}[r] + \frac{1}{2}r^T\frac{d}{dt}\left[M(q)\right] r + 2\tilde{q}^T K\Lambda\frac{d}{dt}\tilde{q} \\
&= r^T \left[-C(q,\dot{q})r - Kr\right] + \frac{1}{2}r^T\frac{d}{dt}\left[M(q)\right] r + +2\tilde{q}^T K\Lambda\frac{d}{dt}\tilde{q} \\
&= -r^T K r + \underbrace{r^T\left[\frac{1}{2}\frac{d}{dt}M(q) - C(q,\dot{q})\right] r}_{=0} + 2\tilde{q}^T K\Lambda\frac{d}{dt}\tilde{q} \\
&= -r^T K r + 2\tilde{q}^T K\Lambda\tilde{q}
\end{aligned}$$

where we can substitute

$$r = \dot{q} - \dot{q}_d + \Lambda\left(q - q_d\right) = \frac{d}{dt}\tilde{q} + \Lambda\tilde{q}$$

with $\tilde{q} = q - q_d$ to obtain

$$\begin{aligned}
\frac{d}{dt}V &= -\left(\frac{d}{dt}\tilde{q} + \Lambda\tilde{q}\right)^T K\left(\frac{d}{dt}\tilde{q} + \Lambda\tilde{q}\right) + 2\tilde{q}^T K\Lambda\frac{d}{dt}\tilde{q} \\
&= -\left[\frac{d}{dt}\tilde{q}\right]^T K\frac{d}{dt}\tilde{q} - \tilde{q}^T\Lambda^T K\Lambda\tilde{q} < 0
\end{aligned}$$

Finally, for the time-dependent closed-loop system

$$M(\tilde{q} + q_d(t))\frac{d}{dt}r + C(\tilde{q} + q_d(t), \dot{\tilde{q}} + \dot{q}_d(t))r + Kr = 0$$

with state variables $\tilde{q}$ and r we have positive definite

$$V = \frac{1}{2}r^T M(q) r + \tilde{q}^T K\Lambda\tilde{q} > 0$$

and negative definite

$$\frac{d}{dt}V = -\left(r - \Lambda\tilde{q})\right]^T K\left(r - \Lambda\tilde{q}\right) - \tilde{q}^T\Lambda^T K\Lambda\tilde{q} < 0$$

Global asymptotic stability follows as well as $\tilde{q} \to 0$ and $r \to 0$.

6.4.2 Adaptive Passivity Based Motion Control

Let us transform a feedback controller above into an adaptive one that can be used whenever the parameters are unknown.

Given a trajectory $q = q_d(t)$, consider the system

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = u$$

with the controller

$$u = \hat{M}(q)a + \hat{C}(q,\dot{q})b + \hat{g}(q) - Kr, \quad K = \text{diag}\left\{k_1, \ldots, k_n\right\}$$

with the variables $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{r}$ defined as above by

$$\boldsymbol{a} = \ddot{\boldsymbol{q}}_d - \boldsymbol{\Lambda}\left(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_d\right), \quad \boldsymbol{b} = \dot{\boldsymbol{q}}_d - \boldsymbol{\Lambda}\left(\boldsymbol{q} - \boldsymbol{q}_d\right), \quad \boldsymbol{r} = \dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_d + \boldsymbol{\Lambda}\left(\boldsymbol{q} - \boldsymbol{q}_d\right)$$

The closed-loop system can be rewritten as

$$\begin{aligned} &\boldsymbol{M}(\boldsymbol{q})\left[\ddot{\boldsymbol{q}} - \boldsymbol{a}\right] + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\left[\dot{\boldsymbol{q}} - \boldsymbol{b}\right] + \boldsymbol{K}\boldsymbol{r} \\ &\quad = \left[\hat{\boldsymbol{M}}(\boldsymbol{q}) - \boldsymbol{M}(\boldsymbol{q})\right]\boldsymbol{a} + \left[\hat{\boldsymbol{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\right]\boldsymbol{b} + \left[\hat{\boldsymbol{g}}(\boldsymbol{p}) - \boldsymbol{g}(\boldsymbol{p})\right] \\ &\quad = \boldsymbol{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{a}, \boldsymbol{b})\left[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right] \end{aligned}$$

which leads to

$$\boldsymbol{M}(\boldsymbol{q})\tfrac{d}{dt}\boldsymbol{r} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{r} + \boldsymbol{K}\boldsymbol{r} = \boldsymbol{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{a}, \boldsymbol{b})\left[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right]$$

To find the update law for $\hat{\boldsymbol{\theta}}$-variable we will use the Lyapunov function candidate

$$\boldsymbol{V} = \tfrac{1}{2}\boldsymbol{r}^T\boldsymbol{M}(\boldsymbol{q})\boldsymbol{r} + \left(\boldsymbol{q} - \boldsymbol{q}_d(t)\right)^T\boldsymbol{K}\boldsymbol{\Lambda}\left(\boldsymbol{q} - \boldsymbol{q}_d(t)\right) + \tfrac{1}{2}\left[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right]^T\boldsymbol{\Gamma}\left[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right]$$

Its time-derivative along a solution of the system is

$$\tfrac{d}{dt}\boldsymbol{V} = -\dot{\tilde{\boldsymbol{q}}}^T\boldsymbol{K}\dot{\tilde{\boldsymbol{q}}} - \tilde{\boldsymbol{q}}^T\boldsymbol{\Lambda}^T\boldsymbol{K}\boldsymbol{\Lambda}\tilde{\boldsymbol{q}} + \left[\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\right]^T\underbrace{\left\{\boldsymbol{\Gamma}\tfrac{d}{dt}\hat{\boldsymbol{\theta}} + \boldsymbol{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{a}, \boldsymbol{b})^T\boldsymbol{r}\right\}}_{=\,0} \leq 0$$

where we have used the following adaptation law

$$\tfrac{d}{dt}\hat{\boldsymbol{\theta}} = -\boldsymbol{\Gamma}^{-1}\boldsymbol{Y}^T(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{a}, \boldsymbol{b})\boldsymbol{r}$$

Similar conclusions about the global stability and convergence to zero of the tracking error can be drown as in the case of feedback-linearization-based adaptive design. However, here we have no need to accurately estimate $\ddot{\boldsymbol{q}}$.

Note that in a similar way as with feedback linearization, a robust controller can be designed based on passivity.

To conclude, we remark that the control laws derived in this chapter are based on the nonlinear dynamical model developed earlier in the form of Euler-Lagrange equations. To the best of our knowledge they are still considered to be too advanced and are not implemented in the currently available on the market industrial robotic manipulators.

Let us now discuss a few approaches that are less complex for implementation. They are to be derived with less rigor and are to be based on various simplifications of the model that might be unacceptable for a particular task to be performed by the robotic manipulator.

Chapter 7

Independent Joint Control

There are several methods that can be used to design feedback control laws making the robots follow a desired working scenario. Let us discuss first the steps that are necessary before using any of the advanced control designs such as the ones described in Chapter 6. After that, we will discuss a few related issues and briefly mention a less mathematically justified approach to feedback control design.

7.1 Development of Control Systems for Robots

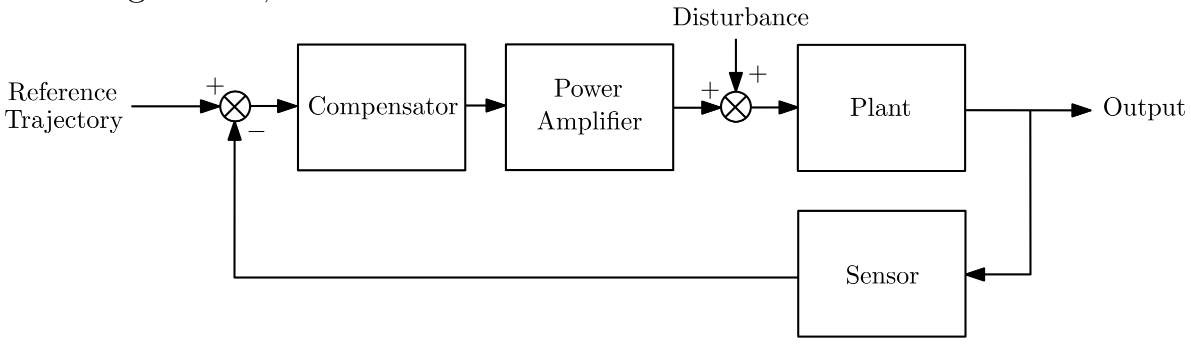
In order to apply any of the advanced methods of Chapter 6, one can proceed as follows.

- Compute appropriate mechanical models for describing possible behavior of the robot, in particular, describe
  - Kinematics (see, e.g., Chapter 2, 3) and use it to specify a scenario in terms of desired time-evolutions of the generalized coordinates such as DH-variables (see, e.g., Chapter 4);
  - Dynamics (see, e.g., Chapter 5);
- Estimate or/and compute from CAD drawings parameters of the kinematic and dynamic models. Here one can figure out necessary specifications of the system based on the dynamical description such as necessary for implementation ranges for joint velocities and accelerations as well as required magnitudes of the generalized forces produced by the actuators.
- Either choose (if possible) actuators and gears according to the specifications above or modify the trajectories designed based on kinematics to make them feasible according to the dynamical model.

- Either choose (if possible) sensors and their allocation or make appropriate models for sensors and actuators according to available specifications to improve the dynamical model for verification of achievable performance via simulation.
- Choose a control architecture, taking into account
  - Possibility to use linear vs. nonlinear methods;
  - Implementation of compensation for friction, delays, lack of velocity measurements, and other features that are not adequately described by the available model;
  - Anti-Windup Methods if integrators are to be used in the control law;
  - Adaptive Mechanisms for On-line Parameters Estimation;
  - Robustness …
- Tune parameters of a controller with a chosen architecture via simulation and experimental study. Here, additional parameter identification is typically required.

7.2 Conceptual Control Loop for a Single Joint

Basic structure of a feedback control system, which is often appropriate for controlling robots, is illustrated below.

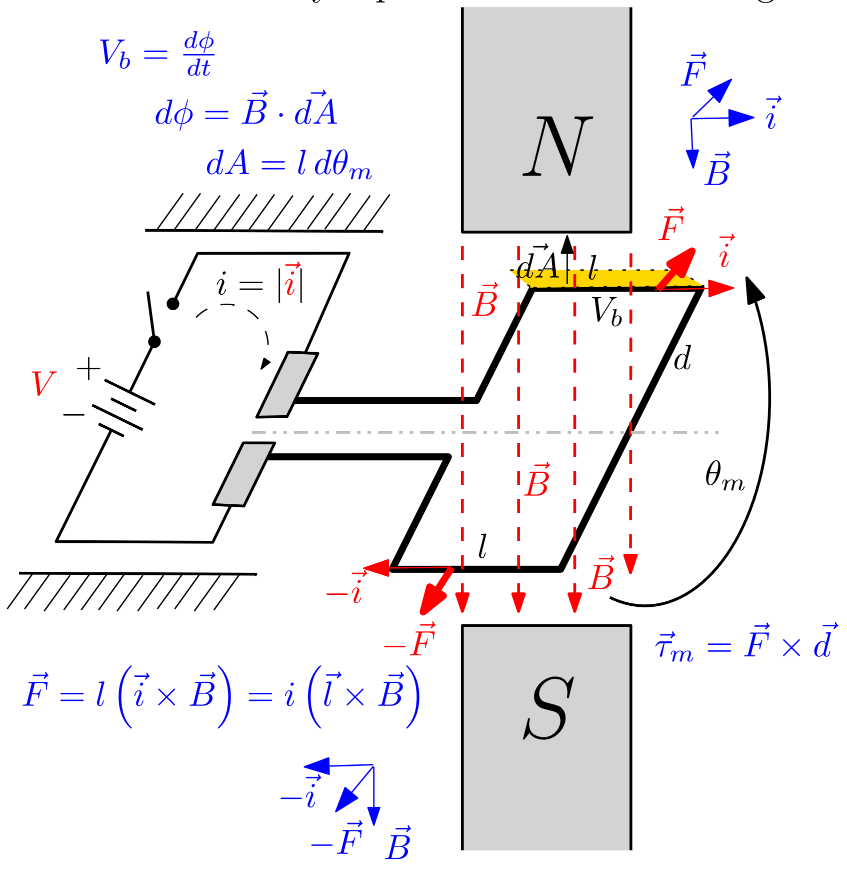


Here the chosen control law is denoted by the "Compensator" block, which, clearly, is driven by the tracking error signal of the single joint variable and produces an output driving the amplifier for the corresponding actuator. The influence of dynamics describing the other DOF is ignored. Such structure often fails in the case when precise or comparably fast motions are targeted. However, sometimes it is sufficient.

7.3 Actuator Dynamics

Let us now briefly discuss slightly more accurate way to improve our model taking into account dynamics of actuators than the modification suggested in the beginning of Chapter 6. More precisely, let us remind the basic principle of operation of a permanent magnet DC motor, assuming that our actuators are of this type.

The motor is schematically represented in the next figure.



Our motor creates a magnetic field $\vec{B}$ characterized by a magnetic flux $\vec{\phi}$ and a current $\vec{i}$ in a conductor. A straight current-carrying conductor of length l in the magnetic field experience a Lorentz force

$$\boxed{\vec{F} = l\left(\vec{i} \times \vec{B}\right)}$$

and, due to an appropriate bearing, starts rotating producing a torque $\vec{\tau}_m$ applied to the mechanical frame.

The magnitude of the the torque is proportional to the magnitude of the created force. In particular, for a rectangle $l \times d$

$$\boxed{\vec{\tau}_m = \vec{F} \times \vec{d}} \qquad \Rightarrow \qquad \boxed{\tau_m = |\vec{\tau}_m| = \left(l\, d\, |\vec{B}|\right) |\vec{i}\,| = K_m\, i}$$

At the same time, whenever a conductor moves in a magnetic field $\vec{B}$, the voltage V_b is induced according to the Faraday's law of induction:

$$V_b = \frac{d\phi}{dt} = \vec{B} \cdot \frac{d\vec{A}}{dt} = \vec{B} \cdot \left(\frac{d\vec{\theta}_m}{dt} \times \vec{l} \right) = K_b \, \tfrac{d}{dt}\theta_m$$

With $\vec{B} = \text{const}$, a sufficiently large number of coil windings N etc., we can relate angular velocity of the rotor with induced voltage by a back emf constant $K_b = N\, l \left|\vec{B}\right|$, which is inverse of the so-called speed constant.

Alternatively, the same relation can be derived from consideration of transformation of electrical power into the mechanical one: The external force causes motion so that mechanical work is done

$$W = \int_{\theta_m(t_1)}^{\theta_m(t_2)} \tau_m \, d\theta_m = \int_{t_1}^{t_2} \tau_m \, \tfrac{d}{dt}\theta_m \, dt \;\; = \sum_{\text{number of coils}} \int_{t_1}^{t_2} V_b(t)\, i(t)\, dt = E$$

which must be equal to a loss of electrical energy in the circuit explaining the induced voltage V_b (back emf) opposite to the current flow, and, therefore $\tau_m \, \tfrac{d}{dt}\theta_m = (K_m \, i) \, \tfrac{d}{dt}\theta_m = \left(K_m \, \tfrac{d}{dt}\theta_m\right) i = N \, V_b \, i$ leading to a similar relation. Of course, for other structures, the constants should be computed according to the particular geometry of the coils.

Now, assuming certain electrical inductance L and resistance R in the armature, we have the following relation between the armature current $i = |\vec{i}|$ and the input voltage V

$$L \left(\tfrac{d}{dt} i\right) + R\, i = V - V_b$$

where, under assumption of constant flux $|\vec{\phi}| = \text{const}$,

$$V_b = K_b \, \tfrac{d}{dt}\theta_m$$

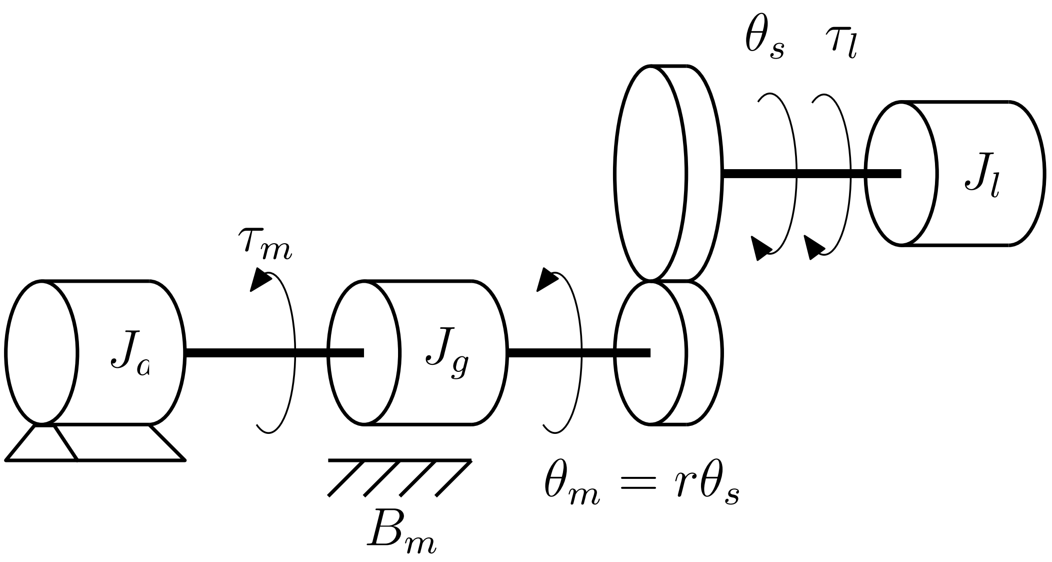
and the produced torque to the mechanical part is

$$\tau_m = K_m \, i$$

Let us use this expression to combine the obtained dynamical equation with a mechanical model, that is taken for simplicity as a single DOF system.

7.4 Dynamical Model for a Single Joint

Consider the next model of a single link with actuator/gear transmission.



In terms of the motor angle $\boldsymbol{\theta_m}$ the equation of motion is

$$\boldsymbol{J_m\ddot{\theta}_m = \tau_m + F_m - B_m\dot{\theta}_m,} \qquad \boldsymbol{\tau_m = K_m\, i,} \quad \boldsymbol{F_m = -\tfrac{1}{r}\,\tau_l,}$$

where $\boldsymbol{J_m = J_a + J_g}$ is the combined moment of inertia for the motor and the gear, $\boldsymbol{B_m}$ is the viscous friction gain, $\boldsymbol{\tau_l}$ and $\boldsymbol{F_m}$ are the reaction forces applied to the link and to the rotor of the motor respectively (note that they have opposite signs and the ratio of their magnitudes is the gear ratio $\boldsymbol{r}$).

Now, one should augment the mechanical model with the electrical one derived above

$$\boldsymbol{L\left(\tfrac{d}{dt}i\right) + R\,i = V - K_b\left(\tfrac{d}{dt}\theta_m\right),}$$

where $\boldsymbol{V}$ is the control input to be assigned.

At the same time, we have

$$\boldsymbol{J_l\ddot{\theta}_s + B_s\dot{\theta}_s = \tau_l + F_s,}$$

where $\boldsymbol{F_s}$ is typically highly nonlinear and can represent the sum of various forces acting on the link, such as gravity or interaction forces with other links of the manipulator. This force can, in general, depend not only on $\boldsymbol{\theta_s}$ but also on the other generalized coordinates as well as on their first and second derivatives.

To proceed with feedback control design one can:

- Assume that the gear box is flexible and so $\boldsymbol{\theta_m \neq r\,\theta_s}$. One of the commonly used assumption is that $\boldsymbol{\tau_l}$ is proportional to $\boldsymbol{(\theta_m - r\,\theta_s)}$. We will consider this situation at the end of this chapter as well as in Chapter 8, where we will derive the so-called dynamical model with flexible joints and use the ideas of geometric nonlinear feedback control theory for design.

- Assume that $\boldsymbol{L \ll R}$ so that $\boldsymbol{L/R \approx 0}$, so that

  $$\mathbf{0}\,\left(\tfrac{d}{dt}\boldsymbol{i}\right)+\boldsymbol{i}=\frac{\mathbf{1}}{\boldsymbol{R}}\boldsymbol{V}-\frac{\mathbf{1}}{\boldsymbol{R}}\boldsymbol{K_b}\left(\tfrac{d}{dt}\boldsymbol{\theta_m}\right) \qquad \Rightarrow \qquad \boldsymbol{\tau_m}=\boldsymbol{u}-\boldsymbol{B_e}\left(\tfrac{d}{dt}\boldsymbol{\theta_m}\right),$$

  where $\boldsymbol{u}=\frac{\boldsymbol{K_m}}{\boldsymbol{R}}\boldsymbol{V}$ and $\boldsymbol{B_e}=\frac{\boldsymbol{K_m}}{\boldsymbol{R}}\boldsymbol{K_b}$. This is, in essence, the assumption made during the derivation of the dynamic model in Chapter 6.

- Assume that the transmission through the gear box is perfectly rigid and there are no additional DOFs due to mechanical part of the motor-gear-transmission system, so that $\boldsymbol{\theta_m \equiv r\,\theta_s}$. Furthermore, one can

  – Assume that the motions are relatively slow and the gear ratio $\boldsymbol{r}$ is relatively high, and $\frac{\mathbf{1}}{\boldsymbol{r}}\boldsymbol{F_s}$ is relatively small with respect to the other terms and can be treated as a small bounded disturbance. The equation should be modified excluding $\boldsymbol{\tau_l}$ and $\boldsymbol{\theta_s}$ and arriving at a similar model with $\boldsymbol{F_s}$ in place of $\boldsymbol{\tau_l}$.

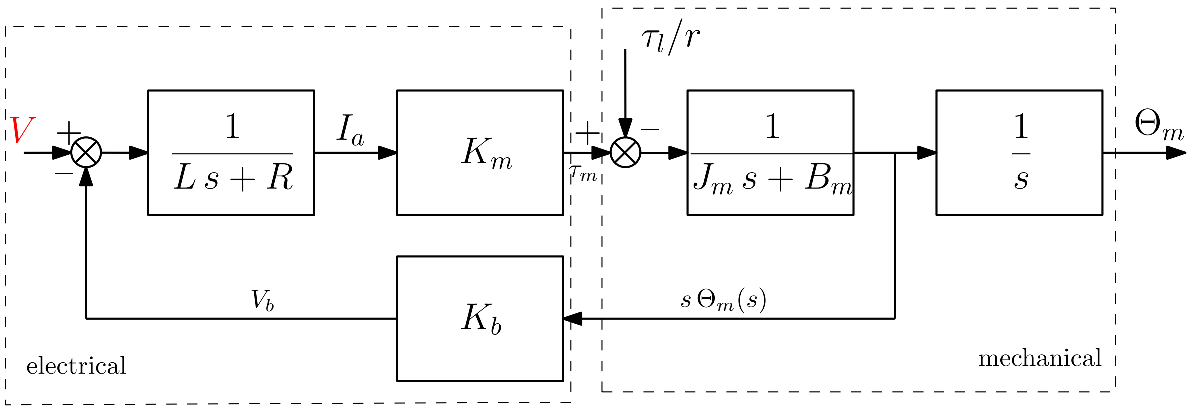
  – Assume that the motions are relatively slow and the gear ratio $\boldsymbol{r}$ is relatively high, and $\frac{\mathbf{1}}{\boldsymbol{r}}\boldsymbol{\tau_l}$ is relatively small with respect to the other terms and can be treated as a small bounded disturbance. We will proceed in this way.

So, for now, suppose that a reasonable combined models is the following

$$\begin{aligned}\boldsymbol{L}\left(\tfrac{d}{dt}\boldsymbol{i}\right)+\boldsymbol{R\,i} &= \boldsymbol{V}-\boldsymbol{K_b}\left(\tfrac{d}{dt}\boldsymbol{\theta_m}\right)\\ \boldsymbol{J_m}\left(\tfrac{d^2}{dt^2}\boldsymbol{\theta_m}\right)+\boldsymbol{B_m}\left(\tfrac{d}{dt}\boldsymbol{\theta_m}\right) &= \boldsymbol{K_m\,i}-\frac{\mathbf{1}}{\boldsymbol{r}}\boldsymbol{\tau_l}\end{aligned}$$

with $\boldsymbol{\tau_l/r}$ being a relatively small signal that can be treated as a small bounded disturbance.

The corresponding system is illustrated in the next figure.



Note that the system is linear! So, we can use classical linear feedback control design methods for it!

The transfer function from the input $V(s)$ to the output $\Theta_m(s)$ is

$$G_{\{V\to\theta\}}(s) = \frac{\Theta_m(s)}{V(s)} = \frac{K_m}{\left[(Ls+R)(J_m s + B_m) + K_m K_b\right] s}$$

while the transfer function from the disturbance $\tau_l(s)$ to the output $\Theta_m(s)$ is

$$G_{\{\tau_l\to\theta\}}(s) = \frac{\Theta_m(s)}{\tau_l(s)} = \frac{1}{r}\,\frac{-(Ls+R)}{\left[(Ls+R)(J_m s + B_m) + K_m K_b\right] s}$$

These transfer functions should be used to achieve certain performance specifications in the closed-loop system as well as a certain level of disturbance attenuation.

Note that in the case when $L \ll R$ so that $L/R \approx 0$ one often can base the design on the reduced transfer functions

$$G_{\{V\to\theta\}}(s) \approx \frac{K_m}{\left[R(J_m s + B_m) + K_m K_b\right] s}$$

and

$$G_{\{\tau_l\to\theta\}}(s) \approx -\frac{1}{r\left[(J_m s + B_m) + K_m K_b\right] s}$$

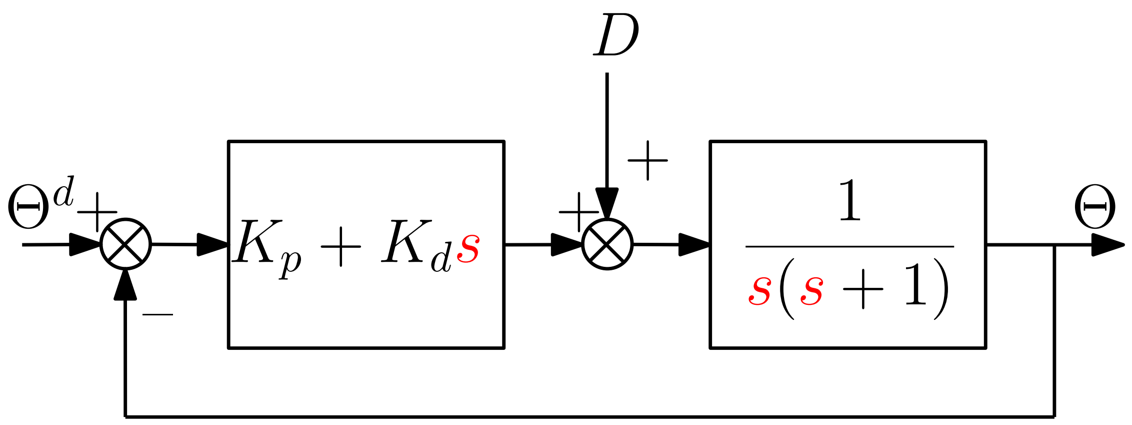
However, the correctness of reduction of the model (i.e. how small L/R should be) should be justified via e.g. **balanced model reduction** method.

7.5 PD and PID Tuning Based on a Simplified Linear Model

Suppose the gear ratio r is so large that $G_{\{\tau_l\to\theta\}} \approx 0$, $L \ll R$, and the physical parameters are such that

$$G_{\{v\to\theta\}}(s) \approx \frac{K_m}{\left[R(J_m s + B_m) + K_m K_b\right] s} = \frac{1}{(s+1)\, s}$$

Let us design a PD-controller, illustrated in the next diagram



such that the closed-loop poles are located at -3 and -1.
The transfer functions for the plant and for the controller are

$$G_{\{v\to\theta\}}(s) = \frac{B(s)}{A(s)} = \frac{1}{(s+1)\,s}, \qquad H(s) = \frac{P(s)}{L(s)} = \frac{K_p + K_d s}{1}$$

and so the pole placement is equivalent to solving the so-called Diophantine equation

$$\begin{aligned} A(s)\cdot L(s) + B(s)\cdot P(s) &= A_{clp}(s) \\ (s+1)s\cdot 1 + 1\cdot(K_p + K_d s) &= (s+3)(s+1) \\ s^2 + (K_d+1)s + K_p &= s^2 + 4s + 3 \end{aligned}$$

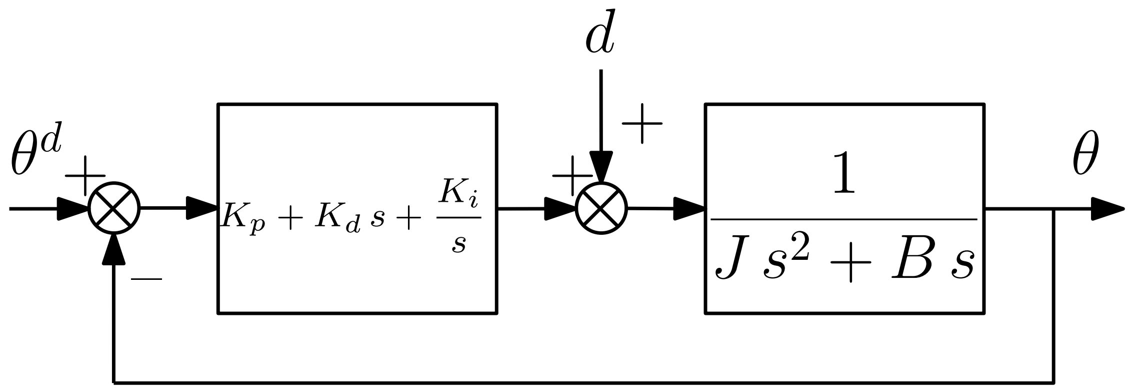
Hence, we should take

$$K_d = 3, \qquad K_p = 3$$

The reader should be able to easily solve the following exercise. Suppose

$$G_{\{V\to\theta\}}(s) \approx \frac{K_m}{[R(J_m s + B_m) + K_m K_b]\,s} = \frac{1}{Js^2 + Bs},$$

find a PID-controller gains such that that closed-loop system illustrated below



is stable.
However, in practice, stability is not sufficient and performance of the

closed-loop system should be reasonable in certain sense. There are several practical techniques for tuning the gains of a PID controller, such as the classical Ziegler-Nichols method.

Often, the gains of a PID-controller are renamed as follows

$$u = P + I + D, \quad P = K\, e(t), \quad I = \frac{K}{T_i}\int_0^\infty e(\tau)\, d\tau, \quad D \approx K\, T_d\, \dot{e}(t),$$

where $e = \theta_d - \theta$ is the tracking error, K is the proportional gain, $T_i > 0$ and $T_d \geq 0$ are the time constants characterizing the integral and differential parts, while D can be computed using an approximation for a differentiating filter or a velocity observer.

The most common sampled discrete-time realization of such a regulator assuming a zero-order-hold (ZOH) model of the digital-to-analog signal converter with the step $h > 0$ is

$$u(t) = u[k], \qquad k\, h \leq t < (k+1)\, h, \qquad k = 0, 1, 2, \ldots$$

where the digital feedback signal on the sampled measurement

$$e[k] = \theta_d(k\, h) - \theta(k\, h)$$

is computed as

$$u[k] = P[k] + I[k] + D[k] \qquad \text{with} \qquad P[k] = K\, e[k]$$

and using approximations by the forward Euler difference for the integral feedback

$$I[k+1] = I[k] + \frac{K\, h}{T_i}\, e[k], \qquad I[0] = 0, \qquad k = 1, 2, \ldots$$

and by the backward Euler difference for the derivative feedback based on the first order differential filter $\frac{T_d}{N}\, \dot{D} + D = -K\, T_d\, \dot{e}$ with $N \gg 1$ (usually, $10 \leq N \leq 100$) realized by

$$D[k] = \frac{T_d}{T_d + N\, h}\Big(D[k-1] - K\, N\, \big(e[k] - e[k-1]\big)\Big), \qquad D[0] = 0$$

for $k = 1, 2, \ldots$

Tuning of the gains can be done using Zigler-Nichols method based either on recorded step response or on periodic response to critical pure proportional feedback.

In the first approach, one record the output in response to $u(t) = 0$ for $t < 0$ and $u(t) = U =$ const for $t \geq 0$, where $U > 0$ should be chosen to obtain a monotonous transition from the initial value to another constant steady-state. After that, one finds on the graph of the response the point with the steepest slope and computes this slope R as well as the value L of the interception of the corresponding tangent line with the time axis.

After that one takes either

$$K = \frac{6}{5\,R\,L}, \qquad T_i = 2\,L, \qquad T_d = \frac{L}{2}$$

or

$$K = \frac{9}{10\,R\,L}, \qquad T_i = 3\,L, \qquad T_d = 0.$$

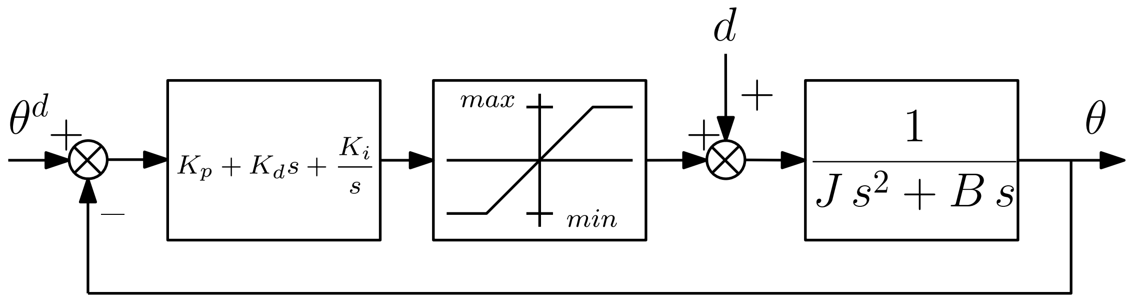
For the other approach one needs to perform a series of experiment with proportional feedback $u(t) = K_{test}\,(\theta_d(t) - \theta(t))$ under slowly changing reference signal $\theta_d(t)$. The coefficient K_{test} should be gradually increased until further increase would lead to instability. The corresponding extremal (critical) value of the proportional gain K_u should be recorded together with an estimate of the period of achieved oscillations T_u right before instability. Based on these values one takes either

$$K = \frac{3\,K_u}{5}, \qquad T_i = \frac{T_u}{2}, \qquad T_d = \frac{T_u}{8}$$

or

$$K = \frac{9\,K_u}{20}, \qquad T_i = \frac{5\,T_u}{6} \qquad T_d = 0.$$

It should be noticed that, often, one have also to address the problem of input saturation. The closed-loop system can be drastically different if we take into account limits on the signals in the loop:

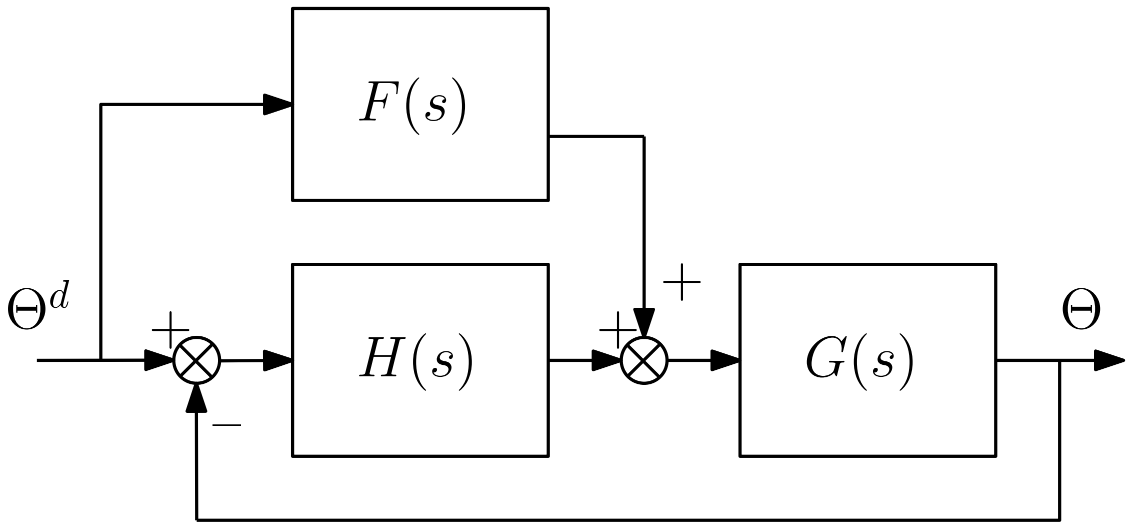


Here, various anti-windup scheme may be useful but in any case, performance of the closed-loop system should be verified taking such limitations into account.

Let us now briefly remind another useful technique from the standard linear control theory.

7.6 Simplified Feedforward Control Design

The following structure of the controller is often useful



Here,

- $\boldsymbol{G(s)}$ represents a simplified model of the control system for a particular link,
- $\boldsymbol{H(s)}$ could be a PID-like controller that is tuned as above,
- while the transfer function $\boldsymbol{F(s)}$ is so-called feedforward term such that
  - $\boldsymbol{F(s)}$ is stable;
  - $\boldsymbol{F(s)}$ is proper;

  A reasonable choice is $\boldsymbol{F(s) \approx 1/G(s)}$, for a particular range of frequencies, provided the plant model is minimum-phase, i.e. there are no unstable zeros as in the case of the transfer functions considered above.

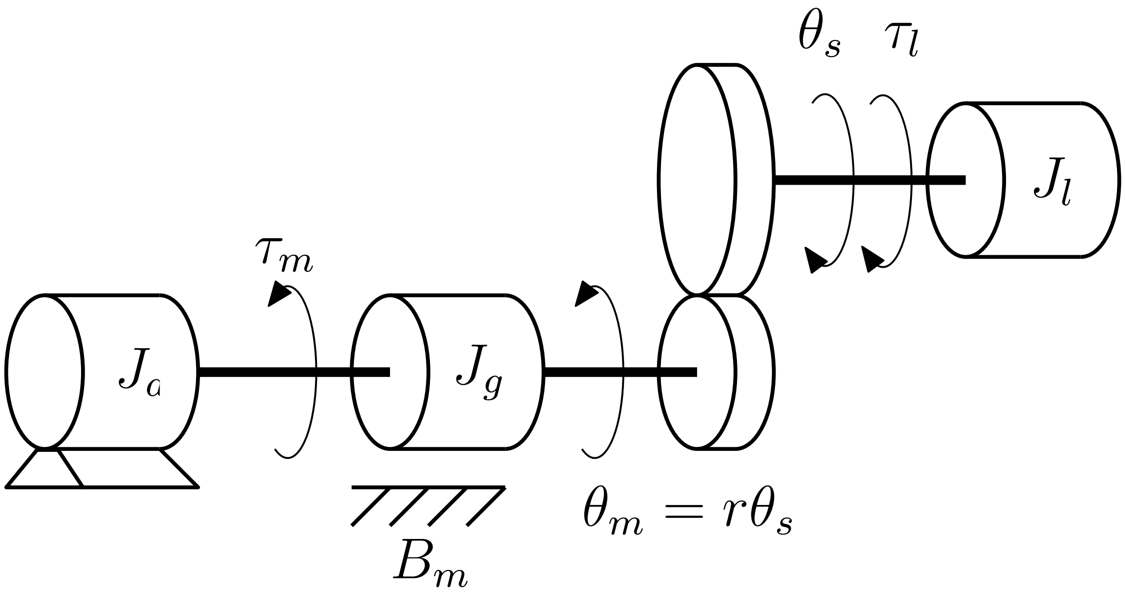
In the considered example with $\boldsymbol{G(s) = \frac{1}{J\,s^2 + B\,s}}$, one can implement the feedforward term $\boldsymbol{F(s)\,\Theta^d(s)}$ as $\boldsymbol{J\,\ddot{\theta}_d(t) + B\,\dot{\theta}_d(t)}$ avoiding approximate differentiation but using generation of not only the reference signal but its two required derivatives.

It is well-known that introduction of a feedforward term can significantly improve performance.

Let us now discuss another issue that may be important to take into account at the control design stage.

7.7 Dealing with Flexible Transmission

Let us re-consider our motor-transmission-gearbox system



but drop the often unrealistic rigidity assumption, so that $\boldsymbol{\theta_m \neq r\,\theta_s}$. Note that

- Gear-box always introduces additional friction and delay;
- Gear-box can complicate the dynamics and limit the performance.

Suppose now that we can ignore the electrical dynamics, i.e. that $\boldsymbol{L \ll R}$ so that $\boldsymbol{L/R \approx 0}$ and

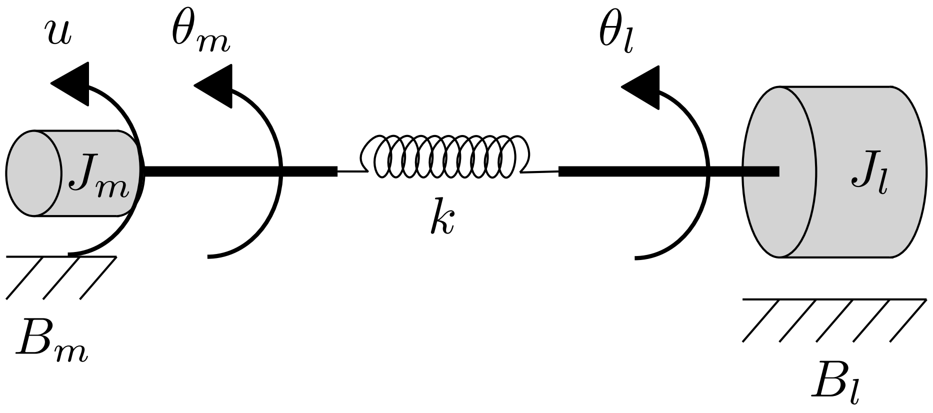
$$\boldsymbol{\tau_m = u - B_e\left(\tfrac{d}{dt}\theta_m\right)},$$

as above.

Let us also assume that the transmission, such as a harmonic drive, can be modeled as a torsional spring, so that

$$\boldsymbol{\tau_l = k\left(\frac{\theta_m}{r} - \theta_s\right)}$$

with some coefficient $\boldsymbol{k > 0}$, as illustrated on the next figure.



The dynamics of the system takes the following form

$$\begin{aligned} J_l\ddot{\theta}_s + B_l\dot{\theta}_s + k\left(\theta_s - \frac{\theta_m}{r}\right) &= F_s \\ J_m\ddot{\theta}_m + B_m\dot{\theta}_m - \frac{k}{r}\left(\theta_s - \frac{\theta_m}{r}\right) &= u \end{aligned}$$

where u is a control torque.

Now, suppose $F_s \approx 0$ and rename the variables as

$$\theta_l \leftarrow \theta_s, \qquad \theta_m \leftarrow \theta_m/r, \qquad u \leftarrow r^2\,u$$

and the constants as

$$J_m \leftarrow r^2\,J_m, \qquad B_m \leftarrow r^2\,B_m,$$

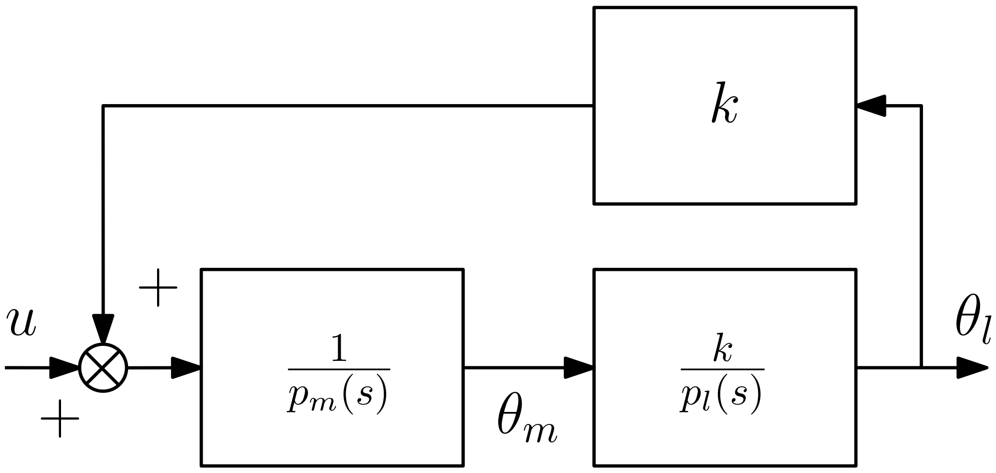
so that the motor related quantities are reflected through the reduction ratios. Then, we can rewrite dynamics as follows

$$\begin{aligned} J_l\ddot{\theta}_l + B_l\dot{\theta}_l + k\theta_l &= k\theta_m \\ J_m\ddot{\theta}_m + B_m\dot{\theta}_m + k\theta_m &= k\theta_l + u \end{aligned}$$

and compute the transfer functions relating the two variables and the control input:

$$\begin{aligned} \Theta_l(s) &= \frac{k}{p_l(s)}\Theta_m(s) = \frac{k}{J_l s^2 + B_l s + k}\Theta_m(s) \\ \Theta_m(s) &= \frac{1}{p_m(s)}\left(k\Theta_l(s) + U(s)\right) = \frac{k\Theta_l(s) + U(s)}{J_m s^2 + B_m s + k} \end{aligned}$$

The result is illustrated on the diagram below

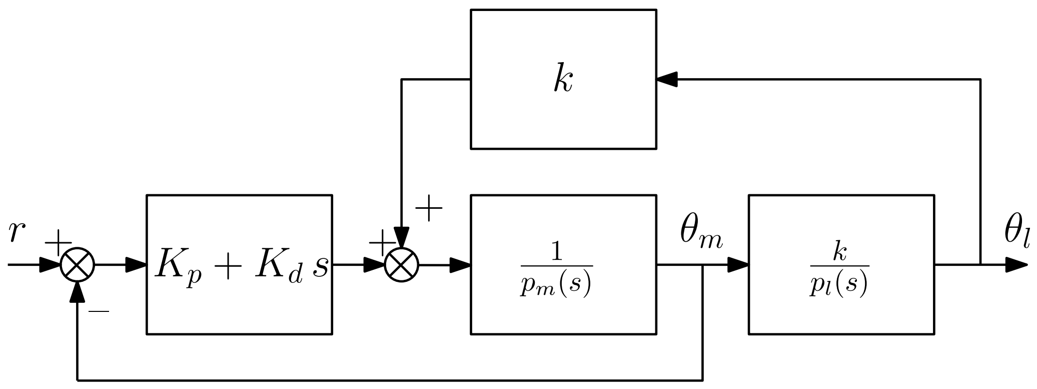


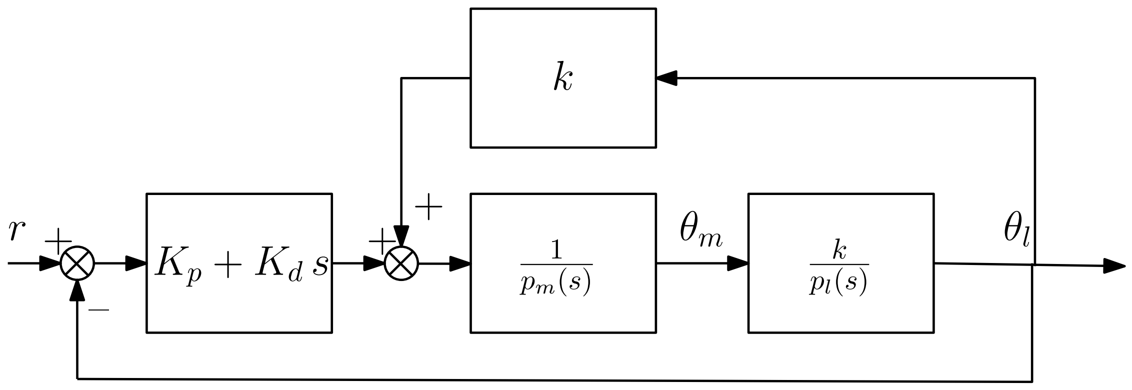
Combining the two expressions we obtain the transfer function from the input to the link position

$$\begin{aligned}\Theta_l(s) &= \frac{k}{p_l(s)p_m(s)-k^2}U(s)\\ &= \frac{k}{J_mJ_ls^4+(J_lB_m+J_mB_l)s^3+(k(J_m+J_l)+B_mB_l)s^2+k(B_m+B_l)s}U(s)\\ &\approx \frac{k}{J_mJ_ls^4+k(J_m+J_l)s^2}U(s)\end{aligned}$$

where the last approximation is valid when the friction coefficients are small.

Obviously, the designer might be able to implement PD controller with respect to either $\boldsymbol{\theta_m}$ or $\boldsymbol{\theta_l}$. The two possibilities are illustrated below

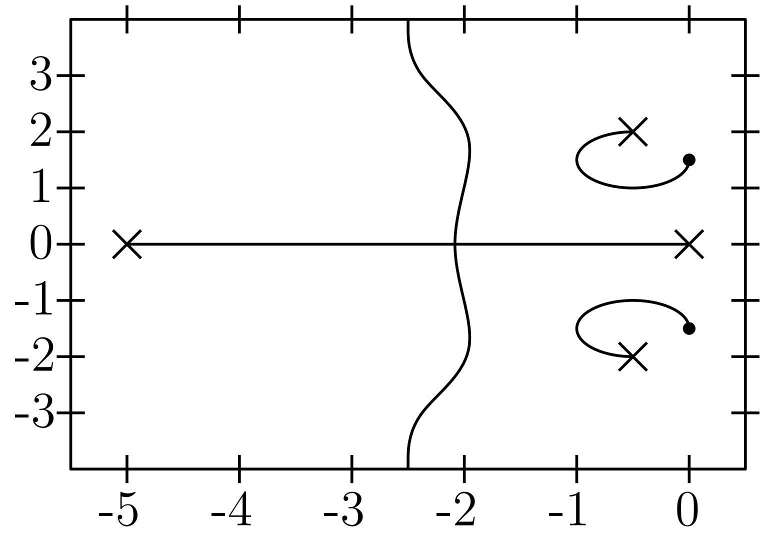




Which design is better? One of the differences can be observed from root-locus analysis. The root-locus of the closed-loop system with **$\boldsymbol{\theta_m}$-feedback**

$$U = -\,(K_P + K_D\, s)\;\Theta_m = K(a+s)\,\Theta_m$$

is shown below

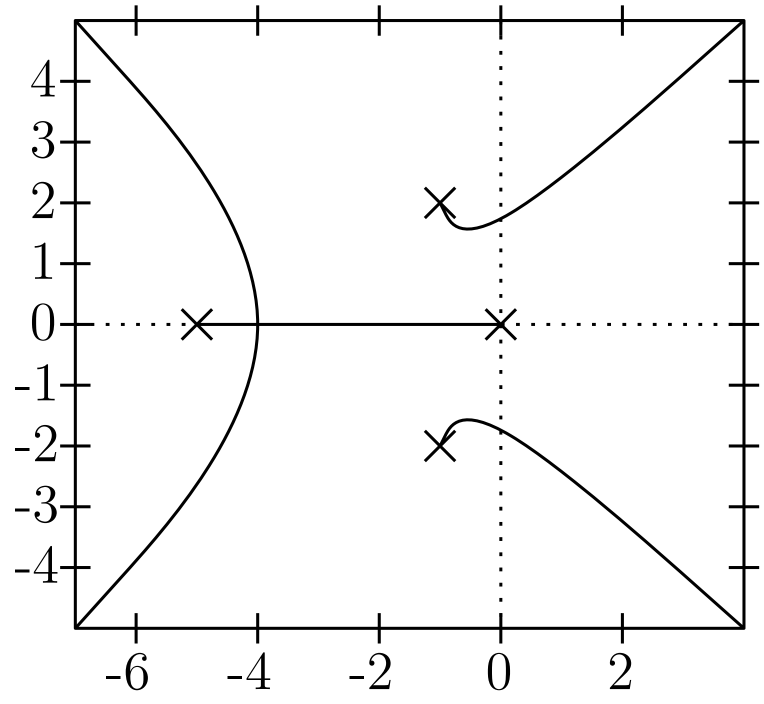


Note that stability is guaranteed for all (positive) values of the gains!

The root-locus of the closed-loop system with $\boldsymbol{\theta_l}$**-feedback**

$$\boldsymbol{U = -\,(K_P + K_D\,s)\;\Theta_l = K(a+s)\,\Theta_l}$$

is shown below



Note that stability is lost when the gains are large. In fact, it can be shown that the system becomes unstable when the proportional gain reaches the value of the coefficient of elasticity.

Chapter 8

Geometric Nonlinear Control

Here we would like to review a particular geometry-based technique for designing feedback control laws for highly nonlinear systems that cannot be linearized by trivial direct cancellation of nonlinearities.

This technique is useful for model-based robot control in case when the rigid model, derived in Chapter 5, does not describe dynamics adequately in relation to a particular class of target motions. In particular, this is the case when the number of DOF that must be taken into account is less than the number of independent control inputs. Such systems are called **underactuated** and cover the two examples that we have briefly mentioned above: the flexible joint case and the case when dynamics of the actuators can not be approximated by static relations.

8.1 Introduction to Differential Geometry

We start with an introduction of a few useful mathematical concepts.

8.1.1 Smooth Embedded Manifold in $\mathbb{R}^n$

From an intuitive point of view, a manifold is something that is locally equivalent to a usual finite-dimensional Euclidean space but may have a more complex global structure. There exists a rigorous mathematical definition involving such notions as topological distances, maps, coordinates, atlases, etc., which are inspired by topography. In what follows, we will give a somewhat simplified definition of what is typically called a smooth manifold embedded into an Euclidean space. This will be sufficient for our goals, while an interested reader may consult a standard textbook on differential geometry for a rigorous overview of the subject.

A smooth (nontrivial) manifold $\mathcal{M}$ of dimension m, with $m \geq 1$, is a subset of points in $\mathbb{R}^n$, with $n \geq 2$, defined as the zero set of a family of $p = n - m \geq 1$ smooth scalar functions

$$\begin{array}{rcl} h_1(x_1, x_2, \ldots, x_n) & = & 0 \\ & \vdots & \\ h_p(x_1, x_2, \ldots, x_n) & = & 0 \end{array}$$

such that the differentials

$$dh_1(\cdot), \quad dh_2(\cdot), \quad \ldots, \quad dh_p(\cdot)$$

are linearly independent for any point of $\mathcal{M}$.

At each point $x \in \mathcal{M}$ we can attach a **tangent space $T_x\mathcal{M}$**, which is an m-dimensional space that specifies the set of possible directions of velocities at this point implying not leaving $\mathcal{M}$. It can be associated with $\mathbb{R}^m$.

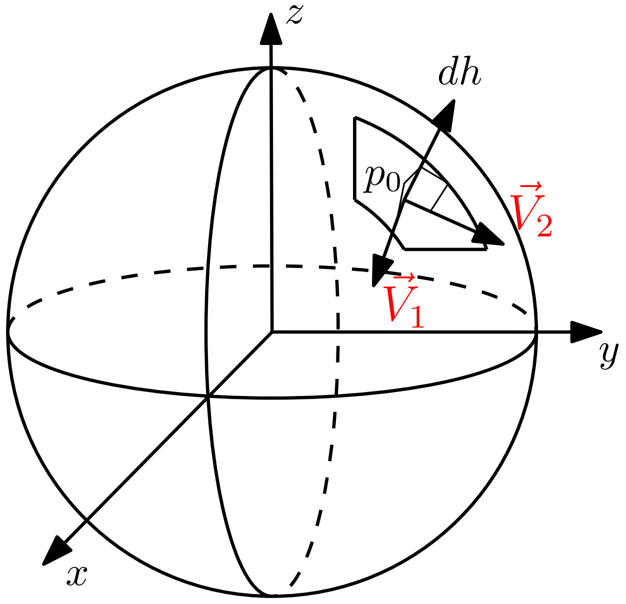
Let us illustrate these notions with an example.

Example: a sphere as a manifold

Consider a unit sphere S^2 in $\mathbb{R}^3$ defined by a single relation

$$h(x, y, z) = x^2 + y^2 + z^2 - 1 = 0$$

on the coordinates in a certain fixed frame and represented on the next figure together with a tangent space at one point.



Since $n = 3$ and there is a single relation, i.e. $p = 1$, the dimension of this manifold is $m = 2 = 3 - 1$.

The differential $dh(\cdot)$ of the function

$$h(x, y, z) = x^2 + y^2 + z^2 - 1 = 0$$

at a point $p_0 = (x_0, y_0, z_0) \in S^2$ is the row vector

$$\begin{aligned} dh(x_0, y_0, z_0) &= \left[\frac{\partial}{\partial x}h,\ \frac{\partial}{\partial y}h,\ \frac{\partial}{\partial z}h\right]\Big|_{x=x_0,\, y=y_0,\, z=z_0} \\ &= \left[2x_0,\ 2y_0,\ 2z_0\right], \end{aligned}$$

where, obviously, $x_0^2 + y_0^2 + z_0^2 = 1$, defines the direction orthogonal to the surface of the sphere at the particular point.

How to compute a tangent space $T_{p_0}S^2$
to the sphere S^2 at $p_0 = (x_0, y_0, z_0) \in S^2$?

By definition

$$\begin{aligned} T_{p_0}S^2 &= \{V = [v_1, v_2, v_3]^T :\ dh(p_0) \perp V\} \\ &= \left\{V = [v_1, v_2, v_3]^T :\ 2\,x_0\,v_1 + 2\,y_0\,v_2 + 2\,z_0\,v_3 = 0\right\} \end{aligned}$$

Suppose for the given $p_0 \in S^2$ we have $z_0 \neq 0$, than we can suggest the following two representative solutions for $V \in T_{p_0}S^2$:

$$V_1 = \left[1, 0, -x_0/z_0\right], \quad V_2 = \left[0, 1, -y_0/z_0\right]$$

that form a basis, i.e.

$$T_{p_0}S^2 = \text{span}\,\{V_1, V_2\} = \{V = \alpha\, V_1 + \beta\, V_2 :\quad \alpha, \beta \in \mathbb{R}\},$$

where here and below **span** $\{\cdot\}$ is defined as the set of all possible linear combinations.

8.1.2 Smooth Vector and Co-vector Fields

The following two notions are useful as well.

- **A smooth vector field** on a manifold $\mathcal{M}$ is a smooth function

$$x \in \mathcal{M} \ \rightarrow\ f(x) = \begin{bmatrix} f_1(x) \\ \vdots \\ f_m(x) \end{bmatrix} \in T_x\mathcal{M}$$

- **A smooth co-vector field** on a manifold $\mathcal{M}$ is a smooth function

$$x \in \mathcal{M} \ \rightarrow\ w(x) = [w_1(x), \cdots, w_m(x)] \in T_x^*\mathcal{M}$$

Here $\boldsymbol{T}^*_{\boldsymbol{x}}\boldsymbol{\mathcal{M}}$ is co-tangent space to the manifold $\boldsymbol{\mathcal{M}}$ at its point $\boldsymbol{x}$.

Every fixed co-vector field $\boldsymbol{w}(\boldsymbol{x})$ can be associated with a linear operator (functional) that associates with every arbitrary vector field $\boldsymbol{f}(\boldsymbol{x})$ a real number:

$$\boldsymbol{\mathcal{A}}_{w(x)} : \quad \boldsymbol{T}_x\boldsymbol{\mathcal{M}} \ni \boldsymbol{f}(\boldsymbol{x}) \mapsto \boldsymbol{\mathcal{A}}_{w(x)}\boldsymbol{f}(\boldsymbol{x}) \in \mathbb{R}$$

for every point in $\boldsymbol{\mathcal{M}}$, computed as the usual product of the row, representing $\boldsymbol{w}(\boldsymbol{x})$, and the column, representing $\boldsymbol{f}(\boldsymbol{x})$:

$$\begin{aligned}\boldsymbol{\mathcal{A}}_{w(x)}\boldsymbol{f}(\boldsymbol{x}) &= [\boldsymbol{w}_1(\boldsymbol{x}), \cdots, \boldsymbol{w}_m(\boldsymbol{x})]\begin{bmatrix}\boldsymbol{f}_1(\boldsymbol{x})\\ \vdots \\ \boldsymbol{f}_m(\boldsymbol{x})\end{bmatrix}\\ &= \boldsymbol{w}_1(\boldsymbol{x})\,\boldsymbol{f}_1(\boldsymbol{x}) + \ldots + \boldsymbol{w}_m(\boldsymbol{x})\,\boldsymbol{f}_m(\boldsymbol{x})\end{aligned}$$

8.1.3 Distributions and Co-distributions

Vector fields and co-vector fields can be grouped.

- Let $\boldsymbol{X}_1(\boldsymbol{x}), \ldots, \boldsymbol{X}_k(\boldsymbol{x})$ be vector fields on $\boldsymbol{\mathcal{M}}$ that are linearly independent for each $\boldsymbol{x} \in \boldsymbol{\mathcal{M}}$. Then, a **distribution** $\triangle(\cdot)$ is a family of linear subspaces of $\boldsymbol{T\mathcal{M}}$ defined for each point $\boldsymbol{x}$ as

$$\triangle(\boldsymbol{x}) = \mathbf{span}\{\boldsymbol{X}_1(\boldsymbol{x}), \ldots, \boldsymbol{X}_k(\boldsymbol{x})\}$$

- Let $\boldsymbol{W}_1(\boldsymbol{x}), \ldots, \boldsymbol{W}_k(\boldsymbol{x})$ be co-vector fields on $\boldsymbol{\mathcal{M}}$ linearly independent for each $\boldsymbol{x} \in \boldsymbol{\mathcal{M}}$. Then a **co-distribution** $\boldsymbol{\Omega}(\cdot)$ is a family of linear subspaces of $\boldsymbol{T}^*\boldsymbol{\mathcal{M}}$ defined for each point $\boldsymbol{x}$ as

$$\boldsymbol{\Omega}(\boldsymbol{x}) = \mathbf{span}\{\boldsymbol{W}_1(\boldsymbol{x}), \ldots, \boldsymbol{W}_k(\boldsymbol{x})\}$$

A vector field $\boldsymbol{f}(\boldsymbol{x})$ can be associated with a differential equation

$$\tfrac{d}{dt}\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x})$$

since by the very definition, whenever $\boldsymbol{x} \in \boldsymbol{\mathcal{M}}$, we have $\tfrac{d}{dt}\boldsymbol{x} \in \boldsymbol{T}_x\boldsymbol{\mathcal{M}}$.

More importantly, a vector field $\boldsymbol{f}(\boldsymbol{x})$, called the drift term, together with a distribution $\boldsymbol{\Omega}(\boldsymbol{x}) = \mathbf{span}\{\boldsymbol{g}_1(\boldsymbol{x}), \ldots, \boldsymbol{g}_m(\boldsymbol{x})\}$ can be associated with a control system defined by an infinite dimensional family of differential equations

$$\tfrac{d}{dt}\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}_1(\boldsymbol{x})\,\boldsymbol{u}_1 + \cdots + \boldsymbol{g}_m(\boldsymbol{x})\,\boldsymbol{u}_m,$$

where $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m$ are control inputs.

We are going to see that some properties of solutions for a control system, defined in this way, can be deduced from analysis of certain properties of the corresponding distribution $\boldsymbol{\Omega}(\boldsymbol{x})$ and the vector field $\boldsymbol{f}(\boldsymbol{x})$, which can be verified by some computation procedures related to the formal operations introduced next.

8.1.4 Lie Bracket of Vector Fields

Given two vector fields $\boldsymbol{f}(\cdot)$ and $\boldsymbol{g}(\cdot)$ on $\boldsymbol{\mathcal{M}} = \mathbb{R}^{n}$, the **Lie bracket** of $\boldsymbol{f}$ and $\boldsymbol{g}$ is the vector field in $\mathbb{R}^{n}$ denoted by $[\boldsymbol{f},\boldsymbol{g}]$ and defined for each $\boldsymbol{x} \in \boldsymbol{\mathcal{M}} = \mathbb{R}^{n}$ by

$$[\boldsymbol{f},\boldsymbol{g}](\boldsymbol{x}) := \left[\tfrac{\partial}{\partial \boldsymbol{x}}\boldsymbol{g}(\boldsymbol{x})\right]\boldsymbol{f}(\boldsymbol{x}) - \left[\tfrac{\partial}{\partial \boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x})\right]\boldsymbol{g}(\boldsymbol{x})$$

For example, suppose $\boldsymbol{f} = \boldsymbol{A}\boldsymbol{x}$ and $\boldsymbol{g} = \boldsymbol{B}\boldsymbol{x}$, then

$$\begin{aligned}[\boldsymbol{f},\boldsymbol{g}](\boldsymbol{x}) &= \left[\tfrac{\partial}{\partial \boldsymbol{x}}\boldsymbol{g}(\boldsymbol{x})\right]\boldsymbol{f}(\boldsymbol{x}) - \left[\tfrac{\partial}{\partial \boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x})\right]\boldsymbol{g}(\boldsymbol{x}) \\ &= \boldsymbol{B}\,\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}\,\boldsymbol{B}\boldsymbol{x} = \underbrace{(\boldsymbol{B}\boldsymbol{A} - \boldsymbol{A}\boldsymbol{B})}_{=\boldsymbol{C}}\boldsymbol{x}\end{aligned}$$

We will also use **repeated Lie brackets** defined recursively as follows

$$\begin{aligned}\boldsymbol{ad}_{\boldsymbol{f}}^{0}(\boldsymbol{g}) &= \boldsymbol{g} \\ \boldsymbol{ad}_{\boldsymbol{f}}^{1}(\boldsymbol{g}) &= [\boldsymbol{f},\boldsymbol{g}] \\ \boldsymbol{ad}_{\boldsymbol{f}}^{2}(\boldsymbol{g}) &= [\boldsymbol{f},[\boldsymbol{f},\boldsymbol{g}]\,] \\ &\vdots \\ \boldsymbol{ad}_{\boldsymbol{f}}^{k}(\boldsymbol{g}) &= \left[\boldsymbol{f},\boldsymbol{ad}_{\boldsymbol{f}}^{(k-1)}(\boldsymbol{g})\right]\end{aligned}$$

For example, suppose for given $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{b} \in \mathbb{R}^{n}$ we define $\boldsymbol{f} = \boldsymbol{A}\boldsymbol{x}$, $\boldsymbol{g} = \boldsymbol{b}$, and a control system $\frac{d}{dt}\boldsymbol{x} = \boldsymbol{A}\,\boldsymbol{x} + \boldsymbol{b}\,\boldsymbol{u}$. Then, $\boldsymbol{ad}_{\boldsymbol{f}}^{0}(\boldsymbol{g}) = \boldsymbol{b}$, $\boldsymbol{ad}_{\boldsymbol{f}}^{1}(\boldsymbol{g}) = -\boldsymbol{A}\boldsymbol{b}$, $\boldsymbol{ad}_{\boldsymbol{f}}^{2}(\boldsymbol{g}) = \boldsymbol{A}^{2}\boldsymbol{b}, \ldots$, $\boldsymbol{ad}_{\boldsymbol{f}}^{n}(\boldsymbol{g}) = (-1)^{n}\boldsymbol{A}^{n}\boldsymbol{b}$, and the rank of the matrix formed by them is the controllability degree.

8.1.5 Lie Derivative

Given

- a scalar function $\boldsymbol{h}(\cdot)$ on $\mathbb{R}^{n}$, i.e. $\boldsymbol{h} : \mathbb{R}^{n} \to \mathbb{R}$ and
- a vector field $\boldsymbol{f}(\cdot)$ on $\mathbb{R}^{n}$, i.e. $\boldsymbol{f} : \mathbb{R}^{n} \to \mathbb{R}^{n}$,

the **Lie derivative** of $h(\cdot)$ with respect to $f(\cdot)$ is the scalar function denoted as $\mathcal{L}_f h$ and defined by

$$\mathcal{L}_f h = \left[\tfrac{\partial}{\partial x} h(x)\right] f(x) = \left[\tfrac{\partial}{\partial x_1} h(x)\right] f_1(x) + \cdots + \left[\tfrac{\partial}{\partial x_n} h(x)\right] f_n(x)$$

Lie derivative $\mathcal{L}_f h$, called also directional derivative, defines the rate of change of a scalar function h of a vector argument x along solutions of the differential equation associated with the vector field $f(x)$. This is exactly derivative that we compute differentiating a Lyapunov function.

We will also use **repeated Lie derivatives** defined recursively as follows

$$\mathcal{L}_f^k h = \mathcal{L}_f \left[\mathcal{L}_f^{(k-1)} h\right], \quad k = 1, 2 \ldots, \quad \mathcal{L}_f^0 h = h$$

The following technical result can be proved via direct computation.

Lemma: Let $h : \mathbb{R}^n \to \mathbb{R}$ be a scalar function and f, g be vector fields on $\mathbb{R}^n$. Then the following identity holds

$$\mathcal{L}_{[f,g]} h \equiv \mathcal{L}_f \left[\mathcal{L}_g h\right] - \mathcal{L}_g \left[\mathcal{L}_f h\right]$$

This will be used later.

Equipped with some basic notions from differential geometry, we can approach some problems of finding solutions of differential equations. This will be used later to define a systematic procedure of feedback control design.

8.2 Frobenius Theorem

Here we will introduce a very useful result known in differential geometry as Frobenius Theorem.

8.2.1 Manifolds as Solutions of PDEs

Consider the following system of two partial differential equations (PDEs)

$$\frac{\partial z}{\partial x} = f(x, y, z), \qquad \frac{\partial z}{\partial y} = g(x, y, z)$$

Suppose, we are looking for a solution of this system of equations, i.e. we would like to express z in the form

$$z = \phi(x, y)$$

such that

$$\frac{\partial}{\partial x}\phi(x,y) = f(x,y,\phi(x,y)), \quad \frac{\partial}{\partial y}\phi(x,y) = g(x,y,\phi(x,y))$$

If we find a solution, then it obviously corresponds to a surface in $\mathbb{R}^3$

$$\mathcal{M} = \{(x,\ y,\ z) \in \mathbb{R}^3 : \quad z - \phi(x,y) = 0\}$$

in other words, to a **2**-dimensional manifold.

How to compute a tangent plane to this surface (manifold)?

8.2.2 Tangent Space to a Solution of a PDE

The differential $dh(\cdot)$ of the function

$$h(x,y,z) = z - \phi(x,y) = 0$$

at the point $p_0 = (x_0, y_0, z_0) \in \mathbb{R}^2$ is the row vector

$$\begin{aligned} dh(p_0) &= \left[\tfrac{\partial}{\partial x}h,\ \tfrac{\partial}{\partial y}h,\ \tfrac{\partial}{\partial z}h\right]\Big|_{x=x_0,\,y=y_0,\,z=z_0} \\ &= \left[-\tfrac{\partial}{\partial x}\phi(x,y),\ -\tfrac{\partial}{\partial y}\phi(x,y),\ 1\right]\Big|_{x=x_0,\,y=y_0,\,z=z_0} \\ &= \left[-f(x_0,y_0,\phi(x_0,y_0)),\ -g(x_0,y_0,\phi(x_0,y_0)),\ 1\right] \end{aligned}$$

Consequently, the tangent space to the manifold $\mathcal{M}$ of the solution of the system of PDEs defined above can be computed as follows

$$\begin{aligned} T_{p_0}\mathcal{M} &= \{V = [v_1, v_2, v_3]^T : dh(p_0) \perp V\} \\ &= \left\{V : \ -f(\cdot)v_1 - g(\cdot)v_2 + 1 \cdot v_3 = 0\right\} \end{aligned}$$

A basis for $T_{p_0}\mathcal{M}$ can be taken as

$$V_1 = [1,\ 0,\ f(\cdot)]^T \qquad \text{and} \qquad V_2 = [0,\ 1,\ g(\cdot)]^T$$

so that

$$T_{p_0}\mathcal{M} = \operatorname{span}\{V_1, V_2\}$$

We see now that the functions $f(\cdot)$ and $g(\cdot)$ appearing in the right-hand sides of the system of PDEs unambiguously and directly define the tangent space to the surface representing a solution. In other words, they define the solution geometrically.

8.2.3 Integral Manifolds

The following conclusion can be drawn:

A search for a

solution $\boldsymbol{z} = \boldsymbol{\phi(x, y)}$ of a system of two partial differential equations

$$\frac{\partial}{\partial x}\boldsymbol{z} = \boldsymbol{f(x, y, z)}, \qquad \frac{\partial}{\partial y}\boldsymbol{z} = \boldsymbol{g(x, y, z)}$$

is equivalent to a search for a

surface $\boldsymbol{\mathcal{M}}$ in $\mathbb{R}^3$ whose tangent space $\boldsymbol{T_p\mathcal{M}}$ is spanned by $\boldsymbol{V_1} = \left[\mathbf{1},\ \mathbf{0},\ \boldsymbol{f(x, y, z)}\right]^{\boldsymbol{T}}$ and $\boldsymbol{V_2} = \left[\mathbf{0},\ \mathbf{1},\ \boldsymbol{g(x, y, z)}\right]^{\boldsymbol{T}}$

If such surface $\boldsymbol{\mathcal{M}}$ can be found then it is called an **integral manifold** of the system while the equations are called **integrable**.

8.2.4 Lie Derivative Representation for PDEs

The problem of finding an integral manifold for a system of **2** PDEs can be rewritten as a problem of finding a function, two Lie derivatives of which vanish simultaneously. These derivatives should be taken along the two vectors computed above.

In other words, the problem of finding a solution $\boldsymbol{z} = \boldsymbol{\phi(x, y)}$ of equations

$$\frac{\partial}{\partial x}\boldsymbol{z} = \boldsymbol{f(x, y, z)}, \qquad \frac{\partial}{\partial y}\boldsymbol{z} = \boldsymbol{g(x, y, z)}$$

can be rewritten as the problem of finding a solution $\boldsymbol{h} = \boldsymbol{h(x, y, z)}$, in the particular form $\boldsymbol{h(x, y, z)} = \boldsymbol{z} - \boldsymbol{\phi(x, y)}$, of the following two equations

$$\boldsymbol{\mathcal{L}_{V_1} h} = \mathbf{0}, \qquad \boldsymbol{\mathcal{L}_{V_2} h} = \mathbf{0}$$

where

$$\boldsymbol{V_1} = \left[\mathbf{1},\ \mathbf{0},\ \boldsymbol{f(x, y, z)}\right]^{\boldsymbol{T}} \quad \text{and} \quad \boldsymbol{V_2} = \left[\mathbf{0},\ \mathbf{1},\ \boldsymbol{g(x, y, z)}\right]^{\boldsymbol{T}}$$

Indeed, by definition, we have

$$\begin{aligned}
\boldsymbol{\mathcal{L}_{V_1} h} &= \left[\frac{\partial}{\partial x}\boldsymbol{h(x, y, z)}\right]\cdot\mathbf{1} + \left[\frac{\partial}{\partial y}\boldsymbol{h(x, y, z)}\right]\cdot\mathbf{0} + \left[\frac{\partial}{\partial z}\boldsymbol{h(x, y, z)}\right]\boldsymbol{f(x, y, z)} \\
&= \frac{\partial}{\partial x}\boldsymbol{h(x, y, z)} + \frac{\partial}{\partial z}\boldsymbol{h(x, y, z)}\,\boldsymbol{f(x, y, z)} \\
&= \frac{\partial}{\partial x}\left(\boldsymbol{z} - \boldsymbol{\phi(x, y)}\right) + \frac{\partial}{\partial z}\left(\boldsymbol{z} - \boldsymbol{\phi(x, y)}\right)\boldsymbol{f(x, y, z)} \\
&= -\frac{\partial}{\partial x}\boldsymbol{\phi(x, y)} + \boldsymbol{f(x, y, z)}
\end{aligned}$$

and

$$\begin{aligned}\mathcal{L}_{V_2}h &= \left[\tfrac{\partial}{\partial x}h(x,y,z)\right]\cdot 0 + \left[\tfrac{\partial}{\partial y}h(x,y,z)\right]\cdot 1 + \left[\tfrac{\partial}{\partial z}h(x,y,z)\right]g(x,y,z)\\ &= \tfrac{\partial}{\partial y}h(x,y,z) + \tfrac{\partial}{\partial z}h(x,y,z)\,g(x,y,z)\\ &= \tfrac{\partial}{\partial y}\big(z-\phi(x,y)\big) + \tfrac{\partial}{\partial z}\big(z-\phi(x,y)\big)\,g(x,y,z)\\ &= -\tfrac{\partial}{\partial y}\phi(x,y) + g(x,y,z)\end{aligned}$$

so that equivalence of the two systems is obvious.

It is now time to notice that the fact that

$$\mathcal{L}_{V_1}h = 0 \qquad \text{and} \qquad \mathcal{L}_{V_2}h = 0$$

immediately implies that

$$\mathcal{L}_{V}h = 0 \qquad \forall\, V \in \triangle = \operatorname{span}\{V_1, V_2\}$$

i.e. h does not change along any direction from a distribution spanned by V_1 and V_2.

We are going to claim next that existence of an integral manifold can be verified algebraically by checking a particular property of this distribution through computation of some repeated Lie brackets.

8.2.5 Integrability and Involutivity for Distributions

Let us introduce two more notions.

- The distribution

$$\triangle = \operatorname{span}\{X_1(x), X_2(x), \dots, X_m(x)\}, \qquad x \in \mathbb{R}^n$$

  is said to be **completely integrable**, if there are functions $h_1(x)$, ..., $h_{n-m}(x)$ such that

  - they are linearly independent for any $x \in \mathbb{R}^n$;
  - they satisfy the system of partial differential equations

$$\mathcal{L}_{X_i}h_j = 0, \qquad \forall\, i \in \{1,\dots,m\},\ \forall\, j \in \{1,\dots,(n-m)\}$$

- The distribution $\triangle$ is said to be **involutive**, if there are scalar functions $\alpha_{ijk} : \mathbb{R}^n \to \mathbb{R}$ such that

$$[X_i, X_j] = \sum_{k=1}^{n} \alpha_{ijk} X_k \qquad \forall\, i,\, j$$

8.2.6 Frobenius Theorem: integrability $\Leftrightarrow$ involutivity

We are ready to formulate the main result that we need from differential geometry.

A distribution $\triangle$ on $\mathbb{R}^n$ is integrable

$$\Updownarrow$$

A distribution $\triangle$ on $\mathbb{R}^n$ is involutive

This statements means that in order to verify existence of a solution for a system of partial differential equations, one can simply compute some Lie brackets and check that they can be expressed as linear combinations of certain vectors.

Note however, that a procedure for finding a solution is not clear from this theorem. Let us show next how this theorem can be used for feedback control design.

8.3 Feedback Linearization

For a given nonlinear control system, it is often not clear whether nonlinearities can be canceled via a feedback transformation. However, possibility for such cancellation is a useful structural property that may allow bringing powerful tools of linear control theory for nonlinear control design. Let us formulate the problem of finding an appropriate transformation that makes a nonlinear control system equivalent to a linear one.

8.3.1 Concept of Feedback Linearization

The control system

$$\tfrac{d}{dt}\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u}, \quad \boldsymbol{x} \in \mathbb{R}^n,\ \boldsymbol{u} \in \mathbb{R}^1,\ \boldsymbol{f}(0) = 0$$

is said to be feedback linearizable if there are

- an invertible change of coordinates $\boldsymbol{y} = \boldsymbol{T}(\boldsymbol{x})$ and
- a feedback transform $\boldsymbol{u} = \boldsymbol{\alpha}(\boldsymbol{x}) + \boldsymbol{\beta}(\boldsymbol{x})\,\boldsymbol{v}$

such that the system dynamics written in new coordinates is

$$\frac{d}{dt}\boldsymbol{y} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & & 0 \\ 0 & 0 & 1 & & 0 \\ \cdot & \cdot & \cdot & \ddots & \cdot \\ 0 & 0 & 0 & & 1 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}}_{=\boldsymbol{A}} \boldsymbol{y} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}}_{=\boldsymbol{B}} \boldsymbol{v}$$

Let us try to exploit this definition to obtain some conditions on the involved transformations.

8.3.2 How to find $\boldsymbol{T}(\cdot)$, $\boldsymbol{\alpha}(\cdot)$ and $\boldsymbol{\beta}(\cdot)$?

If the system

$$\tfrac{d}{dt}\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u}, \quad \boldsymbol{x} \in \mathbb{R}^n,\ \boldsymbol{u} \in \mathbb{R}^1,\ \boldsymbol{f}(0) = 0$$

is feedback linearizable, then there are new coordinates $\boldsymbol{y}$ defined by

$$\boldsymbol{y} = \boldsymbol{T}(\boldsymbol{x}) \quad \Rightarrow \quad \tfrac{d}{dt}\boldsymbol{y} = \tfrac{d}{dt}\left[\boldsymbol{T}(\boldsymbol{x})\right] = \tfrac{\partial}{\partial x}\left[\boldsymbol{T}(\boldsymbol{x})\right]\tfrac{d}{dt}\boldsymbol{x}$$

The last relation can be rewritten as an equation for the unknown $\boldsymbol{T}_i(\boldsymbol{x})$

$$\boldsymbol{A}\boldsymbol{y} + \boldsymbol{B}\boldsymbol{v} \ = \ \tfrac{\partial}{\partial x}\left[\boldsymbol{T}(\boldsymbol{x})\right]\big(\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u}\big) = \begin{bmatrix} \frac{\partial}{\partial x}\boldsymbol{T}_1(\boldsymbol{x}) \\ \vdots \\ \frac{\partial}{\partial x}\boldsymbol{T}_n(\boldsymbol{x}) \end{bmatrix} \big(\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u}\big)$$

$$\boldsymbol{A}\begin{bmatrix} \boldsymbol{T}_1(\boldsymbol{x}) \\ \vdots \\ \boldsymbol{T}_n(\boldsymbol{x}) \end{bmatrix} + \boldsymbol{B}\boldsymbol{v} = \begin{bmatrix} \frac{\partial}{\partial x}\boldsymbol{T}_1(\boldsymbol{x}) \\ \vdots \\ \frac{\partial}{\partial x}\boldsymbol{T}_n(\boldsymbol{x}) \end{bmatrix} \big(\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u}\big)$$

and substituting $\boldsymbol{A}$ and $\boldsymbol{B}$:

$$\begin{bmatrix} 0 & 1 & 0 & & 0 \\ 0 & 0 & 1 & & 0 \\ \cdot & \cdot & \cdot & \ddots & \cdot \\ 0 & 0 & 0 & & 1 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{T}_1(\boldsymbol{x}) \\ \vdots \\ \boldsymbol{T}_n(\boldsymbol{x}) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \boldsymbol{v} \ = \ \begin{bmatrix} \frac{\partial}{\partial x}\boldsymbol{T}_1(\boldsymbol{x}) \\ \vdots \\ \frac{\partial}{\partial x}\boldsymbol{T}_n(\boldsymbol{x}) \end{bmatrix} \big(\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u}\big)$$

$$\begin{bmatrix} T_2(x) \\ \vdots \\ T_n(x) \\ v \end{bmatrix} = \begin{bmatrix} \mathcal{L}_f T_1 + [\mathcal{L}_g T_1]\, u \\ \vdots \\ \mathcal{L}_f T_{n-1} + [\mathcal{L}_g T_{n-1}]\, u \\ \mathcal{L}_f T_n + [\mathcal{L}_g T_n]\, u \end{bmatrix}$$

In other words, we obtain the following system of equations

$$\begin{aligned} \mathcal{L}_f T_1 + \big[\mathcal{L}_g T_1\big]\, u &= T_2, \\ \mathcal{L}_f T_2 + \big[\mathcal{L}_g T_2\big]\, u &= T_3, \\ &\cdots \\ \mathcal{L}_f T_n + \big[\mathcal{L}_g T_n\big]\, u &= v \end{aligned}$$

that must be satisfied after substituting $u = \alpha(x) + \beta(x)\, v$ identically for all values of x and v.

Since we are looking at $T_1(x)$, …, $T_n(x)$ that are independent on u, we must require

$$\mathcal{L}_g T_1 = \mathcal{L}_g T_2 = \cdots = \mathcal{L}_g T_{n-1} = 0, \qquad \mathcal{L}_g T_n \neq 0$$

Therefore, the equations to solve become

$$\mathcal{L}_f T_i = T_{i+1},\ i = 1, \ldots, n-1, \quad \mathcal{L}_f T_n + (\mathcal{L}_g T_n)\,\alpha = 0, \quad (\mathcal{L}_g T_n)\,\beta = 1$$

Hence, to obtain a solution we must solve the system of PDEs

$$\mathcal{L}_g T_1 = 0, \qquad \mathcal{L}_g T_2 = 0, \qquad \ldots, \qquad \mathcal{L}_g T_{n-1} = 0$$

for T_1,…, T_{n-1}, which are defined by

$$T_2 = \mathcal{L}_f T_1, \qquad T_3 = \mathcal{L}_f T_2, \qquad \ldots, \qquad T_n = \mathcal{L}_f T_{n-1}$$

In order to use Frobenius Theorem, let us rewrite it as a system with respect to a single scalar function $T_1(x)$ by recursively substituting the expressions for $T_i(x)$.

It is easy to see that we obtain

$$\mathcal{L}_g T_1 = 0, \quad \mathcal{L}_g \mathcal{L}_f T_1 = 0, \quad \mathcal{L}_g \mathcal{L}_f^2 T_1 = 0, \quad \ldots, \quad \mathcal{L}_g \mathcal{L}_f^{n-1} T_1 = 0$$

It is left to notice that the second equation can be substituted with the following one

$$\mathcal{L}_{[f,g]} T_1 = \mathcal{L}_f\, [\mathcal{L}_g T_1] - \mathcal{L}_g\, [\mathcal{L}_f T_1] = 0 - \mathcal{L}_g T_2 = 0$$

and in a similar way all of them can be substituted by repeated Lie derivatives so that the system to solve becomes

$$\mathcal{L}_g T_1 = 0, \quad \mathcal{L}_{[f,g]} T_1 = 0, \quad \mathcal{L}_{[f,[f,g]]} T_1 = 0, \quad \ldots, \quad \mathcal{L}_{ad_f^{n-1} g} T_1 \neq 0$$

Finally, we have a system of partial differential equations

$$\mathcal{L}_{X_i} T_1 = 0, \quad i = \{1, \ldots, n-1\}, \qquad \mathcal{L}_{X_n} T_1 \neq 0$$

with

$$X_1 = ad_f^0 g, \quad X_2 = ad_f^1 g, \quad \ldots, \quad X_{n-1} = ad_f^{(n-2)} g, \quad X_n = ad_f^{(n-1)} g$$

and to ensure existence of a solution, according to Frobenius Theorem, the distribution $\triangle = \mathrm{span}\{X_1, X_2, \ldots, X_n\}$ should be

- of constant rank, i.e. X_i are to be linearly independent,
- involutive.

If both conditions are satisfied, then the solution $T_1(x)$ exists! As soon as a solution is found, we take

$$T_2 = \mathcal{L}_f T_1, \qquad T_3 = \mathcal{L}_f T_2, \qquad \ldots, \qquad T_n = \mathcal{L}_f T_{n-1}$$

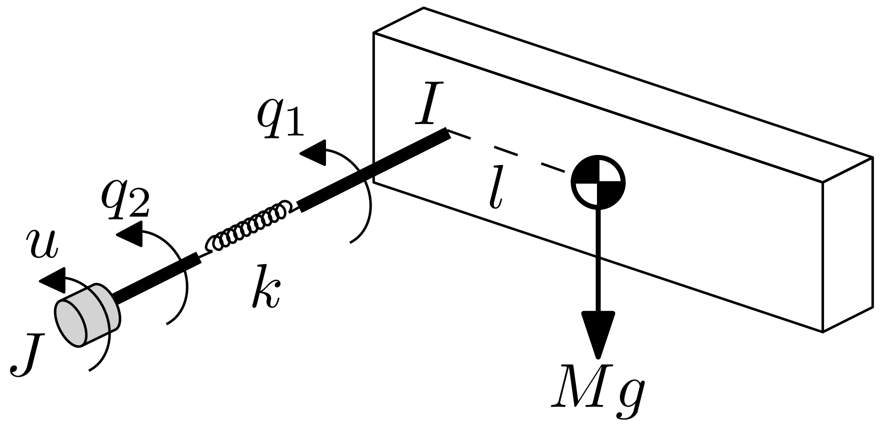
to define the coordinate transformation $y = T(x)$ and solve

$$(\mathcal{L}_g T_n)\,\alpha = -(\mathcal{L}_f T_n), \quad (\mathcal{L}_g T_n)\,\beta = 1$$

to obtain the expression for the control transformation $u = \beta(x) + \alpha(x)\,v$.

8.3.3 Feedback Linearization for Single Link Flexible Joint Robot

Consider the following simple diagram describing a one-revolute-link robot with a gear box modeled as a torsional spring.



The equations of motion are

$$\begin{aligned} I\ddot{q}_1 + Mgl\sin(q_1) + k(q_1 - q_2) &= 0 \\ J\ddot{q}_2 + k(q_2 - q_1) &= u \end{aligned}$$

where M is the mass of the link, l is the distance from the pivot point to the center of mass of the link, g is the gravitational constant, k is the spring constant, I and J are the moments of inertia, q_1 and q_2 are the angles of the motor rotor and the link respectively, and u is the applied control torque produced by the motor.

In order to show that the system is feedback linearizable if $k \neq 0$, $I \neq 0$, $J \neq 0$ we need to compute f, g, a few repeated Lie brackets, and verify involutivity! Let us do this and suggest a solution for the corresponding PDE.

Step 1: Writing Dynamics in a State-Space Form

Two second order equations

$$\begin{aligned} I\ddot{q}_1 + Mgl\sin(q_1) + k(q_1 - q_2) &= 0 \\ J\ddot{q}_2 + k(q_2 - q_1) &= u \end{aligned}$$

should be rewritten a single vector first order equation

$$\tfrac{d}{dt}x = f(x) + g(x)u$$

How to choose the vector x, and how to compute $f(x)$ and $g(x)$?

The most natural choice (among others) for $x = [x_1, x_2, x_3, x_4]^T$ is

$$x_1 = q_1, \quad x_2 = \tfrac{d}{dt}q_1, \quad x_3 = q_2, \quad x_4 = \tfrac{d}{dt}q_2$$

Let us compute derivatives of the chosen state variables

$$\begin{aligned} \tfrac{d}{dt}x_1 &= x_2 \\ \tfrac{d}{dt}x_2 &= -Mgl\sin(x_1)/I - k(x_1 - x_3)/I \\ \tfrac{d}{dt}x_3 &= x_4 \\ \tfrac{d}{dt}x_4 &= k(x_1 - x_3)/J + u/J \end{aligned}$$

This leads to a definition for the vector fields:

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \underbrace{\begin{bmatrix} x_2 \\ -Mgl\sin(x_1)/I - k(x_1 - x_3)/I \\ x_4 \\ k(x_1 - x_3)/J \end{bmatrix}}_{f(x)} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1/J \end{bmatrix}}_{g(x)} u$$

Step 2: Checking the Conditions of Frobenius Theorem

For our system the vector fields $f(x)$ and $g(x)$ are

$$\begin{aligned} f(x) &= \left[x_2,\, -\left(\tfrac{Mgl}{I}\sin(x_1) + \tfrac{k}{I}(x_1 - x_3)\right),\, x_4,\, \tfrac{k}{J}(x_1 - x_3)\right]^T \\ g(x) &= \left[0,\, 0,\, 0,\, \tfrac{1}{J}\right]^T \end{aligned}$$

We need to compute the following repeated Lie brackets

$$X_1 = ad_f^0 g, \qquad X_2 = ad_f^1 g, \qquad X_3 = ad_f^2 g, \qquad X_4 = ad_f^3 g$$

This can be done recursively as follows

$$X_1 = g, \qquad X_2 = [f,\, X_1], \qquad X_3 = [f, X_2], \qquad X_4 = [f, X_3]$$

So, the second one is

$$\begin{aligned} X_2 &= [f,\, X_1] = \left[\tfrac{\partial}{\partial x} X_1(x)\right] f(x) - \left[\tfrac{\partial}{\partial x} f(x)\right] X_1(x) \\ &= \underbrace{\left[\tfrac{\partial}{\partial x} g(x)\right]}_{=0} f(x) - \left[\tfrac{\partial}{\partial x} f(x)\right] g(x) \\ &= -\left[\tfrac{\partial}{\partial x} f(x)\right] g(x) = -\tfrac{1}{J}\left[\tfrac{\partial}{\partial x_4} f(x)\right]^T \\ &= -\tfrac{1}{J}\left[0,\ 0,\ 1,\ 0\right]^T \end{aligned}$$

The third one is

$$\begin{aligned} X_3 &= [f,\, X_2] = \left[\tfrac{\partial}{\partial x} X_2(x)\right] f(x) - \left[\tfrac{\partial}{\partial x} f(x)\right] X_2(x) \\ &= -\left[\tfrac{\partial}{\partial x} f(x)\right] X_2(x) = \tfrac{1}{J}\left[\tfrac{\partial}{\partial x_3} f(x)\right]^T \\ &= \tfrac{1}{J}\left[0,\ \tfrac{k}{I},\ 0,\ -\tfrac{k}{J}\right]^T \end{aligned}$$

The fourth one is

$$\begin{aligned} X_4 &= [f,\, X_3] = \left[\tfrac{\partial}{\partial x} X_3(x)\right] f(x) - \left[\tfrac{\partial}{\partial x} f(x)\right] X_3(x) \\ &= -\left[\tfrac{\partial}{\partial x_2} f(x)\right] \tfrac{k}{JI} + \left[\tfrac{\partial}{\partial x_4} f(x)\right] \tfrac{k}{J^2} \\ &= \left[-\tfrac{k}{JI},\ 0,\ \tfrac{k}{J^2},\ 0\right]^T \end{aligned}$$

Finally,

$$X_1 = \left[0,\ 0,\ 0,\ \tfrac{1}{J}\right]^T, \qquad X_2 = -\tfrac{1}{J}\left[0, 0, 1, 0\right]^T,$$

$$X_3 = \tfrac{1}{J}\left[0, \tfrac{k}{I}, 0, -\tfrac{k}{J}\right]^T, \qquad X_4 = \left[-\tfrac{k}{JI}, 0, \tfrac{k}{J^2}, 0\right]^T$$

Now, it is clear that the distribution $\triangle$ spanned by $X_1, \ldots, X_4$ is of constant dimension since the vector fields are constant

$$\left[X_1, X_2, X_3, X_4\right] = \begin{bmatrix} 0 & 0 & 0 & -\frac{k}{JI} \\ 0 & 0 & \frac{k}{JI} & 0 \\ 0 & -\frac{1}{J} & 0 & \frac{k}{J^2} \\ \frac{1}{J} & 0 & -\frac{k}{J^2} & 0 \end{bmatrix}$$

We need to verify whether it is involutive or not!

By definition, distribution $\triangle$ is said to be **involutive**, if there are scalar functions $\alpha_{ijk} : \mathbb{R}^n \to \mathbb{R}$ such that

$$\left[X_i, X_j\right] = \sum_{k=1}^{n} \alpha_{ijk} X_k \qquad \forall\, i,\, j$$

Obviously, for constant vector fields X_1, X_2, X_3, X_4

$$[X_i, X_j] = 0, \qquad 1 \le i, j \le 4$$

and hence we can take $\alpha_{ijk} = 0$.

To conclude

- Both conditions of Frobenius Theorem are valid!

- The dynamics of the system is **feedback linearizable**!

  Hence, there is a change of coordinates

$$y_1 = T_1(x),\ y_2 = T_2(x),\ y_3 = T_3(x),\ y_4 = T_4(x)$$

and there is a feedback transform

$$u = \alpha(x) + \beta(x)v$$

such that the dynamics written in (y, v) is linear.

- The equations with respect to a scalar function $T_1(x)$

$$\mathcal{L}_g T_1(x) = 0,\ \mathcal{L}_{ad_f g} T_1(x) = 0,\ \mathcal{L}_{ad_f^2 g} T_1(x) = 0,\ \mathcal{L}_{ad_f^3 g} T_1(x) \neq 0$$

has a solution! Functions $T_2(\cdot)$, $T_3(\cdot)$, $T_4(\cdot)$ can be found!

Step 3: Find $T_1(x)$ as a solution for PDEs

The first PDE

$$\boxed{\mathcal{L}_g T_1(x) = 0}$$

becomes

$$\left[\frac{\partial}{\partial x_1}T_1, \frac{\partial}{\partial x_2}T_1, \frac{\partial}{\partial x_3}T_1, \frac{\partial}{\partial x_4}T_1\right] \left[0,\ 0,\ 0,\ \frac{1}{J}\right]^T = 0 \ \Rightarrow\ \boxed{\frac{\partial}{\partial x_4}T_1 = 0}$$

The second PDE

$$\boxed{\mathcal{L}_{ad_f g} T_1(x) = 0}$$

becomes

$$\left[\frac{\partial}{\partial x_1}T_1, \frac{\partial}{\partial x_2}T_1, \frac{\partial}{\partial x_3}T_1, \frac{\partial}{\partial x_4}T_1\right] \left[0,\ 0,\ -\frac{1}{J},\ 0\right]^T = 0 \ \Rightarrow\ \boxed{\frac{\partial}{\partial x_3}T_1 = 0}$$

The third PDE

$$\boxed{\mathcal{L}_{ad_f^2 g} T_1(x) = 0}$$

becomes

$$\left[\frac{\partial}{\partial x_1}T_1, \frac{\partial}{\partial x_2}T_1, \frac{\partial}{\partial x_3}T_1, \frac{\partial}{\partial x_4}T_1\right] \left[0,\ \frac{k}{JI},\ 0,\ -\frac{k}{J^2}\right]^T = 0 \ \Rightarrow\ \boxed{\frac{\partial}{\partial x_2}T_1 = 0}$$

The last inequality

$$\boxed{\mathcal{L}_{ad_f^3 g} T_1(x) \neq 0}$$

becomes

$$\left[\frac{\partial}{\partial x_1}T_1, \frac{\partial}{\partial x_2}T_1, \frac{\partial}{\partial x_3}T_1, \frac{\partial}{\partial x_4}T_1\right] \left[\frac{-k}{JI},\ 0,\ -\frac{k}{J^2},\ 0\right]^T \neq 0 \ \Rightarrow\ \boxed{\frac{\partial}{\partial x_1}T_1 \neq 0}$$

Finally, all the computed requirements for the function $T_1 = T_1(x)$

$$\frac{\partial}{\partial x_1}T_1(x) \neq 0, \quad \frac{\partial}{\partial x_2}T_1(x) = 0, \quad \frac{\partial}{\partial x_3}T_1(x) = 0, \quad \frac{\partial}{\partial x_4}T_1(x) = 0$$

imply that it should depend only on x_1 variable in a nontrivial way!

There are many possible choices. Let us take

$$\boxed{T_1(x) = x_1}$$

Step 4: Compute $T_2(x)$, $T_3(x)$, $T_4(x)$

The functions $T_2(\cdot)$, $T_3(\cdot)$, $T_4(\cdot)$, $\alpha(\cdot)$, and $\beta(\cdot)$ can be easily found from the relations

$$\mathcal{L}_f T_1 = T_2, \quad \mathcal{L}_f T_2 = T_3, \quad \mathcal{L}_f T_3 = T_4,$$

and

$$(\mathcal{L}_g T_n)\,\alpha = -\,(\mathcal{L}_f T_n)\,, \quad (\mathcal{L}_g T_n)\,\beta = 1$$

With $T_1(x_1, x_2, x_3, x_4) = x_1$, we have

$$\begin{aligned} T_2(x) &= \mathcal{L}_f T_1 = \left[\tfrac{\partial}{\partial x_1} T_1(x)\right] \cdot f_1(x) + \cdots + \left[\tfrac{\partial}{\partial x_4} T_1(x)\right] \cdot f_4(x) \\ &= \left[\tfrac{\partial}{\partial x_1} T_1(x)\right] \cdot f_1(x) + 0 \cdot f_2(x) + 0 \cdot f_3(x) + 0 \cdot f_4(x) \\ &= f_1(x) = x_2 \end{aligned}$$

since

$$f(x) = \left[\, x_2,\ -\left(\tfrac{Mgl}{I}\sin(x_1) + \tfrac{k}{I}(x_1 - x_3)\right),\ x_4,\ \tfrac{k}{J}(x_1 - x_3)\,\right]^T$$

Proceeding in the same way

$$\begin{aligned} T_3(x) &= \mathcal{L}_f T_2 = \left[\tfrac{\partial}{\partial x_1} T_2(x)\right] \cdot f_1(x) + \cdots + \left[\tfrac{\partial}{\partial x_4} T_2(x)\right] \cdot f_4(x) \\ &= 0 \cdot f_1(x) + \left[\tfrac{\partial}{\partial x_2} T_2(x)\right] \cdot f_2(x) + 0 \cdot f_3(x) + 0 \cdot f_4(x) \\ &= f_2(x) = -\left(\tfrac{Mgl}{I}\sin(x_1) + \tfrac{k}{I}(x_1 - x_3)\right) \end{aligned}$$

and

$$\begin{aligned} T_4(x) &= \mathcal{L}_f T_3 = \left[\tfrac{\partial}{\partial x_1} T_3(x)\right] \cdot f_1(x) + \cdots + \left[\tfrac{\partial}{\partial x_4} T_3(x)\right] \cdot f_4(x) \\ &= -\left(\tfrac{Mgl}{I}\cos(x_1) + \tfrac{k}{I}\right) \cdot x_2 + \tfrac{k}{I} \cdot x_4 \end{aligned}$$

Step 5: Compute $\alpha(\cdot)$ and $\beta(\cdot)$

The equation for the feedback transform

$$\mathcal{L}_f T_4(x) + \mathcal{L}_g T_4(x) u = v$$

with

$$\begin{aligned} f(x) &= \left[x_2,\, -\left(\tfrac{Mgl}{I}\sin(x_1) + \tfrac{k}{I}(x_1 - x_3)\right),\, x_4,\, \tfrac{k}{J}(x_1 - x_3)\right]^T \\ g(x) &= \left[0,\, 0,\, 0,\, \tfrac{1}{J}\right]^T \\ T_4(x) &= -\left(\tfrac{Mgl}{I}\cos(x_1) + \tfrac{k}{I}\right)\cdot x_2 + \tfrac{k}{I}\cdot x_4 \end{aligned}$$

becomes

$$u = \alpha(x) + \beta(x)v$$

if

$$\begin{aligned} \beta(x) &= JI/k \\ \alpha(x) &= -\frac{JI}{k}\left[\frac{Mgl}{I}\cos(x_1)\cdot x_2^2 + \frac{k}{I}\frac{k}{J}(x_1 - x_3)\right. \\ &\qquad \left. + \left(\frac{Mgl}{I}\cos(x_1) + \frac{k}{I}\right)\left(\frac{Mgl}{I}\sin(x_1) + \frac{k}{I}(x_1 - x_3)\right)\right] \end{aligned}$$

Finally, under the transformation, the equations of motion

$$\begin{aligned} I\ddot{q}_1 + Mgl\sin(q_1) + k(q_1 - q_2) &= 0 \\ J\ddot{q}_2 + k(q_2 - q_1) &= u \end{aligned}$$

become linear

$$\dot{y}_1 = y_2, \quad \dot{y}_2 = y_3, \quad \dot{y}_3 = y_4, \quad \dot{y}_4 = v$$

in the new coordinates: $y_1 = q_1, \quad y_2 = \dot{q}_1$

$$\begin{aligned} y_3 &= -\left(\tfrac{Mgl}{I}\sin(q_1) + \tfrac{k}{I}(q_1 - q_2)\right) \\ y_4 &= -\left(\tfrac{Mgl}{I}\cos(q_1) + \tfrac{k}{I}\right)\dot{q}_1 + \tfrac{k}{I}\dot{q}_2 \end{aligned}$$

and with the new control input v that defines the real control as

$$u = \alpha\left(q_1, \dot{q}_1, q_2, \dot{q}_2\right) + \beta\left(q_1, \dot{q}_1, q_2, \dot{q}_2\right) v$$

Now, v can be designed using the tools of linear control theory.

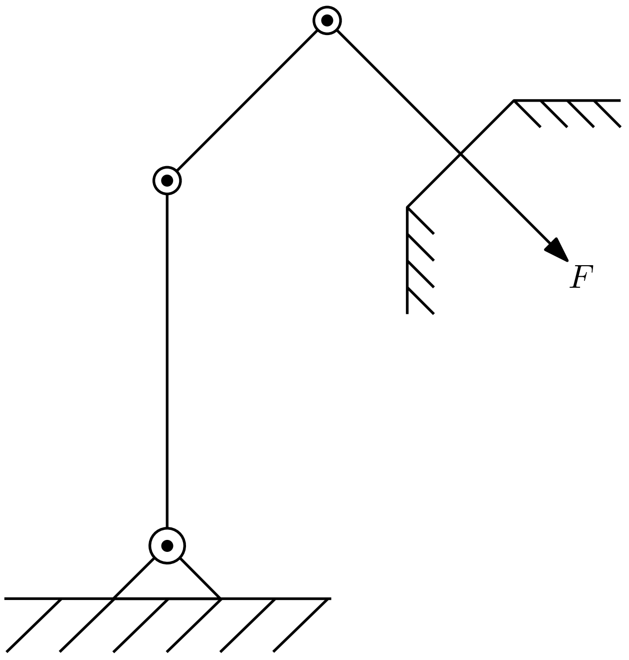
Supplementary Material: Elements of Force and Vision-Based Control

Appendix A

Elements of Force Control

We describe here some basic ideas that are currently used in robotics in various scenarios where a manipulator interacts with the environment or with the object of manipulation. In many cases the interaction forces not only influence the dynamics of the manipulator itself but also must be kept within certain limits in order to avoid distraction or sometimes to ensure completion of a task such as polishing a surface.

Whenever manipulator's tip touches an object, interaction forces appear. A force (or, sometimes, also a torque) will be applied to an object, as illustrated in the following figure.



Simultaneously, an opposite in direction and same in magnitude force (or, sometimes, also a torque) will be applied to the manipulator.

Whenever robot end-effector come into contact with a surface, the model derived in Chapter 5 is not valid from the moment of impact and until the contact is broken; it should be modified. Such a modification can be done in two ways:

1. The environment can be modeled as a separate mechanical system while

contact can be considered (after a brief period of impact) as a holonomic (or of a more general type) constraint that generates forces. Sometimes, it is also necessary to separately model the impact at the moment when impact is initiated. This approach may lead to a quite complicated models of hybrid nature (mixture of continuous and discrete dynamics) with not constant numbers of DOFs.

2. The dynamics of the environment may be completely or partially ignored while the control system relies on special sensors measuring the interaction forces/torques in addition to the usual sensors estimating the kinematic configuration. The interaction forces are added to the dynamical model derived for the interaction-free case: They appear in the dynamical equations through the manipulator Jacobian, which is, obviously, coincides with the Jacobian computed from a description of the contact as a constraint in terms of the manipulator's generalized coordinates.

To the best of our knowledge, all the commercially available systems today are based on the second approach. The main disadvantage of this is that feasible trajectories and interaction forces cannot be neither planned in advance nor, correspondingly, enforced simultaneously even if the advanced reliable sensors are installed. This issue is the subject of active current research.

A.1 Coordinate Frames and Constraints

To describe the interaction, the base frame coordinates of the vectors of linear and angular velocities and of the vectors of force and torque acting on the end-effector

$$\boldsymbol{\xi} = \left[\boldsymbol{v}^T,\, \boldsymbol{\omega}^T\right]^T, \qquad \boldsymbol{F} = \left[\boldsymbol{f}^T,\, \boldsymbol{n}^T\right]^T$$

are typically used.

Let $\boldsymbol{\mathcal{M}}$ be a set of coordinates of vectors of linear and angular velocities of the end-effector in the $\mathbf{0}$ frame

$$\boldsymbol{\xi} = \left[\boldsymbol{v}^T,\, \boldsymbol{\omega}^T\right]^T \in \boldsymbol{\mathcal{M}}$$

known also as **twists**. They can be computed as in Chapter 3.

Let $\boldsymbol{\mathcal{F}}$ be a similar set for vectors of forces and torques applied to the end-effector

$$\boldsymbol{F} = \left[\boldsymbol{f}^T,\, \boldsymbol{n}^T\right]^T \in \boldsymbol{\mathcal{F}}$$

known also as **wrenches**. They are often assumed to be measured by a special sensor. Sometimes, they can be approximately computed under certain assumptions about the environment and of the contact.

It is a common practice to describe the desired interaction task of a manipulator with the environment in terms of specifications on the twists and wrenches. When such specifications are described, it is often the case that the following property is valid. A twist $\boldsymbol{\xi} \in \boldsymbol{\mathcal{M}}$ and a wrench $\boldsymbol{F} \in \boldsymbol{\mathcal{F}}$ are called **reciprocal** if

$$\boldsymbol{\xi}^T \boldsymbol{F} = \boldsymbol{v}^T \boldsymbol{f} + \boldsymbol{\omega}^T \boldsymbol{n} = 0$$

This can be interpreted as zero elementary work during the target motion when performing the task and, in particular, assumes that the friction at the contact can be ignored.

A.2 Natural and Artificial Constraints

It is usually more natural to describe a task involving interaction in a particular frame that is attached to the surface of interaction or to the object of interaction. The frame, in which task is readily defined, is called a **compliance frame** or a constraint frame.

With respect to this frame, the task is specified in terms of constraints that can be grouped as

- the **natural** ones, which are implied by mechanics and cannot be changed by the behavior of the robot,
- the **artificial** ones, which are implied by our description of the task and can be assigned differently.

This is illustrated on an example next.

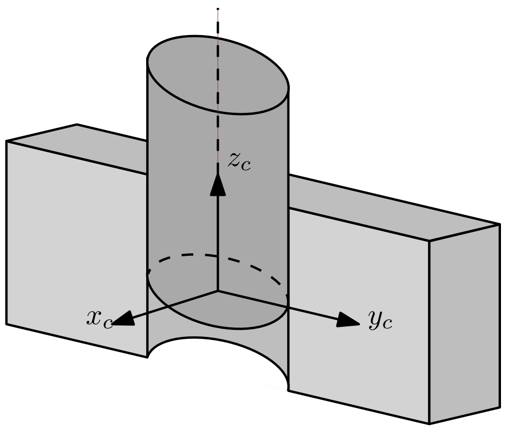
A.2.1 Inserting Task

Consider the task of inserting a cylindrical peg into a matching cylindrical hole in a static fixed rigid object. Let us assume that the walls of the hole and the peg itself are perfectly rigid and there is no friction between them.

When such a task is performed by a manipulator, it is clear that during the insertion the end-effector is prevented to have linear velocities in the directions orthogonal to the walls as well as angular velocities along such directions. At the same time, the object can produce neither a resiting force nor a resiting torque along the direction of insertion. We have just described the set of the **natural constraints**, which appear due to the interaction.

Now, in order not to damage both the object and the manipulator, the robot should not produce any forces or torques along the directions orthogonal to the walls of the hole. At the same time, insertion should be performed with a certain linear speed and a certain angular speed, so the target velocity along the hole should be a specified constant while the target angular velocity should be zero since this is a peg and not a screw. We have just described the set of the **artificial constraints**, which appear due to the task description.

The two types of constraints are summarized on the next figure.



Natural Constraints	Artificial Constraints
$\boldsymbol{v_x} = 0$	$\boldsymbol{f_x} = 0$
$\boldsymbol{v_y} = 0$	$\boldsymbol{f_y} = 0$
$\boldsymbol{f_z} = 0$	$\boldsymbol{v_z} = \boldsymbol{v_d}$
$\boldsymbol{\omega_x} = 0$	$\boldsymbol{n_x} = 0$
$\boldsymbol{\omega_y} = 0$	$\boldsymbol{n_y} = 0$
$\boldsymbol{n_z} = 0$	$\boldsymbol{\omega_z} = 0$

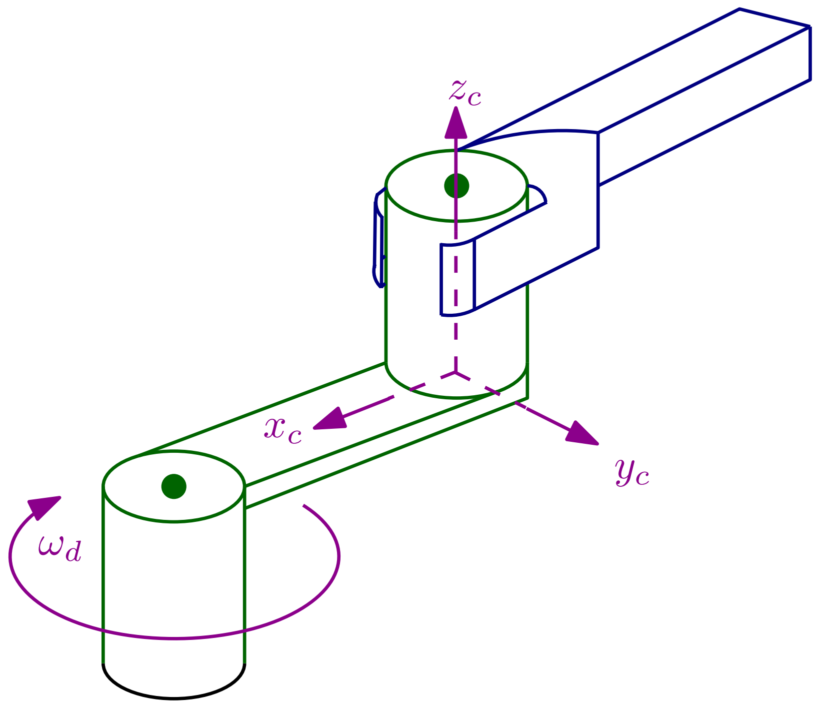
Note that for this example,

$$\boldsymbol{\xi}^T \boldsymbol{F} = \boldsymbol{v_x f_x} + \boldsymbol{v_y f_y} + \boldsymbol{v_z f_z} + \boldsymbol{\omega_x n_x} + \boldsymbol{\omega_y n_y} + \boldsymbol{\omega_z n_z} = = 0$$

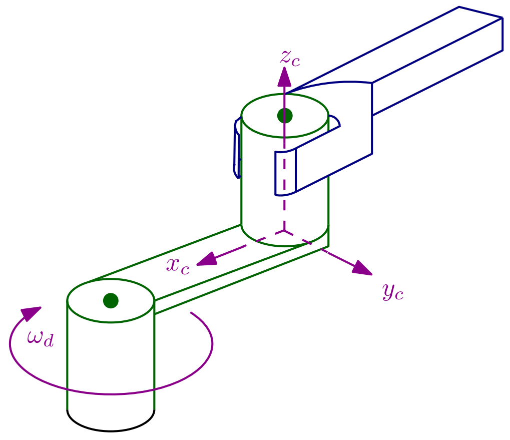
that is the constraints are reciprocal. Note that this is partially due to our assumption of frictionless contact.

A.2.2 Turning a Crank Task

Consider now the task illustrated below.



Which constraints are natural and which ones are artificial in this case? The answer is given next.



Natural Constraints	Artificial Constraints
$v_x = 0$	$f_x = 0$
$f_y = 0$	$v_y = 0$
$v_z = 0$	$f_z = 0$
$\omega_x = 0$	$n_x = 0$
$\omega_y = 0$	$n_y = 0$
$n_z = 0$	$\omega_z = \omega_d$

Note that the total elementary work is again zero and so the constraints are reciprocal.

A.3 Network Models and Impedance

In the situation when total elementary work along a target task is zero, that is the task describing constraints are reciprocal, it is clear how to formulate a control design task: In certain directions the position and the orientation of the end-effector should be achieved, while in the complimentary directions forces and torques should be produced.

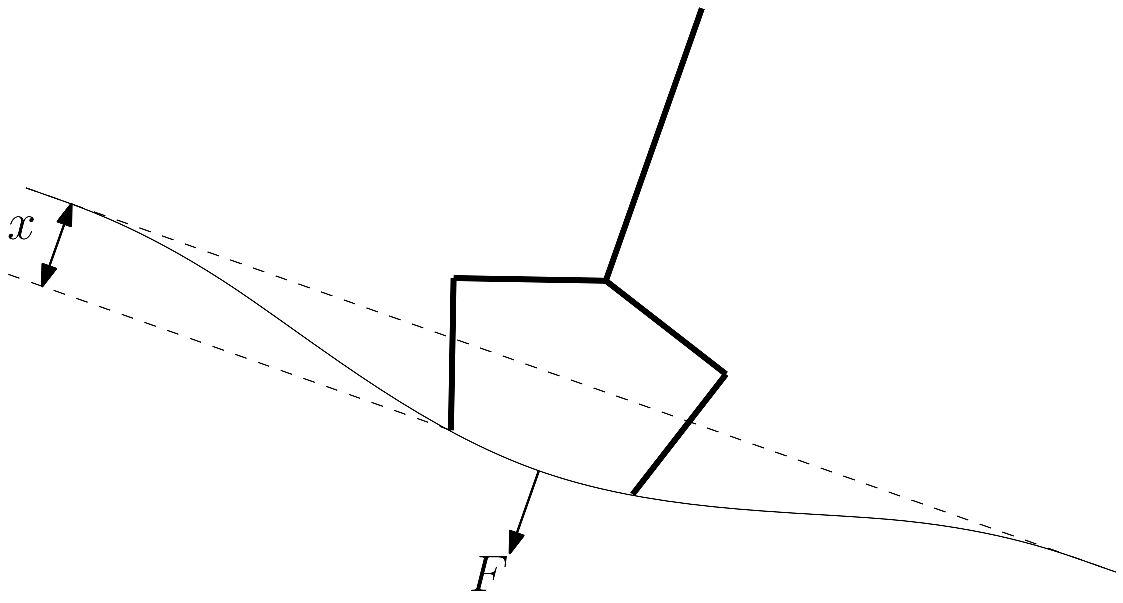
However, in practice, such separation in the task description is often not possible without introducing models of the environment and of the interaction. To deal with the situation, it has been suggested to introduce partial intuition-based models of the environment and of the corresponding interaction forces. This models are to allow generation of target trajectories (or just target velocities) and target interaction forces/torques that are not contradictory to each other.

A.3.1 Network Models

Let us introduce a useful concept of mechanical impedance.

Flexible surface

Suppose the end-effector comes into contact with a surface that is not rigid. Instead of attempting to generate a mechanical model of the object, surface of which is touched, let us introduce assumptions about the induced reaction force. Suppose first that the surface acts as a perfect resisting spring, i.e. whenever deformation of the surface appear as illustrated in the figure below



a reaction force proportional to the deformation in the direction normal to the surface appears as well. Introduce the stiffness coefficient of the surface $\boldsymbol{k}$ and suppose

$$\boldsymbol{F} = [\boldsymbol{f_x},\ \boldsymbol{f_y},\ \boldsymbol{f_z}]^T = [\boldsymbol{k\,x},\ \boldsymbol{0},\ \boldsymbol{0}]^T, \qquad \boldsymbol{n} = \boldsymbol{0}$$

then, clearly,

$$\boldsymbol{\xi}^T\boldsymbol{F}(t) = \boldsymbol{v_x f_x} + \boldsymbol{v_y f_y} + \boldsymbol{v_z f_z} + \boldsymbol{\omega_x n_x} + \boldsymbol{\omega_y n_y} + \boldsymbol{\omega_z n_z} \neq \boldsymbol{0}$$

and non-zero elementary work must be done to introduce deformation of the surface.

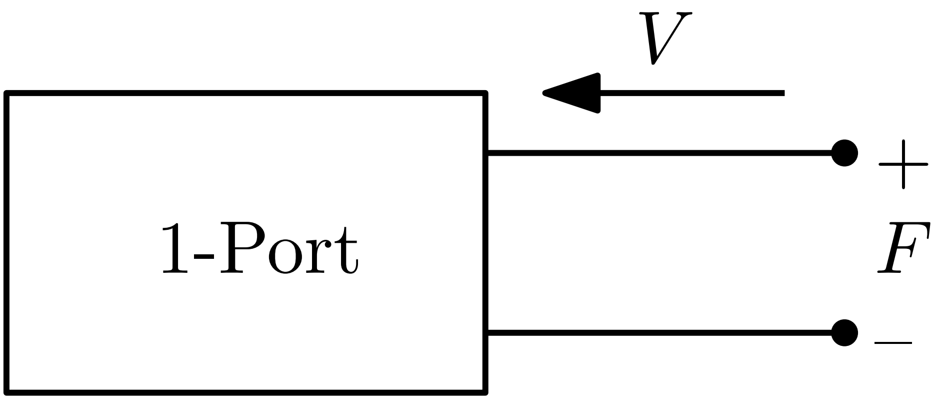
The work done along the motion of certain duration $\boldsymbol{T}$ can be computed as follows

$$\begin{aligned}\int_0^T \boldsymbol{\xi}(t)^T \boldsymbol{F}(t)dt &= \int_0^T [v_x]\, f_x\, dt = \int_0^T [v_x]\, kx(t)\, dt \\ &= \int_0^T \tfrac{d}{dt}\big[x(t)\big]\, kx(t)\, dt = k\int_0^T \tfrac{d}{dt}\big[\tfrac{1}{2}\, x^2(t)\big]\, dt \\ &= \tfrac{1}{2}\, k\left[x^2(T) - x^2(0)\right]\end{aligned}$$

Note that this work is defined by the initial and final configurations.

One-Port Network

The above description of flexible surface of contact can be generalized as a network with velocity $\boldsymbol{V}$ (or twists) treated as an input and the law of change of the resistance force $\boldsymbol{F}$ (or wrenches) characterized by dissipated power during the interaction. The conceptual diagram is given below.



The variables $\boldsymbol{V}$ and $\boldsymbol{F}$ are such that

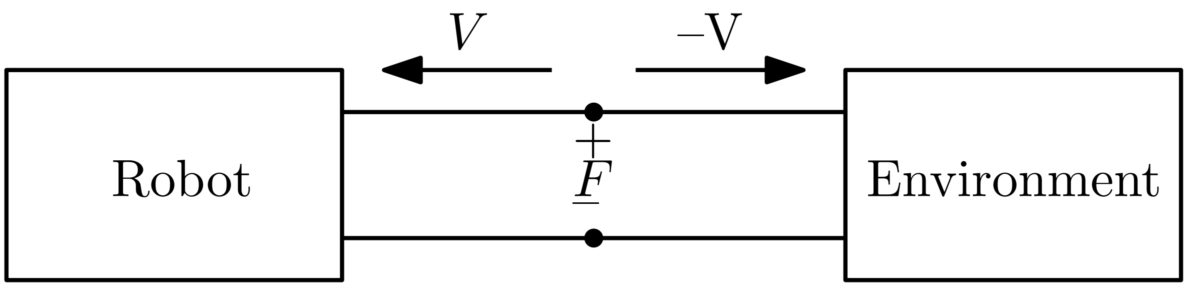
- $\boldsymbol{V}^T(t)\boldsymbol{F}(t)$ represents an instantaneous (elementary) power,
- $\displaystyle\int_0^T \boldsymbol{V}(t)^T\boldsymbol{F}(t)\, dt$ is the energy dissipated by the network on the time interval $t \in [0, T]$.

These variables have special names:

- $\boldsymbol{V}$ is called the **flow** variable,
- $\boldsymbol{F}$ is called the **effort** variable.

Interconnection of One-Port Networks

It is possible to describe interaction between the robot and the environment as a interconnection of two one-port networks as illustrated below



Note that in order to describe the energy exchange between the robot and the environment, it is convenient to consider connection of the two subsystems with opposite in sign flow variables and equal effort variables.

A.3.2 Mechanical Impedance

A network model of interaction allows to introduce the concept of (mechanical) impedance.

Impedance Operator for One-Port Network

If the system is linear, then the (steady-state) relation between the flow variable ($\boldsymbol{V}(\boldsymbol{t})$) and the effort variable ($\boldsymbol{F}(\boldsymbol{t})$) for a one-port network can be described by a transfer function, which is called the **impedance operator**

$$\boldsymbol{Z}(\boldsymbol{s}) = \frac{\tilde{\boldsymbol{F}}(\boldsymbol{s})}{\tilde{\boldsymbol{V}}(\boldsymbol{s})}$$

For example, in the case of flexible surface, we have

$$\boldsymbol{F}(\boldsymbol{t}) = \boldsymbol{k}\boldsymbol{x}(\boldsymbol{t}) \quad \Rightarrow \quad \tfrac{d}{dt}\boldsymbol{F} = \boldsymbol{k}\boldsymbol{V}(\boldsymbol{t}) \quad \Rightarrow \quad \boldsymbol{s}\tilde{\boldsymbol{F}}(\boldsymbol{s}) = \boldsymbol{k}\tilde{\boldsymbol{V}}(\boldsymbol{s}) \quad \Rightarrow \quad \boldsymbol{Z}(\boldsymbol{s}) = \frac{\boldsymbol{k}}{\boldsymbol{s}}$$

where with some abuse of notation we have dropped the zero components redefining $\boldsymbol{V} \leftarrow \boldsymbol{v}_{\boldsymbol{x}}$ and $\boldsymbol{F} \leftarrow \boldsymbol{f}_{\boldsymbol{x}}$.

Let us consider now another example. Suppose dynamics of a one-port network can be described as a 1-DOF linear time-invariant mechanical system

$$\boldsymbol{M}\,\ddot{\boldsymbol{x}} + \boldsymbol{B}\,\dot{\boldsymbol{x}} + \boldsymbol{K}\,\boldsymbol{x} = \boldsymbol{f}$$

Then, the impedance operator from $\boldsymbol{V} = \dot{\boldsymbol{x}}$ to $\boldsymbol{F} = \boldsymbol{f}$ is

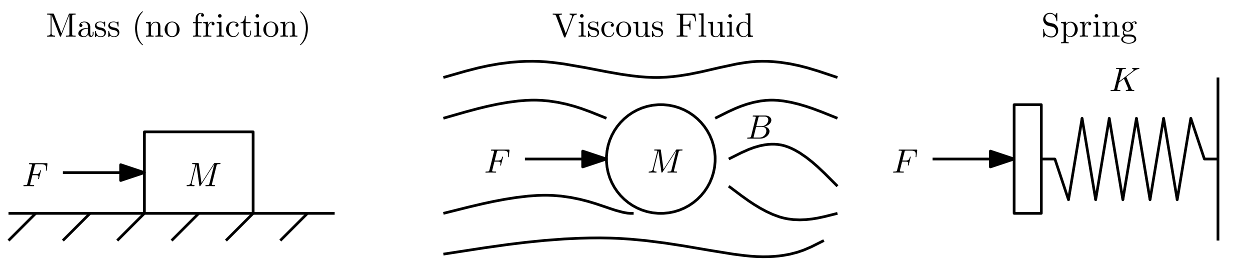
$$\boldsymbol{Z}(\boldsymbol{s}) = \boldsymbol{M}\boldsymbol{s} + \boldsymbol{B} + \boldsymbol{K}\,\frac{\boldsymbol{1}}{\boldsymbol{s}}$$

Classification of Impedance Operators

An impedance operator $\boldsymbol{Z}(\boldsymbol{s})$ is called

- **Inertial**, if $\left|\boldsymbol{Z}(\boldsymbol{0})\right| = \boldsymbol{0}$
- **Resistive**, if $\left|\boldsymbol{Z}(\boldsymbol{0})\right| = \boldsymbol{B}$ with $\boldsymbol{0} < \boldsymbol{B} < \infty$
- **Capacitive**, if $\left|\boldsymbol{Z}(\boldsymbol{0})\right| = +\infty$

This classification is illustrated in the following diagram



(a) Inertial (b) Resistive (c) Capacitive

The type of the impedance operator that is appropriate for description of the environment within a particular task have some consequences on how to approach design of a control law.

A.4 Task Space Dynamics and Impedance Control

Impedance description may be helpful in designing a control law.

A.4.1 Dynamics of a Rigid Manipulator in Contact with an Environment

Dynamics of the system can be obtained by modifying the model derived in Chapter 5. We just need to add the external forces due to interaction with the environment to obtain

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}^{T}(\boldsymbol{q})\boldsymbol{F}_{e} = \boldsymbol{u}$$

where the generalized force $\boldsymbol{J}^{T}(\boldsymbol{q})\boldsymbol{F}_{e}$ is computed as product of the transposed manipulator Jacobian (see Chapter 3) and the external generalized reaction force.

Now, in order to design a feedback control law to achieve a particular task involving interaction with the environment one have to know how the interaction force is formed. Avoiding modeling the environment as an interacting with the robot mechanical system, one may attempt to proceed in the following way:

- Assume that there is a description of the target trajectory $\boldsymbol{q} = \boldsymbol{q}_d(\boldsymbol{t})$ and the target interaction force $\boldsymbol{F} = \boldsymbol{F}_d(\boldsymbol{t})$ that are **consistent**, i.e. simultaneously possible without loosing the contact between the robot and the environment or the object of manipulation.

- Introduce feedback not only on the state variables but also on the generalized interaction force measurable by an appropriate sensor.

To understand the consequences and importance of the consistency assumption one needs to realize that the internal (reaction) forces between various pairs of links of a mechanical system, such as a manipulator, are completely defined by the values of the generalized coordinates, generalized velocities and generalized accelerations. The forces and the values of the coordinates can be computed along any target motion (e.g. by introducing appropriate excessive coordinates and writing dynamics in the form of a system of algebraic-differential equations with Lagrange multipliers representing magnitudes of the internal forces) but *can not be chosen independently!*

Nevertheless, let us assume that a particular task is described appropriately. For example, it seems obvious that zero interaction force can be achieved provided the target trajectory avoids collisions with the environment. Similarly, it seems to be reasonable to assume possibility to achieve constant interaction force together with a constant deformation of the environment accurately approximated by a flexible surface model.

As a possible control design strategy, one may modify the feedback-linearization-based control transformation described in Chapter 6 as follows

$$\boldsymbol{u} = \boldsymbol{M}(\boldsymbol{q})\boldsymbol{a}_{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}(\boldsymbol{q})^T \boldsymbol{a}_{\boldsymbol{f}}$$

Then, we obtain the transformed dynamics

$$\boldsymbol{M}(\boldsymbol{q})\,[\ddot{\boldsymbol{q}} - \boldsymbol{a}_{\boldsymbol{q}}] + \boldsymbol{J}^T(\boldsymbol{q})\,[\boldsymbol{F}_e - \boldsymbol{a}_{\boldsymbol{f}}] = \boldsymbol{0}$$

and may try to design the two new control inputs:

We can design feedback implying certain relation between $\boldsymbol{a}_{\boldsymbol{q}}$ and $\boldsymbol{a}_{\boldsymbol{f}}$ to reach a particular interplay (or accuracy trade-off): tracking *vs* force assignment

This approach seems to be particularly attractive in the case when the constraints describing the task are reciprocal, i.e. when the directions important for accurate tracking and for accurate interaction force assignment are naturally distinguished. However, in this case we need to overcome the difficulty related to the fact that the task is described in the compliance frame and not in the base frame of the manipulator.

A.4.2 Task Space Dynamics and Impedance Control

Consider the vector of linear and angular velocities of the end-effector

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{v}_e \\ \boldsymbol{\omega}_e \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{x}} \\ \boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} \boldsymbol{J}_v(\boldsymbol{q}) \\ \boldsymbol{J}_\omega(\boldsymbol{q}) \end{bmatrix} \dot{\boldsymbol{q}} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}},$$

related to the generalized velocities via the manipulator Jacobian.

Compute its time derivative

$$\tfrac{d}{dt}\boldsymbol{\xi} = \boldsymbol{J}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \tfrac{d}{dt}\left[\boldsymbol{J}(\boldsymbol{q})\right]\dot{\boldsymbol{q}}$$

and assume, for simplicity, that orientation dynamics is of no interest and can be dropped

$$\tfrac{d^2}{dt^2}\boldsymbol{x} = \boldsymbol{J}_v(\boldsymbol{q})\ddot{\boldsymbol{q}} + \tfrac{d}{dt}\left[\boldsymbol{J}_v(\boldsymbol{q})\right]\dot{\boldsymbol{q}}$$

Introduce the following control transformation

$$\boldsymbol{a}_x = \boldsymbol{J}(\boldsymbol{q})\boldsymbol{a}_q + \tfrac{d}{dt}\left[\boldsymbol{J}(\boldsymbol{q})\right]\dot{\boldsymbol{q}}$$

Combining this new transformation with

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{F}_e = \boldsymbol{u},$$

$$\boldsymbol{u} = \boldsymbol{M}(\boldsymbol{q})\boldsymbol{a}_q + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{a}_f$$

we obtain

$$\tfrac{d^2}{dt^2}\boldsymbol{x} = \boldsymbol{a}_x + \boldsymbol{W}(\boldsymbol{q})\left[\boldsymbol{F}_e - \boldsymbol{a}_f\right]$$

where

$$\boldsymbol{W}(\boldsymbol{q}) = \boldsymbol{J}(\boldsymbol{q})\boldsymbol{M}^{-1}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q})$$

is called the mobility tensor.

Assuming that our task can be described as achieving certain impedance in the closed-loop system, a possible choice of the new control inputs is

$$\boldsymbol{a}_f = \boldsymbol{F}_e$$

to cancel the current interaction force, and

$$\boldsymbol{a}_x = \ddot{\boldsymbol{x}}_d - \boldsymbol{M}_d^{-1}\left(\boldsymbol{B}_d\left(\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_d(t)\right) + \boldsymbol{K}_d\left(\boldsymbol{x} - \boldsymbol{x}_d(t)\right) + \boldsymbol{F}_e\right)$$

to achieve certain impedance of the end-effector dynamics in the closed-loop system

$$\boldsymbol{M}_d\ddot{\boldsymbol{e}} + \boldsymbol{B}_d\dot{\boldsymbol{e}} + \boldsymbol{K}_d\dot{\boldsymbol{e}} = -\boldsymbol{F}_e$$

where $\boldsymbol{e} = \boldsymbol{x} - \boldsymbol{x}_d(t)$ is the task-space tracking error.

Note that using this approach zero tracking error can not be achieved simultaneously with a nonzero interaction force.

Under assumption of accuracy of impedance-based description of interaction with the environment, it is possible to use a more advanced design in order to achieve certain specifications of the closed-loop system impedance together with partial assignment of interaction forces simultaneously with accurate partial tracking under some additional assumptions. However, we will not go deeper into this subject.

Appendix B

Elements of Computer Vision

We present here a brief introduction into computer vision, i.e. into various issues that should be address to make a robotic system detect and identify various objects in the workspace using video cameras.

B.1 Digital Image

Information provided by a standard video camera is organized in the form of digital images.

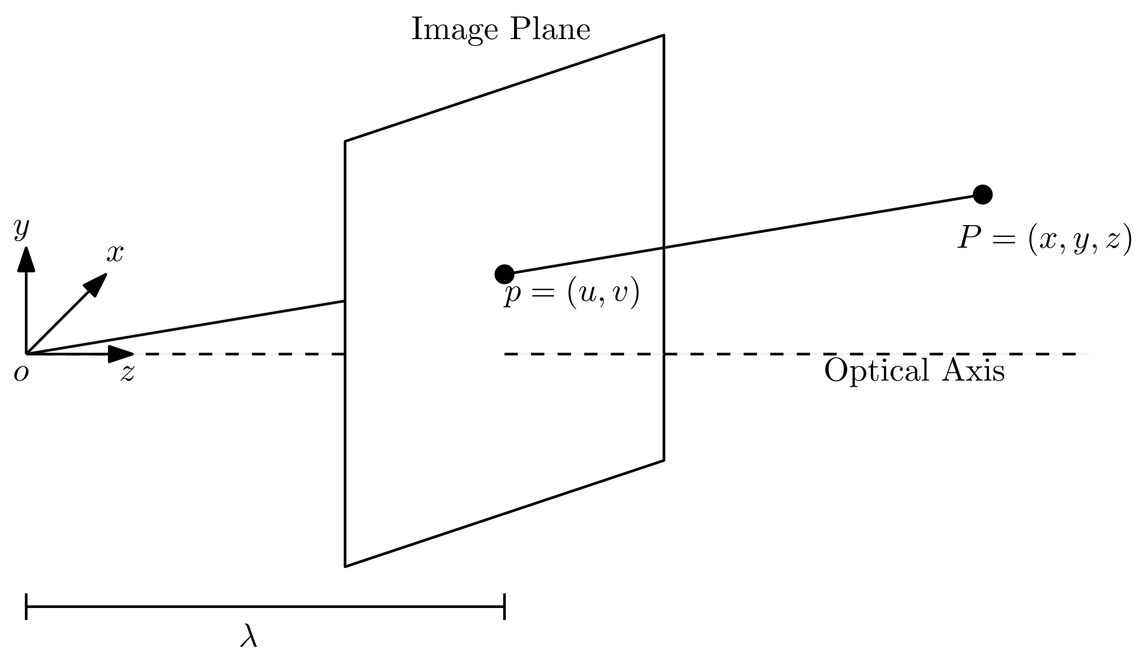
Digital image is a **two dimensional array**: $\boldsymbol{N_{rows}} \times \boldsymbol{N_{cols}}$ and the following remarks are in order.

- Elements of an image are called **pixels** (comes from "picture elements").
- The image is formed by focusing light on a two-dimensional array of **sensing elements**.
- Each pixel has a numerical value that corresponds to the **intensity of light** incident on a particular sensing element.
- A lens is used to focus the light onto the sensing array, which is usually composed of CCD (charge-couple device) sensors.
- Frame grabber transfers the data from matrix of sensing elements to the pixel array (digital image).

In order to understand how the obtained information on the environment can be used, one needs to understand the basic principle of image formation.

B.1.1 Geometry of Image Formation

An image is always formed from a camera view relative to the coordinate frame attached to the camera. It is done via a projection onto a plane, called the **image plane**. The schematic is given below.



The following terminology is used.

- The origin of the coordinate frame is called the **center of projection**.
- The point, at which the optical axis intersects the image plane, is called the **principal point**.
- The distance from the center of projection to the image plane, $\boldsymbol{\lambda}$, is called the **focal length**.
- Any point on the image plane has coordinates: $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\lambda})$ in the camera frame and represents a point $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ by the light intensity.

Let us start our analysis deriving the relation between the two sets of coordinates $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\lambda})$ and $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$.

B.1.2 Perspective Projection

Consider a point $\boldsymbol{P} = (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ in the World (represented by coordinates in an inertial frame) and an induced by it point $\boldsymbol{p} = (\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\lambda})$ in the image plane. Obviously, it is produced by a projection.

We see from the picture that the vectors $\overrightarrow{\boldsymbol{oP}}$ and $\overrightarrow{\boldsymbol{op}}$ from the origin $\boldsymbol{o}$ to the point $\boldsymbol{P}$ and to the point $\boldsymbol{p}$, respectively, have the same directions.

Therefore, there exists $\boldsymbol{k}$ such that

$$\boldsymbol{k}\,\overrightarrow{\boldsymbol{oP}} = \overrightarrow{\boldsymbol{op}}$$

or, in coordinates,

$$\boldsymbol{kx} = \boldsymbol{u}, \qquad \boldsymbol{ky} = \boldsymbol{v}, \qquad \boldsymbol{kz} = \boldsymbol{\lambda}$$

Eliminating the distance value (the depth of the point), one obtains the following

$$\boxed{\boldsymbol{u} = \frac{\boldsymbol{\lambda}}{\boldsymbol{z}}\,\boldsymbol{x}}, \qquad \text{and} \qquad \boxed{\boldsymbol{v} = \frac{\boldsymbol{\lambda}}{\boldsymbol{z}}\,\boldsymbol{y}}$$

Now, we know the relation between coordinates on the image plane $(\boldsymbol{u}, \boldsymbol{v})$ for a point with the coordinates $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ and would like to relate these values to the corresponding pixel of the digital image.

B.1.3 Coordinates on Image Plane vs Sensor Array

A digital image is a **two dimensional array** of pixels with an integer number of rows, $\boldsymbol{N_{rows}}$, and an integer number of columns, $\boldsymbol{N_{cols}}$.

Each pixel is defined by coordinates $(\boldsymbol{r}, \boldsymbol{c})$ with $\boldsymbol{r} \in [\boldsymbol{1}, \boldsymbol{N_{rows}}]$ and $\boldsymbol{c} \in [\boldsymbol{1}, \boldsymbol{N_{cols}}]$.

It is important to know the coordinates of the principal point $(\boldsymbol{r_o}, \boldsymbol{c_o})$ as well as the horizontal and vertical dimensions of a pixel, denoted below by $\boldsymbol{s_x}$ and $\boldsymbol{s_y}$.

With such information at hand, the relation between the coordinates of the point $\boldsymbol{p} = (\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\lambda})$ on the image plane and the indexes of the associated pixel in the array $(\boldsymbol{r}, \boldsymbol{c})$ can be represented as follows

$$\boxed{-\frac{\boldsymbol{u}}{\boldsymbol{s_x}} \approx \boldsymbol{r} - \boldsymbol{r_o}}, \qquad \boxed{-\frac{\boldsymbol{v}}{\boldsymbol{s_y}} \approx \boldsymbol{c} - \boldsymbol{c_o}}$$

Here, the "$-$" sign corresponds to the most frequently used direction of counting pixels.

The exact relations hold if we round the left-hand sides so that

$$\boxed{\boldsymbol{r} = \boldsymbol{r_o} + \text{round}\left[-\frac{\boldsymbol{u}}{\boldsymbol{s_x}}\right]}, \qquad \boxed{\boldsymbol{c} = \boldsymbol{c_o} + \text{round}\left[-\frac{\boldsymbol{v}}{\boldsymbol{s_y}}\right]}$$

Note that the frame grabber transfers the data from matrix of sensing elements to the pixel array in according to the formulae above.

However, one should realize by now that several parameters must be known in order to correctly interpret the information presented in a digital image. Procedure of obtaining such information is called **camera celebration**; and we address it next.

B.2 Camera Calibration

Our ultimate goal is to relate the information recorded in a pixel $(\boldsymbol{r}, \boldsymbol{c})$ with the information observed at a point $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$. To make such relation unambiguous, one needs to calibrate the camera, i.e. find various parameters of a particular camera. The parameters are typically split into two groups. Let us describe them.

B.2.1 Extrinsic Camera Parameters

Let us remind our goal:

Given coordinates $(\boldsymbol{x}^w, \boldsymbol{y}^w, \boldsymbol{z}^w)$ *of a point in the* **world frame**,
we need an algorithm for computing
the coordinates $(\boldsymbol{r}, \boldsymbol{c})$ *of the corresponding point on the* **digital image**.

First of all, note that the camera frame and the world frame are typically different. In fact, the camera may even be attached to a moving link of the robot so that the camera frame orientation may continuously change with time. Independently on whether both frames are stationary or not, at each moment of time there is a homogeneous transform that relates them (see Chapter 2)

$$\boldsymbol{q}^w = \begin{bmatrix} \boldsymbol{x}^w \\ \boldsymbol{y}^w \\ \boldsymbol{z}^w \end{bmatrix} = \boldsymbol{R}_c^w \begin{bmatrix} \boldsymbol{x}^c \\ \boldsymbol{y}^c \\ \boldsymbol{z}^c \end{bmatrix} + \boldsymbol{c}_o^w = \boldsymbol{R}_c^w \boldsymbol{q}^c + \boldsymbol{c}_o^w$$

Here, coordinates of the point $\boldsymbol{q}$ are denoted by $\boldsymbol{q}^c$ in the camera frame and by $\boldsymbol{q}^w$ in the world frame.

Hence,

$$\boldsymbol{q}^c = \left[\boldsymbol{R}_c^w\right]^T \boldsymbol{q}^w - \left[\boldsymbol{R}_c^w\right]^T \boldsymbol{c}_o^w = \boldsymbol{R}\, \boldsymbol{q}^w + \boldsymbol{T}$$

In the most common situations, a camera is mounted on a tripod so that its orientation is conveniently defined by two parameters:

$$\boldsymbol{R} = \boldsymbol{R}_{z,\theta}\, \boldsymbol{R}_{x,\alpha}\,, \quad \text{where} \quad \boldsymbol{\theta} \text{ is the pan angle,} \quad \boldsymbol{\alpha} \text{ is the tilt angle.}$$

The components of the vector $\boldsymbol{T}$ together with the angles $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ are called **extrinsic parameters** that must be estimated for a particular set-up.

B.2.2 Intrinsic Camera Parameters

With the values of the **extrinsic parameters** at hands, one can relate the coordinates (x^w, y^w, z^w) of a point in the world frame with the coordinates (x^c, y^c, z^c) of the point in the camera frame.

$$\begin{bmatrix} x^c \\ y^c \\ z^c \end{bmatrix} = R \begin{bmatrix} x^w \\ y^w \\ z^w \end{bmatrix} + T$$

We need now to find relation between (x^c, y^c, z^c) and (r, c) of the corresponding point of the **digital image**.

Earlier we have found the following relations

$$\text{round}\left[-\frac{u}{s_x}\right] = r - o_x, \qquad u = \frac{\lambda}{z^c} x^c,$$
$$\text{round}\left[-\frac{v}{s_y}\right] = c - o_y, \qquad v = \frac{\lambda}{z^c} y^c$$

Combining them, we have

$$r \approx -\frac{\lambda}{s_x}\frac{x^c}{z^c} + r_o$$

and

$$c \approx -\frac{\lambda}{s_y}\frac{y^c}{z^c} + c_o$$

In other words,

$$r \approx -f_x \frac{x^c}{z^c} + r_o, \qquad c \approx -f_y \frac{y^c}{z^c} + c_o$$

The values f_x and f_y are called the **intrinsic parameters**.

B.2.3 Steps in Camera Calibration

Finally, we know that we need to identify all the (extrinsic and intrinsic) parameters:

$$R, \quad T, \quad f_x, \quad r_o, \quad f_y, \quad c_o$$

This can be done as follows:

- Determine the position (r_o, c_o) of the principle point.
- Collect a data set $\{r_i,\ c_i,\ x_i^w,\ y_i^w,\ z_i^w\}_{i=1}^n$ from various measurements of test points with known locations in the world frame.

- Use the equations with respect to the unknown entries of $\boldsymbol{R}$ and $\boldsymbol{T}$

$$\begin{aligned} x^c &= r_{11}x^w + r_{12}y^w + r_{13}z^w + T_x \\ y^c &= r_{21}x^w + r_{22}y^w + r_{23}z^w + T_y \\ z^c &= r_{31}x^w + r_{32}y^w + r_{33}z^w + T_z \end{aligned}$$

to rewrite the equations

$$r - r_o = -f_x \frac{x^c}{z^c}, \qquad c - c_o = -f_y \frac{y^c}{z^c}$$

as

$$z^c = -\frac{f_x}{r - r_o} x^c, \qquad z^c = -\frac{f_y}{c - c_o} y^c$$

to obtain

$$[c - c_o]\, f_x\, x^c = [r - r_o]\, f_y\, y^c$$

that can be rewritten as

$$\begin{aligned} &[c - c_o]\, f_x\, (r_{11}x^w + r_{12}y^w + r_{13}z^w + T_x) \\ &\quad = [r - r_o]\, f_y\, (r_{21}x^w + r_{22}y^w + r_{23}z^w + T_y) \end{aligned}$$

- Use the relations

$$\begin{aligned} &[c - c_o]\, f_x\, (r_{11}x^w + r_{12}y^w + r_{13}z^w + T_x) \\ &\quad - [r - r_o]\, f_y\, (r_{21}x^w + r_{22}y^w + r_{23}z^w + T_y) = 0 \end{aligned}$$

to find the parameters

$$r_{21}, \qquad r_{22}, \qquad r_{23}, \qquad T_y, \qquad \dots$$

searching for the best fit for the collected data set, i.e. make the following substitutions

$$r = r_i, \; c = c_i, \; x^w = x_i^w, \; y^w = y_i^w, \; z^w = z_i^w \quad i = 1, \dots, n$$

to obtain a system of equations for the parameters.

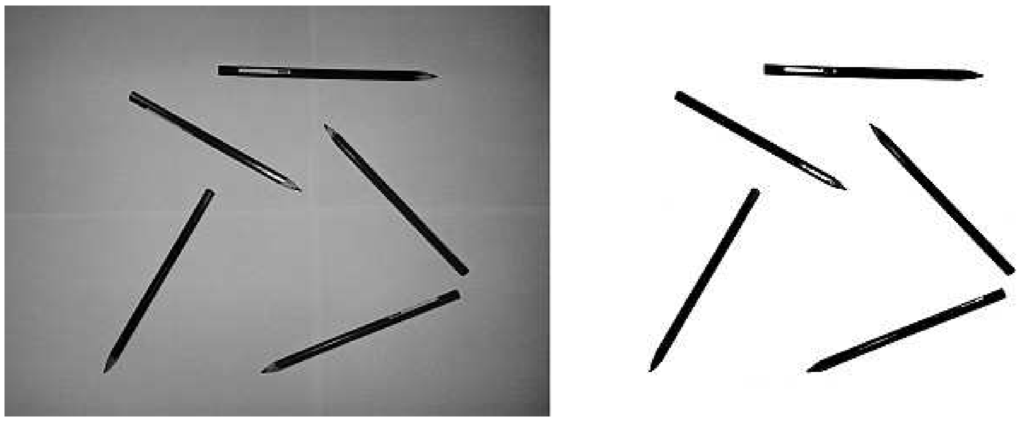
B.3 Image Processing: Segmentation

Each digital image from a calibrated camera should be processed to extract the information about various objects.

The first typical task is to identify groups of pixels that correspond to the same object using the intensity information. This procedure is called segmentation.

B.3.1 Segmentation of Image

The basic idea is to separate objects from the background by choosing a trash-hold value and finding connected groups of pixels with intensity below it. A typical outcome of segmentation is identification of different parts of the image with different objects as illustrated below



B.3.2 Tools for Image Processing

Typically, for a given digital $\boldsymbol{N_{rows}} \times \boldsymbol{N_{cols}}$ image, each pixel is one of the following integers

$$\boldsymbol{z \in \{0,\ 1,\ 2,\ 3,\ \ldots,\ (N-1) \quad (= 2^8 - 1 = 255)\}}$$

The image can be processed as follows.

- Compute the frequency of each value on the image

$$\boldsymbol{P(z) = \frac{H(z)}{N_{rows} \times N_{cols}}}, \quad \boldsymbol{H(z)} \text{ is the number of times } \boldsymbol{z} \text{ occurs.}$$

- Compute the **mean value** for the whole image

$$\boldsymbol{\mu = \sum_{z=0}^{N-1} z\, P(z)}$$

- Compute the **variance** for the whole image

$$\sigma^2 = \sum_{z=0}^{N-1} (z-\mu)^2\, P(z)$$

- Find possible **segmentation of the image**.

  To this end, note that *mean values of pixel values for the areas occupied by an object is different from the mean values of pixel values for the background.* So, we can do the following.

  – Compute the frequency of each value not for the whole image, but for various sub-areas

  $$P_i(z) = \frac{H_i(z)}{\sum_{z=0}^{N-1} H_i(z)}, \qquad H_i(z) \text{ is the number of times } z \text{ occurs}$$

  here $i = 0$ means pixels of (possible) background and $i = 1$ means pixels of a (possible) object.

  – Compute **mean values** and **variances** for various areas

  $$\mu_i = \sum_{z=0}^{N-1} z\, P_i(z) \qquad \sigma_i^2 = \sum_{z=0}^{N-1} (z-\mu_i)^2\, P_i(z)$$

  and distinguish the areas on the image, e.g.
  $\mu_1 \neq \mu_0$ (an object) or $\sigma_i \approx 0$ (background).

  Note that if we do not know segmentation of image into object and background areas, we can search for an **optimal** threshold value z_t

  $$\mu_1 > z_t > \mu_0 \quad \text{or} \quad \mu_1 < z_t < \mu_0$$

  such that the **variance** for each area

  $$\sigma_i^2 = \sum_{z=0}^{N-1} (z-\mu_i)^2\, P_i(z), \quad i = 0, 1$$

  is as small as possible!

  Alternatively, one can search for an optimal threshold z_t by minimizing a weighted sum of conditional variances

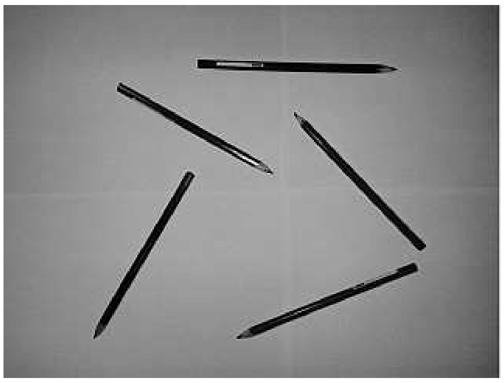
  $$\Phi(z_t) = q_0(z_t)\sigma_0^2(z_t) + q_1(z_t)\sigma_1^2(z_t)$$

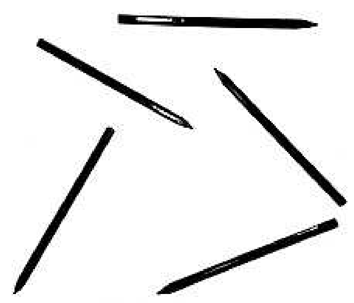
where weights are

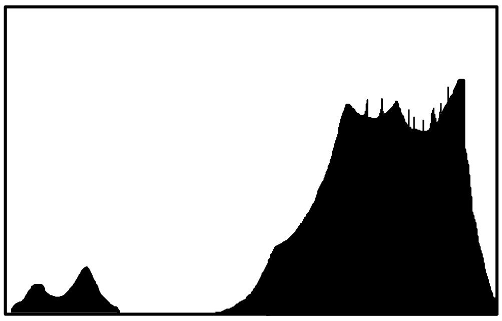
$$\boldsymbol{q_0(z_t) = \frac{\sum_{z=0}^{z_t} H(z)}{N_{rows} \times N_{cols}}} \qquad \boldsymbol{q_1(z_t) = \frac{\sum_{z=z_t+1}^{N-1} H(z)}{N_{rows} \times N_{cols}}}$$

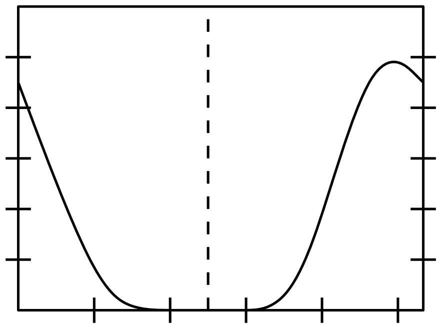
The function $\boldsymbol{\Phi(\cdot)}$ is known as the **within-group variance**.

The result of the *segmentation of image* procedure described above is illustrated below


(a) the image


(b) the binary image

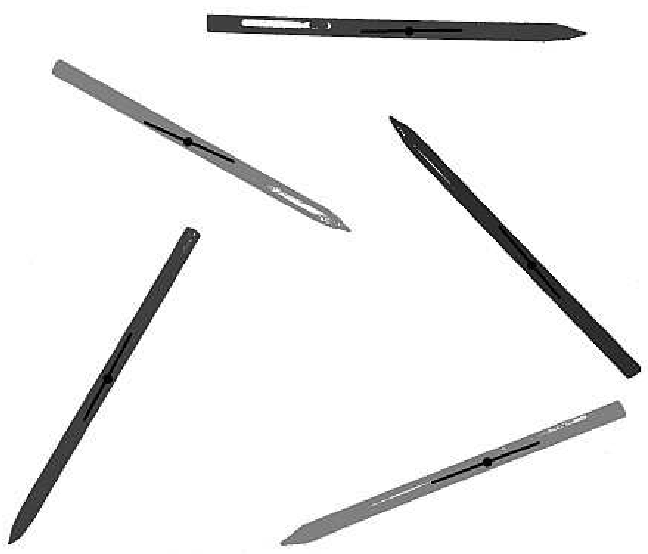

(c) the histogram


(d) within-group variance as function of threshold

B.4 Estimating Positions and Orientations

After an image is segmented, one can approach the problem of extracting the information on various properties of the objects.

As an example, suppose we are interested in estimating geometry of detected objects in terms of centers of masses (centroids) and orientations represented by points and lines in the next figure



Let a function $\mathcal{I}_i(r,c)$ be an indicator of the object i, i.e.

$$\mathcal{I}_i(r,c) = \begin{cases} 1 & : \text{ pixel with coordinates } (r,c) \text{ belongs to an object } i \\ 0 & : \text{ otherwise} \end{cases}$$

which is obtained from segmentation.

Then, location of the **center of mass** of the object can be computed as follows

$$\bar{r}_i = \frac{\sum_{r,c} r\,\mathcal{I}_i(r,c)}{\sum_{r,c} \mathcal{I}_i(r,c)}, \qquad \bar{c}_i = \frac{\sum_{r,c} c\,\mathcal{I}_i(r,c)}{\sum_{r,c} \mathcal{I}_i(r,c)}$$

On the other hand, the **orientation** of the object is defined by finding **a line**, along which the value of the function

$$\mathcal{L} = \sum_{r,c} d^2(r,c)\,\mathcal{I}_i(r,c)$$

where $d(r,c)$ is the distance from a pixel to the line, is **minimal**.

Appendix C

Elements of Vision Based Control

With a working computer vision system, it is possible to actively react on changes in the robot's environment. To use the information extracted from images taken by a camera, it is important to make an appropriate choice of camera configuration. Clearly, there are two reasonable choices for possible installation of a camera; it could be

- Fixed to some predefined position in the world frame;
- Attached to a robot at a particular place; this is called the "eye-in-hand" configuration.

In both situations, there are different ways to use digital images for feedback control.

C.1 Position *vs* Imaged-Based Control

There are two basic approaches for vision-based control:

- The first one is **Position-Based** and means that the system
  - Reconstruct **3**-D representation of the world from an image;
  - Control a robot to reach a desired configuration of the robot based on this **3**-D reconstruction with feedback based on the configuration error.
- The other one is **Image-Based** and means that the system
  - Does NOT reconstruct a **3**-D estimate of the world;
  - Control the robot based on the image data with feedback based on the mismatch with a desired configuration but as it would be originally described in terms of quantities on the image plane.

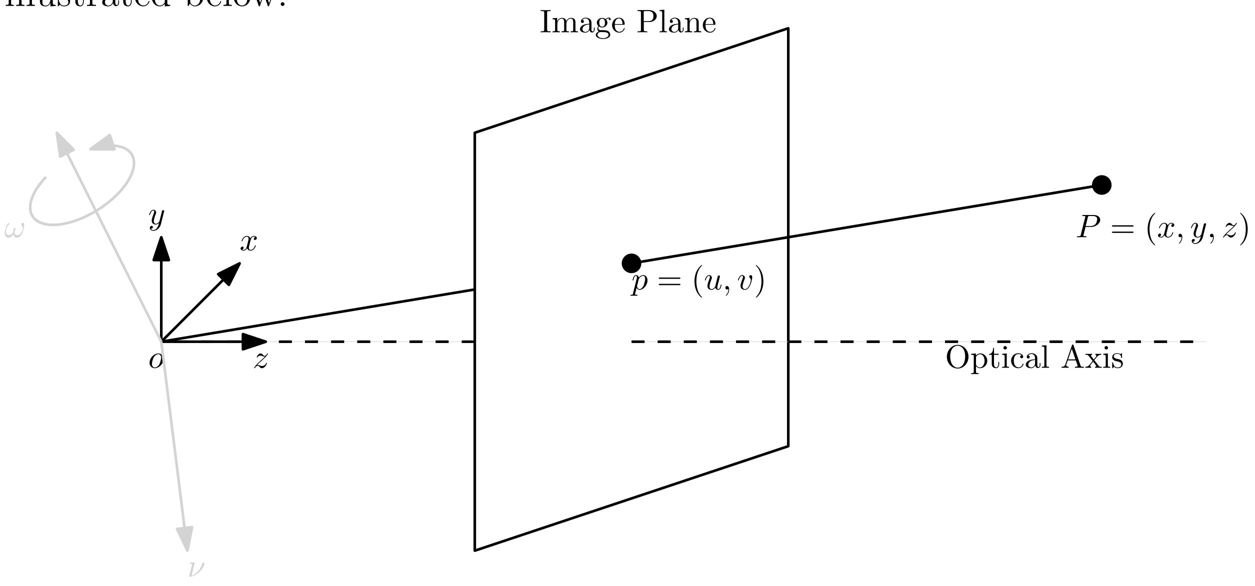
C.2 Camera Motion and Interaction Matrix

Let us consider the "eye-in-hand" configuration and position-based approach.

C.2.1 Camera Motion Description

To organize an appropriate feedback action, one needs to describe the changes of locations of various points of an image that are associated with a motion of the camera.

Suppose we know from computation of velocity kinematics for the camera frame that $\vec{v}$ is the velocity of its origin while $\vec{\omega}$ is its angular velocity as illustrated below.



Let $\boldsymbol{s(t)}$ be a point feature on the image with coordinates

$$\boldsymbol{s(t)} = \left[\boldsymbol{s_{x^c}(t),\ s_{y^c}(t),\ s_{z^c}(t)}\right] = \left[\boldsymbol{u(t),\ v(t),\ \lambda}\right]$$

that is associated with a fixed point of an object of interest $\boldsymbol{p^0}$ with coordinates $\boldsymbol{(x, y, z)}$ in a fixed stationary frame.

We can directly estimate $\boldsymbol{s(t)}$ using certain image processing technique and can estimate its rate of change $\boldsymbol{\frac{d}{dt}s(t)}$ from a time series of digital images. This should allow us to describe the motion of the camera.

C.2.2 Definition of the Interaction Matrix

The question is whether based on $\boldsymbol{s(t)}$ and $\boldsymbol{\frac{d}{dt}s(t)}$ we can compute the velocity $\vec{\boldsymbol{v}}\boldsymbol{(t)}$ of the origin of the camera frame and its angular velocity $\vec{\boldsymbol{\omega}}\boldsymbol{(t)}$ in a particular frame?

In other words, the relation between the quantities $\left[\boldsymbol{s(t),\ \frac{d}{dt}s(t)}\right]$ and $\left[\vec{\boldsymbol{v}}\boldsymbol{(t)},\ \vec{\boldsymbol{\omega}}\boldsymbol{(t)}\right]$ is of interest.

It is reasonable to guess that such a relation is linear

$$\tfrac{d}{dt}s(t) = L(\cdot) \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix}$$

The operator $L(\cdot)$ is called the **interaction matrix**. To compute this matrix, one can proceed as follows:

- Relate coordinates of a fixed point $p_c(t)$ and $\tfrac{d}{dt}p_c(t)$ with the camera frame velocities;
- Relate $s(t)$ and $\tfrac{d}{dt}s(t)$ with $p_c(t)$ and $\tfrac{d}{dt}p_c(t)$;
- Combine these two relations.

C.2.3 Velocities of the Camera Frame

Note that when a point p with coordinates p^0 is fixed in the inertial frame, but the camera moves:

- In the camera frame, the coordinates of p are changing with time: $p^c = p^c(t)$.
- The coordinates of p in the inertia and camera frames are related by a homogeneous transformation (see Chapter 2)

$$p^0 = R_c^0(t)p^c(t) + o^0(t)$$

Hence,

$$p^c(t) = \left[R_c^0(t)\right]^T \left(p^0 - o^0(t)\right)$$

Differentiating this equation we obtain velocity of the point p written in the camera frame

$$\tfrac{d}{dt}p^c(t) = \tfrac{d}{dt}\left(\left[R_c^0(t)\right]^T\right)\left(p^0 - o^0(t)\right) + \left[R_c^0(t)\right]^T \tfrac{d}{dt}\left(p^0 - o^0(t)\right)$$

Using the Poisson equation $\tfrac{d}{dt}\left(R_c^0(t)\right) = S(\omega^0)\, R_c^0(t)$ we obtain

$$\tfrac{d}{dt}\left(\left[R_c^0(t)\right]^T\right) = \left[R_c^0(t)\right]^T S(\omega^0)^T = -\left[R_c^0(t)\right]^T S(\omega^0)$$

and therefore

$$\begin{aligned} \tfrac{d}{dt}p^c(t) &= -\left[R_c^0(t)\right]^T S(\omega^0)\left(p^0 - o^0(t)\right) - \left[R_c^0(t)\right]^T \dot{o}^0(t) \\ &= -\left[R_c^0(t)\right]^T S(\omega^0) R_c^0(t)\left[R_c^0(t)\right]^T \left(p^0 - o^0(t)\right) - \left[R_c^0(t)\right]^T \dot{o}^0(t) \end{aligned}$$

Now, using the standard change of coordinates identity

$$\left[R_c^0\right]^T S(\omega^0)\, R_c^0 = S\left(\left[R_c^0\right]^T \omega^0\right) = \left(\left[R_c^0\right]^T \omega^0\right) \times I$$

we obtain

$$\begin{aligned}\frac{d}{dt}p^c(t) &= -\underbrace{\left[R_c^0(t)\right]^T \omega^0}_{\omega^c} \times \underbrace{\left[R_c^0(t)\right]^T \left(p^0 - o^0(t)\right)}_{p^c(t)} - \underbrace{\left[R_c^0(t)\right]^T \dot{o}^0(t)}_{\frac{d}{dt}o^c(t) = v^c} \\ &= -\omega^c \times p^c(t) - v^c\end{aligned}$$

Example

For illustrative purposes suppose for example that

- The optical axis z^c of a camera is parallel and opposite to the world z-axis;
- The camera motions are constrained to
  - rotation about the optical axis;
  - translation parallel to the image (x, y) plane.

It follows that

$$p^0 = \begin{bmatrix} \cos\theta(t) & -\sin\theta(t) & 0 \\ \sin\theta(t) & \cos\theta(t) & 0 \\ 0 & 0 & 1 \end{bmatrix} p^c(t) + \begin{bmatrix} x_{co}^0(t) \\ y_{co}^0(t) \\ z_{co}^0 \end{bmatrix}$$

and

$$\dot{p}^c = -\omega^c \times p^c - \dot{o}^c, \quad \omega^c = \begin{bmatrix} 0 \\ 0 \\ \dot{\theta} \end{bmatrix}, \quad \dot{o}^c = v^c = \begin{bmatrix} v_x^c \\ v_y^c \\ 0 \end{bmatrix}$$

C.2.4 Constructing the Interaction Matrix

The relation

$$\begin{aligned}\begin{bmatrix} \dot{x}^c \\ \dot{y}^c \\ \dot{z}^c \end{bmatrix} = \dot{p}^c = -\omega^c \times p^c - v^c &= -\begin{bmatrix} \omega_x^c \\ \omega_y^c \\ \omega_z^c \end{bmatrix} \times \begin{bmatrix} x^c \\ y^c \\ z^c \end{bmatrix} - \begin{bmatrix} v_x^c \\ v_y^c \\ v_z^c \end{bmatrix} \\ &= -\begin{bmatrix} \omega_y^c\, z_c - \omega_z^c\, y_c + v_x^c \\ \omega_z^c\, z_c - \omega_x^c\, z_c + v_y^c \\ \omega_x^c\, y_c - \omega_y^c\, z_c + v_z^c \end{bmatrix}\end{aligned}$$

combined with the projection equations

$$u = k\,x^c, \quad v = k\,y^c, \quad \lambda = k\,z^c,$$

allow us to compute the operator $L(\cdot)$ in

$$\begin{bmatrix} \dot u(t) \\ \dot v(t) \end{bmatrix} = L(\cdot) \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix}$$

Indeed,

$$\begin{aligned} \dot u &= \tfrac{d}{dt}\left[\frac{\lambda\, x^c}{z^c}\right] = \lambda\,\frac{z^c\,\dot x^c - x^c\,\dot z^c}{(z^c)^2} \\ &= -\frac{\lambda}{z^c}\,v^c_x + \frac{u}{z^c}\,v^c_z + \frac{u\,v}{\lambda}\omega^c_x - \frac{\lambda^2+u^2}{\lambda}\,\omega^c_y + v\,\omega^c_z \end{aligned}$$

and

$$\dot v = \tfrac{d}{dt}\left[\frac{\lambda\, y^c}{z^c}\right] = -\frac{\lambda}{z^c}\,v^c_y + \frac{v}{z^c}\,v^c_z - \frac{u\,v}{\lambda}\omega^c_y + \frac{\lambda^2+v^2}{\lambda}\,\omega^c_x - u\,\omega^c_z$$

Combining these two equations we have

$$\begin{bmatrix} \dot u \\ \dot v \end{bmatrix} = \begin{bmatrix} -\dfrac{\lambda}{z^c} & 0 & \dfrac{u}{z^c} & \dfrac{u\,v}{\lambda} & -\dfrac{\lambda^2+u^2}{\lambda} & v \\ 0 & -\dfrac{\lambda}{z^c} & \dfrac{v}{z^c} & \dfrac{\lambda^2+v^2}{\lambda} & -\dfrac{u\,v}{\lambda} & -u \end{bmatrix} \begin{bmatrix} v^c_x \\ v^c_y \\ v^c_z \\ \omega^c_x \\ \omega^c_y \\ \omega^c_z \end{bmatrix}$$

Clearly, the interaction matrix L depends on u, v, and z^c, which is the distance to the point feature along the optical axis. The equation can be rewritten as

$$\begin{bmatrix} \dot u \\ \dot v \end{bmatrix} = L(u, v, z^c) \begin{bmatrix} v^c \\ \omega^c \end{bmatrix} = \frac{1}{z^c}\,L_v(u,v)\,v^c + L_\omega(u,v)\,\omega^c$$

to separate the dependence on z^c.

C.2.5 Properties of the Interaction Matrix

It is not hard to show that the equation

$$\begin{bmatrix} \dot u \\ \dot v \end{bmatrix} = L(u, v, z^c) \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix}$$

has a non-trivial kernel that is

$$\exists\, \xi \in \mathbb{R}^6 : \quad L(u, v, z^c)\, \xi = 0, \quad \xi \neq 0$$

Moreover, the basis of the kernel subspace is

$$\begin{bmatrix} u \\ v \\ \lambda \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ u \\ v \\ \lambda \end{bmatrix}, \quad \begin{bmatrix} uvz^c \\ -(u^2+\lambda^2)z^c \\ \lambda v z^c \\ -\lambda^2 \\ 0 \\ u\lambda \end{bmatrix}, \quad \begin{bmatrix} \lambda(u^2+v^2+\lambda^2)z^c \\ 0 \\ -u(u^2+v^2+\lambda^2)z^c \\ uv\lambda \\ -(u^2+\lambda^2)z^c \\ u\lambda^2 \end{bmatrix}$$

C.3 Reconstructing a Camera Motion

Let us assume that we have processed coordinates of n different feature points $P_1, \ldots, P_n$.

Combining the interaction matrix based relations for them we have

$$\begin{bmatrix} \dot{u}_1 \\ \dot{v}_1 \end{bmatrix} = \begin{bmatrix} -\dfrac{\lambda}{z_1^c} & 0 & \dfrac{u_1}{z_1^c} & \dfrac{u_1\, v_1}{\lambda} & -\dfrac{\lambda^2+u_1^2}{\lambda} & v_1 \\ 0 & -\dfrac{\lambda}{z_1^c} & \dfrac{v_1}{z_1^c} & \dfrac{\lambda^2+v_1^2}{\lambda} & -\dfrac{u_1\, v_1}{\lambda} & -u_1 \end{bmatrix} \begin{bmatrix} v_x^c \\ v_y^c \\ v_z^c \\ \omega_x^c \\ \omega_y^c \\ \omega_z^c \end{bmatrix}$$

$$\vdots$$

$$\begin{bmatrix} \dot{u}_n \\ \dot{v}_n \end{bmatrix} = \begin{bmatrix} -\dfrac{\lambda}{z_n^c} & 0 & \dfrac{u_n}{z_n^c} & \dfrac{u_n\, v_n}{\lambda} & -\dfrac{\lambda^2+u_n^2}{\lambda} & v_n \\ 0 & -\dfrac{\lambda}{z_n^c} & \dfrac{v_n}{z_n^c} & \dfrac{\lambda^2+v_n^2}{\lambda} & -\dfrac{u_n\, v_n}{\lambda} & -u_n \end{bmatrix} \begin{bmatrix} v_x^c \\ v_y^c \\ v_z^c \\ \omega_x^c \\ \omega_y^c \\ \omega_z^c \end{bmatrix}$$

Clearly these combined equations for n-feature points

$$\begin{bmatrix} \dot{u}_1 \\ \dot{v}_1 \\ \vdots \\ \dot{u}_n \\ \dot{v}_n \end{bmatrix} = \begin{bmatrix} L(u_1, v_1, z_1^c) \\ \vdots \\ L(u_n, v_n, z_n^c) \end{bmatrix} \begin{bmatrix} v_x^c \\ v_y^c \\ v_z^c \\ \omega_x^c \\ \omega_y^c \\ \omega_z^c \end{bmatrix}$$

can be used for computation of the velocities v^c and ω^c!

For instance, if there are **3**-feature points, then

$$\begin{bmatrix} v_x^c \\ v_y^c \\ v_z^c \\ \omega_x^c \\ \omega_y^c \\ \omega_z^c \end{bmatrix} = \begin{bmatrix} L(u_1, v_1, z_1^c) \\ L(u_2, v_2, z_2^c) \\ L(u_3, v_3, z_3^c) \end{bmatrix}^{-1} \begin{bmatrix} \dot{u}_1 \\ \dot{v}_1 \\ \dot{u}_2 \\ \dot{v}_2 \\ \dot{u}_3 \\ \dot{v}_3 \end{bmatrix}$$

C.4 An Idea for Image-Based Control Design

Let us consider now the **Image-Based Control** design.

In this settings,

- the target configuration is not given in terms of the **3**D World;
- it is specified in terms of a location of the image features s^d on the image plane.

Suppose we can extract the current status of the feature $s(t)$ and compute the image error

$$e(t) = s(t) - s^d$$

A possible idea for feedback control design based on the value of the error $e(t)$ is the following. Let us produce a control signal to decrease the mismatch.

Ignoring dynamics, let us assume for simplicity that

- the control variables are the vectors of linear velocity $\vec{v}$ of the origin of the camera frame and
- its angular velocity $\vec{\omega}$.

We know that

$$\frac{d}{dt}e(t) = \frac{d}{dt}\left[s(t) - s^d\right] = \frac{d}{dt}s(t) = L(\cdot)\begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix}$$

Hence, for example, the control variables $\vec{v}(t)$, $\vec{\omega}(t)$ can be chosen to ensure

$$L(\cdot)\begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} := -\gamma\, e(t), \qquad \gamma > 0$$

so that the dynamics of the error variable becomes

$$\frac{d}{dt}e(t) = -\gamma e(t)$$

and exponentially stable tracking is achieved.

Index