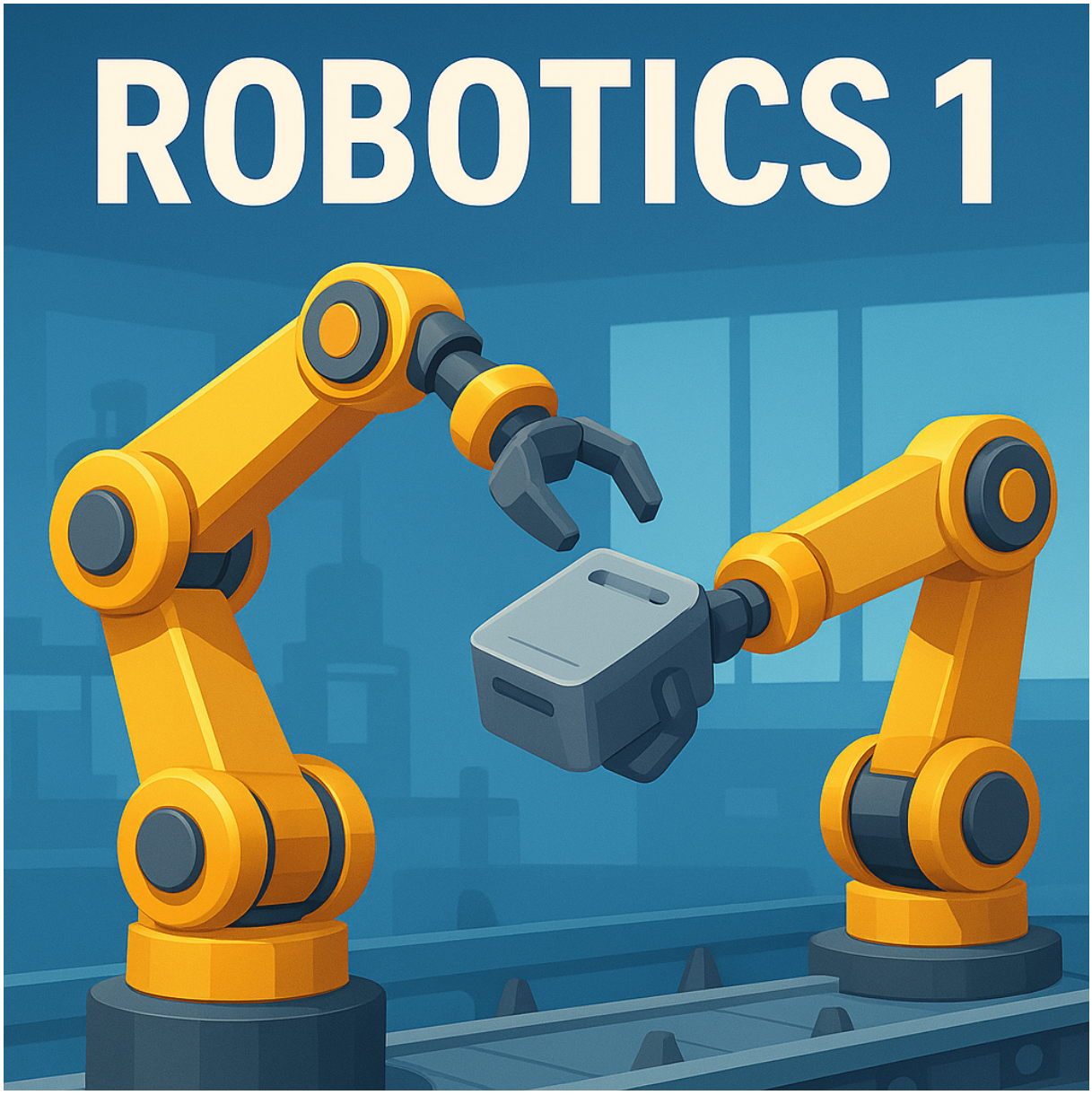
Marco Casu

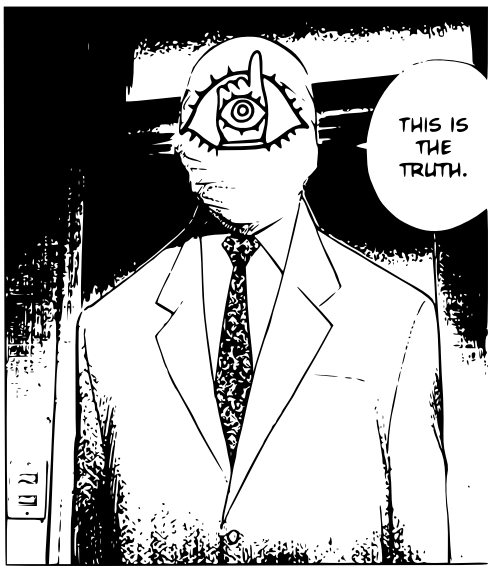
ROBOTICS 1



SAPIENZA
UNIVERSITÀ DI ROMA

Faculty of Information Engineering, Computer Science and Statistics
Department of Computer, Control and Management Engineering
Master's degree in Artificial Intelligence and Robotics

This document summarizes and presents the topics for the Robotics 1 course for the Master's degree in Artificial Intelligence and Robotics at Sapienza University of Rome. The document is free for any use. If the reader notices any typos, they are kindly requested to report them to the author.



CONTENTS

CHAPTER

1

INTRODUCTION

In this chapter we will see a brief introduction to the mathematical tools used in the main topics of the course. The topics presented in this section may seem somewhat unclear, as many concepts and definitions are only briefly introduced and deliberately not elaborated upon. They will be discussed in detail in their respective chapters.

1.1 About the End Effector Pose

A robot is made up of a series of arms connected to one another by joints, these joints can be **revolut** or **prismatic** (as shown in figure 1.1), a revolut joint rotate the link connected along 1 axis, the prismatic joint can make the link extend or contract, making them translate along 1 axis.

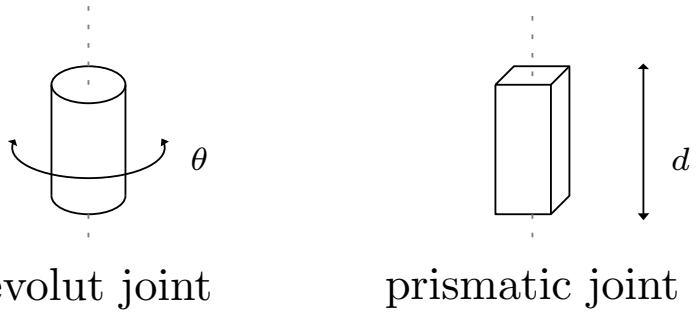


Figure 1.1: two types of joints (spatial representation)

It is important to know that that if the angle θ increase the joint is rotating counter clock wise. In a planar drawing, the joints are denoted as shown in image 1.2.

In the mathematical/geometrical model of a robotic arms, it is important the *kinematic skeleton*, the quantities involved are

- the current angle of the joints
- the length of the links

everything is defined respect to the base frame, usually denoted as Σ_0.

The robot shown in figure 1.3 is an *R4 robot* (4 revolut joints) with three links. With ${}^0\mathbf{p}_e$ and Σ_e we denote the position and the reference frame of the **end effector**, if there are a 0 supscript to a vector, we mean that is expressed in the base reference frame.

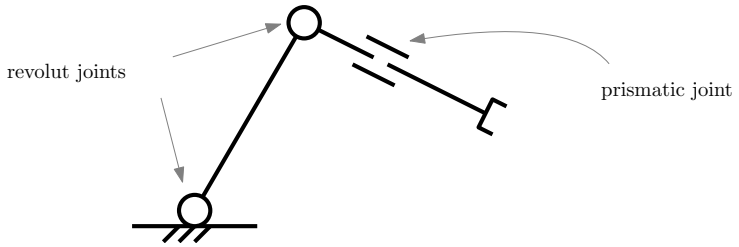


Figure 1.2: planar representation of the joints

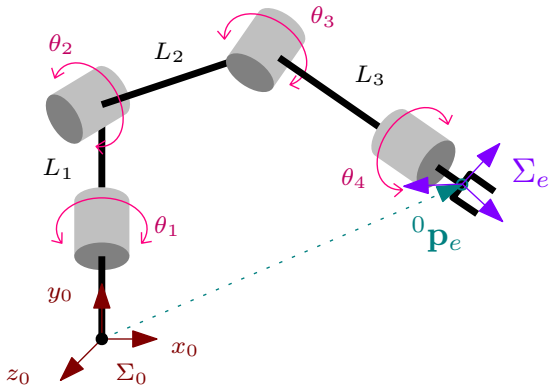


Figure 1.3: spatial R4 robot

With **Direct Kinematics**, we define the problem to find what are the **pose** (position and orientation) of the end effector, in function of the joint's angles.

$$Kin_p(\boldsymbol{\theta}) : \Sigma_0 \rightarrow \Sigma_e \tag{1.1}$$

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_1 & \theta_2 & \theta_3 & \theta_4 \end{pmatrix}^T \tag{1.2}$$

With Σ_e is denoted the reference frame of the end effector. How can we compute $Kin_p(\boldsymbol{\theta})$? This is given by an homogeneous 4×4 matrix defined as follows:

$${}^0T_e = \begin{pmatrix} & {}^0R_e & & {}^0\mathbf{p}_e \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1.3}$$

where

- ${}^0R_e \in SO(3)$ is the rotation matrix, and depends from $\boldsymbol{\theta}$
- ${}^0\mathbf{p}_e \in \mathbb{R}^3$ is the translation vector.

Recall: $SO(3)$ is the group of all the orthogonal 3×3 matrices with determinant equals to 1.

The matrix ${}^0T_e$ is obtained by multiplying n matrix (where n is the number of joints)

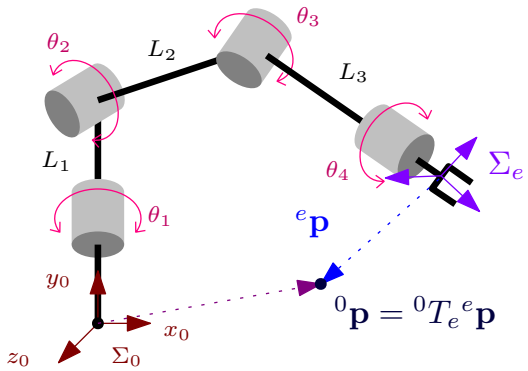
$${}^0T_e = {}^0A_1(\theta_1)\,{}^1A_2(\theta_2)\ldots{}^{n-1}A_n(\theta_n) = \tag{1.4}$$

$$\prod_{i=0}^{n-1} {}^iA_{i+1}(\theta_{i+1}). \tag{1.5}$$

Each *homogeneous matrix* ${}^{i-1}A_i$ describe the pose of the i-th joint's frame respect to the previous joint's frame, and depends from θ_i (the i-th joint's angle). A more detailed description of the matrix describing the direct kinematics will be given later.

If there another frame Σ_w, the new matrix can be computed as follows

$${}^wT_e = {}^wT_0\,{}^0T_e. \tag{1.6}$$



the end effector position respect to the base frame is ${}^0T_e\mathbf{0}$

The **Inverse Kinematics** is the opposite problem, given a position ${}^0\mathbf{p}_e$ for the end effector, we want to find the values of $\boldsymbol{\theta}$ such that

$$ {}^0\mathbf{p}_e = Kin_p(\boldsymbol{\theta}) \tag{1.7} $$

to find $\boldsymbol{\theta}$, we have to solve a non-linear system of equations, this is generally an undecidable problem, but for some specific cases, there exists a closed form, that can be found analytically, there are also numerical methods. Clearly, for the positions out of the work space, the system does not admit solutions (also this can be checked analytically).

1.2 About the End Effector Velocity

Let's now consider **Differential Kinematics**, that is the problem to find the end effector velocity in the workspace given the velocity of the joint's angles. Since the superposition principle is valid, the components resulting from the movement of each individual joint, which constitute the final velocity of the end effector, can be considered separately. It is important to know that the velocity component of the end effector given by a joint, is always orthogonal to the rotation axis of that joint.

The end effector have

- a linear velocity, usually denoted $\mathbf{v}$
- an angular velocity, usually denoted $\boldsymbol{\omega}$.

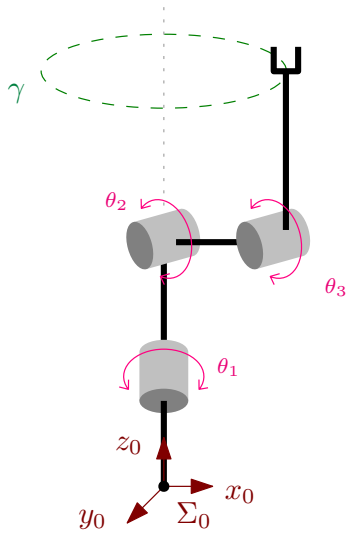


Figure 1.4: possible trajectory by moving θ_1

In the figure 1.4 the curve γ represent all possible positions where the end effector could lie if the angle θ_1 change, the linear velocity of the end effector is orthogonal to the z_0 axis. The velocity of the end effector doesn't depend only from the angular velocity, but also from the current configuration of the angles $\boldsymbol{\theta}$.

Even if the end effector is a rigid body, is sufficient to know the linear velocity of only one point and his angular velocity to compute the velocity of all the other points, since the following relation holds:

$$\mathbf{v}_2 = \mathbf{v}_1 + \boldsymbol{\omega} \times \mathbf{r}_{12} \tag{1.8}$$

where

- $\mathbf{v}_1$ is the velocity of the first point
- $\mathbf{v}_2$ is the velocity of the second point
- $\boldsymbol{\omega}$ is the angular velocity of the rigid body
- $\mathbf{r}_{12}$ is the difference between the positions of the two points.

Let's analyze the velocity components of the end effector. If the i-th joint is changing is angle, the linear velocity of the end effector will have one component that is

$$\mathbf{v}_i = \mathbf{j}_i(\boldsymbol{\theta})\dot{\theta}_i \tag{1.9}$$

where $\mathbf{j}_i$ is a 3 components vector describing the direction of the velocity. Since the direction depends from the configuration, the vector $\mathbf{j}_i$ is in function of $\boldsymbol{\theta}$. This holds for all the angles θ_i, the resultant linear velocity of the end effector will be

$$\mathbf{v} = \sum_{i=1}^{n} \mathbf{j}_i(\boldsymbol{\theta})\dot{\theta}_i \tag{1.10}$$

it can be written in matrix form

$$\mathbf{v} = J_L(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} = \begin{pmatrix} \mathbf{j}_1(\boldsymbol{\theta}) & \dots & \mathbf{j}_n(\boldsymbol{\theta}) \end{pmatrix} \begin{pmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_n \end{pmatrix} \tag{1.11}$$

where $J_L(\boldsymbol{\theta})$ is a $3 \times n$ matrix called the **Jacobian Matrix**, where n is the number of joints. This description were given in terms of the linear velocity, but it holds also for the angular velocity of the end effector, indeed we have two Jacobian Matrix:

- we denote $J_L(\boldsymbol{\theta})$ the Jacobian matrix for the linear velocity
- we denote $J_A(\boldsymbol{\theta})$ the Jacobian matrix for the angular velocity

$$\boldsymbol{\omega} = J_A(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \tag{1.12}$$

the matrix

$$J(\boldsymbol{\theta}) = \begin{pmatrix} J_L(\boldsymbol{\theta}) \\ J_A(\boldsymbol{\theta}) \end{pmatrix} \in Mat(6 \times n) \tag{1.13}$$

it's called **basic Jacobian**.

The Jacobian matrix is a mapping from the joint velocity space to the end effector velocity space. Let's ignore the angular velocity for now, suppose that we want to impose to the end effector a desired linear velocity (in a specific time instant)

$$\mathbf{v} = \mathbf{v}_d \in \mathbb{R}^3 \tag{1.14}$$

we need to find the values for the vector $\dot{\boldsymbol{\theta}}$ such that

$$\mathbf{v}_d = J_L(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \tag{1.15}$$

if we have 3 joints, the matrix J is squared and can be inverted

$$\dot{\boldsymbol{\theta}} = J_L^{-1}(\boldsymbol{\theta})\mathbf{v}_d \tag{1.16}$$

but this is not the general case, if $n > 3$, the system of equations given in (1.15) could

- have zero solutions

- have infinite solutions

if the determinant of J_L is zero, the system admit infinite solutions if and only if the desired velocity vector $\mathbf{v}_d$ is in the range space of J_L

$$\det J_L = 0 \implies \exists \text{ inf. sol.} \iff \mathbf{v}_d \in \text{Range}(J_L) \tag{1.17}$$

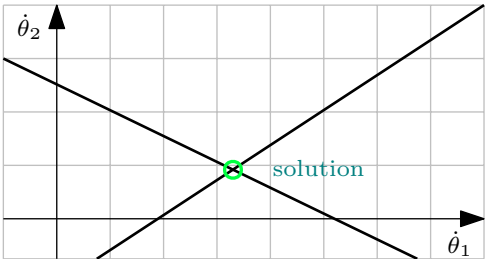
we remind that the range space of a matrix, is the set of all the linear combinations of the matrix's columns. If this isn't true, the system does not admit any solution, it means that no possible combination of velocity $\dot{\boldsymbol{\theta}}$ could realize the desired end effector velocity.

1.2.1 Singularity

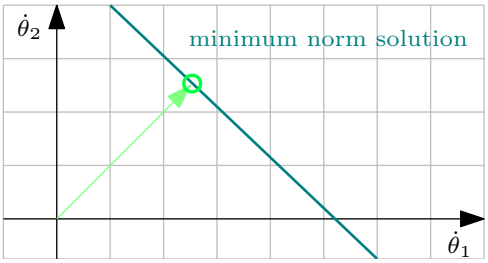
Let's talk about **singularities** in the joint velocity space, we will give a geometric example. Let's consider a 2R planar robot, with a fixed joints configuration $\boldsymbol{\theta}$, the Jacobian is a 2×2 matrix. Let $\mathbf{v}_d$ to be the desired velocity, the system is the following

$$\begin{pmatrix} v_d^x \\ v_d^y \end{pmatrix} = \begin{pmatrix} \mathbf{j}_1^T \\ \mathbf{j}_2^T \end{pmatrix} \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{pmatrix} \tag{1.18}$$

the two linear equation of the system is represented on the plane as two lines. If $\det J_L \neq 0$, there are only one solution, and is the intersection between the two lines.



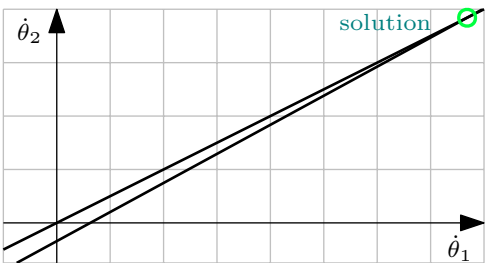
If $\det J_L = 0$, the two lines are parallel, so either they have no intersection, or they are the same line. If there are infinite solutions, we can choose the one with the smallest norm, since represents the "minimum energy" solution (the solution that requires the least joint rotation speed intensity).



We have a *singularity* when the determinant approaches zero

$$\det J_L \to 0 \tag{1.19}$$

The closer the determinant (in absolute value) gets to zero, the more "nearly" parallel the row vectors (and thus the lines they represent) become, which means the angle of intersection approaches zero. In this case the norm of the solution could be large.



This is true also because the following relations holds

$$J_L = \begin{pmatrix} j_{11} & j_{12} \\ j_{21} & j_{22} \end{pmatrix} \implies \tag{1.20}$$

$$J_L^{-1} = \frac{1}{\det J_L} \begin{pmatrix} j_{22} & -j_{12} \\ -j_{21} & j_{11} \end{pmatrix} \tag{1.21}$$

$$\mathbf{v}_d = J_L \dot{\boldsymbol{\theta}} \tag{1.22}$$

$$\dot{\boldsymbol{\theta}} = J_L^{-1} \mathbf{v}_d \tag{1.23}$$

$$\dot{\boldsymbol{\theta}} = \frac{1}{\det J_L} \begin{pmatrix} j_{22} & -j_{12} \\ -j_{21} & j_{11} \end{pmatrix} \mathbf{v}_d \tag{1.24}$$

with $\det J_L \to 0$ the term $\frac{1}{\det J_L}$ (and with it, also $\dot{\boldsymbol{\theta}}$) became bigger and bigger. In this case, the required joint rotation velocity might not be achievable by the robotic arm's motors.

The previous example showed how certain algebraic relationships are connected to physical problems in robot joint control. Another similar example is the following, consider the robotic arm shown in figure 1.5.

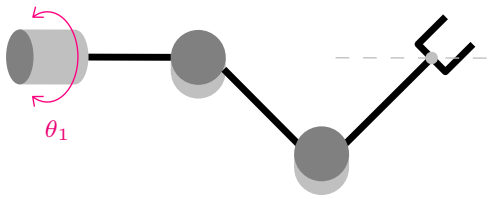


Figure 1.5: 3R spatial robot

Geometrically, it can be seen that by rotating only the first joint θ_1, the position of the end effector will not change, this condition holds when

$$J_L(\boldsymbol{\theta}) \begin{pmatrix} \dot{\theta}_1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{1.25}$$

this is true if the vector $\begin{pmatrix} \dot{\theta}_1 & 0 & 0 \end{pmatrix}^T$ is in the kernel of the Jacobian matrix

$$\begin{pmatrix} \dot{\theta}_1 \\ 0 \\ 0 \end{pmatrix} \in \ker J_L(\boldsymbol{\theta}). \tag{1.26}$$

Therefore, the vectors contained in the kernel of the Jacobian matrix for the linear (or angular) velocity represent all possible combinations of individual joint velocities that would not change the position (or orientation) of the end effector.

1.3 Brief Overview of Planning and Control

When we want to control the end effector of a robotic arm, we want to know how to move the joints to get a specific position for the end effector, and also how to control the joints over the time to get a particular *trajectory* in the working space.

Consider a 2R planar robot, as shown in figure 1.6, where $\boldsymbol{\theta}$ is the angular position of the joints, and $\mathbf{p}_e = f(\boldsymbol{\theta})$ is the position of the end effector for some $f_{\mathbb{R}}^2 \to \mathbb{R}^2$.

We would like to move the end effector from a certain starting point $\mathbf{p}_a \in \mathbb{R}^2$ to an another point $\mathbf{p}_b \in \mathbb{R}^2$. We could consider the segment line from $\mathbf{p}_a$ to $\mathbf{p}_b$ defined as follows:

$$\mathbf{p}(s) = s\mathbf{p}_b + (1-s)\mathbf{p}_a \qquad s \in [0,1]. \tag{1.27}$$

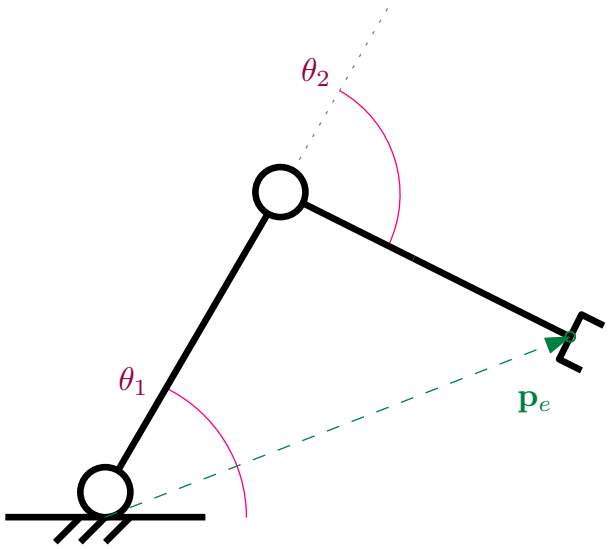


Figure 1.6: a 2R planar robot

Such a trajectory can be represented by a time-dependent function that starts from an initial time $t_0 = 0$ until a final time T, making s a monotonically increasing function of $t \in [0, T]$:

$$s : [0, T] \longmapsto [0, 1] \tag{1.28}$$
$$s(0) = 0 \tag{1.29}$$
$$s(T) = 1 \tag{1.30}$$
$$\mathbf{p}(s) = \mathbf{p}(s(t)) \tag{1.31}$$

We say that a trajectory is rest-to-rest if the velocity of the end effector at the start and at the end of that trajectory is zero:

$$\dot{\mathbf{p}}(s(0)) = \dot{\mathbf{p}}(s(T)) = \mathbf{0} \tag{1.32}$$

we need to include boundary conditions. Considering the chain rule, the derivative of $\mathbf{p}$ respect to the time t is

$$\dot{\mathbf{p}} = \frac{d\mathbf{p}}{dt} = \frac{d\mathbf{p}}{ds}\frac{ds}{dt} = \frac{d\mathbf{p}}{ds}\dot{s} \tag{1.33}$$

since

$$\frac{d\mathbf{p}}{ds} = \frac{d}{ds}\left(s\mathbf{p}_b + (1-s)\mathbf{p}_a\right) = \mathbf{p}_b - \mathbf{p}_a \tag{1.34}$$

we have

$$\dot{\mathbf{p}} = \frac{d\mathbf{p}}{ds}\dot{s} = \dot{s}(\mathbf{p}_b - \mathbf{p}_a) \tag{1.35}$$

the acceleration is

$$\ddot{\mathbf{p}} = \ddot{s}(\mathbf{p}_b - \mathbf{p}_a) + \dot{s} \cdot \mathbf{0} = \ddot{s}(\mathbf{p}_b - \mathbf{p}_a) \tag{1.36}$$

we have that

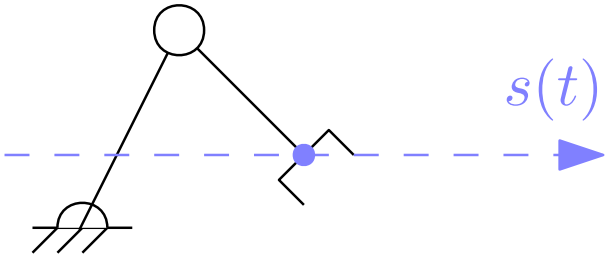
$$\dot{\mathbf{p}}(s(0)) = 0 \iff \dot{s}(0)(\mathbf{p}_b - \mathbf{p}_a) \iff \dot{s}(0) = 0 \tag{1.37}$$
$$\dot{\mathbf{p}}(s(T)) = 0 \iff \dot{s}(T)(\mathbf{p}_b - \mathbf{p}_a) \iff \dot{s}(T) = 0 \tag{1.38}$$

The starting velocity and the final velocity is zero, so the variation of the velocity is zero, this can be seen by the integral of the acceleration

$$\int_0^T \ddot{\mathbf{p}}dt = \int_0^T \ddot{s}(\mathbf{p}_b - \mathbf{p}_a)dt = (\mathbf{p}_b - \mathbf{p}_a)\int_0^T \ddot{s}dt = (\mathbf{p}_b - \mathbf{p}_a)(\dot{s}(T) - \dot{s}(0)) = 0. \tag{1.39}$$

Let's see an example, let's say that the linear position of the end effector along 1 axis is given by the law

$$s(t) \tag{1.40}$$



we have the following profile for the acceleration:

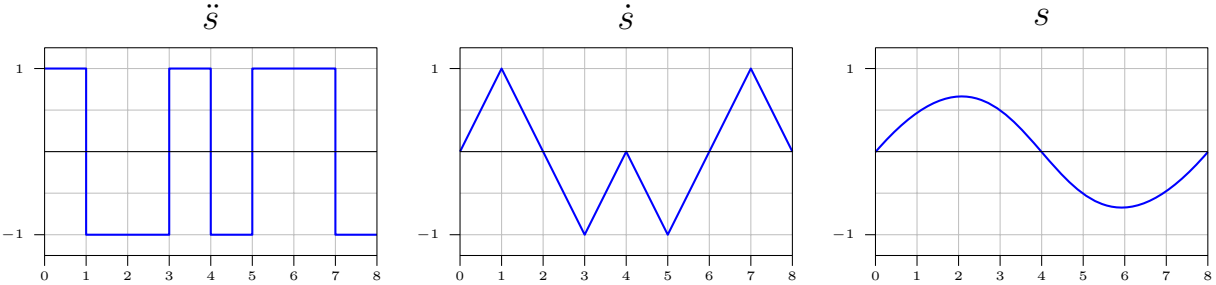
$$\ddot{s}(t) = \begin{cases} 1 \text{ if } t \in [0,1] \cup [\frac{3}{8}, 4] \cup [\frac{5}{8}, \frac{7}{8}] \\ -1 \text{ if } t \in [1, \frac{3}{8}] \cup [4, \frac{5}{8}] \cup [\frac{7}{8}, 8] \end{cases} \tag{1.41}$$

Let's assume that

$$\dot{s}(0) = s(0) = 0 \tag{1.42}$$

- Question: What will be the speed of the end effector at $t = 8$? We can easily see that the integral of $\ddot{s}$ over $t \in [0, 8]$ is 0, so $\Delta\dot{s} = 0 \implies \dot{s}(8) = 0$.
- Question: What will be the position of the end effector at $t = 8$? We can easily see that the integral of $\dot{s}$ over $t \in [0, 8]$ is 0, so $\Delta s = 0 \implies s(8) = 0$.

The acceleration, speed and position profiles are the following:



Now we consider the *control aspects* of the problem, we denote $\mathbf{p}_e(t)$ the position of the end effector at the time t, and $\mathbf{p}_d(t)$ the **desired position** at time t.

$$\mathbf{p}_d(0) = \mathbf{p}_a. \tag{1.43}$$

We define the **error** such as the difference between the current position and the desired position:

$$\mathbf{e}(t) = \mathbf{p}_d(t) - \mathbf{p}_e(t) \tag{1.44}$$

The aim of the *control system* of the robot is to maintain $\mathbf{e}$ as close to zero as possible. This can be done by computing the initial error, and by giving to the system a new command $\dot{\boldsymbol{\theta}}(t)$ to correct it such that $\mathbf{e}(t) \to \mathbf{0}$. Let's denotate the error as follows

$$\mathbf{e}(t) = \begin{pmatrix} e_x(t) \\ e_y(t) \end{pmatrix} \tag{1.45}$$

For now, we will not discuss in detail how to control the error through the control of joint velocities $\dot{\boldsymbol{\theta}}$; it is sufficient to know that the following condition is required:

$$\dot{\mathbf{e}}(t) = -K\mathbf{e}(t) = \begin{pmatrix} -k_x & 0 \\ 0 & -k_y \end{pmatrix} \begin{pmatrix} e_x(t) \\ e_y(t) \end{pmatrix} \tag{1.46}$$

with $k_x, k_y > 0$. Why this conditions is required?

- if $e_x(t)$ is greater than zero, the condition $\dot{e}_x(t) = -k_x e_x(t)$ describes a decrease in error, making it approach zero
- if $e_x(t)$ is smaller than zero, the condition $\dot{e}_x(t) = -k_x e_x(t)$ describes an increase in error, making it approach zero

- same for e_y.

The system of equations

$$\dot{\mathbf{e}}(t) = \begin{pmatrix} -k_x & 0 \\ 0 & -k_y \end{pmatrix} \begin{pmatrix} e_x(t) \\ e_y(t) \end{pmatrix} \implies \begin{cases} \dot{e}_x(t) = -k_x e_x(t) \\ \dot{e}_y(t) = -k_y e_y(t) \end{cases} \tag{1.47}$$

admits exponential functions as a solution

$$e_x(t) = e_x(0)e^{-k_x t} \tag{1.48}$$

$$e_y(t) = e_y(0)e^{-k_y t} \tag{1.49}$$

if the initial error $\mathbf{e}(0)$ is not zero, then the error will approaches zero, without never reaching it. For practical applications it goes sufficiently fast to values very close to zero.

We introduce now an important concept in linear differential equations systems.

Definition 1 *Let $A \in M_{n,n}(\mathbb{R})$ to be a squared real-valued matrix. The **matrix exponential**, denoted e^A, is the $n \times n$ matrix defined as follows:*

$$e^A = \sum_{k=0}^{\infty} \frac{A^k}{k!} \tag{1.50}$$

Given a linear system

$$\dot{\mathbf{x}} = A\mathbf{x} \tag{1.51}$$

the solution is

$$\mathbf{x}(t) = e^{At}\mathbf{x}(0) \tag{1.52}$$

In some cases the exponential matrix can be computed easily, let's assume that A is diagonal

$$A = \begin{pmatrix} a_1 & 0 & \cdots & 0 \\ 0 & a_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_n \end{pmatrix} \tag{1.53}$$

in this case we have that

$$A^k = \begin{pmatrix} a_1 & 0 & \cdots & 0 \\ 0 & a_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_n \end{pmatrix} \times \cdots \times \begin{pmatrix} a_1 & 0 & \cdots & 0 \\ 0 & a_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_n \end{pmatrix} = \begin{pmatrix} a_1^k & 0 & \cdots & 0 \\ 0 & a_2^k & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_n^k \end{pmatrix} \tag{1.54}$$

so

$$e^A = \sum_{k=0}^{\infty} \frac{1}{k!} \begin{pmatrix} a_1^k & 0 & \cdots & 0 \\ 0 & a_2^k & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_n^k \end{pmatrix} = \begin{pmatrix} \sum_{k=0}^{\infty} \frac{a_1^k}{k!} & 0 & \cdots & 0 \\ 0 & \sum_{k=0}^{\infty} \frac{a_2^k}{k!} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sum_{k=0}^{\infty} \frac{a_n^k}{k!} \end{pmatrix} \tag{1.55}$$

$$= \begin{pmatrix} e^{a_1} & 0 & \cdots & 0 \\ 0 & e^{a_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & e^{a_n} \end{pmatrix}. \tag{1.56}$$

So, if a system is described by a diagonalizable matrix A there exists a diagonal matrix Λ and an invertible matrix T such that

$$A = T\Lambda T^{-1} \tag{1.57}$$

in this case we can easily calculate the exponential matrix

$$e^{At} = T^{-1}e^{\Lambda t}T. \tag{1.58}$$

Note: The Sections 1.4, 1.6, 1.7, 1.5, 1.8 are written with the aid of Gemini, by having the model process the information taken from the professor's slides and the lecture notes.

1.4 Defining Robots

The concept of a robot is formalized through several key definitions, spanning from strict industrial standards to more encompassing theoretical perspectives.

1.4.1 Standardized Definitions

1. Industrial Definition (Robotic Institute of America - RIA)

The RIA defines a robot as a **re-programmable, multi-functional manipulator** designed to move material, parts, tools, or specialized devices through variable programmed motions for the performance of a variety of tasks. A crucial element of this definition is the requirement that the robot must **acquire information from the environment** and move intelligently in response, setting it apart from simpler automated machinery.

2. ISO 8373:2012 Definition

The International Organization for Standardization (ISO) provides a formal, international standard: an industrial robot is an **automatically controlled, re-programmable, multi-purpose manipulator** programmable in **three or more axes**. The manipulator may be either **fixed in place or mobile** and is intended for use in industrial automation applications. This specific definition helps to delineate complex, versatile machines from basic mechanical devices.

1.4.2 The Visionary Definition: Perception and Action

A broader, more "visionary" definition emphasizes the cognitive and functional aspects of robotic systems: the **intelligent connection between perception and action**.

- **Perception:** This is the process of acquiring and processing sensing information from the environment.
- **Action:** This involves not just controlling the robot's current state, but actively **making some changes in the physical world** to achieve a goal.

This relationship forms a continuous feedback loop that governs autonomous behavior:

$$\text{percept} \longrightarrow \text{action} \longrightarrow \text{percept}$$

The robot's understanding of its environment (*percept*) drives its movement or operation (*action*), which modifies the environment, leading to a new cycle of perception.

1.5 Notable Robot Examples

Throughout history, various robots have exemplified different facets of the definition of robotics, from pure industrial work to exploration and human interaction.

- **Comau H4 (1995):** Representing the industrial segment, these manipulators were widely used in **automotive industries** (e.g., owned by Fiat at the time). They are a classic example of fixed, multi-functional, re-programmable automation.
- **Waseda WAM-8 (1984):** This famous **humanoid robot** from a Japanese university demonstrated early cognitive abilities. It was capable of complex tasks such as playing an organ and reading music from a sheet, combining perception (reading) and fine manipulation.
- **Spirit Rover (2002):** An excellent example of **autonomous mobile robotics and exploration**.
  - It was landed on Mars, featuring articulated wheels and solar panels.
  - Its mission was to move, analyze material, and send gathered information back to Earth.
  - While the **global target** is **specified remotely**, the rover must operate with significant **local autonomy** because of the approximately 8-minute communication delay required to receive instructions from Earth. This necessity highlights the critical role of the onboard perception-action loop for mission success.

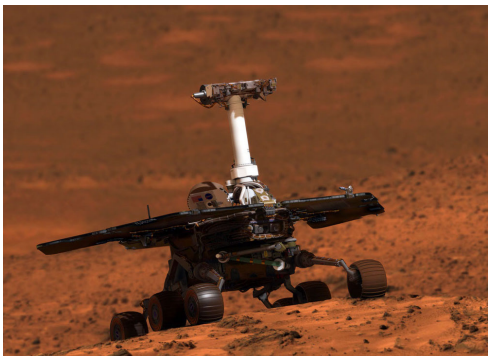


Figure 1.7: Spirit Rover (2002)

According to the rigorous **ISO 8373:2012** definition, certain devices and systems are **not considered robots**. These exclusions are typically applied to specific devices with only **one or two degrees of freedom (DOF)** and complex software systems lacking the physical manipulator required by the standard. Systems that are not classified as robots under the ISO 2012 standard include:

- Software "bots", Artificial Intelligence (AI), and Robotic Process Automation (RPA).
- Voice assistants.
- Automatic Teller Machines (ATMs).
- Cooking machines, smart washing machines, and similar appliances.

Furthermore, advanced mobile systems like drones and autonomous cars are generally not classified as robots under this definition. However, in a **2021 revision**, the term **robotic device** was introduced to encompass these increasingly sophisticated, automated machines that fall outside the strict definition of an industrial robot.

The word "robot" has an historical and literary origin that is foundational to the field:

- The term derives from the Slavic word **Robota**, meaning "work" or "forced labor."
- The first recorded use of the word "Robot" in a theatrical context was in **1920** by Czech writer **Karel Čapek** in his science-fiction play, *Rossum's Universal Robots (R.U.R.)*. In the play, "robots" are artificial, human-like creatures created to be inexpensive workers.

1.6 The Ethical Framework: Asimov's Three Laws of Robotics

The science fiction author Isaac Asimov defined a set of foundational ethical rules for robotics in his short stories.

1. **First Law:** A robot may not injure a human being or, through inaction, allow a human being to come to harm.
    - This law is fundamental to modern **collaborative robotics** (cobots), ensuring human safety is prioritized as robots and humans work in close proximity.
2. **Second Law:** A robot must obey orders given to it by human beings, except where such orders would conflict with the First Law.
    - This establishes the robot's subordination to human command. Situations like robots used in war or faulty programming clearly violate this rule.
3. **Third Law:** A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.
    - The robot's self-preservation is conditional, being secondary to human safety and commands.

1.7 Evolution and Characteristics of Robot Manipulators

The journey toward the industrial robot began around the 1950s with the convergence of **Computerized Numerically Controlled (CNC) machines** and **mechanical telemanipulators**. This synthesis led to the development of the first **robot manipulators**, with the **Unimation PUMA (1970)** being a key early example.

Unlike the early mechanical telemanipulators, which often required continuous human control, true robot manipulators offered several distinct advantages:

- **Absence of Position Memory:** The robot operates based on its programmed coordinates, not needing to remember past states.
- **Adaptivity** to conditions previously unknown.
- High **Accuracy** in positioning.
- Superior **Repeatability** of operation (the ability to return consistently to a programmed point).

For an industrial manipulator, it is often noted that **repeatability** and **compliance** (adaptivity to variations) are more fundamental for task success than absolute accuracy.

The history of industrial robotics is marked by key patented designs and technological firsts: **The First Industrial Robot** The very first industrial robot was installed at a General Motors plant in **1961**. It was developed by **George Devol and Joseph Engelberger** of Unimation.

- **Kinematics:** This design featured a total of **6 Degrees of Freedom (DOF)**, comprising five revolute (rotational) joints and one prismatic (linear) joint. This combination was considered the optimal solution at the time to achieve **full control over the end effector's pose** (position and orientation).

Key Successor Robot Manipulators Following the first installation, several robots introduced foundational innovations:

- **ASEA IRB-6 (1973):** The first robot where all axes were driven by **electric motors** (all-electric drives), featuring 5 DOF.
- **Cincinnati Milacron T3 (1974):** Recognized as the first industrial robot to be controlled by a **micro-computer**.
- **Hirata AR-300 (1978):** Introduced the first **SCARA (Selective Compliance Assembly Robot Arm)** robot, which has a distinct cylindrical workspace, prioritizing speed and rigidity in the vertical axis.
- **Unimation PUMA 560 (1979):** Characterized by its 6 revolute joints, this was the first truly **'anthropomorphic'** robot, offering human-like dexterity.

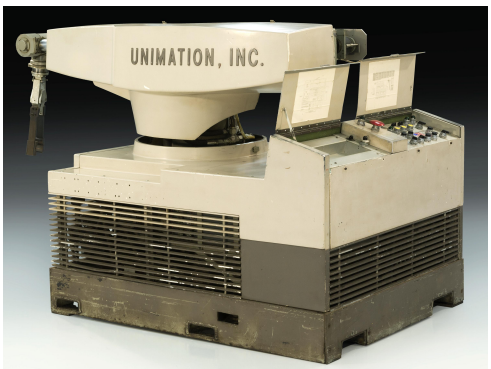


Figure 1.8: Unimation PUMA 560 (1979)

Actuators power the robot and sustain its payload. **Electric motors** are the most common choice, converting electrical energy to torque. However, when required to sustain a **heavy payload**, a **hydraulic actuator** generally works better due to its higher power-to-weight ratio.

1.8 Global Industrial Robotics Market Statistics

The following statistics are sourced from the **International Federation of Robotics (IFR)** World Robotics documents (Executive Summary for 2025 statistics). These figures illustrate the rapid global expansion of industrial automation.

1.8.1 Operational Stock and Growth Rates

The total worldwide operational stock of industrial robots reached **4.6 million units** at the end of 2024. This represents a substantial growth of **+9%** compared to 2023. Over the five-year period from 2019 to 2024, the market demonstrated a robust Compound Annual Growth Rate (**CAGR**) of **+11%**.

The Compound Annual Growth Rate is calculated as:

$$\text{CAGR} = \left(\frac{V_{\text{end}}}{V_{\text{begin}}} \right)^{1/\text{years}} - 1$$

New robot sales in 2024 reached **542,000 units**, maintaining stability (±0%) compared to 2023, and demonstrating a +7% CAGR from 2019–2024. This marks the **fourth consecutive year** that annual new installations have surpassed 500,000 units. The estimated average service life of an industrial robot is between **12 and 15 years**.

Regarding market size, the value of the robot market (excluding software and peripherals) was **$15.7 billion** in 2022. The value of the total robotic systems market, which includes surrounding equipment and services, is estimated to be approximately **four times** this core market value.

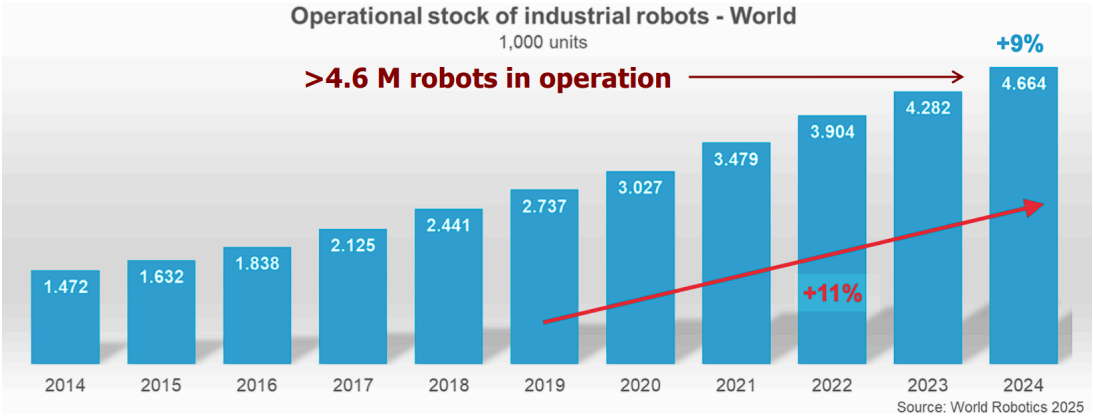
1.8.2 Sectoral and Geographic Distribution

The market growth is primarily driven by the **electronics** and **automotive** industries, with collaborative robots (cobots) also contributing significantly to market expansion.

The global distribution of installations is highly concentrated:

- **China** is the world's largest market for new installations, a position it has held since 2013. China installs **every other robot** globally, accounting for **54%** of all new annual installations.
- A vast majority—**80%** of all new robot installations—occur in just five countries: **China, Japan, USA, Korea, and Germany**.
- Within Europe, **Italy** stands out as the second-largest European country for new installations.

The scale of growth is highlighted by the historical operational stock data: starting from a modest 3,000 units in 1973, stock grew to 66K in 1983, 575K in 1993, and 800K in 2003, culminating in the 4.6 million units recorded in 2024.



CHAPTER

2

SENSORS AND ACTUATORS

From an high level prospective, a robot is a system composed by different units, that takes commands and responds with actions in a working environment, as shown in figure 2.1.

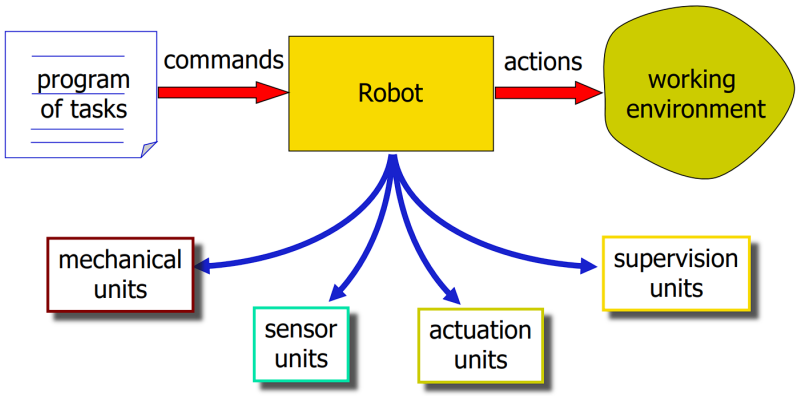


Figure 2.1: Robot as a system

The functional units are the following:

- mechanical units (robot arms): We see the links as rigid bodies, connected by rotational or prismatic joints, with an end effector attached at the end of the serial structure.
- actuation units: we have motors for the joints, that can be electrical, hydraulic or pneumatic, and eventually transmissions (i.e. a belt). We also consider the motion control algorithm as a part of that unit.
- sensor units: proprioceptive sensors that measure the internal state of the robot (position and velocity of the joints) and exteroceptive that measure the external environment.
- supervision units: AI and reasoning, task planning and control.

Let's focus on the actuation system, we can consider the scheme in figure 2.2. We see how the power is measured in different units in different stage of the actuation process:

- Electrical : power = voltage $\times$ current
- Hydraulic : power = pressure $\times$ flow rate

- Linear Mechanics : power = force × speed
- Rotational Mechanics : power = torque × angular speed

fig:system.

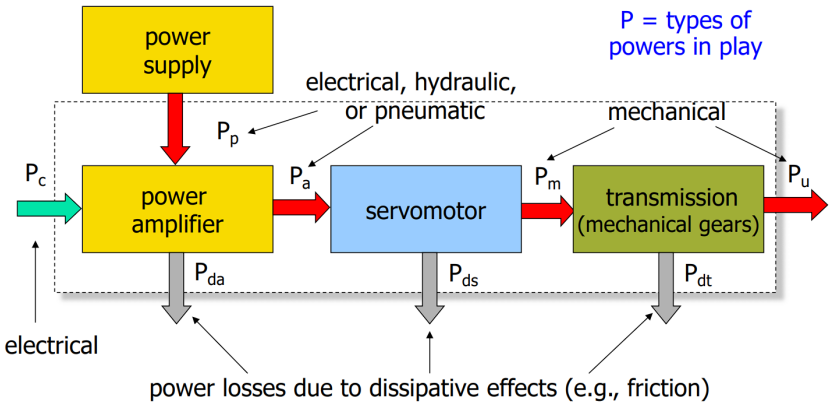


Figure 2.2: Actuation system

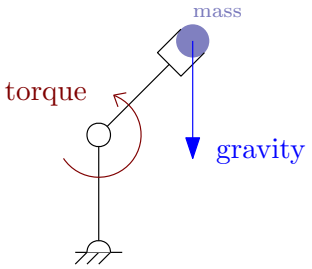
We define the efficiency as the ratio between the output and input power:

$$\text{efficiency} = \frac{P_u}{P_c}. \tag{2.1}$$

2.1 Electrical Motors

Since pneumatic system have difficulties to guarantee high levels of precision, and the hydraulic actuation systems are expensive and needs hydraulic supply, in most of the cases electrical servo motors are mounted on the robots.

- advantages of electrical motors:
  - power supply available everywhere
  - low cost
  - large variety of products
  - high power conversion efficiency
  - easy maintenance
  - no pollution in working environment
- disadvantages
  - overheating in static conditions (in the presence of gravity)
  - need special protection in flammable environments
  - some advanced models require more complex control laws



If a robot should keep an object in a still position, the force of gravity tends to rotate the arms, so torque from the motors must be applied to keep that positions (this does not happen with hydraulic motors).

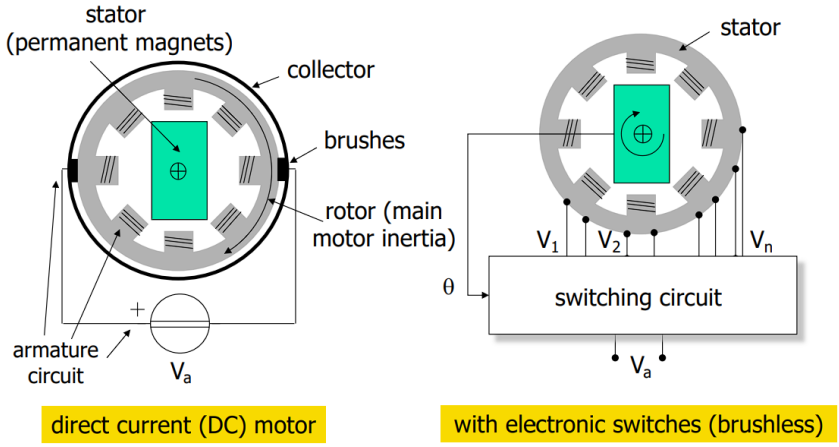
Definition 2 *A **servomotor** is a type of electric motor (AC or DC) specifically designed for the precision control of position, speed, and acceleration. It uses a closed-loop system with a feedback component (such as an encoder) that continuously monitors the motor's current position and compares it to the desired position. If there is a difference (an error signal), the controller corrects it, ensuring extremely accurate and dynamic motion.*

Desired characteristics for a servomotor mounted on a robot are:

- low inertia and high power-to-weight ratio
- high acceleration capabilities and large range of operational velocities (1 to 2000 round per minute)
- low torque ripple[1] and high accuracy in positioning
- power: 10 W to 10 kW.

There are two types of electrical motors

- Brushed DC: Has windings on the rotor (rotating part) and permanent magnets on the stator (stationary part). It uses physical brushes rubbing against a commutator to mechanically switch the current direction and maintain rotation.
- Brushless DC (BLDC): Has permanent magnets on the rotor and windings on the stator. It uses an electronic controller (instead of brushes/commutator) to electrically switch the current to the windings for rotation.



The DC motor has a complex construction, but the mathematical model that describes it is simple. The rotor (the rotating part) includes copper windings (coils) through which current circulates in a determined direction; power is supplied thanks to the contact of brushes that rub against the motor. The entire assembly is immersed in a magnetic field provided by permanent magnets, as shown in figure 2.3.

The electrons will move in the direction of the vector $\bar{i}$ (current density), with intensity I, and due to the presence of a magnetic field $\bar{B}$, they will be subjected to the **Lorentz force**

$$\bar{F} = I(\bar{l} \times \bar{B}) \tag{2.2}$$

where $\bar{l}$ represents the section of wire. In this way, a torque will be generated, causing the circuit to rotate. The brushes, colored yellow in figure 2.3, by making contact, ensure that the current circulates. Note how the slip ring is divided into two parts; it acts as a *commutator*. This construction allows the current to reverse direction with every half-turn. If this were not the case, the Lorentz force would alternate, causing the rotor to oscillate until it stopped, without rotating, as shown in figure 2.4.

The **torque** is equal to

$$\bar{T} = (\bar{r}_1 \times \bar{F}_1 + \bar{r}_2 \times \bar{F}_2) \tag{2.3}$$

We denote $\tau = \pm|\bar{T}|$ the *scalar torque*.

[1] Torque ripple is an issue that arises in motors due to their construction, and it will be explained later.

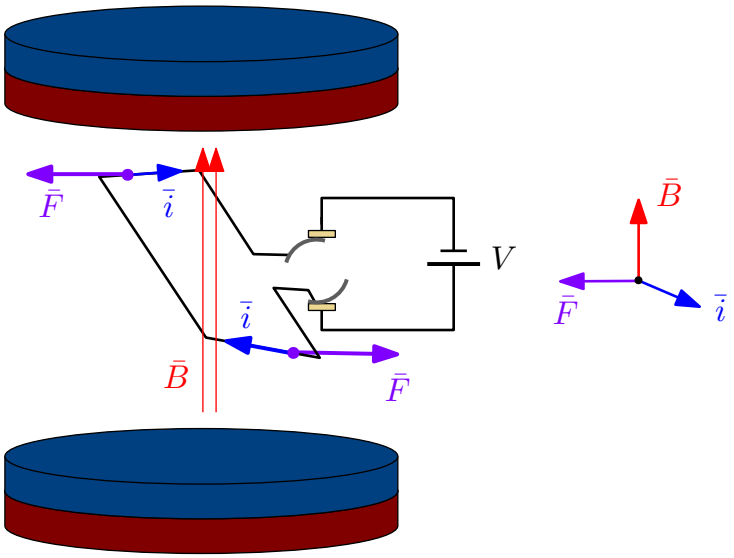


Figure 2.3: brush DC motor

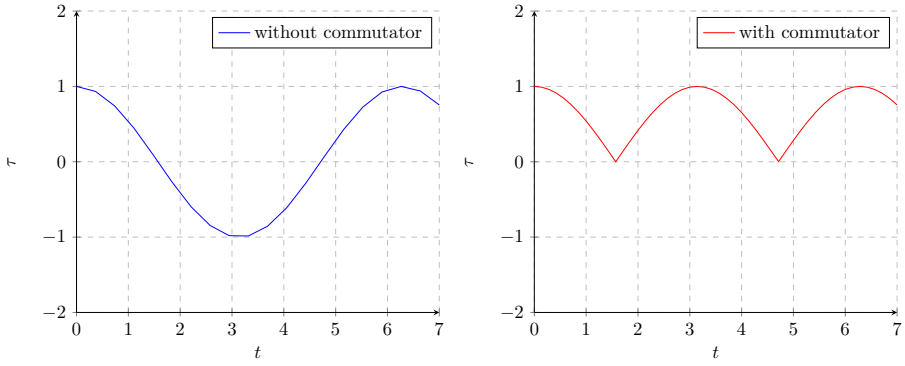
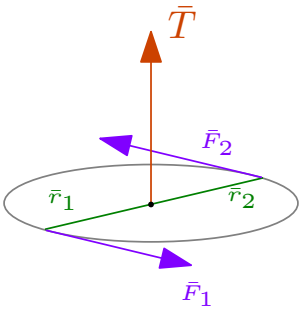
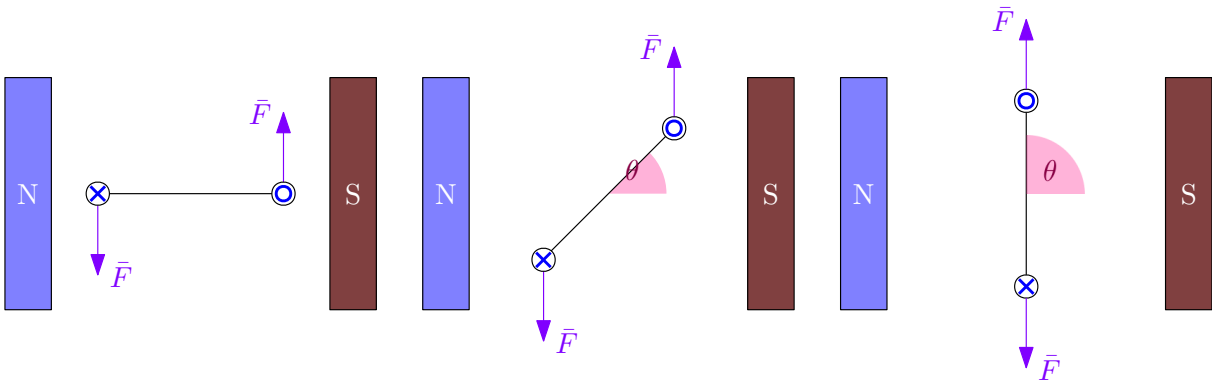


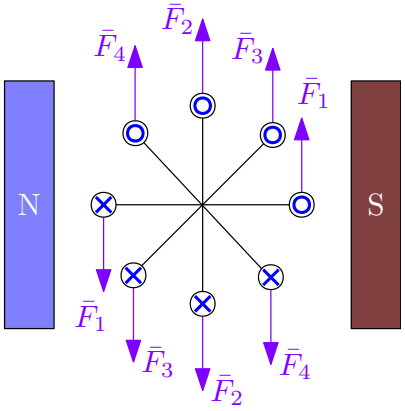
Figure 2.4: Torque profile τ



The force $\bar{F}$ depends on the angle θ of the motor's rotation, because the change in the direction of the current $\bar{i}$ can cause the force to attenuate. The torque becomes zero every time the angle between the generated force and the lever arm of rotation is $k\pi$, where $k \in \mathbb{N}$.



maximum torque null torque



Because of this, the torque has an oscillatory behavior and is not constant; this phenomenon is called **torque ripple**. To limit this (and increase the torque), multiple coils can be added so that, at the instant one coil produces zero torque, another coil produces maximum torque.

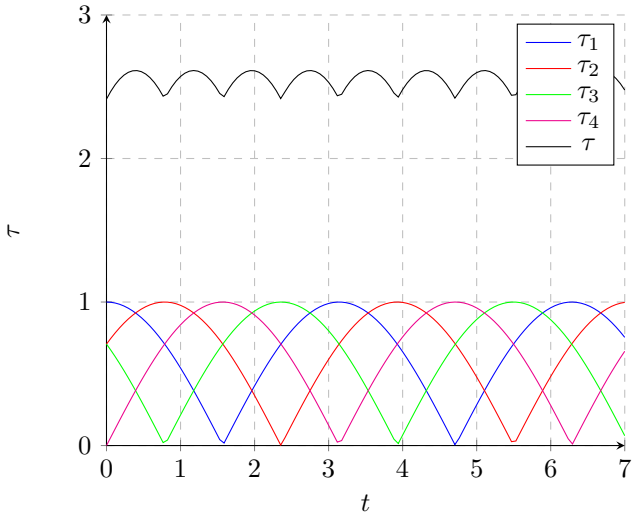


Figure 2.5: torque generated by multiple coils.

The greater the number of coils included, the more the torque "flattens out", tending towards a constant behavior.

2.1.1 Dynamical Model of the Motor

Now let's model an electric motor as a dynamical system. The motor is composed by two main sub-model

- An electromagnetic model
- A mechanical model

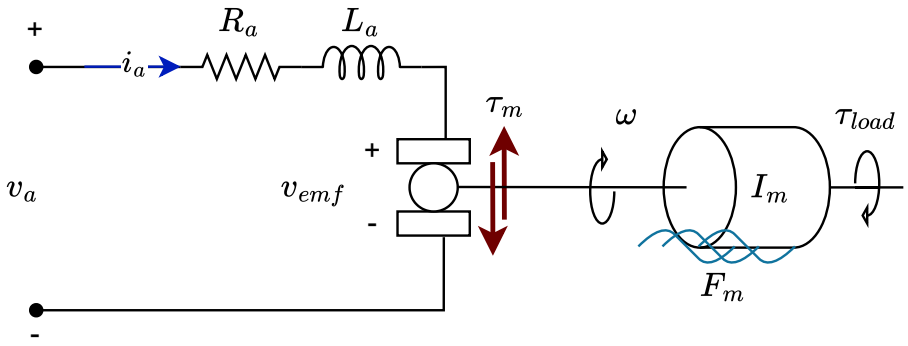


Figure 2.6: Dynamical Model

The commutator circuit have in series a resistor and an inductor, the model is shown in figure 2.6.
in the electromagnetic sub model, the voltage v_a is given by the Kirchhoff laws

$$v_a(t) = R_a i_a(t) + L_a \frac{di_a}{dt} + v_{emf}(t) \tag{2.4}$$

where

- R_a is the resistance coefficient
- L_a is the inductance.

The term $v_{emf}(t)$ is called **Back Electromotive Force** and is a voltage generated within the rotating armature of an electric motor, based on Faraday's Law of Induction, when the motor's armature coils $\omega(t)$ rotate and cut across the magnetic field (produced by the field coils or permanent magnets), a voltage is induced across the armature windings. Since a motor in motion is essentially acting as a generator, this is often called generator action. This voltage is proportional to the angular speed of the motor

$$v_{emf}(t) = k_v \omega(t) \tag{2.5}$$

Let's consider the mechanical sub model, we apply the newton law on torques, the scalar torque $\tau_m(t)$ of the rotating body is given by

$$\tau_m(t) = I_m \frac{d\omega}{dt} + F_m \omega(t) + \tau_{load}(t) \tag{2.6}$$

where

- I_m is the moment of inertia of the motor, a larger moment of inertia means the motor will take more torque and time to accelerate or decelerate to a given speed.
- F_m is the viscous friction coefficient (since the motor never rotate in the void), and is proportional to the angular speed. It has a damping action on the rotational speed
- $\tau_{load}(t)$ is the external mechanical torque load given by the weight of the object that the motor is trying to rotate, and it opposes to the input torque.

The two sub models are related by an equation, it states that the torque τ_m of the motor is proportional to the applied current i_a on the circuit:

$$\tau_m(t) = k_t i_a(t) \tag{2.7}$$

Let's define the state variables of our model, we can apply a voltage $v_a(t)$, and we are interested in the resulting angular position θ and speed $\dot{\theta} = \omega$.

$$\begin{cases} v_a(t) = R_a i_a(t) + L_a \dfrac{di_a}{dt} + v_{emf}(t) \\ \tau_m(t) = I_m \dfrac{d\omega}{dt} + F_m \omega(t) + \tau_{load}(t) \end{cases} \tag{2.8}$$

by using the equations (2.7) and (2.5) we get

$$\begin{cases} v_a(t) = R_a i_a(t) + L_a \dfrac{di_a}{dt} + v_{emf}(t) \\ k_t i_a(t) = I_m \dfrac{d\omega}{dt} + F_m \omega(t) + \tau_{load}(t) \end{cases} \implies \tag{2.9}$$

$$\begin{cases} L_a \dfrac{di_a}{dt} = v_a(t) - R_a i_a(t) - k_v \omega(t) \\ I_m \dfrac{d\omega}{dt} = k_t i_a(t) - F_m \omega(t) - \tau_{load}(t) \\ \dfrac{d\theta}{dt} = \omega(t) \end{cases} \tag{2.10}$$

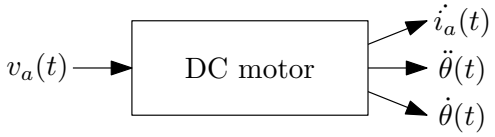
in the dot notation for derivatives:

$$\begin{cases} L_a \dot{i_a} = v_a(t) - R_a i_a(t) - k_v \omega(t) \\ I_m \ddot{\theta} = k_t i_a(t) - F_m \omega(t) - \tau_{load}(t) \\ \dot{\theta} = \omega(t) \end{cases} \tag{2.11}$$

the state variables are

$$\mathbf{x} = \begin{pmatrix} \dot{i_a} \\ \ddot{\theta} \\ \dot{\theta} \end{pmatrix} \tag{2.12}$$

the position θ can be found by integrating ω. The motor is a system that given a input voltage over time $v_a(t)$, returns the electric current (more precisely, it's derivative, i_a can be found by integrating over time), the angular speed and the acceleration of the motor.



About the notation: if $f(t)$ is a function in the time domain, we denote $\mathcal{L}[f](s) = f(s)$ it's Laplace transform in the complex variable s. For example:

$$\mathcal{L}[i_a](s) = i_a(s) \tag{2.13}$$

Let's analyze the model in the Laplace domain by considering the Laplace transform, and then draw the block diagram of the system. For the electrical model we have

$$\mathcal{L}\left[L_a \dot{i_a} = v_a(t) - R_a i_a(t) - k_v \omega(t)\right] \implies \tag{2.14}$$

$$sL_a i_a(s) = v_a(s) - R_a i_a(s) - k_v \omega(s) \implies \tag{2.15}$$

$$sL_a i_a(s) + R_a i_a(s) = v_a(s) - k_v \omega(s) \implies \tag{2.16}$$

$$(sL_a + R_a) i_a(s) = v_a(s) - k_v \omega(s) \implies \tag{2.17}$$

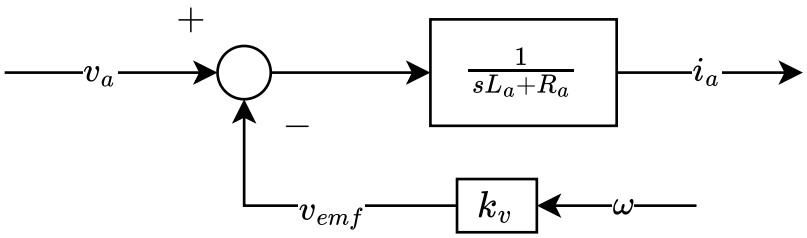
$$i_a(s) = \frac{v_a(s) - k_v \omega(s)}{sL_a + R_a} \tag{2.18}$$

assuming that the quantities are null in $t = 0$. In the end, we get:

$$i_a(s) = \frac{v_a(s) - k_v \omega(s)}{sL_a + R_a} \tag{2.19}$$

v_a and v_{emf} are external input to the model, so the transfer function for the electromagnetic sub model is

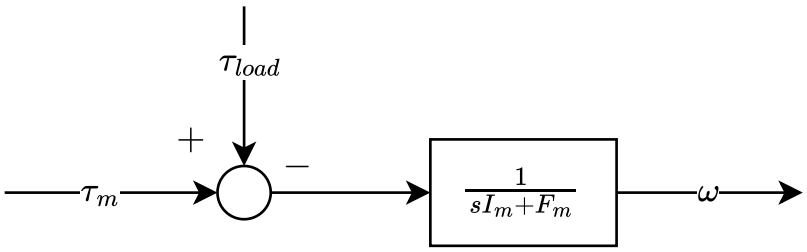
$$\frac{1}{sL_a + R_a} \tag{2.20}$$



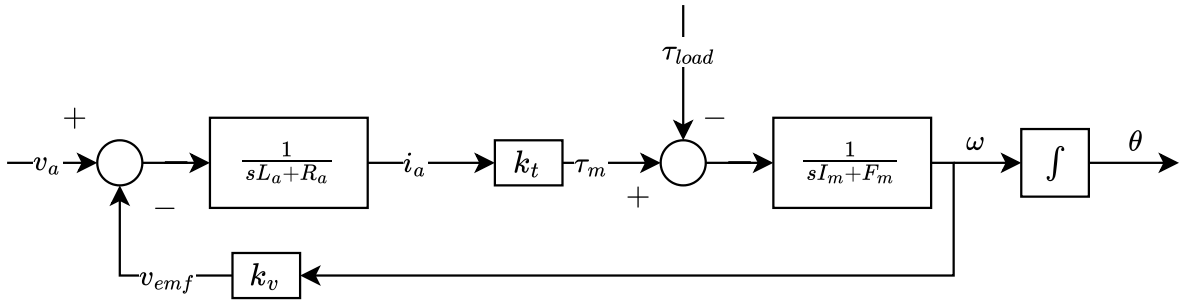
By analogous steps, we find that the transfer function for the mechanical sub model is

$$\frac{1}{sI_m + F_m} \tag{2.21}$$

where the input is the torque τ_m from which we subtract the torque load τ_{load}.



We can chain the models since $\tau_m = k_t i_a$



Proposition 1 *The two constants k_v and k_t are equals*

$$k_v = k_t. \tag{2.22}$$

Proof: This is trivially true. The electric power generated by the EM sub model is

$$v_{emf} i_a \tag{2.23}$$

the mechanical power is

$$\tau_m \omega \tag{2.24}$$

since the conservation of power holds in energy transformations:

$$v_{emf} i_a = \tau_m \omega \tag{2.25}$$

applying equations (2.5),(2.7) we get

$$v_{emf} i_a = \tau_m \omega \implies \tag{2.26}$$
$$k_v \omega i_a = k_t i_a \omega \implies \tag{2.27}$$
$$k_v = k_t. \tag{2.28}$$

■

2.2 Transmissions

We make use of motion transmission gears to:

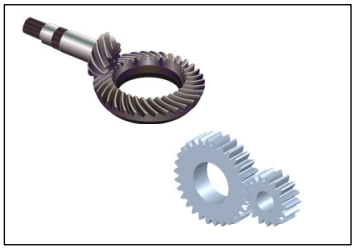
- optimize the transfer of mechanical torque from actuating motors to driven links
- quantitative transformation of torque/velocity, since these two can be controlled, by letting constant the ratio
- qualitative transformation from rotational motion to linear (and vice versa)

- allow improvement of static and dynamic performance by reducing the weight of the actual robot structure in motion (locating the motors remotely, closer to the robot base)

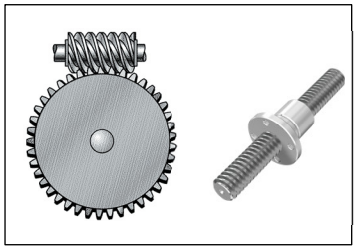
There are different kinds of transmissions in robotics:

- **spur/bevel gears**: modify direction and/or translate axis of motor displacement.
  - *Problems*: deformations, backlash
- **lead screws, worm gearing**: convert rotational into translational motion (prismatic joints)
  - *Problems*: friction, elasticity, backlash
- **toothed belts and chains**: dislocate the motor with respect to the joint axis
  - *Problems*: belts are subject to compliance and larger mass at high speed can induce vibrations to the chains.
- **harmonic drives**: compact, in-line, power efficient, with high reduction ratio (up to 150-200:1)
  - *Problems*: elasticity.
- **transmission shafts**: long, inside the links, with flexible couplings for alignment

spur/bevel gears

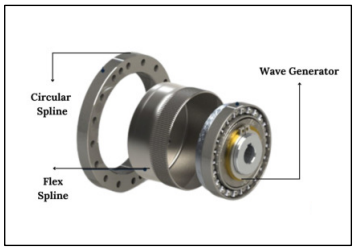


lead screws, worm gearing



toothed belts and chains

harmonic drives



transmission shafts

2.2.1 Harmonic Drives

A Harmonic Drive, also known as a Strain Wave Gear, is a high-precision, compact gear system capable of achieving very high reduction ratios in a single stage, with near-zero backlash. They are widely used in robotics and aerospace.

The core of a Harmonic Drive mechanism involves the elastic deflection of a flexible gear component. It consists of three main components:

1. Wave Generator (Input): This is typically an elliptical plug fitted with a specialized, thin-raced ball bearing. It serves as the input element, usually attached to the motor shaft. As it rotates, its elliptical shape is transmitted to the next component.
2. Flexspline (Output): A thin-walled, flexible metal cup with external gear teeth around its open edge. When the wave generator is inserted, the flexspline conforms to the elliptical shape, causing its teeth to engage with the circular spline at two opposite regions (the major axis of the ellipse) and disengage at the minor axis. The output shaft is typically connected to the rigid base of this cup.

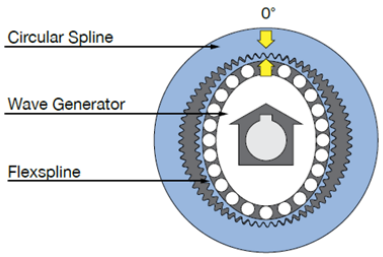
3. Circular Spline (Fixed): A rigid ring with internal teeth that meshes with the flexspline's external teeth. The key is that the Circular Spline has two more teeth than the Flexspline. It is usually fixed to the housing.

Given

- N_{CS} : number of inner teeth
- N_{FS} : number of outer teeth, $N_{CS} = N_{FS} + 2$ since the Circular Spline typically has two more teeth than the Flexspline

the reduction ratio n is

$$n = \frac{N_{FS}}{N_{CS} - N_{FS}} = \frac{N_{FS}}{2} \tag{2.29}$$



The video in the following link:

`https://www.youtube.com/watch?v=bzRh672peNk`

shows in details how a harmonic drives works.

2.2.2 Reduction Ratio

We can see the transmission gear system as a black box that takes in input mechanical power P_m from the motors, and gives in output mechanical power to the links P_u, the mechanical power in this context is given by the product of the torque generated by the motors and the angular speed:

$$P_m = \tau_m \dot{\theta}_m \tag{2.30}$$
$$P_u = \tau_u \dot{\theta}_u \tag{2.31}$$

there are some power that are dissipated due to the friction:

$$P_u = P_m - P_d, \qquad P_d > 0 \tag{2.32}$$

in the ideal case there are no friction:

$$P_m = \tau_m \dot{\theta}_m = \tau_u \dot{\theta}_u = P_u \tag{2.33}$$

every transmission gear have a reduction ratio n, such that

$$\dot{\theta}_m = n\dot{\theta}_u \tag{2.34}$$
$$\tau_u = n\tau_m \tag{2.35}$$

reducing the angular speed will increase the torque and vice versa.

Let's assume that we would like to provide a desired angular acceleration $\ddot{\theta}_u = a$ to the link trough the transmission gear, we have

$$\ddot{\theta}_u = a \implies \ddot{\theta}_m = na$$

Let

- J_m to be the moment of inertia of the motor
- J_u is the moment of inertia of the load

to achieve $\ddot{\theta}_u = a$ the motor should provide a torque

$$\tau_m = J_m \ddot{\theta}_m + \frac{1}{n}(J_u \ddot{\theta}_u) = \left(J_m n + \frac{J_u}{n} \right) a \tag{2.36}$$

we want to choose the ratio n that minimize the sufficient torque τ_m to achieve $\ddot{\theta}_u = a$. We set

$$\frac{\partial \tau_m}{\partial n} = 0 \implies \tag{2.37}$$

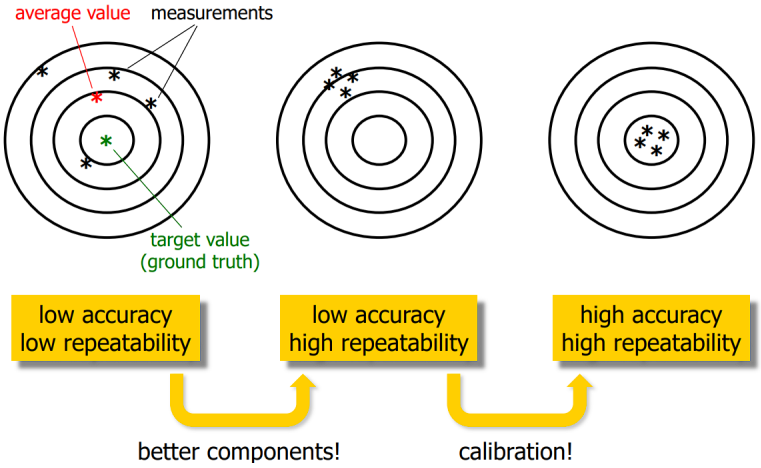
$$\left(J_m + \frac{J_u}{n^2} \right) a = 0 \implies \tag{2.38}$$

$$n = \sqrt{\frac{J_u}{J_m}} \tag{2.39}$$

2.3 Sensors

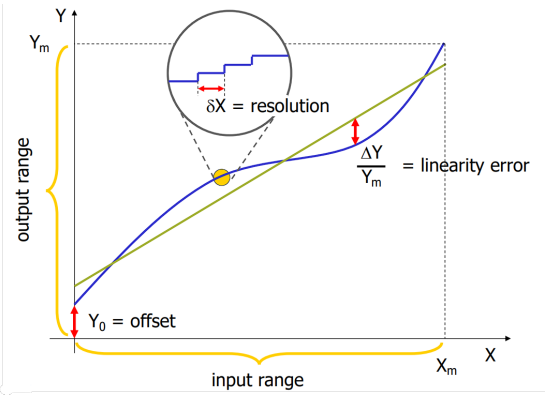
A measurement system should be **accurate**, the measured values should agree with the reference standard (usually given in ideal cases). Consecutive measurements of the same constant input quantity should reproduce similar measurement output (**repeatability**), and it should be **stable**, by keeping the same measuring characteristics over time. Usually

- better components leads to better repeatability
- calibration leads to high accuracy



There are other properties of a measurement system:

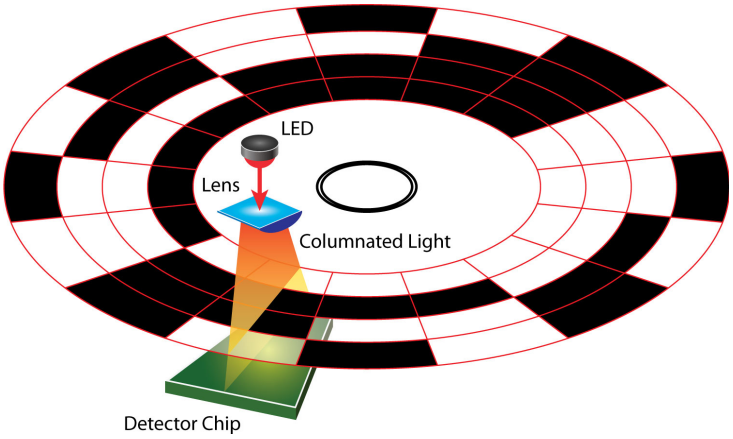
- the linearity error is the maximum deviation of the measured output from the straight line that best fits the real characteristics, this is the "error" of the measurement system if we assume that the process that we are measuring produces output that are linear in the input.
- the offset error is the value of the measured output for the null input
- the resolution error is maximum variation (measured in %) of the input quantity producing no variation of the measured output



2.3.1 Absolute and Incremental Encoders

The **absolute encoders** are angular position sensors, highly used in robotics, these are optical sensor, that uses light pulses to measure the absolute angular position. An encoder works in the following way:

- there are a rotating optical disk, with alternate transparent and opaque sectors on multiple concentric tracks
- a light beam are emitted, it passes trough these opaque and transparent sensor and then hits a photo-receiver
- the intensity of the lights received depends on the tracks, in this way, the intensity is a measure of the position of the rotating part
- light pulses are converted into electrical pulses, electronically processed and transmitted in output



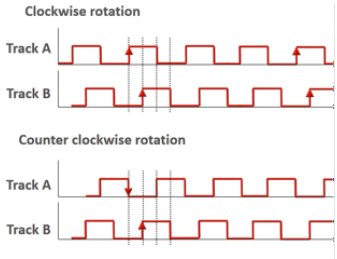
We denote N_t the number of tracks, there is a digital encoding of the absolute position, stored in N_t bits. The resolution of an absolute encoder is

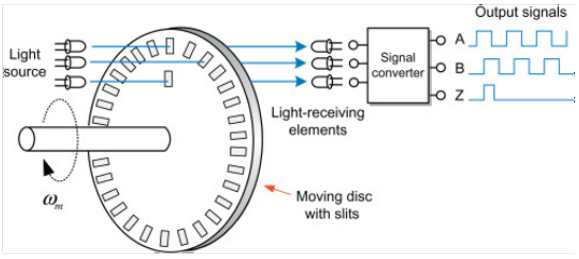
$$\frac{360}{2^{N_t}} \text{ degrees} \tag{2.40}$$

The alternation of opaque and transparent sectors leads to 2^{N_t} possible combination that can be detected.

The **incremental encoders** works differently, they have only 3 tracks, usually called A, B and Z, alternating transparent and opaque areas. Instead of measuring the angular position, the lights detects angular displacements, each displacement (change of the light received) is a pulse, we denote N_e the number of pulses per turn.

- The first two tracks A and B of the encoder are "shifted" and necessary to detect the rotation direction.
- The lights passes trough the two tracks and are received as two signals (one for each tracks).





The third track Z is used to define a zero reference position with a reset of the counter of the pulses. We define one **electrical degree** as

$$\frac{1 \text{ mechanical degree}}{N_e} \tag{2.41}$$

360 electrical degrees (one full electrical turn) corresponds to $\frac{360}{N_e}$ mechanical degrees. The two signals A and B are shifted of 90° electrical degrees. In that case, the signals A and B are always out of phase of 90 degrees, if A anticipates B, the rotation direction is clockwise, else, if B anticipates A, is clockwise.

The resolution of the incremental encoder depends on the number of slits on the rotating disc.

- since the incremental encoder can count the pulses per second, we can count the pulses C every Δt seconds. The "pulses speed" is
$$f = \frac{C}{\Delta T}$$
- the resolution of the encoder, denoted PPR is the number of pulses per each turn.
- we can compute the degrees for each pulses as
$$AR = \frac{360°}{PPR}$$
- at this point f is the number of pulses per second, AR is the degrees per pulses, angular velocity is the number of degrees per second:
$$\omega = f \cdot AR = \frac{C}{\Delta T} \cdot \frac{360°}{PPR} \quad \frac{\text{degrees}}{\text{seconds}} \tag{2.42}$$

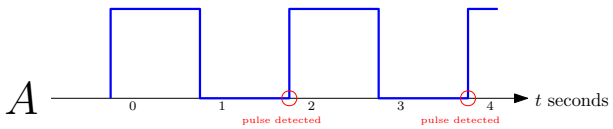
For example, if an encoder have a resolution of $PPR = 1000$ (every 1000 pulses, the motor performs one complete turn), the angular resolution is $\frac{360°}{1000} = 0.36°$, if in an interval of $\Delta t = 0.1$ seconds, we count $C = 200$ pulses, the angular speed of the motor is

$$\frac{200}{0.1}\frac{360}{1000} = 720 \quad \frac{\text{degrees}}{\text{seconds}} \tag{2.43}$$

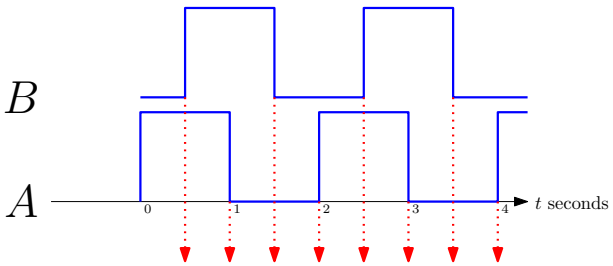
So every second the motor performs two complete revolutions.

2.3.2 Quadrature

For now, we described the "pulses" per second, in details, the encoder counts the rising edges of the signal A:



in the image, in the interval $t \in [0, 4]$, 2 pulses are detected, so the velocity is $\frac{2}{4}$ pulses per seconds. The angular resolution AR is computed as the number of pulses (rising edges of the signal A) per round. We can increase the angular resolution, instead of counting the rising edges of A, we can count the rising and descending edges of the two signals A and B, increasing the angular resolution by 4 times.



In that case instead of counting 2 pulses, we detected 8 pulses by the two signals, in that case the angular resolution become

$$AR = \frac{360°}{PPR \cdot 4} \tag{2.44}$$

Since the pulses per revolut are 4 times greater then the PPR without counting the two signals. We recap the general procedure:

1. Every incremental encoder have a fixed number of slits, we denote PPR (pulses per revolution) this number. This is the number of square waves generated for each complete revolution of the motor.
2. The angular resolution depends on the quadrature

- if we count as pulses the rising edges of the signal A, we have a $D = 1$ encoding
- if we count as pulses the rising edges of A and B, we have $D = 2$ as encoding
- if we count as pulses the rising and descending edges of A and B, we have $D = 4$ as encoding

3. the angular resolution (degrees over pulses) is

$$AR = \frac{360°}{PPR \cdot D} \tag{2.45}$$

2.3.3 Force Sensors

TODO

2.3.4 Vision

TODO

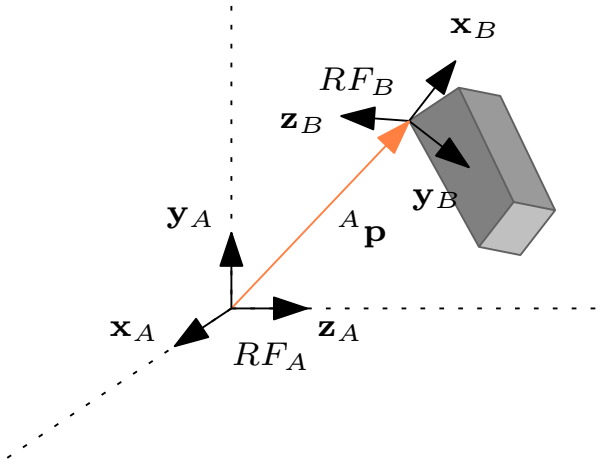
CHAPTER

3

DESCRIBING ORIENTATION

3.1 Position and Orientation of a Rigid Body

We have a base reference frame $RF_A = (\mathbf{x}_A, \mathbf{y}_A, \mathbf{z}_A)$, and a rigid body B in the space, let's say that there is a fixed reference frame $RF_B = (\mathbf{x}_B, \mathbf{y}_B, \mathbf{z}_B)$ attached to a point of B, a vector ${}^A\mathbf{p} \in \mathbb{R}^3$ can describe the position of the whole body respect to the frame RF_A, and a 3×3 matrix is sufficient to describe the orientation of that body.



The orientation of the reference frame RF_B respect to RF_A is described by a matrix R of the following type

- R is orthonormal, the columns vector are orthogonal from each other and have unitary length.
- $R^T = R^{-1} \implies RR^T = I$
- $\det R = 1$

These are the matrices in the orthogonal group $SO(3)$. More precisely, we denote ${}^AR_B$ the matrix that describe the orientation of RF_B respect to RF_A.

$$ {}^AR_B = \begin{pmatrix} {}^A\mathbf{x}_B & {}^A\mathbf{y}_B & {}^A\mathbf{z}_B \end{pmatrix} \tag{3.1} $$

the components of ${}^{A}R_{B}$ are the direction cosines of the axes of RF_B with respect to RF_A. In general, let's say the reference vector of two different frames are

$$(\mathbf{x}_A, \mathbf{y}_A, \mathbf{z}_A) \tag{3.2}$$
$$(\mathbf{x}_B, \mathbf{y}_B, \mathbf{z}_B) \tag{3.3}$$

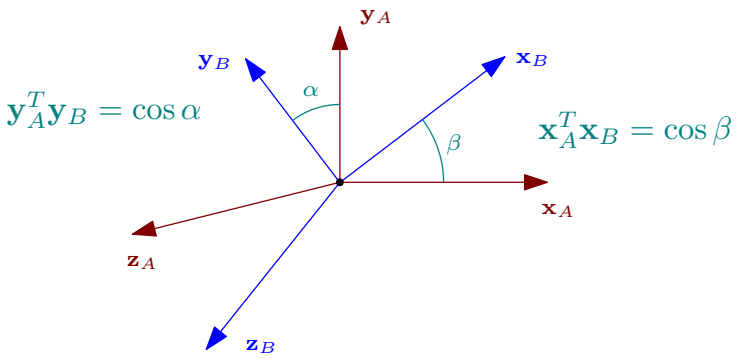
the matrix ${}^{A}R_{B}$ is the following

$${}^{A}R_{B} = \begin{pmatrix} \mathbf{x}_A^T\mathbf{x}_B & \mathbf{x}_A^T\mathbf{y}_B & \mathbf{x}_A^T\mathbf{z}_B \\ \mathbf{y}_A^T\mathbf{x}_B & \mathbf{y}_A^T\mathbf{y}_B & \mathbf{y}_A^T\mathbf{z}_B \\ \mathbf{z}_A^T\mathbf{x}_B & \mathbf{z}_A^T\mathbf{y}_B & \mathbf{z}_A^T\mathbf{z}_B \end{pmatrix} \tag{3.4}$$

since all the basis vectors are unitary, and

$$\mathbf{u}^T\mathbf{v} = \|\mathbf{u}\| \cdot \|\mathbf{v}\| \cdot \cos\alpha \tag{3.5}$$

we have that each component represent the angle between the two considered axis.



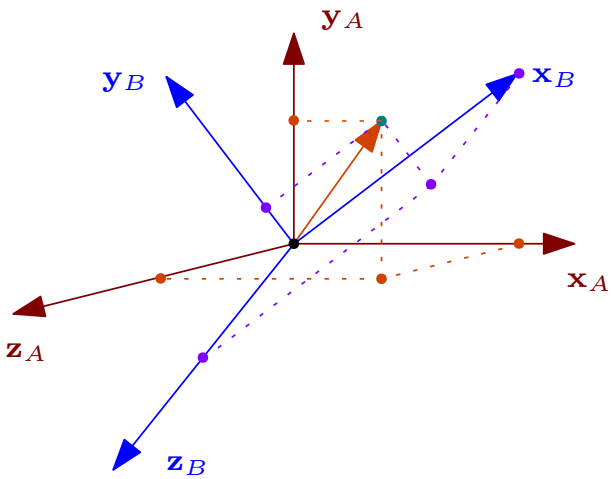
for three different reference frame RF_i, RF_j, RF_k, the **chain rule** holds:

$${}^{k}RF_i\,{}^{i}RF_j = {}^{k}RF_j \tag{3.6}$$

so, if ${}^{A}\mathbf{p}$ is a vector in the frame reference RF_A, the same vector in the reference frame RF_B is

$${}^{B}\mathbf{p} = {}^{B}RF_A\,{}^{A}\mathbf{p} \tag{3.7}$$

so the matrix is used for the **change of coordinates** between the two frames with different orientation.



The position of a rigid body can be expressed in cartesian, cylindrical or spherical coordinates. A point in $\mathbb{R}^3$ in cylindrical coordinates is described by

$$r, \theta, h \tag{3.8}$$

where the transformation from cylindrical to cartesian is

$$x = r\cos\theta \tag{3.9}$$
$$y = r\sin\theta \tag{3.10}$$
$$z = h \tag{3.11}$$

the inverse transformation (assuming $r \geq 0$) is

$$r = \sqrt{x^2 + y^2} \tag{3.12}$$
$$\theta = \text{atan2}(y, x) \tag{3.13}$$
$$h = z \tag{3.14}$$

The atan2(y, x) function is a two-argument arctangent that returns the angle θ in standard position whose terminal side passes through the point (x, y), correctly placing the angle in the correct quadrant (π to π or 0 to 2π). Is defined as follows;

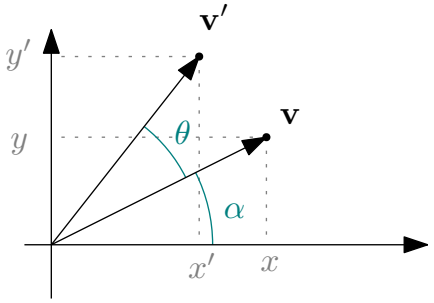
$$\text{atan2}(y,x) = \begin{cases} \text{atan}\left(\frac{y}{x}\right) & x > 0 \\ \pi + \text{atan}\left(\frac{y}{x}\right) & y \geq 0, x < 0 \\ -\pi + \text{atan}\left(\frac{y}{x}\right) & y < 0, x < 0 \\ \frac{\pi}{2} & y > 0, x = 0 \\ -\frac{\pi}{2} & y < 0, x = 0 \\ \text{undefined} & y = 0, x = 0 \end{cases} \tag{3.15}$$

We saw that the rotation matrices can describes the change of coordinates between two frames, and the orientation of a given frame with respect to another. These matrices can also describes the rotation of a vector in $\mathbb{R}^3$, let's consider a rotation around the z axis. We have a vector $\mathbf{v}$ that lies in the xy plane, with coordinates

$$\mathbf{v} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \|\mathbf{v}\|\cos\alpha \\ \|\mathbf{v}\|\sin\alpha \\ z \end{pmatrix}$$

where α is the angle between $\mathbf{v}$ and the x axis. We rotate $\mathbf{v}$ by θ radiant around the z axis, obtaining a new vector $\mathbf{v}'$ with coordinates

$$\mathbf{v}' = \begin{pmatrix} \|\mathbf{v}\|\cos(\alpha+\theta) \\ \|\mathbf{v}\|\sin(\alpha+\theta) \\ z \end{pmatrix} = \begin{pmatrix} x\cos\theta - y\sin\alpha \\ x\sin\theta + y\cos\alpha \\ z \end{pmatrix}$$



Notice how the relation between $\mathbf{v}$ and $\mathbf{v}'$ is given by a matrix $\mathbf{v}' = R_z(\theta)\mathbf{v}$ called the **elementary rotation around** z by θ radiant.

$$R_z(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.16}$$

similarly, are defined elementary rotation around x and y too:

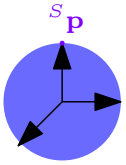
$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix} \qquad R_y(\theta) = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \tag{3.17}$$

So an orthonormal matrix have 3 possible interpretations:

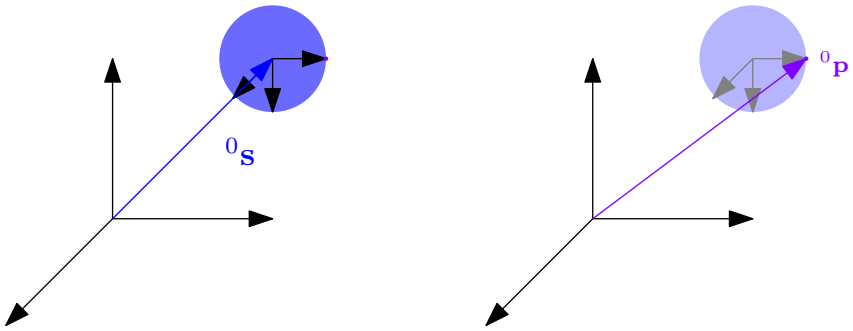
- it can describe the orientation of a rigid body with respect to a reference frame
- it can describe the change of coordinates between two different frames one rotated respect to the other
- it can describe the rotation operator on a vector.

Example

Let's consider a sphere of radius 1 as a rigid body, we have a reference frame RF_S for the sphere fixed in the center, clearly the top pole in that frame is in position ${}^S\mathbf{p} = \begin{pmatrix}0 & 1 & 0\end{pmatrix}^T$.



We have a base frame RF_0 with the canonical base $\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z$, the sphere is placed in position $\begin{pmatrix}3 & 3 & 0\end{pmatrix}^T$, we denote ${}^0\mathbf{s}$ the position of the frame RF_S respect to RF_0. The sphere, is also rotated by 90 degrees clock wise around the z axis.



to find the position of the north pole $\mathbf{p}$ respect to RF_0, we have to apply the rotation matrix $R_z(-\pi/2)$ to ${}^S\mathbf{p}$ and consider the translation.

$$ {}^0\mathbf{p} = R_z(-\pi/2)\,{}^S\mathbf{p} + {}^0\mathbf{s} = \tag{3.18} $$

$$ \begin{pmatrix}0 & 1 & 0\\ -1 & 0 & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}0\\1\\0\end{pmatrix} + \begin{pmatrix}3\\3\\0\end{pmatrix} = \begin{pmatrix}4\\3\\0\end{pmatrix}. \tag{3.19} $$

Let's consider 4 reference frames RF_0, RF_1, RF_2, RF_3, with different orientations, we have 3 matrix

$$ {}^0R_1, {}^1R_2, {}^2R_3 $$

let's consider a position $\mathbf{p}$, in the coordinate system of RF_3, it's denoted ${}^3\mathbf{p}$, to express this position in RF_0 we can consider the matrix

$$ {}^0R_1\,{}^1R_2\,{}^2R_3 = {}^0R_3 $$

and we have

$$ {}^0\mathbf{p} = {}^0R_3\,{}^3\mathbf{p} $$

by getting this matrix products, we perform in total 63 products an 42 summations, we may calculate ${}^0\mathbf{p}$ by deriving ${}^1\mathbf{p}$ and ${}^2\mathbf{p}$

$$ {}^2\mathbf{p} = {}^2R_3\,{}^3\mathbf{p} \tag{3.20} $$

$$ {}^1\mathbf{p} = {}^1R_2\,{}^2\mathbf{p} \tag{3.21} $$

$$ {}^0\mathbf{p} = {}^0R_1\,{}^1\mathbf{p} \tag{3.22} $$

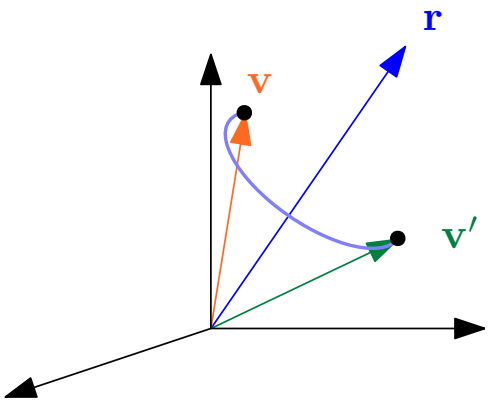
in this case, we perform in total 27 products an 18 summations, this is computationally better, in case we have to change the coordinates of multiple vector, is convenient to calculate the matrix ${}^0R_3$.

3.2 Generalizing Rotation

We defined the basic rotation along the three axis by an angle θ

$$R_x(\theta), R_y(\theta), R_z(\theta) \tag{3.23}$$

we want to generalize this concept by considering a rotation of θ radiants along an arbitrary direction $\mathbf{r}$ (with $\|\mathbf{r}\| = 1$).



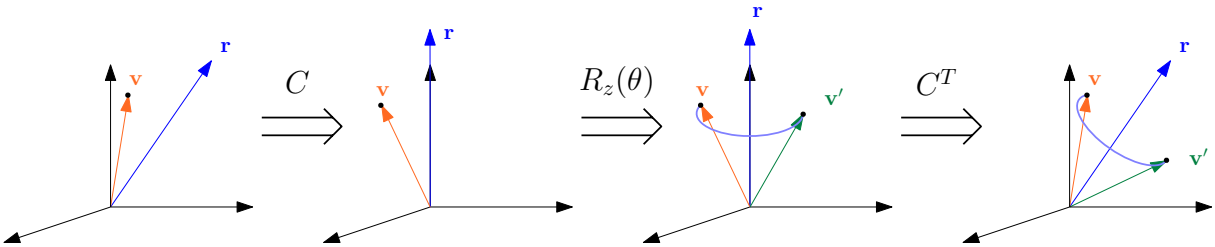
Given $\mathbf{r}, \theta$, we want to find the rotation matrix $R(\theta, \mathbf{r})$ that describes the rotation of a vector of θ radiants along r, or the change of coordinates from a reference frame that is rotated in that orientation respect to the base frame.

$$\mathbf{v}' = R(\theta, \mathbf{r})\mathbf{v} \tag{3.24}$$

3.2.1 Direct Problem

We have to calculate $R(\theta, \mathbf{r}) \in SO(3)$ given $\theta, \mathbf{r}$. We can apply the following process:

1. We consider a transformation denoted C that change the coordinate of our system, by making to set $\mathbf{r}$ aligned with the axis z.
2. We consider the rotation along the z axis by θ radiants.
3. We consider the inverse transformation of C (since $C \in SO(3)$, $C^{-1} = C^T$).



In fact, we are looking for a rotation matrix C such that

$$R(\theta, \mathbf{r}) = CR_z(\theta)C^T \tag{3.25}$$

decomposing $R(\theta, \mathbf{r})$ in a sequence of three rotations. It's relevant to know that the third columns of C is the vector $\mathbf{r}$

$$C = \begin{pmatrix} \mathbf{n} & \mathbf{s} & \mathbf{r} \end{pmatrix} \tag{3.26}$$

Since $C \in SO(3)$, the vectors $\mathbf{n}, \mathbf{s}, \mathbf{r}$ are orthogonal, and it holds that

$$\mathbf{n} \times \mathbf{s} = \mathbf{r}. \tag{3.27}$$

Now we show how to get to C, by the fact that the column vectors are orthogonal, the inner product C^TC is the identity matrix

$$\begin{pmatrix} \mathbf{n}^T \\ \mathbf{s}^T \\ \mathbf{r}^T \end{pmatrix} \begin{pmatrix} \mathbf{n} & \mathbf{s} & \mathbf{r} \end{pmatrix} = I \tag{3.28}$$

the **outer product** of two vectors $\mathbf{v}, \mathbf{u} \in \mathbb{R}^n$ is defined as follows

$$\mathbf{v}\mathbf{u}^T \in Mat(n \times n) \tag{3.29}$$

let $\mathbf{e}_i$ to be a vector of the canonical basis, the outer product

$$\mathbf{e}_i\mathbf{e}_j^T \tag{3.30}$$

is the $n \times n$ matrix with all entries equal to zeros, except for the i, j entry, that is one, for example:

$$\mathbf{e}_1\mathbf{e}_3^T = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{3.31}$$

We define the **dyadic expansion** of an $n \times n$ matrix A as the sum of n^2 matrices, in term of the entrance of that matrix

$$A = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}\mathbf{e}_i\mathbf{e}_j^T \tag{3.32}$$

such that a_{ij} is the (i, j) entry of A. Note how

$$\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}\mathbf{e}_i\mathbf{e}_j^T = IAI^T \tag{3.33}$$

the product of three matrices B, A, B^T can be expressed in dyadic expansion as

$$BAB^T = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}\mathbf{b}_i\mathbf{b}_j^T \tag{3.34}$$

Since $C \in SO(3)$, $CC^T = I$, with dyadic expansion:

$$CC^T = I = \begin{pmatrix} \mathbf{n}^T \\ \mathbf{s}^T \\ \mathbf{r}^T \end{pmatrix} \begin{pmatrix} \mathbf{n} & \mathbf{s} & \mathbf{r} \end{pmatrix} = \begin{pmatrix} \mathbf{n}^T \\ \mathbf{s}^T \\ \mathbf{r}^T \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{n} & \mathbf{s} & \mathbf{r} \end{pmatrix} = \mathbf{n}\mathbf{n}^T + \mathbf{s}\mathbf{s}^T + \mathbf{r}\mathbf{r}^T \tag{3.35}$$

the outer product of different vectors cancels out since are orthogonal from each other. We have that

$$\mathbf{n}\mathbf{n}^T + \mathbf{s}\mathbf{s}^T + \mathbf{r}\mathbf{r}^T = I. \tag{3.36}$$

Definition 3 *A matrix $S \in Mat(n \times n)$ is **skew symmetric** if*

$$S^T = -S \tag{3.37}$$

The diagonal of a skew symmetric matrix have null entries. There are some notable properties:

- Any square matrix A can be decomposed in a sum of two matrices, one symmetric, and one skew symmetric:

  $$A = \frac{A + A^T}{2} + \frac{A - A^T}{2} \tag{3.38}$$

  where $\frac{A+A^T}{2}$ is symmetric and $\frac{A-A^T}{2}$ is skew symmetric.

- In quadratic forms, only the symmetric part of a matrix influences the function:

  $$\mathbf{x}^TA\mathbf{x} = \frac{1}{2}(\mathbf{x}^TA\mathbf{x} + (\mathbf{x}^TA\mathbf{x})^T) = \frac{1}{2}(\mathbf{x}^TA\mathbf{x} + \mathbf{x}^TA^T\mathbf{x}) = \mathbf{x}^T\frac{A + A^T}{2}\mathbf{x} \tag{3.39}$$

  so only the symmetric part matters

  $$\mathbf{x}^TA\mathbf{x} = \mathbf{x}^T\frac{A + A^T}{2}\mathbf{x} \tag{3.40}$$

  if follows that if S is skew symmetric

  $$\mathbf{x}^TS\mathbf{x} = \mathbf{0}. \tag{3.41}$$

The canonical form of a skew symmetric matrix is the following

$$S = \begin{pmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{pmatrix} \tag{3.42}$$

so to describe a matrix S is sufficient a single vector

$$\mathbf{v} = \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} \tag{3.43}$$

in this way, given an arbitrary vector $\mathbf{v} \in \mathbb{R}^n$, we can construct the associated skew symmetric matrix $S(\mathbf{v}) \in Mat(n \times n)$.

Proposition 2 *Given two vectors* $\mathbf{v}, \mathbf{u}$*, the dot product* $\mathbf{v} \times \mathbf{u}$ *is equal to the product from the skew symmetric associated matrix* $S(\mathbf{v})$ *and* $\mathbf{u}$*:*

$$\mathbf{v} \times \mathbf{u} = S(\mathbf{v})\mathbf{u} \tag{3.44}$$

Since $\mathbf{v} \times u = -\mathbf{u} \times \mathbf{v}$ and $-\mathbf{u} \times \mathbf{v} = -S(\mathbf{u})\mathbf{v}$ it follows that

$$S(\mathbf{v})\mathbf{u} = S^T(\mathbf{u})\mathbf{v}. \tag{3.45}$$

Let's go back to the original problem, given that $R(\theta, \mathbf{r}) = CR_z(\theta)C^T$, we want to derive C. Expanding the form we can see how

$$\begin{pmatrix} & & \\ \mathbf{n} & \mathbf{s} & \mathbf{r} \\ & & \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} & \mathbf{n}^T & \\ & \mathbf{s}^T & \\ & \mathbf{r}^T & \end{pmatrix} = \tag{3.46}$$

$$\mathbf{r}\mathbf{r}^T + (\mathbf{n}\mathbf{n}^T + \mathbf{s}\mathbf{s}^T)\cos\theta + (\mathbf{s}\mathbf{n}^T - \mathbf{n}\mathbf{s}^T)\sin\theta \tag{3.47}$$

Since $CC^T = \mathbf{r}\mathbf{r}^T + \mathbf{n}\mathbf{n}^T + \mathbf{s}\mathbf{s}^T = I$, it implies that $\mathbf{n}\mathbf{n}^T + \mathbf{s}\mathbf{s}^T = I - \mathbf{r}\mathbf{r}^T$.

$$\mathbf{r}\mathbf{r}^T + (\mathbf{n}\mathbf{n}^T + \mathbf{s}\mathbf{s}^T)\cos\theta + (\mathbf{s}\mathbf{n}^T - \mathbf{n}\mathbf{s}^T)\sin\theta = \tag{3.48}$$

$$\mathbf{r}\mathbf{r}^T + (I - \mathbf{r}\mathbf{r}^T)\cos\theta + (\mathbf{s}\mathbf{n}^T - \mathbf{n}\mathbf{s}^T)\sin\theta \tag{3.49}$$

It's important to notice that $\mathbf{r}$ is orthogonal to $\mathbf{s}$ and $\mathbf{n}$, and all three vectors are unit 1, it has to be that

$$\mathbf{n} \times \mathbf{s} = \mathbf{r} \tag{3.50}$$

so

$$\mathbf{n} \times \mathbf{s} = \begin{pmatrix} n_y s_z - s_y n_z \\ n_z s_x - s_z n_x \\ n_x s_y - s_x n_y \end{pmatrix} = \begin{pmatrix} r_x \\ r_y \\ r_z \end{pmatrix} = \mathbf{r} \tag{3.51}$$

we can see how

$$\mathbf{s}\mathbf{n}^T - \mathbf{n}\mathbf{s}^T = \begin{pmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{pmatrix} = S(\mathbf{r}) \tag{3.52}$$

So $\mathbf{s}\mathbf{n}^T - \mathbf{n}\mathbf{s}^T$ is equal to the skew symmetric matrix associated to $\mathbf{r}$. Given these propositions, the following theorem holds.

Theorem 1 *Given a vector* $\mathbf{r} \in \mathbb{R}^3$ *and an angle* θ*, the rotation matrix that describes a rotation of* θ *radiants along the vector* $\mathbf{r}$ *is*

$$R(\theta, \mathbf{r}) = \mathbf{r}\mathbf{r}^T + (I - \mathbf{r}\mathbf{r}^T)\cos\theta + S(\mathbf{r})\sin\theta. \tag{3.53}$$

By developing all the computation we can expand the form of that matrix:

$$R(\theta,\mathbf{r}) = \begin{pmatrix} r_x^2(1-\cos\theta)+\cos\theta & r_x r_y(1-\cos\theta)-r_z\sin\theta & r_x r_z(1-\cos\theta)+r_y\sin\theta \\ r_x r_y(1-\cos\theta)+r_z\sin\theta & r_y^2(1-\cos\theta)+\cos\theta & r_y r_z(1-\cos\theta)-r_x\sin\theta \\ r_x r_z(1-\cos\theta)-r_y\sin\theta & r_y r_z(1-\cos\theta)+r_x\sin\theta & r_z^2(1-\cos\theta)+\cos\theta \end{pmatrix} \tag{3.54}$$

The trace of that matrix is $1+2\cos\theta$:

$$Trace(R(\theta,\mathbf{r})) = 1+2\cos\theta \tag{3.55}$$

and we have that

$$R(\theta,\mathbf{r}) = R(-\theta,-\mathbf{r}) = R^T(-\theta,\mathbf{r}) \tag{3.56}$$

There are some properties of $R(\theta,\mathbf{r})$:

- Since is the rotation along $\mathbf{r}$, is invariant to $\mathbf{r}$

$$R(\theta,\mathbf{r})\mathbf{r} = \mathbf{r} \tag{3.57}$$

- the map $(\theta,\mathbf{r}) \to R(\theta,\mathbf{r})$ is not injective since $R(\theta,\mathbf{r}) = R(-\theta,-\mathbf{r})$
- if $\lambda_1,\lambda_2,\lambda_3$ are the eigenvalues of $R(\theta,\mathbf{r})$, the determinant is

$$\det R = \lambda_1\lambda_2\lambda_3 = 1 \tag{3.58}$$

the trace is

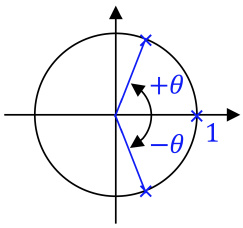
$$\lambda_1+\lambda_2+\lambda_3 = 1+2\cos\theta \tag{3.59}$$

one of the eigenvalues of R is 1. Since $\lambda_2+\lambda_3 = 2\cos\theta$ to find the other eigenvalues we can set the quadratic equation

$$\lambda^2 + 2\cos\theta\lambda + 1 = 0 \tag{3.60}$$

the two complex roots are

$$e^{\pm i\theta} \tag{3.61}$$



Since R is orthonormal, all eigenvalues have unitary norm.

3.2.2 Inverse Problem

Given a rotation matrix $R = (R_{ij})$, we want to find the unit vector $\mathbf{r}$ and the angle θ such that

$$R(\theta,\mathbf{r}) = (R_{ij}) \tag{3.62}$$

Since the trace is $R_{11}+R_{22}+R_{33} = 1+2\cos\theta$, a possible solution for θ is

$$\theta = \arccos\frac{R_{11}+R_{22}+R_{33}-1}{2} \tag{3.63}$$

this is not a good solution since arccos $\in [0,\pi]$, and the function, when implemented, lacks of precision for θ near to 0. We can consider from the data R that

$$R - R^T = \begin{pmatrix} 0 & R_{12}-R_{21} & R_{13}-R_{31} \\ R_{21}-R_{12} & 0 & R_{23}-R_{32} \\ R_{31}-R_{13} & R_{32}-R_{23} & 0 \end{pmatrix} = 2\sin\theta \begin{pmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{pmatrix} \tag{3.64}$$

and since $\|\mathbf{r}\|$ we derive

$$\sin\theta = \pm\frac{1}{2}\sqrt{(R_{12}-R_{21})^2+(R_{13}-R_{31})^2+(R_{23}-R_{32})^2} \tag{3.65}$$

so we can use the atan2 function

$$\text{atan2}(\pm\frac{1}{2}\sqrt{(R_{12}-R_{21})^2+(R_{13}-R_{31})^2+(R_{23}-R_{32})^2}, R_{11}+R_{22}+R_{33}-1) \tag{3.66}$$

if $\sin\theta \neq 0$, $\mathbf{r}$ is

$$\mathbf{r} = \frac{1}{2\sin\theta}\begin{pmatrix} R_{32}-R_{23} \\ R_{13}-R_{31} \\ R_{21}-R_{12} \end{pmatrix} \tag{3.67}$$

if $\sin\theta = 0$, we have two cases

- if $\theta = 0$, there is no solution for $\mathbf{r}$
- if $\theta = \pm\pi$, we have $\sin\theta = 0,\ \cos\theta = -1$ and we solve

$$R = 2\mathbf{r}\mathbf{r}^T - I \tag{3.68}$$

for $\mathbf{r}$.

3.2.3 Quaternion Representation

We can use quaternions to represent rotations, we will not enter in the algebraic details of the quaternions field, in this context, is sufficient to know that a quaternion is an element that can be described with a scalar and a vector in $\mathbb{R}^3$

$$q = (\eta, \boldsymbol{\epsilon}) \tag{3.69}$$
$$\eta \in \mathbb{R} \tag{3.70}$$
$$\boldsymbol{\epsilon} \in \mathbb{R}^3 \tag{3.71}$$

We can represent a rotation along $\mathbf{r}$ of θ radiants by considering the quaternion

$$q = (\cos\frac{\theta}{2}, \sin\frac{\theta}{2}\mathbf{r}) \tag{3.72}$$

notice how $(\theta, \mathbf{r})$ and $(-\theta, -\mathbf{r})$ are associated to the same quaternion. Given q, the rotation matrix is

$$R(q) = R((\eta,\boldsymbol{\epsilon})) = \begin{pmatrix} 2(\eta^2+\epsilon_x^2)-1 & 2(\epsilon_x\epsilon_y-\eta\epsilon_z) & 2(\epsilon_x\epsilon_z+\eta\epsilon_y) \\ 2(\epsilon_x\epsilon_y+\eta\epsilon_z) & 2(\eta^2+\epsilon_y^2)-1 & 2(\epsilon_y\epsilon_z-\eta\epsilon_x) \\ 2(\epsilon_x\epsilon_z-\eta\epsilon_y) & 2(\epsilon_y\epsilon_z+\eta\epsilon_x) & 2(\eta^2+\epsilon_z^2)-1 \end{pmatrix} \tag{3.73}$$

the inverse rotation of $q = (\eta, \boldsymbol{\epsilon})$ is $(\eta, -\boldsymbol{\epsilon})$, and the null rotation is $(1, \mathbf{0})$. There is a binary operation on quaternions $*$ such that

$$q_1 * q_2 = (\eta_1, \boldsymbol{\epsilon}_1) * (\eta_2, \boldsymbol{\epsilon}_2) = (\eta_1\eta_2 - \boldsymbol{\epsilon}_1^T\boldsymbol{\epsilon}_2, \eta_1\boldsymbol{\epsilon}_2 + \eta_2\boldsymbol{\epsilon}_1 + \boldsymbol{\epsilon}_1 \times \boldsymbol{\epsilon}_2) \tag{3.74}$$

We also want to find the quaternion q associated to a given rotation matrix $R = (R_{ij})$. Summing the elements on the diagonal of the matrix shown in (3.73) we get

$$R_{11} + R_{22} + R_{33} = 4\eta^2 - 1 \tag{3.75}$$

the positive roots of this quadratic equation is the value of η

$$\eta = \frac{1}{2}\sqrt{R_{11}+R_{22}+R_{33}+1} \geq 0 \tag{3.76}$$

without entering in the details, from the equation (3.73) we can derive the values for $\boldsymbol{\epsilon}$ as follows:

$$\epsilon_x = \frac{1}{2}\text{sign}(R_{32}-R_{23})\sqrt{R_{11}-R_{22}-R_{33}+1} \tag{3.77}$$
$$\epsilon_y = \frac{1}{2}\text{sign}(R_{13}-R_{32})\sqrt{R_{22}-R_{11}-R_{33}+1} \tag{3.78}$$
$$\epsilon_z = \frac{1}{2}\text{sign}(R_{21}-R_{12})\sqrt{R_{33}-R_{11}-R_{22}+1} \tag{3.79}$$

3.3 Minimal Representations of Orientation

We discussed many representation of rotation, quaternions, rotation axis and angle and rotation matrices, we want to describe an orientation with the minimum number of variables, indeed, the group $SO(3)$ is a 3-manifold, to describe an element in $SO(3)$ we need just three independent variables. We define three rotation along the base axis with three angles $\alpha_1, \alpha_2, \alpha_3$.

There are two ways to use 3 angles to describe a rotation, one along moving axis, and one along fixed axis.

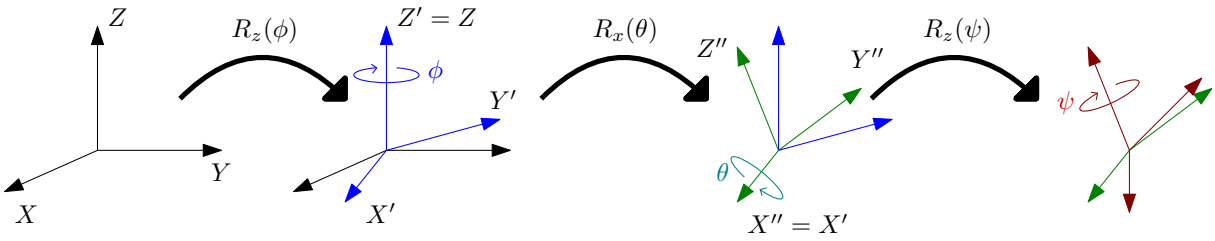
3.3.1 Euler Angles

We describe a rotation as a triple of angles $(\alpha_1, \alpha_2, \alpha_3)$, each angle is associated to an axis X, Y or Z. The angle α_1 describe a rotation of the frame along his associated axis, this will provide a new set of axis X', Y', Z' with a different orientation respect to the initial one. The angle α_2 describe a rotation of the frame along his associated axis in the new frame X', Y', Z', this will provide a new frame X'', Y'', Z'', and α_3 describe a rotation of the frame along his associated axis in this last frame.

There are 12 possible sequence of axis for a rotation, for example

- $XY'X''$ describe a rotation along X, then Y', and then X''. We use the quotation mark $'$ do emphasizes the fact that the axis is part of a frame that has been rotated.
- $ZX'Z''$ describe a rotation along Z, then X', and then Z''.

We exclude contiguous repetitions of axes, like $XX'Z''$ or $YZ'Z''$. Let's consider an example, we have the three euler angles (ϕ, θ, ψ) associated to the axis $ZX'Z''$.



The rotation matrix that describes this rotation is

$$R_z(\phi)R_x(\theta)R_z(\psi) \tag{3.80}$$

Definition 4 *Given three euler angles $\alpha_1, \alpha_2, \alpha_3$ and three axis*

$$(a_1, a_2, a_3) \;:\; a_i \in \{x, y, z\},\; a_1 \neq a_2,\; a_2 \neq a_3$$

the rotation matrix describing the rotation along this moving axes is

$$R_{a_1}(\alpha_1)R_{a_2}(\alpha_2)R_{a_3}(\alpha_3) \tag{3.81}$$

For the example with the axis $ZX'Z''$, the expanded version of the matrix is the following

$$R = \begin{pmatrix} \cos\phi\cos\psi - \sin\phi\cos\theta\sin\psi & -\cos\phi\sin\psi - \sin\phi\cos\theta\cos\psi & \sin\phi\sin\theta \\ \sin\phi\sin\psi + \cos\phi\cos\theta\sin\psi & -\sin\phi\sin\psi + \cos\phi\cos\theta\cos\psi & -\cos\phi\sin\theta \\ \sin\theta\sin\psi & \sin\theta\cos\psi & \cos\theta \end{pmatrix} \tag{3.82}$$

Given a rotation matrix $R = (R_{ij})$, we can derive the values for (ϕ, θ, ψ) by looking at the matrix in equation (3.82). Since

$$R_{13} = \sin\phi\sin\theta \tag{3.83}$$
$$R_{23} = -\cos\phi\sin\theta \tag{3.84}$$

we have that

$$R_{13}^2 + R_{23}^2 = \sin^2\phi \sin^2\theta + (-\cos\phi)^2 \sin^2\theta = \tag{3.85}$$

$$\sin^2\theta(\sin^2\phi + (-\cos\phi)^2) = \sin^2\theta(\sin^2\phi + \cos^2\phi) = \sin^2\theta \tag{3.86}$$

$$\implies \sin\theta = \pm\sqrt{R_{13}^2 + R_{23}^2} \tag{3.87}$$

and

$$R_{33} = \cos\theta \tag{3.88}$$

we can derive θ with the atan2 function:

$$\theta = \text{atan2}\left(\pm\sqrt{R_{13}^2 + R_{23}^2},\ R_{33}\right). \tag{3.89}$$

We have two possible values for θ since we have to consider that $\sin\theta = \pm\sqrt{R_{13}^2 + R_{23}^2}$. We consider the case without singularities, where $\sin\theta \neq 0$, since $R_{31} = \sin\theta\sin\psi$ and $R_{32} = \sin\theta\cos\psi$, we have that

$$\sin\psi = {}^{R_{31}}/_{\sin\theta} \tag{3.90}$$

$$\cos\psi = {}^{R_{32}}/_{\sin\theta} \tag{3.91}$$

$$\implies \psi = \text{atan2}\left(\frac{R_{31}}{\sin\theta}, \frac{R_{32}}{\sin\theta}\right). \tag{3.92}$$

By analogous steps, we can derive ϕ:

$$\phi = \text{atan2}\left(\frac{R_{13}}{\sin\theta}, -\frac{R_{23}}{\sin\theta}\right). \tag{3.93}$$

Since we have two different values for θ, this process provides two different triplets for the angles (ϕ, θ, ψ). We have a singularity when

$$\theta = 0 \vee \theta = \pm\pi \tag{3.94}$$

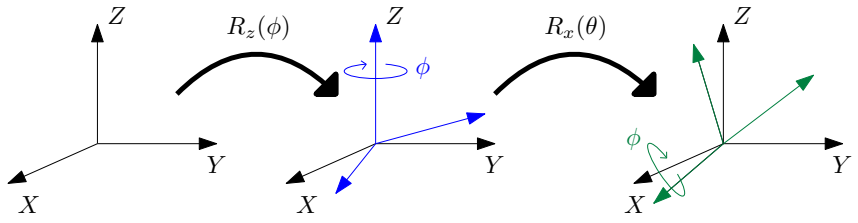
in that case, $\sin\theta = 0$, so we can't derive the values for ϕ, ψ. When $\sin\theta = 0$, the rotation along the second axis X' is irrelevant, we can't derive fixed values for ψ, ϕ, but we can determine only the sum $\phi + \psi$ or the difference $\phi - \psi$. This fact should be intuitive, since we are not rotating along X', we are doing two consecutive rotation along $Z = Z''$.

$$R = R_z(\phi + \psi)$$

This discussion concerns the specific case of rotations along $ZX'Z''$, analogous considerations can be made for all other cases.

3.3.2 Roll-Pitch-Yaw Angles

We describe a rotation as a triple of angles $(\alpha_1, \alpha_2, \alpha_3)$, each angle is associated to an axis X, Y or Z. The frame gets rotated always along fixed axis, so, if we have a rotation along ZXZ with angles (ϕ, θ, ψ), the frame gets rotated by ϕ along Z, then by θ along the original axis X and not the rotated one X'.



In this case

1. We perform a rotation along Z with $R_z(\phi)$
2. We perform a rotation along the original X, that is not coincident with the current axis X', so the rotation matrix is $C_1 R_x(\theta) C_1^T$

3. We perform a rotation along the original Z, that is not coincident with the current axis Z'', so the rotation matrix is $C_2 R_z(\psi) C_2^T$

The final matrix is

$$R_z(\phi) C_1 R_x(\theta) C_1^T C_2 R_z(\psi) C_2^T \tag{3.95}$$

Theorem 2 $R_z(\phi) C_1 R_x(\theta) C_1^T C_2 R_z(\psi) C_2^T = R_z(\psi) R_x(\theta) R_z(\phi)$

Proof: The rotation C_1 align the axis X' of the frame after the first rotation along Z with the original axis X, so

$$C_1 = R_z^T(\phi).$$

Similarly, the transformation C_2 align the axis Z'' after two rotation with the original axis Z, so

$$C_2 = R_z^T(\phi) R_x^T(\theta).$$

We can expand the matrix

$$R_z(\phi) C_1 R_x(\theta) C_1^T C_2 R_z(\psi) C_2^T = \tag{3.96}$$
$$R_z(\phi) R_z^T(\phi) R_x(\theta) R_z(\phi) R_z^T(\phi) R_x^T(\theta) R_z(\psi) R_x(\theta) R_z(\phi) = \tag{3.97}$$
$$R_z(\phi) R_z^T(\phi) R_x(\theta) R_z(\phi) R_z^T(\phi) R_x^T(\theta) R_z(\psi) R_x(\theta) R_z(\phi) = \tag{3.98}$$
$$R_x(\theta) R_x^T(\theta) R_z(\psi) R_x(\theta) R_z(\phi) = \tag{3.99}$$
$$R_z(\psi) R_x(\theta) R_z(\phi) \qquad \blacksquare \tag{3.100}$$

Definition 5 *Given three roll-pitch-yaw angles $\alpha_1, \alpha_2, \alpha_3$ and three axis*

$$(a_1, a_2, a_3) \;:\; a_i \in \{x, y, z\},\; a_1 \neq a_2,\; a_2 \neq a_3$$

the rotation matrix describing the rotation along this fixed axes is

$$R_{a_3}(\alpha_3) R_{a_2}(\alpha_2) R_{a_1}(\alpha_1) \tag{3.101}$$

The roll-pitch-yaw angles rotation is equivalent to an euler angles rotation, we get this equivalence by inverting the order of the sequence of axis:

Euler angles		**Roll-Pitch-Yaw angles**
first rotation : by α_1 radiants along a_1		first rotation : by α_3 radiants along a_3
second rotation : by α_2 radiants along a_2	$\equiv$	second rotation : by α_2 radiants along a_2
third rotation : by α_3 radiants along a_3		third rotation : by α_1 radiants along a_1

Let's consider now a rotation along XYZ, by the angles (ψ, θ, ϕ). The rotation matrix that describes this rotation is

$$R = R_z(\phi) R_y(\theta) R_x(\psi) \tag{3.102}$$

the expanded form is

$$\begin{pmatrix} \cos\phi\cos\theta & \cos\phi\sin\psi - \sin\phi\cos\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \sin\phi\cos\theta & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\sin\theta\cos\phi - \cos\phi\sin\psi \\ -\sin\theta & \cos\theta\sin\psi & \cos\theta\cos\psi \end{pmatrix} \tag{3.103}$$

The inverse problem is to find (ψ, θ, ϕ) given $R = (R_{ij})$, we can determine the values of the angles by considering the matrix in equation (3.103). Since

$$R_{32}^2 + R_{33}^2 = \cos^2\theta \tag{3.104}$$
$$R_{31} = -\sin\theta \tag{3.105}$$

we get the two possible values for θ:

$$\theta = \text{atan2}\left(-R_{31},\; \pm\sqrt{R_{32}^2 + R_{33}^2}\right) \tag{3.106}$$

if $\cos\theta \neq 0$, we have

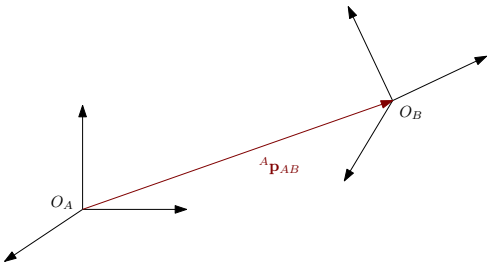
$$\psi = \text{atan2}\left(\frac{R_{32}}{\cos\theta},\; \frac{R_{33}}{\cos\theta}\right) \tag{3.107}$$
$$\phi = \text{atan2}\left(\frac{R_{21}}{\cos\theta},\; \frac{R_{11}}{\cos\theta}\right) \tag{3.108}$$

We have singularities when $\theta = \pm\frac{\pi}{2}$, in these cases, we can determine only the sum $\phi + \psi$ or the difference $\phi - \psi$. This discussion concerns the specific case of rotations along XYZ, analogous considerations can be made for all other cases.

3.4 Homogeneous Transformations

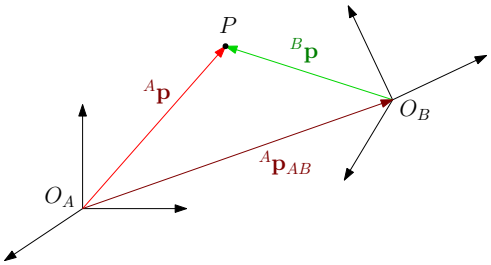
For now, we discussed about frames with different orientation, but fixed in the same spot. Now we consider frames with different orientation and position. Let's consider two frames, O_A and O_B, and let's say that

- ${}^{A}R_B$ is the rotation matrix that act as a change of coordinates from O_A to O_B
- ${}^{A}\mathbf{p}_{AB}$ is the vector that starts from A and goes to B, it describes the position of the frame O_B respect to O_A.



Let's consider a point P, let ${}^{B}\mathbf{p}$ the vector that describes the position P in the frame O_B. To determine the vector ${}^{A}\mathbf{p}$ that describes the position P in the frame O_A, we have to apply the translation and the rotation as follows:

$$ {}^{A}\mathbf{p} = {}^{A}\mathbf{p}_{AB} + {}^{A}R_B\,{}^{B}\mathbf{p} \tag{3.109} $$



Instead of considering a sum and a multiplication by a matrix, we can consider the **affine relationship** as a linear relationship by considering a 4×4 homogeneous matrix and transforming ${}^{A}\mathbf{p}$ and ${}^{B}\mathbf{p}$ in the homogeneous coordinates as follows:

$$ {}^{A}\mathbf{p}_{hom} = \begin{pmatrix} {}^{A}\mathbf{p} \\ 1 \end{pmatrix} \in \mathbb{R}^4 \tag{3.110} $$

$$ {}^{B}\mathbf{p}_{hom} = \begin{pmatrix} {}^{B}\mathbf{p} \\ 1 \end{pmatrix} \in \mathbb{R}^4 \tag{3.111} $$

$$ {}^{A}\mathbf{p}_{hom} = \begin{pmatrix} & {}^{A}R_B & & {}^{A}\mathbf{p}_{AB} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} {}^{B}\mathbf{p} \\ 1 \end{pmatrix} = {}^{A}\mathbf{p}_{AB} + {}^{A}R_B\,{}^{B}\mathbf{p} \tag{3.112} $$

we denote this **homogeneous transformation** ${}^{A}T_B$

$$ {}^{A}\mathbf{p}_{hom} = {}^{A}T_B\,{}^{B}\mathbf{p}_{hom} \tag{3.113} $$

- A homogeneous transformation T describes the relation between two reference frames relative to the pose.
- Transforms the representation of a position vector (applied vector starting from the origin of the frame) from one frame to another frame.
- It is a roto-translation operator on vectors in the three dimensional space.
- It is invertible $({}^{A}T_B)^{-1} = {}^{B}T_A$
- and can be composed

$$ {}^{A}T_B\,{}^{B}T_C = {}^{A}T_C. \tag{3.114} $$

If

$$ {}^{A}T_{B} = \begin{pmatrix} {}^{A}R_{B} & {}^{A}\mathbf{p}_{AB} \\ \mathbf{0}^{T} & 1 \end{pmatrix} $$

the inverse

$$ {}^{B}T_{A} = \begin{pmatrix} {}^{B}R_{A} & {}^{B}\mathbf{p}_{BA} \\ \mathbf{0}^{T} & 1 \end{pmatrix} $$

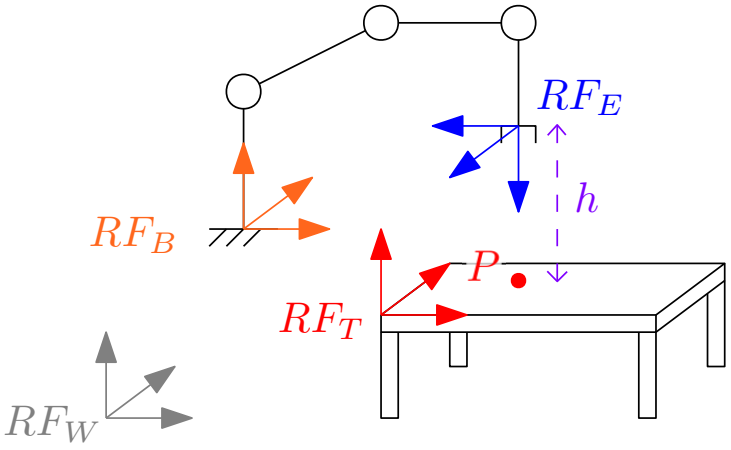
can be written in term of the original vector/matrices as follows

$$ {}^{B}T_{A} = ({}^{A}T_{B})^{-1} = \begin{pmatrix} {}^{A}R_{B}^{T} & -{}^{A}R_{B}^{T}{}^{A}\mathbf{p}_{AB} \\ \mathbf{0}^{T} & 1 \end{pmatrix} \tag{3.115} $$

3.4.1 Example of a Robotic Task

Let's consider the following scenario

- There is a world reference frame RF_W
- There is a robotic arm with his reference frame RF_B
- There is a reference frame RF_E attached to the end effector of the robot
- There is a table with a reference frame RF_T and an object placed on the table, in position P.



The end effector is placed in a way such that, is pointing downward to the table, so the axis z of RF_T is in the same direction of the axis z of RF_E, but with opposite orientation

$$ {}^{E}R_{T} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} = R_{x}(\pi) \tag{3.116} $$

Since the point P lies on the table, the z value for the vector that describes P in RF_T will be null

$$ {}^{T}\mathbf{p} = \begin{pmatrix} p_{x} \\ p_{y} \\ 0 \end{pmatrix} \tag{3.117} $$

Let's say that the end effector is pointing P, in that case the vector that describes P in RF_E will be

$$ {}^{E}\mathbf{p} = \begin{pmatrix} 0 \\ 0 \\ h \end{pmatrix} \tag{3.118} $$

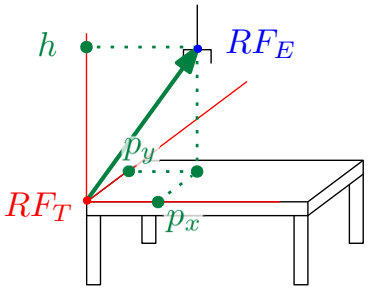
where h is the distance between P and RF_E. To determine the position of the end effector respect to the table, we have to consider

$$ {}^{T}R_{E} = {}^{E}R_{T}^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}^{T} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} = {}^{E}R_{T} \tag{3.119} $$

we need the vector ${}^T\mathbf{p}_{TE}$ that describes the position of the end effector respect to the frame RF_T. This is given by

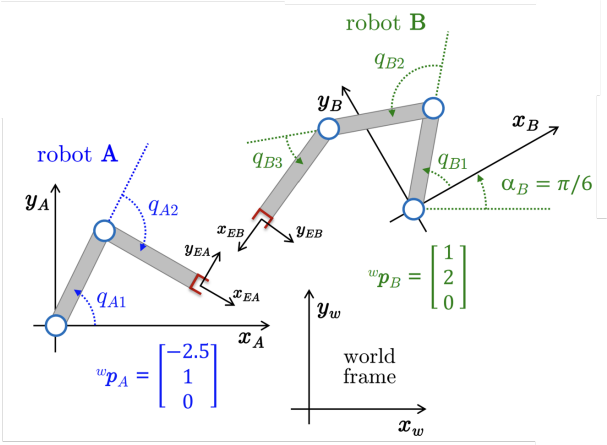
$$ {}^T\mathbf{p}_{TE} = {}^T\mathbf{p} - {}^T R_E \, {}^E\mathbf{p} = \tag{3.120} $$

$$ \begin{pmatrix} p_x \\ p_y \\ 0 \end{pmatrix} - \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ h \end{pmatrix} = \begin{pmatrix} p_x \\ p_y \\ h \end{pmatrix} \tag{3.121} $$



3.4.2 Example of An Exercise

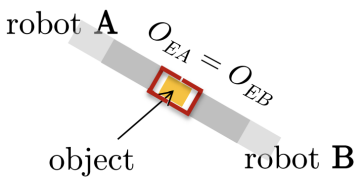
Let's consider the following configuration:



We have two robots, robot A with the base in the frame RF_A, and robot B with the base in RF_B. RF_B is rotated along the z axis respect to our base world frame RF_w. We also have the frames of the end effectors RF_{EA} and RF_{EB}. The joints $\mathbf{q}_A$ are fixed

$$ \mathbf{q}_A = \begin{pmatrix} \frac{\pi}{3} \\ -\frac{\pi}{2} \end{pmatrix} \tag{3.122} $$

We want to find the values for $\mathbf{q}_B$ such that the two end effectors are in the same positions and points each other:



In that case, the rotation matrix that describe the orientation of RF_{EA} respects to RF_{EB} is

$$ {}^{EA}R_{EB} = R_z(\pi) = \begin{pmatrix} \cos\pi & -\sin\pi & 0 \\ \sin\pi & \cos\pi & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.123} $$

so

$$ {}^{EA}T_{EB} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.124} $$

Since RF_w and RF_A have the same orientation, the homogeneous transformation ${}^{w}T_A$ is just a translation

$$ {}^{w}T_A = \begin{pmatrix} 1 & 0 & 0 & -2.5 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.125} $$

the frame RF_B is rotated along z by $\frac{\pi}{6}$ radiants respects to the frame RF_w so

$$ {}^{w}R_B = R_z(\pi/6) \tag{3.126} $$

the homogeneous transformation is

$$ {}^{w}T_B = \begin{pmatrix} & R_z(\pi/6) & & 1 \\ & & & 2 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0.866 & -0.5 & 0 & 1 \\ 0.5 & 0.866 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.127} $$

The pose of the end effector RF_{EA} respect his base RF_A is calculated with direct kinematics, and it's in function of $\mathbf{q}_A$

$$ {}^{A}T_{EA} = \begin{pmatrix} 0.866 & 0.5 & 0 & 1.366 \\ -0.5 & 0.866 & 0 & 0.366 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.128} $$

For now, it is not important how ${}^{A}T_{EA}$ is calculated, since we didn't discussed about direct kinematics yet. The pose of the end effector of the robot A respect to our world frame is

$$ {}^{w}T_{EA} = {}^{w}T_A \, {}^{A}T_{EA} \tag{3.129} $$

The position of the end effector of B can be computed by "passing" through the robot A as follows

$$ {}^{w}T_{EB} = {}^{w}T_A \, {}^{A}T_{EA} \, {}^{EA}T_{EB} \tag{3.130} $$

can be also computed by passing through B

$$ {}^{w}T_{EB} = {}^{w}T_B \, {}^{B}T_{EB} \tag{3.131} $$

the transformation ${}^{B}T_{EB}$ is unknown and it is in function of $\mathbf{q}_B$. We have to solve the system of non linear equations

$$ {}^{w}T_A \, {}^{A}T_{EA} \, {}^{EA}T_{EB} = {}^{w}T_B \, {}^{B}T_{EB}(\mathbf{q}_B) \tag{3.132} $$

$$ {}^{B}T_{EB}(\mathbf{q}_B) = ({}^{w}T_B)^{-1} \, {}^{w}T_A \, {}^{A}T_{EA} \, {}^{EA}T_{EB} \tag{3.133} $$

for $\mathbf{q}_B$ with inverse kinematics. We do not discuss the full procedure to find the solutions (there are two).

CHAPTER

4

DIRECT AND INVERSE KINEMATICS

In this chapter we study the geometric and timing aspects of robot motion, without considering the forces that causes that motion (since we are not considering the dynamics of a robotic manipulator, but just the kinematics).

A robot is defined as an open kinematic chain of rigid bodies, interconnected by joints, these joints can be revolute or prismatic. To plan a robotic task, we consider three abstract spaces:

- the actuation space, where we give commands to the actuators to generate torque and forces to make the joints move
- the joint space, where we define the degree of freedom of a robotic system
- the tasks space, such as the cartesian space where the end effector lies.

We don't take in account the actuation space for now, we assume that we can directly control the joint space, the *direct kinematics* is a map from that space and the task space, we usually denote

$$\mathbf{q} = (q_1 \dots, q_n) \tag{4.1}$$

the variables of the joint space, and

$$\mathbf{r} = (r_1 \dots, r_m) \tag{4.2}$$

the variables of the task space. The direct kinematics is a map f such that

$$\mathbf{r} = f_r(\mathbf{q}) \tag{4.3}$$

the r as a subscript denotes that the function f depends on the characterization of the task space, this should be *minimal* and *unambiguous*.

A non minimal characterization of the configuration of a mechanical system is the following: Let's consider a simple pendulum with a stick of length L, we want to describe the position of each point of the stick, we define the vector (p_x, p_y), as shown in figure 4.1.

Since there is a constraint

$$p_x^2 + p_y^2 = L^2 \tag{4.4}$$

only one variable will be sufficient to describe the entire configuration, a minimal representation is to consider the angle θ between the stick and the y axis, as shown in figure 4.2.

We have an ambiguous choice of the representation when the variables of the task space doesn't define one single configuration, let's consider the robotic arm shown in figure 4.3, if the configuration is represented by the angle of the third joint q_3 and the position of that joint

$$\mathbf{r} = (p_x, p_y, q_3)$$

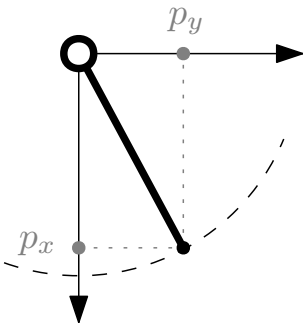


Figure 4.1: Non minimal representation of the configuration

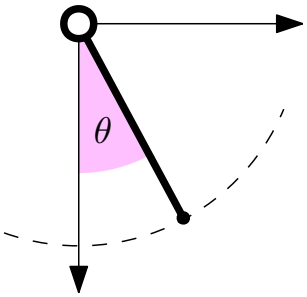


Figure 4.2: Minimal representation of the configuration

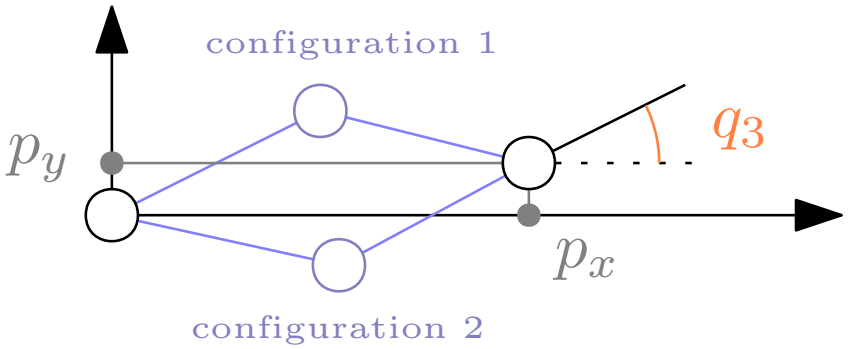


Figure 4.3: ambiguous representation of the configuration

a single value for $\mathbf{r}$ identifies two configurations.

Usually, the number of variables of the configuration space are less or equals than the number of degree of freedom:

$$m \leq n$$

to describe the pose of an object we need a configuration space with $m = 6$ variables (3 for the position and 3 for the orientation).

For simple robotic system, as the one shown in figure 4.4, the direct kinematics can be computed geometrically by inspecting the planar diagram.

With simple consideration we can easily compute the configuration $\mathbf{r} = (p_x, p_y, \phi)$:

$$p_x = l_1 \cos q_1 + l_2 \cos(q_1 + q_2) \tag{4.5}$$
$$p_y = l_1 \sin q_1 + l_2 \sin(q_1 + q_2) \tag{4.6}$$
$$\phi = q_1 + q_2 \tag{4.7}$$

For more general case, we need a systematic method that assigns frames to each joints.

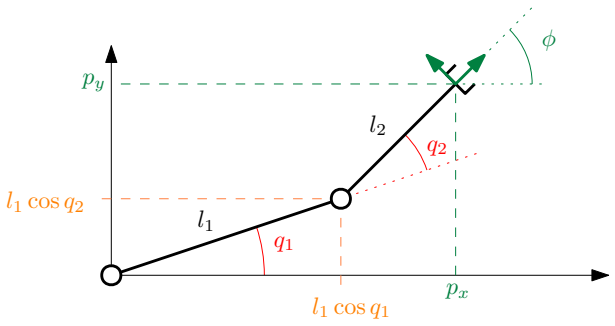


Figure 4.4: Planar manipulator

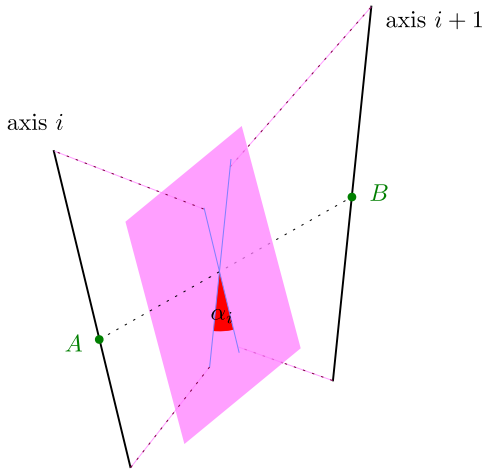
4.1 Denavit-Hartenberg Frames

Definition 6 *Given two axes in $\mathbb{R}^3$, described by two functions*

$$f_1(t) = \mathbf{p}_1 + \mathbf{v}_1 t \tag{4.8}$$
$$f_2(t) = \mathbf{p}_2 + \mathbf{v}_2 t \tag{4.9}$$

*the **common normal** is the segment passing trough f_1 and f_2 that is orthogonal to both axes. Is the shortest segment that connects the two axis.*

Definition 7 *Given two axes f_1, f_2 and the common normal AB, let Π to be the plane orthogonal to AB, and let f_1' and f_2' to be the projection of f_1, f_2 on Π. The angle between f_1' and f_2' is called the **twist angle** between the axes.*

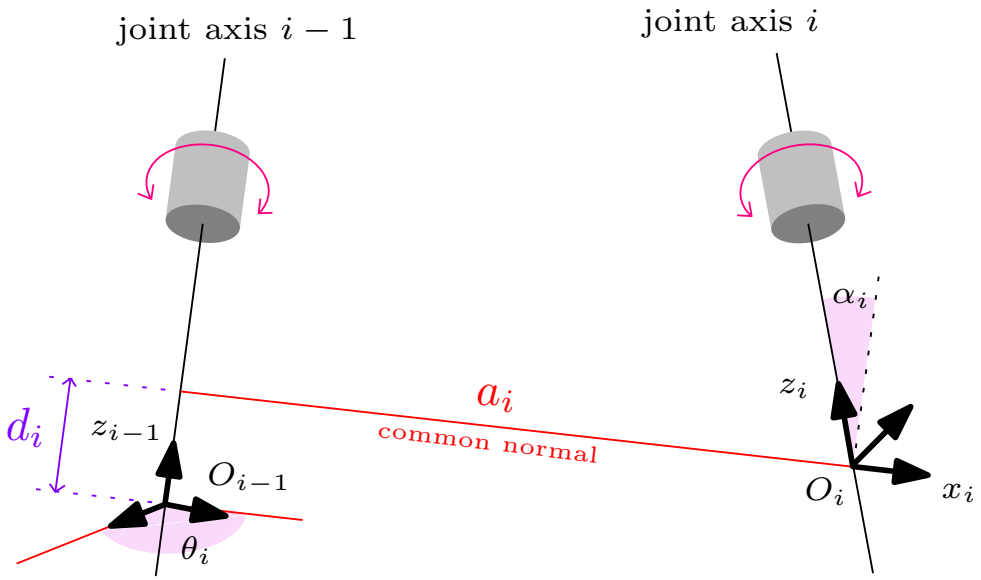


In our representation of a robotics arm, given two consecutive joints i and $i+1$, and given the axis of these joints, we consider the common normal, and denote α_i the twisted angle between the axis, and a_i the *displacement*, the length of the common normal.

Now we have to describe how to assign a frame to each joints of a robotic arm. We recall that

- Given a joint axis i, if the joint is revolut, the rigid body associated can rotate around the axis.
- Given a joint axis i, if the joint is prismatic, the rigid body associated can translate along the axis.

We describe two consecutive joints ($i-1$ and i), we consider the axis of these joints. The frame O_{i-1} of the joint $i-1$ will have the z axes (denoted z_{i-1}) aligned with the joint axis, and we consider the common normal between the two joints axes, denoting with a_i the length.



We denote d_i the distance between O_{i-1} and the point that is the intersection between the axis $i-1$ and the common normal. α_i is the twisted angle between the two axes $i-1$ and i. The x_i axis of the frame O_i is aligned with the common normal.

θ_i is the axis between the common normal a_{i-1} and a_i. If the joint $i-1$ is prismatic, d_i is variable and θ_i is fixed, if it is revolut, d_i is fixed and θ_i is variable. α_i and a_i are fixed and depends on the construction of the robotic arm.

The homogeneous transformation from O_{i-1} to O_i is given by

1. a translation of d_i along z_{i-1}
2. a rotation of θ_i radiants around z_{i-1}. Once these rotation is performed, x_{i-1} and x_i will be parallel.
3. a translation of a_i along x_i
4. a rotation of α_i radiants around x_i.

The first rotation around z_{i-1} is given by

$$\begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.10}$$

the translation along z_{i-1} is given by

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.11}$$

the roto-translation around and along x_i is given by

$$\begin{pmatrix} 1 & 0 & 0 & a_i \\ 0 & \cos\alpha_i & -\sin\alpha_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.12}$$

the final homogeneous transformation to describe the pose of O_i respects to O_{i-1} is

$$^{i-1}A_i(q_i) = \begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & a_i \\ 0 & \cos\alpha_i & -\sin\alpha_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \tag{4.13}$$

$$\begin{pmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.14}$$

if the joint is revolut, $q_i = \theta_i$, if it is prismatic, then $q_i = d_i$. This matrix is called the **DH matrix**. When we consider a full robotic arm, the first frame O_0 is placed arbitrary, with the only condition to align the z_0 axis with the first joint axis. To describe the pose of the last n-th joint frame O_n respect to the first joint frame O_0, we perform a product of DH matrices

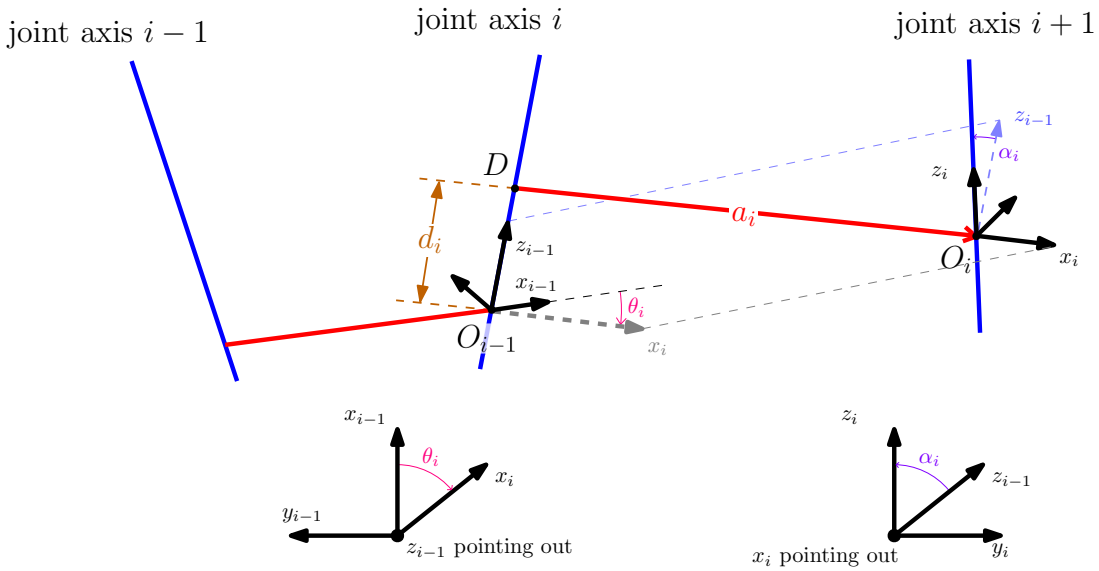
$$^0A_n = {}^0A_1(q_1)\,{}^1A_2(q_2)\dots{}^{n-1}A_n(q_n) \tag{4.15}$$

Usually, if the word frame RF_w is not in the same pose of the first joint frame, we have a constant transformation ${}^wT_0$. The transformation from the n-th joint frame O_n and the end effector frame RF_E is constant (we will discuss about this transformation later). In general, the final direct kinematics transformation is

$$^wT_E(\mathbf{q}) = {}^wT_0\,{}^0A_1(q_1)\,{}^1A_2(q_2)\dots{}^{n-1}A_n(q_n)\,{}^nT_E. \tag{4.16}$$

We recap the parameters of the DH frames:

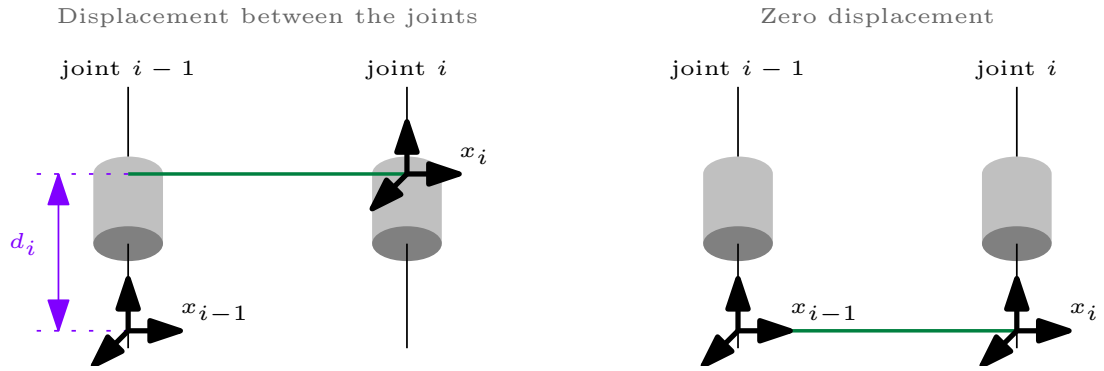
- the unit vector z_i is along the axis of the joint $i+1$
- the unit vector x_i is placed along the common normal to joint i and $i+1$
- a_i is the distance DO_i oriented as x_i, is always constant and is the length of the common normal that intersects the joint axis i and $i+1$. We recall that the point D is the point of the intersection between the common normal and the joint's axis $i-1$.
- d_i is the distance $O_{i-1}D$, oriented as z_{i-1}, this value is variable if the joint i is prismatic
- α_i is the twist angle from z_{i-1} to z_i around x_i, is constant.
- θ_i is the angle from x_{i-1} to x_i around z_{i-1}, this value is variable if the joint i is revolute.



There are some caveats in the assignments of the DH frames:

- the frame 0 origin and x_0 axis are arbitrary, z_0 must be on the first joint axis.
- the origin of frame n should be chosen conveniently (to zero out displacements), x_n must intersect and be chosen orthogonal to z_{n-1}.
- the direction of z_{i-1} is arbitrary, is better to avoid to flip over consecutive z axes.
- if z_{i-1} and z_i are parallel, there are infinitely many common normal. We should try to zero out parameters.

- if z_{i-1} and z_i are coincident, x_i axis can be chosen at will.



4.2 Workspace

The *reachable workspace* of a robot manipulator is the set of all points that can be reached by the origin of the end effector (realized by changing che joint's variables) with at least one orientation. Let $f_p(\mathbf{q})$ to be the positional part of the direct kinematics map, that describe the position of the origin of the end effector, it is a restriction of f_r. The workspace is defined as follows:

$$WS_1 = \{f_p(\mathbf{q}) \ : \ \mathbf{q} \in \mathbb{R}^n \ \wedge \ q_{im} \leq q_i \leq q_{iM}, \ i = 1, 2 \ldots, n\} \tag{4.17}$$

where n is the number of joints and $[q_{im}, q_{iM}]$ is the range of the joint's variable q_i. We can define the joint's space as

$$Q = \{\mathbf{q} \ : \ \mathbf{q} \in \mathbb{R}^n \ \wedge \ q_{im} \leq q_i \leq q_{iM}, \ i = 1, 2 \ldots, n\} \tag{4.18}$$

and rewrite the reachable workspace as follows:

$$WS_1 = \{f_p(\mathbf{q}) \ : \ \mathbf{q} \in Q\} \tag{4.19}$$

Since the joint's can be spherical, revolut and prismatic, the workspace WS_1 can always be defined as the intersection of planar, toroidal, spherical and cylindrical surfaces. In general, Q is a differentiable manifold, and WS_1 is finite, closed and connected, because the function f_p is continuous on Q.

The space WS_1 is called **primary workspace**. We define also the **secondary workspace** WS_2 as the set of positions $\mathbf{p}$ that can be reached by the origin of the end effector in any possible orientation. Clearly,

$$WS_2 \subseteq WS_1$$

if $f_r : Q \to \mathbb{R}^6$ is the direct kinematics map that assigns at each joint's configuration $\mathbf{q}$ a pose $\mathbf{r} \in \mathbb{R}^6$, we can describe f_r as a couple of two functions, one for the position and one for the orientation:

$$f_r(\mathbf{q}) = \begin{pmatrix} f_p(\mathbf{q}) \\ f_\phi(\mathbf{q}) \end{pmatrix} \tag{4.20}$$

The secondary workspace is defined as follows:

$$WS_2 = \left\{ \mathbf{p} \in \mathbb{R}^3 \ : \ \forall \boldsymbol{\phi} \in [0, 2\pi]^3, \ \exists \mathbf{q} \in Q \text{ such that } \begin{pmatrix} f_p(\mathbf{q}) \\ f_\phi(\mathbf{q}) \end{pmatrix} = \begin{pmatrix} \mathbf{p} \\ \boldsymbol{\phi} \end{pmatrix} \right\} \tag{4.21}$$

Usually, side views of the workspace of a robot manipulator are shown in the technical sheets of the robot as shown in figure 4.5.

Let's consider a planar 2R arm, with two links of length l_1, l_2. The workspace is

- if $l_1 \neq l_2$
  - $WS_1 = \{\mathbf{p} \in \mathbb{R}^2 \ : \ |l_1 - l_2| \leq \|\mathbf{p}\| \leq l_1 + l_2\}$
  - $WS_2 = \emptyset$

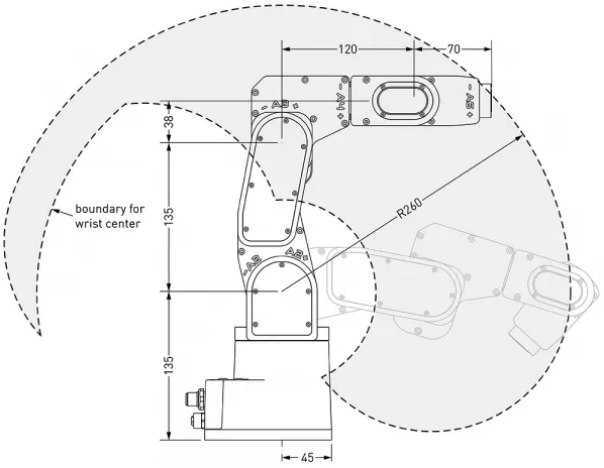


Figure 4.5: side view workspace of a robotic manipulator

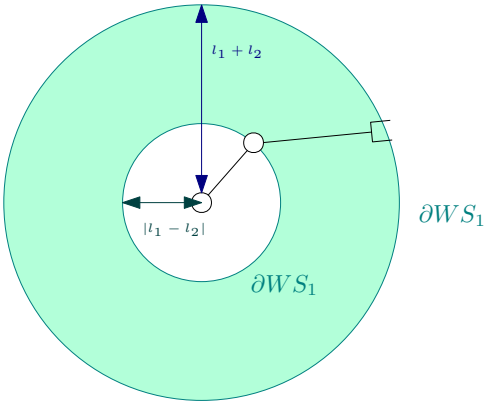


Figure 4.6: workspace of a planar 2R robotic manipulator

- if $l_1 = l_2 = l$
  - $WS_1 = \{\mathbf{p} \in \mathbb{R}^2 \ : \|\mathbf{p}\| \leq 2l\}$
  - $WS_2 = \{\mathbf{0}\}$

The workspace is the difference between a closed and an open ball in $\mathbb{R}^2$, as shown in figure 4.6.

We denote ∂WS_1 the boundaries of WS_1. The end effector can lie in only one orientation in ∂WS_1 and in two possible orientation in $WS_1 \backslash \partial WS_1$.

4.3 The Inverse Kinematics Problem

The inverse kinematics problem is to find the joint's configuration $\mathbf{q}$ that realize a desired task configuration $\mathbf{r}$. The direct kinematics is always unique, given $\mathbf{q} \in Q$, we can always identify one configuration $\mathbf{r}$. This is not true for the inverse kinematics, since, given a desired pose $\mathbf{r}$, they may be zero, infinite, or a finite number of joint's configuration that realize $\mathbf{r}$.

This is in general a non linear problem, due to the trigonometric functions inside the homogeneous transformation. We have to determine if there are solutions (workspace definition), the uniqueness and multiplicity of these solutions, and the methods for finding them.

For example, given an anthropomorphic arm (3 revolut joints, the first orthogonal to the second, the second and the third parallel) we can identify four possible joint's configuration that realize a given end-effector position, as shown in figure 4.7
For the UR10 robotic arm, shown in figure 4.8, there are 8 IK solutions that realize the desired pose:

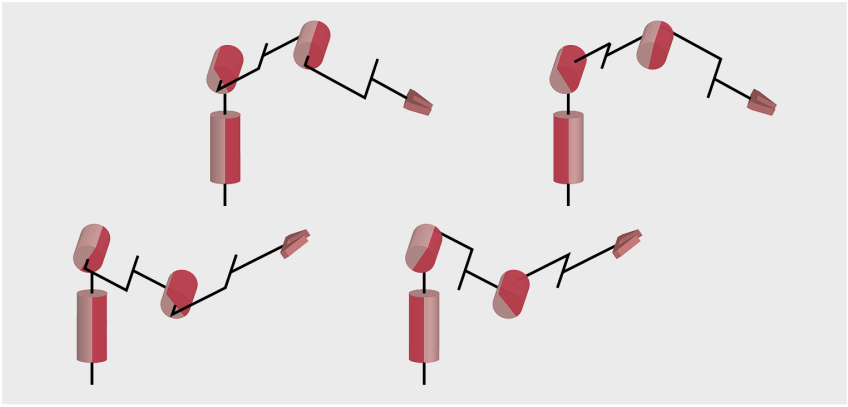


Figure 4.7: IK solutions of an anthropomorphic arm

$$\mathbf{p} = \begin{pmatrix} -0.2373 \\ -0.0832 \\ 1.3224 \end{pmatrix} \text{ meters} \tag{4.22}$$

$$R = \begin{pmatrix} \sqrt{3}/2 & 0.5 & 0 \\ -0.5 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.23}$$

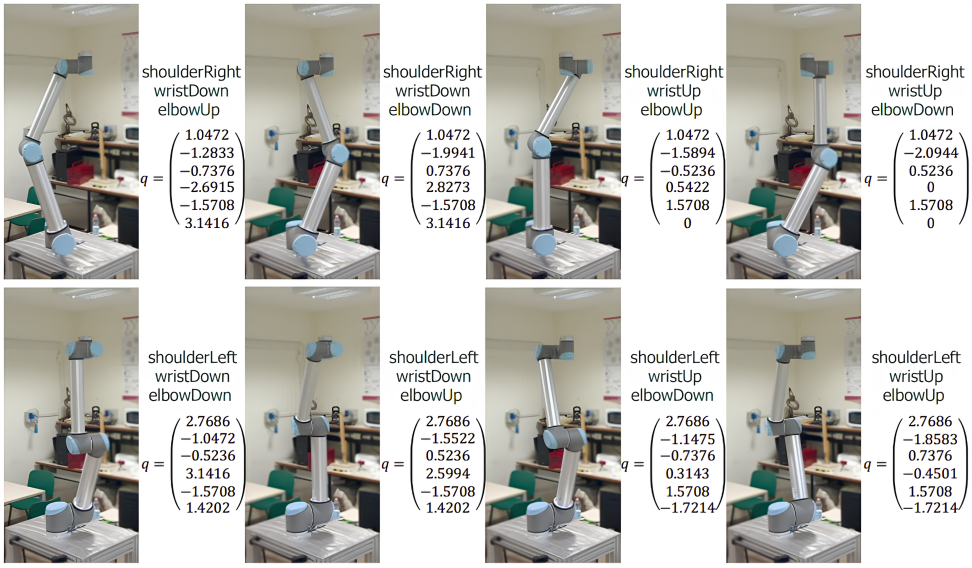


Figure 4.8: IK solutions for the UR10 robotic manipulator

The multiplicity of solutions depends on the kinematic structure of the manipulator, a few examples are the following:

- For a desired position of the end effector of a 2R planar robot ($m = n = 2$)[1] we have
  - 2 regular solutions in $WS_1 \backslash \partial WS_1$
  - 1 solution on ∂WS_1
  - if $l_1 = l_2$ (the link's lengths are equals) there are infinite solutions on the origin $WS_2 = \{\mathbf{0}\}$.
- For a desired position of the end effector of a elbow-type 3R robot ($m = n = 3$) we have 4 regular solutions on WS_1. The singular cases have not been investigated yet.

[1] We denote with n the number of joints (DOF) and with m the number of variables of the configuration space.

- For a spatial 6R robot ($m = n = 6$) we have
  - at most 16 distinct solutions. This upper bound of 16 is attained by a particular instance of an orthogonal robotic arm.
  - the analysis of the solutions is based on algebraic transformations of the robot kinematics, transcendental equations are transformed into a single polynomial equation in one variable, or we seek for a transformed polynomial equation of the least possible degree.

Let's see an example of these algebraic transformations. In some cases, we may have to solve for θ the equation

$$a \sin\theta + b \cos\theta = c \tag{4.24}$$

we define a variable $u = \tan \theta/2$, in this case:

$$\sin\theta = \frac{2u}{1+u^2} \tag{4.25}$$

$$\cos\theta = \frac{1-u^2}{1+u^2} \tag{4.26}$$

and

$$\tan\theta = \tan\frac{2\theta}{2} = \frac{2\tan\theta/2}{1-\tan^2\theta/2} = \frac{2u}{1-u^2} \tag{4.27}$$

We substitute in equation (4.24):

$$a\frac{2u}{1+u^2} + b\frac{1-u^2}{1+u^2} = c \tag{4.28}$$

this is a second degree polynomial equation:

$$a\frac{2u}{1+u^2} + b\frac{1-u^2}{1+u^2} = c \implies \tag{4.29}$$

$$\frac{b-u^2b+2au}{1+u^2} = c \implies \tag{4.30}$$

$$b - u^2b + 2au = c + cu^2 \implies \tag{4.31}$$

$$b - u^2b + 2au - c - cu^2 = 0 \implies \tag{4.32}$$

$$-b + u^2b - 2au + c + cu^2 = 0 \implies \tag{4.33}$$

$$u^2(c+b) - 2au - (b-c) = 0 \implies \tag{4.34}$$

the solutions are

$$u_{1,2} = \frac{a \pm \sqrt{a^2+b^2-c^2}}{b+c} \tag{4.35}$$

if $a^2 + b^2 \geq c^2$ we have

$$u_1 = \tan\frac{\theta_1}{2} \tag{4.36}$$

$$u_2 = \tan\frac{\theta_2}{2} \tag{4.37}$$

and we can solve for $\theta_{1,2}$

$$\theta_{1,2} = 2\arctan(u_{1,2}) \tag{4.38}$$

4.3.1 Workspace of a Planar 3R Arm

Let's consider a 3R planar robotic manipulator ($n = 3$, $m = 2$), we consider the case where the 3 links have the same length $l_1 = l_2 = l_3 = l$. The primary workspace is the closed ball of radius $3l$ centered in the origin

$$WS_1 = \{\mathbf{p} \in \mathbb{R}^2 \ : \ \|\mathbf{p}\| \leq 3l\} \tag{4.39}$$

The end effector could lie in any possible orientation in the closed ball of radius l:

$$WS_2 = \{\mathbf{p} \in \mathbb{R}^2 \ : \ \|\mathbf{p}\| \leq l\} \tag{4.40}$$

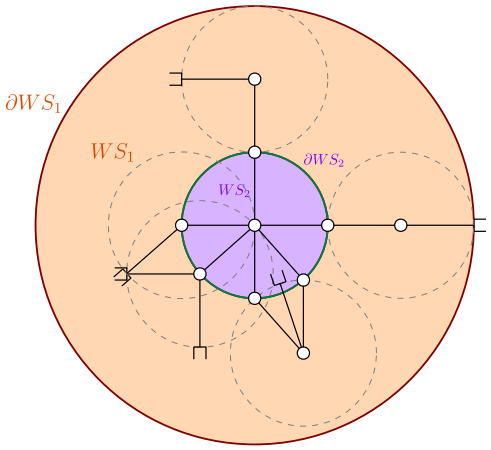
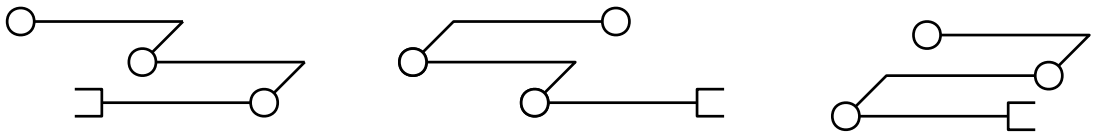


Figure 4.9: workspace of a planar 3R robotic manipulator

- In the space $WS_1 \backslash \{\partial WS_1 + \partial WS_2\}$ (the orange region), the end effector can lie in any position with a continuum number of possible orientation $\phi \in \Phi \subset [0, 2\pi]$, but *not all possible* orientation.
- In the region ∂WS_1 ($\|\mathbf{p}\| = 3l$) the end effector can lie in only one possible orientation (*singular case*).
- In the region ∂WS_2 ($\|\mathbf{p}\| = l$), the end effector can lie in any position and in any orientation, there are infinite solution for an IK problem in that region, but 3 of that solutions are singular:



These configurations are singular since the arm loses degree of freedom: In that particular configuration, the arm cant move in any arbitrary direction.

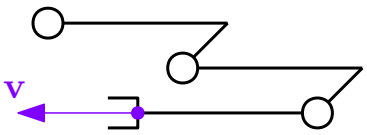


Figure 4.10: in the current configuration, the end effector can't move in the direction $\mathbf{v}$.

- In the region $WS_2 \backslash \partial WS_2$ ($\|\mathbf{p}\| < l$), the end effector can lie in any position and in any orientation, each solution is non singular.

In general, l_1, l_2, l_3 are not are not necessarily equal, the reachable points that are the most far from the first joint are at distance $l_1 + l_2 + l_3$, the closest one are at distance $\max\{0, l_{max} - (l_{med} + l_{min})\}$ where

$$l_{max} = \max\{l_1, l_2, l_3\} \tag{4.41}$$
$$l_{min} = \min\{l_1, l_2, l_3\} \tag{4.42}$$
$$l_{med} \text{ is the remaining one} \tag{4.43}$$

In general:

- if $m = n$, for an IK problem they may:
  - doesn't exist solutions (out of workspace)
  - exist a finite number of solution (regular case)
  - infinite set of solutions, or a finite set, in a different number respects to the regular case, these are the **singular case**.

- if $m < n$, the robot are mor DOF than the dimension of the task space, the robot is *kinematically redundant*, in that case for an IK problem they may:
  - doesn't exist solutions (out of workspace)
  - infinitely many solutions, in details, we say that there are ∞^{n-m} solutions, if the set Q' of joint's configuration that is a solution for the inverse kinematics, is a $n-m$-manifold.
  - they may be a finite or infinite number of **singular** solutions.

The formal definition of a **singular solution/case** will be formalized in chapter 5 in the context of differential kinematics.

4.3.2 Solution Methods

There are two ways to solve the inverse kinematics for a robotic arm:

- **Analytical Solution**: It is preferred, since it doesn't require too much computation, can be found algebraically by solving some polynomial equation derived from the DK equations. There are systematic ways to generate a reduced set of equations to solve. The following condition (for a 6 dof arm) are sufficient to find an analytical solution
  - there are 3 consecutive rotational joints that are parallel
  - there are 3 consecutive rotational joints that are incidents (spherical wrist).
- **Numerical Solution**: If the problem is too complex (i.e. in the redundant case where $n > m$) an iterative method to solve system of non linear equations can be used. These methods are easy to set up, but computationally expensive, the methods usually consider the Jacobian matrix of the direct kinematics map[2]
$$J_r(\mathbf{q}) = \left(\frac{\partial f_r^i}{\partial q_j}\right)$$
commons methods are Newton method, Quasi Newton Method, Gradient descent and so on.

4.3.3 Inverse Kinematics of a Planar 2R

We will inspect the IK for some common cases. The simplest one is the 2R planar arm, in figure 1.6. We will denote q_1 and q_2 the joint's angles. The direct kinematics for the position is

$$\mathbf{p}(\mathbf{q}) = \begin{pmatrix} p_x \\ p_y \end{pmatrix} = \begin{pmatrix} l_1 \cos q_1 + l_2 \cos(q_1 + q_2) \\ l_1 \sin q_1 + l_2 \sin(q_1 + q_2) \end{pmatrix} \tag{4.44}$$

we have to solve the system

$$\begin{cases} p_x = l_1 \cos q_1 + l_2 \cos(q_1 + q_2) \\ p_y = l_1 \sin q_1 + l_2 \sin(q_1 + q_2) \end{cases} \tag{4.45}$$

in the unknown q_1, q_2. In this case, one variable can be isolated by squaring and summing the two equations:

$$\begin{cases} p_x = l_1 \cos q_1 + l_2 \cos(q_1 + q_2) \\ p_y = l_1 \sin q_1 + l_2 \sin(q_1 + q_2) \end{cases} \implies \tag{4.46}$$

$$p_x^2 + p_y^2 = l_1^2 \cos^2 q_1 + l_2^2 \cos^2(q_1 + q_2) + 2 l_1 l_2 \cos q_1 \cos(q_1 + q_2) + \tag{4.47}$$

$$l_1^2 \sin^2 q_1 + l_2^2 \sin^2(q_1 + q_2) + 2 l_1 l_2 \sin q_1 \sin(q_1 + q_2) \implies \tag{4.48}$$

$$p_x^2 + p_y^2 - (l_1^2 + l_2^2) = 2 l_1 l_2 (\cos q_1 \cos(q_1 + q_2) + \sin q_1 \sin(q_1 + q_2)) = 2 l_1 l_2 \cos q_2 \implies \tag{4.49}$$

$$\cos q_2 = \frac{p_x^2 + p_y^2 - (l_1^2 + l_2^2)}{2 l_1 l_2} \tag{4.50}$$

We used the identity

$$\cos(x)\cos(x+y) + \sin(x)\sin(x+y) = \cos(y) \tag{4.51}$$

[2] with f_r^i is denoted the i-th component of the direct kinematics map f_r

Since we have $\cos q_2$, we can evaluate the two possible values for $\sin q_2$ since

$$\sin q_2 = \pm\sqrt{1 - \cos^2 q_2} \tag{4.52}$$

At this point we can evaluate the two possible values of q_2:

$$q_2^+ = \text{atan2}(\sqrt{1 - \cos^2 q_2}, \cos q_2) \tag{4.53}$$

$$q_2^- = \text{atan2}(-\sqrt{1 - \cos^2 q_2}, \cos q_2) \tag{4.54}$$

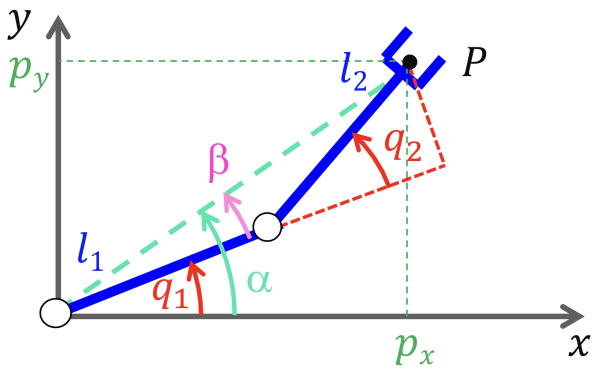
Since

$$\cos q_2 = \frac{p_x^2 + p_y^2 - (l_1^2 + l_2^2)}{2 l_1 l_2} \tag{4.55}$$

it is necessary that

$$\frac{p_x^2 + p_y^2 - (l_1^2 + l_2^2)}{2 l_1 l_2} \in [-1, 1] \tag{4.56}$$

if not, there is no value for q_2, the desired position p_x, p_y is outside of the workspace. Once we have evaluated q_2^+ and q_2^-, we can evaluate q_1 by inspecting the geometry of the problem.



The angle α is given by

$$\alpha = \text{atan2}(p_y, p_x) \tag{4.57}$$

the angle β can be found by considering the reference frame where the link l_1 is parallel to the x axis:

$$\beta = \text{atan2}(l_2 \sin(q_2), l_1 + l_2 \cos(q_2)) \tag{4.58}$$

q_1 is given by $\alpha - \beta$:

$$q_1 = \text{atan2}(p_y, p_x) - \text{atan2}(l_2 \sin(q_2), l_1 + l_2 \cos(q_2)) \tag{4.59}$$

Since there are two possible values for q_2, there are also two possible values for q_1:

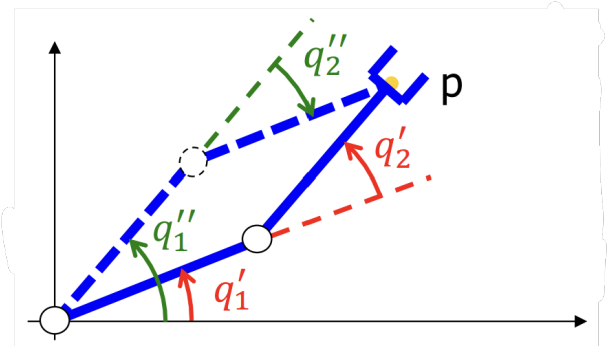
$$q_1^+ = \text{atan2}(p_y, p_x) - \text{atan2}(l_2 \sin(q_2^+), l_1 + l_2 \cos(q_2^+)) \tag{4.60}$$

$$q_1^- = \text{atan2}(p_y, p_x) - \text{atan2}(l_2 \sin(q_2^-), l_1 + l_2 \cos(q_2^-)) \tag{4.61}$$

Note: When the difference of these atan2 functions are evaluated, the result may be outside the interval $(-\pi, \pi]$, these value have to be re-expressed in the interval.

In the regular case, there are two possible solution for the IK problem:

$$\mathbf{q}' = \begin{pmatrix} q_1^+ \\ q_2^+ \end{pmatrix}, \qquad \mathbf{q}'' = \begin{pmatrix} q_1^- \\ q_2^- \end{pmatrix} \tag{4.62}$$



There is a singularity when the desired position is on the boundaries of WS_1, in that case, we have that

$$\sqrt{p_x^2 + p_y^2} = l_1 + l_2 \tag{4.63}$$

so

$$\cos q_2 = \frac{p_x^2 + p_y^2 - (l_1^2 + l_2^2)}{2l_1 l_2} = \frac{(l_1 + l_2)^2 - (l_1^2 + l_2^2)}{2l_1 l_2} = \frac{l_1^2 + l_2^2 + 2l_1 l_2 - l_1^2 - l_2^2}{2l_1 l_2} = 1 \tag{4.64}$$

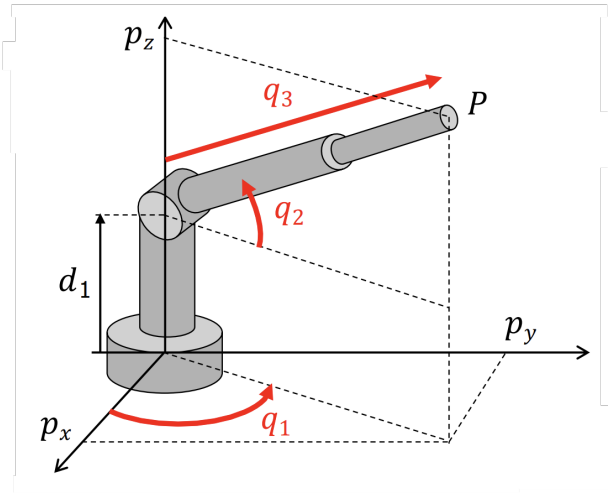
if $\cos q_2 = 1$, there is only one value for $\sin q_2$:

$$\sin q_2 = \pm\sqrt{1 - \cos^2 q_2} = \pm 0 = 0 \implies q_2 = 0 \tag{4.65}$$

There is also another singularity when $l_1 = l_2$ and the desired position is the origin, in that case, there are infinite solution (with $q_2 = \pi$ fixed).

4.3.4 Inverse Kinematics of a Polar RRP arm

We consider a robotic arm of the following type:



Note that q_2 is not a DH variable. The direct kinematics for this type of manipulator is the following

$$\begin{cases} p_x = q_3 \cos q_2 \cos q_1 \\ p_y = q_3 \cos q_2 \sin q_1 \\ p_z = d_1 + q_3 \sin q_2 \end{cases} \tag{4.66}$$

By summing and squaring the equations we can find the value of q_3

$$p_x^2 + p_y^2 + (p_z - d_1)^2 = q_3^2 \cos^2 q_2 \cos^2 q_1 + q_3^2 \cos^2 q_2 \sin^2 q_1 + q_3^2 \sin^2 q_2 \tag{4.67}$$

$$p_x^2 + p_y^2 + (p_z - d_1)^2 = q_3^2 (\cos^2 q_2 \cos^2 q_1 + \cos^2 q_2 \sin^2 q_1 + \sin^2 q_2) \tag{4.68}$$

$$p_x^2 + p_y^2 + (p_z - d_1)^2 = q_3^2 [\cos^2 q_2 (\cos^2 q_1 + \sin^2 q_1) + \sin^2 q_2] \tag{4.69}$$

$$p_x^2 + p_y^2 + (p_z - d_1)^2 = q_3^2 [\cos^2 q_2 + \sin^2 q_2] \tag{4.70}$$

$$q_3^2 = p_x^2 + p_y^2 + (p_z - d_1)^2 \tag{4.71}$$

$$q_3 = \pm\sqrt{p_x^2 + p_y^2 + (p_z - d_1)^2} = \pm\gamma \tag{4.72}$$

For simplicity, we denote $\gamma = \sqrt{p_x^2 + p_y^2 + (p_z - d_1)^2}$. If $q_3 = 0$, the end effector is on the point $(0, 0, d_1)$, in such case, every value of q_1 and q_2 works as a solution, we can say that the set of solutions for an IK problem where the end effector lies on $(0, 0, d_1)$ is a 2-manifold. We can solve the third equation for $\sin q_2$

$$p_z = d_1 \pm \gamma \sin q_2 \tag{4.73}$$

$$\sin q_2 = \pm\frac{p_z - d_1}{\gamma} \tag{4.74}$$

$$\tag{4.75}$$

We can find the two values for $\cos q_2$:

$$\cos q_2 = \pm\sqrt{1 - \left(\pm\frac{p_z - d_1}{\gamma}\right)^2} \tag{4.76}$$

Note how we have two values for $q_3 = \pm\gamma$ and two values for $\cos q_2$, at this point, we are identifying 4 possible solutions for (q_2, q_3). We can extract q_2 :

$$q_2 = \text{atan2}\left(\pm\frac{p_z - d_1}{\gamma}, \pm\sqrt{1 - \left(\pm\frac{p_z - d_1}{\gamma}\right)^2}\right) \tag{4.77}$$

The four solutions for now are

$$q_3 = \gamma, \quad q_2 = \text{atan2}\left(\frac{p_z - d_1}{\gamma}, \sqrt{1 - \left(\frac{p_z - d_1}{\gamma}\right)^2}\right) \tag{4.78}$$

$$q_3 = \gamma, \quad q_2 = \text{atan2}\left(\frac{p_z - d_1}{\gamma}, -\sqrt{1 - \left(\frac{p_z - d_1}{\gamma}\right)^2}\right) \tag{4.79}$$

$$q_3 = -\gamma, \quad q_2 = \text{atan2}\left(\frac{p_z - d_1}{-\gamma}, \sqrt{1 - \left(\frac{p_z - d_1}{-\gamma}\right)^2}\right) \tag{4.80}$$

$$q_3 = -\gamma, \quad q_2 = \text{atan2}\left(\frac{p_z - d_1}{-\gamma}, -\sqrt{1 - \left(\frac{p_z - d_1}{-\gamma}\right)^2}\right) \tag{4.81}$$

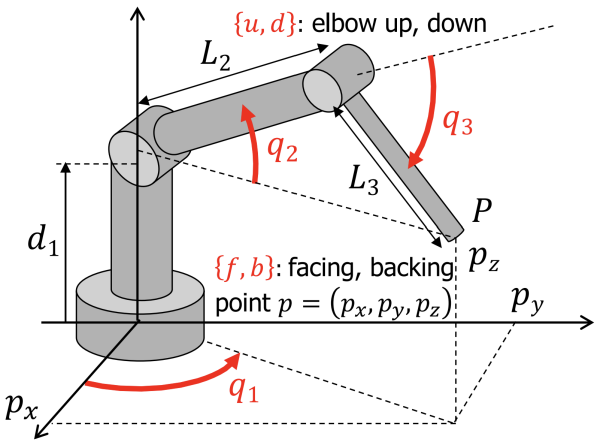
If we fix q_2, q_3, by the system of equation

$$\begin{cases} p_x = q_3 \cos q_2 \cos q_1 \\ p_y = q_3 \cos q_2 \sin q_1 \end{cases} \implies q_1 = \text{atan2}\left(\frac{p_y}{\cos q_2}, \frac{p_x}{\cos q_2}\right) \tag{4.82}$$

we find one value for q_1 (by solving it for $\cos q_1$ and $\sin q_1$), at the end, for an IK problem (where the task vector is the pose of the end effector) of this polar robot, there are 4 solutions in the regular case.

4.3.5 Inverse Kinematics of a 3R Elbow-type Arm

We now consider a manipulator with 3 rotational joints, as shown in the following picture.



The direct kinematics is given by the following system of equations:

$$\begin{cases} p_x = \cos q_1 (L_2 cos q_2 + L_3 \cos(q_2 + q_3)) \\ p_y = \sin q_1 (L_2 cos q_2 + L_3 \cos(q_2 + q_3)) \\ p_z = d_1 + L_2 \sin q_2 + L_3 \sin(q_2 + q_3) \end{cases} \tag{4.83}$$

In the third equation we can move d_1 to the left side

$$\begin{cases} p_x = \cos q_1(L_2 cosq_2 + L_3\cos(q_2+q_3)) \\ p_y = \sin q_1(L_2 cosq_2 + L_3\cos(q_2+q_3)) \\ p_z - d_1 = L_2\sin q_2 + L_3\sin(q_2+q_3) \end{cases} \tag{4.84}$$

then we square and sum the three equations:

$$p_x^2 + p_y^2 + p_z^2 = \tag{4.85}$$
$$\cos^2 q_1(L_2 cosq_2 + L_3\cos(q_2+q_3))^2 + \sin^2 q_1(L_2 cosq_2 + L_3\cos(q_2+q_3))^2 + \tag{4.86}$$
$$(L_2\sin q_2 + L_3\sin(q_2+q_3))^2 = \tag{4.87}$$
$$(L_2 cosq_2 + L_3\cos(q_2+q_3))^2 + (L_2\sin q_2 + L_3\sin(q_2+q_3))^2 = \tag{4.88}$$
$$L_2^2\sin^2 q_2 + L_3^2\sin^2(q_2+q_3) + 2L_2\sin q_2 L_3\sin(q_2+q_3) + \tag{4.89}$$
$$L_2^2\cos^2 q_2 + L_3^2\cos^2(q_2+q_3) + 2L_2\cos q_2 L_3\cos(q_2+q_3) = \tag{4.90}$$
$$L_2^2 + L_3^2 + 2L_2L_3(\sin q_2\sin(q_2+q_3) + \cos q_2\cos(q_2+q_3)) = \tag{4.91}$$

We use the identity $\sin q_2\sin(q_2+q_3) + \cos q_2\cos(q_2+q_3) = \cos q_3$:

$$p_x^2 + p_y^2 + p_z^2 = L_2^2 + L_3^2 + 2L_2L_3\cos q_3 \implies \tag{4.92}$$
$$2L_2L_3\cos q_3 = p_x^2 + p_y^2 + p_z^2 - L_2^2 - L_3^2 \implies \tag{4.93}$$
$$\cos q_3 = \frac{p_x^2 + p_y^2 + p_z^2 - L_2^2 - L_3^2}{2L_2L_3} \tag{4.94}$$

if $\frac{p_x^2+p_y^2+p_z^2-L_2^2-L_3^2}{2L_2L_3} \notin [-1,1]$, the point is out of the workspace. Else, we can consider

$$\sin q_3 = \pm s_3 = \pm\sqrt{1-\cos^2 q_3} \implies \begin{cases} q_3^{\{+\}} = \text{atan2}(s_3, \cos q_3) \\ q_3^{\{-\}} = \text{atan2}(-s_3, \cos q_3) \end{cases} \tag{4.95}$$

If we square and sum the first two equations we get

$$p_x^2 + p_y^2 = (L_2\cos q_2 + L_3\cos(q_2+q_3))^2 \implies \tag{4.96}$$
$$L_2\cos q_2 + L_3\cos(q_2+q_3) = \pm\sqrt{p_x^2+p_y^2} \tag{4.97}$$

So we rearrange the first two equations:

$$\begin{cases} p_x = \pm\cos q_1\sqrt{p_x^2+p_y^2} \\ p_y = \pm\sin q_1\sqrt{p_x^2+p_y^2} \end{cases} \implies \begin{cases} \cos q_1 = \frac{p_x}{\pm\sqrt{p_x^2+p_y^2}} \\ \sin q_1 = \frac{p_y}{\pm\sqrt{p_x^2+p_y^2}} \end{cases} \tag{4.98}$$

If $p_x^2 + p_y^2 = 0$, the sine and cosine of q_1 are undefined, so there are infinite solutions (singular case), else

$$\begin{cases} q_1^{\{+\}} = \text{atan2}(p_y, p_x) \\ q_1^{\{-\}} = \text{atan2}(-p_y, -p_x) \end{cases} \tag{4.99}$$

To find q_2, we have to combine the two equations and rearrange the last one in the following way:

$$\begin{cases} p_x = \cos q_1(L_2 cosq_2 + L_3\cos(q_2+q_3)) \\ p_y = \sin q_1(L_2 cosq_2 + L_3\cos(q_2+q_3)) \\ p_z = d_1 + L_2\sin q_2 + L_3\sin(q_2+q_3) \end{cases} \tag{4.100}$$

$$\begin{cases} p_x + p_y = \cos q_1(L_2 cosq_2 + L_3\cos(q_2+q_3)) + \sin q_1(L_2 cosq_2 + L_3\cos(q_2+q_3)) \\ p_z - d_1 = L_2\sin q_2 + L_3\sin(q_2+q_3) \end{cases} \tag{4.101}$$

$$\begin{cases} \cos q_1 p_x + \sin q_1 p_y = (L_2 + L_3\cos q_3)\cos q_2 - L_3\sin q_3\sin q_2 \\ p_z - d_1 = L_3\sin q_3\cos q_2 + (L_2 + L_3\cos q_3)\sin q_2 \end{cases} \tag{4.102}$$

Since q_1 and q_3 are known, this became a linear system in the variables $\cos q_2, \sin q_2$

$$\begin{cases} \cos q_1 p_x + \sin q_1 p_y = (L_2 + L_3 \cos q_3) \cos q_2 - L_3 \sin q_3 \sin q_2 \\ p_z - d_1 = L_3 \sin q_3 \cos q_2 + (L_2 + L_3 \cos q_3) \sin q_2 \end{cases} \tag{4.103}$$

In details, there are 4 linear systems for the 4 possible combinations of q_1, q_3

$$(q_1^{\{+\}}, q_3^{\{+\}}), \qquad (q_1^{\{+\}}, q_3^{\{-\}}), \qquad (q_1^{\{-\}}, q_3^{\{+\}}), \qquad (q_1^{\{-\}}, q_3^{\{-\}}) \tag{4.104}$$

q_2 is defined only if the system have the determinant different than zero:

$$p_x^2 + p_y^2 + (p_z - d_1)^2 \neq 0 \tag{4.105}$$

In the end, by solving the four systems we will obtain four regular solutions.

4.4 Numerical Solutions

Numerical solutions are applied in the most part in real life cases, if a task is redundant ($n > m$), it's impossible to find an analytical solution, since there are infinitely many in the regular case.

4.4.1 Newton Method

The newton method is an optimization iterative algorithm that aims to find the roots of a multi-variate real function. It's easy to understand the case with a function $f : \mathbb{R} \to \mathbb{R}$. This method aims fo find x^* such that $f(x^*) = 0$. This method can be applied to find an x that realize any real value r_d by setting $\hat{f} = f - r_d$ and applying the method to find x^* such that $\hat{f}(x^*) = 0 \implies f(x^*) = r_d$.

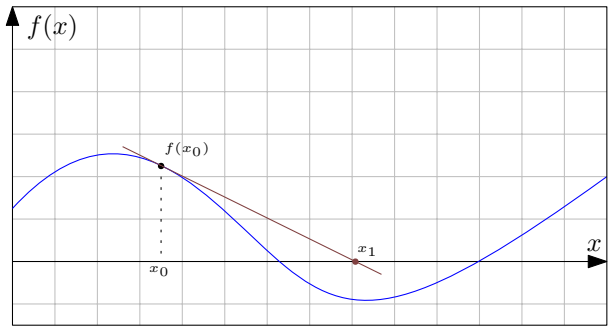
The method needs an initial guess x_0 to start, and consider a linear approximation of the function f in x_0:

$$f(x) \simeq f(x_0) + f'(x_0)(x - x_0) \tag{4.106}$$

we denote f' the derivative of f. To reach the next step, we consider the root of that linear approximation:

$$f(x_0) + f'(x_0)(x^* - x_0) = 0 \implies x^* = x_0 - \frac{f(x_0)}{f'(x_0)} \tag{4.107}$$

The points $x^* = x_1$ and x_0 (initial guess) is represented in the following picture.



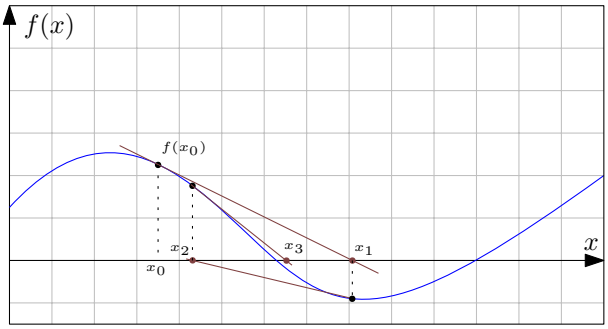
Then, we consider the next guess x_1 and linearize the function f on x_1, to get

$$f(x) \simeq f(x_1) + f'(x_1)(x - x_1) \tag{4.108}$$

the root of this linearized function will be the next guess x_2:

$$x_2 - \frac{f(x_1)}{f'(x_1)} \tag{4.109}$$

The step 2 and 3 is represented in the following picture.



In general, the method is applied iterative, with the $n+1$-th guess as

$$x_{n+1} = \frac{f(x_n)}{f'(x_n)} \tag{4.110}$$

Under sufficient conditions, the method will converge to the root of the function $f(x)$.

Let's now consider the robotic task where $n = m$ (no redundant case), we have the direct kinematics equation

$$\mathbf{r} = f_r(\mathbf{q}) = \begin{pmatrix} f_r^1(\mathbf{q}) \\ \vdots \\ f_r^m(\mathbf{q}) \end{pmatrix} \tag{4.111}$$

In a given initial guess $\mathbf{q}^0$, the linearize version of f_r as a function of $\mathbf{q}$ is

$$f_r(\mathbf{q}^0) + J_r(\mathbf{q}^0)(\mathbf{q} - \mathbf{q}^0) \tag{4.112}$$

where J_r is the analytical Jacobian of f_r

$$J_r = \left(\frac{\partial f_r^i}{\partial q_j}\right) = \begin{pmatrix} \frac{\partial f_r^1}{\partial q_1} & \cdots & \frac{\partial f_r^1}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_r^m}{\partial q_1} & \cdots & \frac{\partial f_r^m}{\partial q_n} \end{pmatrix} \tag{4.113}$$

Let $\mathbf{r}_d$ to be the desired solution for the inverse kinematics problem. The iterative method consider the $k+1$-th guess as:

$$\mathbf{q}^{k+1} = \mathbf{q}^k + J_r^{-1}(\mathbf{q}^k)(\mathbf{r}_d - f_r(\mathbf{q}^k)) \tag{4.114}$$

So, the Jacobian matrix J_r must be invertible during the iterations, in case of singularities, there will be problems. In case of redundancy ($m < n$), instead of the inverse, we consider the pseudo inverse of the Jacobian matrix

$$\mathbf{q}^{k+1} = \mathbf{q}^k + J_r^{\#}(\mathbf{q}^k)(\mathbf{r}_d - f_r(\mathbf{q}^k)) \tag{4.115}$$

where

$$J_r^{\#} = (J_r^T J_r)^{-1} J_r^T \tag{4.116}$$

We have singularities when $J_r^T J_r$ is not invertible. Without entering in the mathematical and computational complexity details, the Newton method converge when the initial guess is "close enough" to some $\mathbf{q}^*$ that realize $f_r(\mathbf{q}^*) = \mathbf{r}_d$, and the convergence rate is quadratic.

We will now discuss about singularities during the iterations. Let's assume that we iterate $k-1$ times, and we obtained a series of guesses that are converging to a solution $\mathbf{q}^*$

$$\{\mathbf{q}^0, \mathbf{q}^1, \mathbf{q}^2, \ldots, \mathbf{q}^{k-1}\} \longrightarrow \mathbf{q}^* \tag{4.117}$$

Let's assume that we are getting closer to the solution:

$$\|\mathbf{q}^i - \mathbf{q}^*\| > \|\mathbf{q}^{i+1} - \mathbf{q}^*\|, \quad \forall i \tag{4.118}$$

We are near a singularity at the k-th step, so, the matrix $J_r(\mathbf{q}^{k-1})$ is almost (but not) singular:

$$\det J_r(\mathbf{q}^{k-1}) \simeq 0 \tag{4.119}$$

so the determinant is not zero, but really close, we calculate the k-th guess by considering

$$\mathbf{q}^k = \mathbf{q}^{k-1} + J_r^{-1}(\mathbf{q}^{k-1})(\mathbf{r}_d - f_r(\mathbf{q}^{k-1})) \tag{4.120}$$

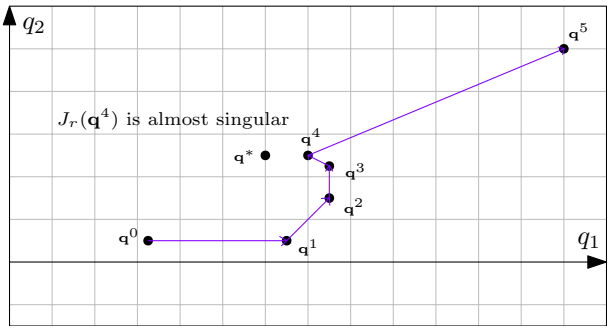
we use the identity

$$J_r^{-1} = \frac{1}{\det J_r} J_r^+ \tag{4.121}$$

where J_r^+ is the classical adjoint matrix of J_r. The k-th guess is

$$\mathbf{q}^k = \mathbf{q}^{k-1} + \frac{1}{\det J_r(\mathbf{q}^{k-1})} J_r^+(\mathbf{q}^{k-1})(\mathbf{r}_d - f_r(\mathbf{q}^{k-1})) \tag{4.122}$$

Since $\det J_r(\mathbf{q}^{k-1})$ is really small, matrix J_r^{-1} will have terms that grows really fast to infinity with $\det J_r(\mathbf{q}^{k-1}) \to 0$, and the next guess $\mathbf{q}^k$ will be really far from $\mathbf{q}^{k-1}$, and far from the solution $\mathbf{q}^*$, so this methods might diverge.



An useful visualization of the newton method in case $m = n = 1$ is presented to the following link.

`https://upload.wikimedia.org/wikipedia/commons/e/e0/NewtonIteration_Ani.gif`

4.4.2 Gradient Descent

The gradient descent is an iterative algorithm that aims to minimize a given function. In our context, we want to find a $\mathbf{q}^*$ that realize

$$f_r(\mathbf{q}^*) = \mathbf{r}_d \tag{4.123}$$

for a given task vector $\mathbf{r}_d$. We can define the following *error function*:

$$H(\mathbf{q}) = \frac{1}{2}\|\mathbf{r}_d - f_r(\mathbf{q})\|^2 = \frac{1}{2}(\mathbf{r}_d - f_r(\mathbf{q}))^T(\mathbf{r}_d - f_r(\mathbf{q})) \tag{4.124}$$

that, for a given $\mathbf{q}$, measure how far it is from a solution $\mathbf{q}^*$ (this distance is expressed in the task space). After defining H, we apply the gradient descent to minimize it, since, if $H(\mathbf{q}^*) = 0 \implies f_r(\mathbf{q}^*) = \mathbf{r}_d$. We start with an initial guess $\mathbf{q}^0$, the $k+1$-th guess is given by

$$\mathbf{q}^{k+1} = \mathbf{q}^k - \alpha \nabla H(\mathbf{q}^k) \tag{4.125}$$

$\nabla H(\mathbf{q}^k)$ is the gradient of H respect to $\mathbf{q}$, by expanding it we get

$$\mathbf{q}^{k+1} = \mathbf{q}^k + \alpha J_r^T(\mathbf{q}^k)(\mathbf{r}_d - f_r(\mathbf{q}^k)) \tag{4.126}$$

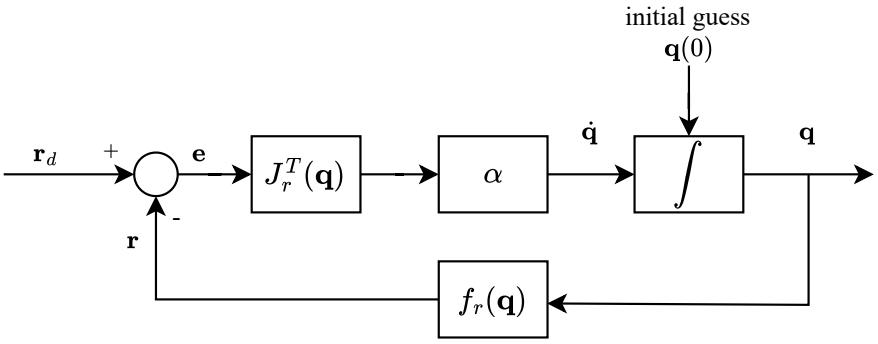
$\alpha \in (0, +\infty)$ is a real value called **step size**, and have to be chosen as an hyper parameter of the optimization.

- a small choice of α may leads to extremely slow convergence
- a large choice of α may lead the method to "miss" the minimum

Notice how this method is a good choice since at each step we don't have to invert the Jacobian matrix. We may also see revisit the gradient method as a feedback scheme, by considering continuous steps in time t instead of discrete steps k, in that case:

$$\mathbf{q}(t + dt) = \mathbf{q}(0) + \int_0^t K J_r^T(\mathbf{q}(\tau))(\mathbf{r}_d - f_r(\mathbf{q}(\tau)))d\tau \tag{4.127}$$

K is a gain that depends on α. The feedback scheme is the following:



$\mathbf{e}$ is the error $\mathbf{r}_d - f_r(\mathbf{q})$, this system is asymptotically stable and $\mathbf{e}$ will converge to 0. The function

$$V = \frac{1}{2}\mathbf{e}^T\mathbf{e} = H(\mathbf{q}) \tag{4.128}$$

is a *Lyapunov candidate function.*

Definition 8 *For a dynamical system, a* ***Lyapunov function*** *is a scalar function defined on the state space of the system, that proves the stability of an equilibrium point. A function V for the equilibrium point $\mathbf{x} = \mathbf{0}$ is a Lyapunov function if*

- $V(\mathbf{x}) > 0$ *for all* $\mathbf{x} \neq 0$ *and* $V(\mathbf{0}) = 0$
- $\dot{V}(\mathbf{x}) = \frac{dV}{dt} \leq 0$*, the time derivative must be less than or equal to zero.*

For now, we will not enter in the details of Lyapunov stability. Let's consider our Lyapunov candidate

$$V = \frac{1}{2}\mathbf{e}^T\mathbf{e} = H(\mathbf{q})$$

the time derivative is

$$\dot{V} = \dot{H} = -\alpha\mathbf{e}^T J_r(\mathbf{q}) J_r^T(\mathbf{q})\mathbf{e} \leq 0 \tag{4.129}$$

notice how $\dot{V} = 0 \iff \mathbf{e} \in \ker(J_r^T(\mathbf{q}))$. The Gradient descent is a good optimization method because:

- It never diverges, may converge to a point that is not a solution
- Have the same use for regular case $n = m$ and redundant case $n > m$
- It is computationally simpler since it uses the Jacobian transpose and not the inverse

CHAPTER

5

DIFFERENTIAL KINEMATICS

In this chapter we will explain the relation between the velocity of the joints $\dot{\mathbf{q}}$ and the velocity of the task vector $\dot{\mathbf{r}}$. This relation is explained by the Jacobian matrix, can be obtain by deriving the direct kinematics function respects to time, or by geometric reasoning if the task vector represents the pose of the end effector.

First, we recap some basics concepts about the velocity of a rigid body. Let B to be a set of points describing a rigid body, let $\mathbf{r}_i$ and $\mathbf{r}_j$ to be two points of B, since it is rigid, the distance between this two points is constant in time

$$\|\mathbf{r}_i - \mathbf{r}_j\| = \text{constant} \tag{5.1}$$

Let's consider two distinct points $\mathbf{r}_1, \mathbf{r}_2$, let $\mathbf{v}_1 = \dot{\mathbf{r}_1}$ and $\mathbf{v}_2 = \dot{\mathbf{r}_2}$ to be the velocities of the points. It is always true that the vector

$$\mathbf{v}_2 - \mathbf{v}_1$$

is orthogonal to $\mathbf{r}_{12}$, where this last one is the vector that goes from $\mathbf{r}_1$ to $\mathbf{r}_2$. Since it is orthogonal, there exists a vector $\boldsymbol{\omega}_1$ such that

$$\mathbf{v}_2 - \mathbf{v}_1 = \boldsymbol{\omega}_1 \times \mathbf{r}_{12} \tag{5.2}$$

Let $\mathbf{r}_3$ to be another point, it is also holds (for the same reason) that

$$\mathbf{v}_3 - \mathbf{v}_1 = \boldsymbol{\omega}_1 \times \mathbf{r}_{13} \tag{5.3}$$

and

$$\mathbf{v}_3 - \mathbf{v}_2 = \boldsymbol{\omega}_2 \times \mathbf{r}_{23} \tag{5.4}$$

We can subtract the first two equations

$$\mathbf{v}_2 - \mathbf{v}_1 - (\mathbf{v}_3 - \mathbf{v}_1) = \boldsymbol{\omega}_1 \times \mathbf{r}_{12} - (\boldsymbol{\omega}_1 \times \mathbf{r}_{13}) \implies \implies \tag{5.5}$$

$$\mathbf{v}_2 - \mathbf{v}_3 = \boldsymbol{\omega}_1 \times (\mathbf{r}_{12} - \mathbf{r}_{13}) \implies \tag{5.6}$$

$$\mathbf{v}_3 - \mathbf{v}_2 = \boldsymbol{\omega}_2 \times \mathbf{r}_{23} \tag{5.7}$$

So, it follows that

$$\boldsymbol{\omega}_1 = \boldsymbol{\omega}_2 \tag{5.8}$$

In fact, even if all the points of B have different velocities, they all shares the same $\boldsymbol{\omega}$, this is the *angular velocity* associated to the whole body (not to a single point). If $\boldsymbol{\omega} = \mathbf{0}$ but the velocities of the points are non zero, the object is moving in a pure translation motion. If some points have non zero velocity but there exists two distinct points with zero velocity, then the object is in a pure rotation motion (the points with zero velocity is on the axis of rotation).

Since

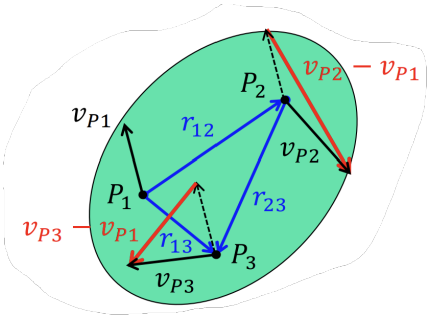
$$\mathbf{v}_j = \mathbf{v}_i + \boldsymbol{\omega} \times \mathbf{r}_{ij} \tag{5.9}$$

we can use the skew symmetric matrix to write

$$\mathbf{v}_j = \mathbf{v}_i + S(\boldsymbol{\omega})\mathbf{r}_{ij} \tag{5.10}$$

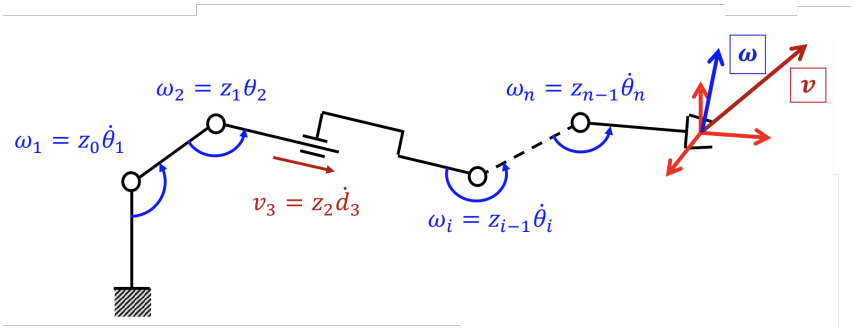
and this holds if and only if

$$\dot{\mathbf{r}}_{ij} = \boldsymbol{\omega} \times \mathbf{r}_{ij} \tag{5.11}$$



5.1 End Effector Velocities

The end effector have an angular velocity $\boldsymbol{\omega}$ and a linear velocity $\mathbf{v}$.



These two quantities are vectors. The pose of the end effector is described by 6 scalars:

$$\mathbf{r} = (\mathbf{p}, \phi) \tag{5.12}$$

The first, $\mathbf{p}$ is the position, and is a vector of $\mathbb{R}^3$, the velocity $\mathbf{v}$ is just the time derivative of $\mathbf{p}$.

$$\dot{\mathbf{p}} = \mathbf{v}$$

The other 3 scalars $\phi = \langle \phi_1, \phi_2, \phi_3 \rangle$ are the minimal representation of the orientation of the end effector. ϕ is not a vector since is not an element of any vector space, in fact:

$$\dot{\phi} \neq \boldsymbol{\omega} \tag{5.13}$$

The pose can also be described by an homogeneous matrix

$$T = \begin{pmatrix} R & \mathbf{p} \\ \mathbf{0}^T & 1 \end{pmatrix} \tag{5.14}$$

So, the orientation is described by R or ϕ, these quantities are not vectors. The angular speed is described by a vector $\boldsymbol{\omega}$, and this is not directly the time derivative of ϕ. We want to explicit the relation between $\boldsymbol{\omega}$ and the time derivative of R. $R(\mathbf{q})$ is variable in time if we give a time law $\mathbf{q}(t)$ to the joints, so it is well defined the time derivative of the matrix R.

$$R = (R_{ij}(\mathbf{q})) \implies \dot{R} = \left(\frac{dR_{ij}}{dt}\right) \tag{5.15}$$

Let's consider now a finite displacement of a rigid body (or a point)

$$\begin{pmatrix}\Delta x\\ \Delta y\\ \Delta z\end{pmatrix} \tag{5.16}$$

we know that, by applying the displacement in series by different coordinates, the final position of the rigid body will be independent from the order in which we apply these displacements.

$$\mathbf{p} + \Delta x \begin{pmatrix}1\\0\\0\end{pmatrix} + \Delta z \begin{pmatrix}0\\0\\1\end{pmatrix} = \mathbf{p} + \Delta z \begin{pmatrix}0\\0\\1\end{pmatrix} + \Delta x \begin{pmatrix}1\\0\\0\end{pmatrix} \tag{5.17}$$

this holds also for infinitesimal displacement dx, dy, dz.

> translating a point by Δx on the x axis and then by Δz on the z axis is the same that translating by Δz on the z axis and then by Δx on the x axis.

This doesn't holds for finite rotation, in general:

$$R_x(\alpha)R_z(\gamma)\mathbf{p} \neq R_z(\gamma)R_x(\alpha)\mathbf{p} \tag{5.18}$$

This is true because the function $\boldsymbol{\omega}(t)$ is not an exact *differential form*, in particular, the integral of $\boldsymbol{\omega}$ over time depends on the choice of the integration path, and not only on the initial and final value of $\boldsymbol{\omega}$, for example, in $t \in [0, 2]$ seconds, if we have these time law for $\boldsymbol{\omega}$:

$$\boldsymbol{\omega}^1(t) = \begin{cases}\omega_x^1(t) = 0 \text{ if } t < 1,\ 90^\circ \text{ else}\\ \omega_y^1(t) = 0\\ \omega_z^1(t) = 90^\circ \text{ if } t < 1,\ 0 \text{ else}\end{cases} \qquad \boldsymbol{\omega}^2(t) = \begin{cases}\omega_x^2(t) = 90^\circ \text{ if } t < 1,\ 0 \text{ else}\\ \omega_y^2(t) = 0\\ \omega_z^2(t) = 0 \text{ if } t < 1,\ 90^\circ \text{ else}\end{cases} \tag{5.19}$$

it is true that

$$\int_0^2 \boldsymbol{\omega}^1(t)dt \text{ and } \int_0^2 \boldsymbol{\omega}^2(t)dt \tag{5.20}$$

will lead to different final orientation. In particular, the results of the two integrals will be the same:

$$\begin{pmatrix}90^\circ\\0\\90^\circ\end{pmatrix} \tag{5.21}$$

but the final orientations (described by ϕ or R) will be different.

Differently from the finite case, infinitesimal rotations commute, since

$$\cos(dx) = 1 \tag{5.22}$$

$$\sin(dx) = dx \tag{5.23}$$

we have that

$$R_x(d\phi_x) = \begin{pmatrix}1 & 0 & 0\\ 0 & 1 & -d\phi_x\\ 0 & d\phi_x & 1\end{pmatrix} \tag{5.24}$$

$$R_y(d\phi_y) = \begin{pmatrix}1 & 0 & d\phi_y\\ 0 & 1 & 0\\ -d\phi_y & 0 & 1\end{pmatrix} \tag{5.25}$$

$$R_z(d\phi_z) = \begin{pmatrix}1 & -d\phi_z & 0\\ d\phi_z & 1 & 0\\ 0 & 0 & 1\end{pmatrix} \tag{5.26}$$

In that case all the 12 possible products in any order of the three matrices leads to the same results:

$$\begin{pmatrix}1 & -d\phi_z & d\phi_y\\ d\phi_z & 1 & -d\phi_x\\ -d\phi_y & d\phi_x & 1\end{pmatrix} = I + S(d\phi) \tag{5.27}$$

where S is the skew symmetric matrix associated to the triplets $d\phi = (d\phi_x, d\phi_y, d\phi_z)$.

We introduce now the time derivative of the rotation matrix, and we will describe the relation between that quantity and the angular velocity vector $\boldsymbol{\omega}$. Let's say that $R(\mathbf{q})$ is the orientation matrix of the end effector in function of the joint's angles $\mathbf{q}(t)$, that depends on time variable t, we can directly define $R(t)$ in function of time. Since $R \in SO(3)$, $R^T = R^{-1} \implies RR^T = I$. Since $R(t)R^T(t)$ is constant, it's time derivative is zero

$$R(t)R^T(t) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \implies \frac{d}{dt}(R(t)R^T(t)) = \begin{pmatrix} \frac{d}{dt}1 & \frac{d}{dt}0 & \frac{d}{dt}0 \\ \frac{d}{dt}0 & \frac{d}{dt}1 & \frac{d}{dt}0 \\ \frac{d}{dt}0 & \frac{d}{dt}0 & \frac{d}{dt}1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} = M_0 \tag{5.28}$$

It's also true, for the chain rule, that

$$\frac{d}{dt}(R(t)R^T(t)) = \frac{d}{dt}(R(t))R^T(t) + R(t)\frac{d}{dt}(R^T(t)) \tag{5.29}$$

from equations (5.28) and (5.29) we get

$$\frac{d}{dt}(R(t))R^T(t) + R(t)\frac{d}{dt}(R^T(t)) = M_0 \tag{5.30}$$

where M_0 denotes the 3×3 matrix with all the components equal to zero. Since $R(t)\frac{d}{dt}(R^T(t)) = (\frac{d}{dt}(R(t))R^T(t))^T$ we have

$$\frac{d}{dt}(R(t))R^T(t) + (\frac{d}{dt}(R(t))R^T(t))^T = M_0 \implies \tag{5.31}$$

$$\frac{d}{dt}(R(t))R^T(t) = -(\frac{d}{dt}(R(t))R^T(t))^T \tag{5.32}$$

So $\frac{d}{dt}(R(t))R^T(t)$ is a skew symmetric matrix, that we denote $S(t)$

$$S(t) = \frac{d}{dt}(R(t))R^T(t) \implies \frac{d}{dt}(R(t)) = S(t)R(t) \tag{5.33}$$

Let's now consider a vector $\mathbf{p}(t)$ with constant norm that is being rotated over time:

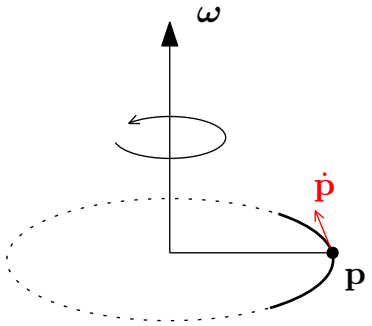
$$\mathbf{p}(t) = R(t)\mathbf{p}' \tag{5.34}$$

where $\mathbf{p}'$ is constant and $R(t)$ is a time dependent rotation matrix. Let's consider the time derivative of $\mathbf{p}$

$$\dot{\mathbf{p}}(t) = \frac{dR}{dt}\mathbf{p}' \tag{5.35}$$

from (5.33) we get to:

$$\dot{\mathbf{p}}(t) = S(t)R(t)\mathbf{p}' = S(t)\mathbf{p}(t) \tag{5.36}$$



Since $\mathbf{p}$ it exists a vector $\boldsymbol{\omega}(t)$ such that

$$\dot{\mathbf{p}}(t) = \boldsymbol{\omega}(t) \times \mathbf{p}(t) \tag{5.37}$$

the cross product can be rewritten with the skew symmetric matrix associated to $\boldsymbol{\omega}$

$$\dot{\mathbf{p}}(t) = S(\boldsymbol{\omega}(t))\mathbf{p}(t) \tag{5.38}$$

it follows that

$$S(t) = S(\boldsymbol{\omega}(t)) \tag{5.39}$$

So at the end we derive the following relations between the time derivative of a rotation matrix and the angular velocity vector:

$$\dot{R} = S(\boldsymbol{\omega})R \tag{5.40}$$

$$S(\boldsymbol{\omega}) = \dot{R}R^T \tag{5.41}$$

The orientation over time can be described by a rotation matrix R or by 3 angles (α, β, γ) (RPY), we derived the relation between the derivative of R and the angular velocity, we now consider the time derivative of the RPY angles.

Given the time derivatives of the angles

$$\begin{pmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{pmatrix} \tag{5.42}$$

around three axis, we consider the roll pitch yaw matrix $T(\alpha, \beta, \gamma)$ (depends on the order of the axis). The following holds:

$$\boldsymbol{\omega} = T(\alpha, \beta, \gamma) \begin{pmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{pmatrix} \tag{5.43}$$

5.1.1 The Jacobian Matrix

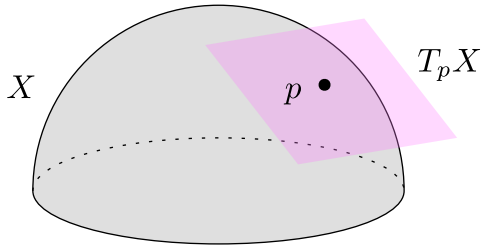
Given two differential manifolds X, Y, and a function $f : X \to Y$, the Jacobian is a linear application defined between the tangent spaces of X and Y in a given point, it would be overly long to introduce the space of derivation and the tangent space for differential manifold, so we will give a non general definition of the Jacobian without entering in the differential geometry field. It's important to know that the joint space Q (the set of all possible configuration $\mathbf{q}$) and the task space R (the set of all possible task vectors $\mathbf{r}$) are manifolds, and the direct kinematics is a differentiable function defined on them

$$f_r : Q \to R \tag{5.44}$$

A manifold, is (briefly) a topological space that is locally homeomorphic to an open set of $\mathbb{R}^n$, the joint space of a planar 2R robot is a torus

$$Q = S^1 \times S^1 \tag{5.45}$$

where S^1 denotes the unit sphere in $\mathbb{R}^2$ (circle). The torus is a 2-manifold because is locally homeomorphic to $\mathbb{R}^2$. We can imagine (not in a formal way) the tangent space of a manifold in a point p as the tangent plane in that point (this holds for manifolds that are embedded in $\mathbb{R}^n$). Usually, if X is a manifold and $p \in X$ is a point, the tangent space in p is denoted T_pX.



The Jacobian J of a differentiable function $f : X \to Y$ between two manifolds in a given point p is a linear mapping

$$J : T_pX \to T_pY \tag{5.46}$$

If $X \subseteq \mathbb{R}^n$ and $Y \subseteq \mathbb{R}^m$, the Jacobian is a matrix with m rows and n columns. In the robotics domain, the tangent space of Q in a given point is the set of all possible joint's velocities $\dot{\mathbf{q}}$, the tangent space in a given point of R is set of all possible task vector velocities, if the task vector is the pose of the end effector, this is the set of all possible linear and angular velocities of the end effector.

The Jacobian matrix J_r of the direct kinematics function f_r in a given point is a linear map that describe the relation between joint's velocities and task vector velocities.

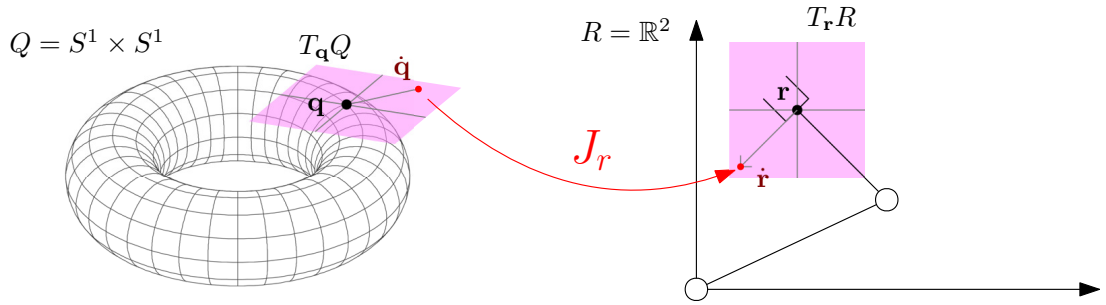
Since $\mathbf{r} = f_r(\mathbf{q})$, by deriving respect to time, due to the chain rule we get

$$\dot{\mathbf{r}} = \left(\frac{\partial f_r}{\partial \mathbf{q}}\right)\dot{\mathbf{q}} = J_r(\mathbf{q})\dot{\mathbf{q}} \tag{5.47}$$

the **analytical Jacobian** is the following matrix

$$J_r = \left(\frac{\partial f_r^i}{\partial q_j}\right) = \begin{pmatrix} \frac{\partial f_r^1}{\partial q_1} & \cdots & \frac{\partial f_r^1}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_r^m}{\partial q_1} & \cdots & \frac{\partial f_r^m}{\partial q_n} \end{pmatrix} \tag{5.48}$$

and depends on the actual configuration $\mathbf{q}$.



Let's see an example for a planar 2R, the direct kinematics for the end effector pose is

$$f_r(q_1, q_2) = \mathbf{r} = \begin{cases} p_x = l_1 \cos q_1 + l_2 \cos(q_1 + q_2) \\ p_y = l_1 \sin q_1 + l_2 \sin(q_1 + q_2) \\ \phi = q_1 + q_2 \end{cases} \tag{5.49}$$

The Jacobian matrix is

$$J_r(q_1, q_2) = \begin{pmatrix} -l_1 \sin q_1 - l_2 \sin(q_1 + q_2) & -l_2 \sin(q_1 + q_2) \\ l_1 \cos q_1 + l_2 \cos(q_1 + q_2) & l_2 \cos(q_1 + q_2) \\ 1 & 1 \end{pmatrix} \tag{5.50}$$

the joint's velocity $\dot{\mathbf{q}}$ will produce an end effector velocity of

$$\dot{\mathbf{r}} = J_r(\mathbf{q})\dot{\mathbf{q}} = \begin{pmatrix} -l_1 \sin q_1 - l_2 \sin(q_1 + q_2) & -l_2 \sin(q_1 + q_2) \\ l_1 \cos q_1 + l_2 \cos(q_1 + q_2) & l_2 \cos(q_1 + q_2) \\ 1 & 1 \end{pmatrix} \begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{\phi} \end{pmatrix} \tag{5.51}$$

The analytical Jacobian is defined for an arbitrary task vector, in case $\mathbf{r}$ is the pose of the end effector, there is a parallel description of the Jacobian that can be defined without evaluating the partial derivatives of the direct kinematics. We call that matrix the **geometric Jacobian**, if n are de degrees of freedom, is always a $6 \times n$ matrix.

- The analytical Jacobian is denoted J_r
- The geometric Jacobian is denoted J, but can be defined as two $3 \times n$ matrices J_L, J_A

$$J(\mathbf{q}) = \begin{pmatrix} J_L(\mathbf{q}) \\ J_A(\mathbf{q}) \end{pmatrix} = \begin{pmatrix} J_{L1}(\mathbf{q}) & \cdots & J_{Ln}(\mathbf{q}) \\ J_{A1}(\mathbf{q}) & \cdots & J_{An}(\mathbf{q}) \end{pmatrix} \tag{5.52}$$

J_L is related to the linear velocity, J_A to the angular velocity. The velocity of the pose is given by

$$\dot{\mathbf{r}} = \begin{pmatrix} \mathbf{v}_E \\ \boldsymbol{\omega}_E \end{pmatrix} = \begin{pmatrix} J_{L1}(\mathbf{q}) & \cdots & J_{Ln}(\mathbf{q}) \\ J_{A1}(\mathbf{q}) & \cdots & J_{An}(\mathbf{q}) \end{pmatrix} \dot{\mathbf{q}} \tag{5.53}$$

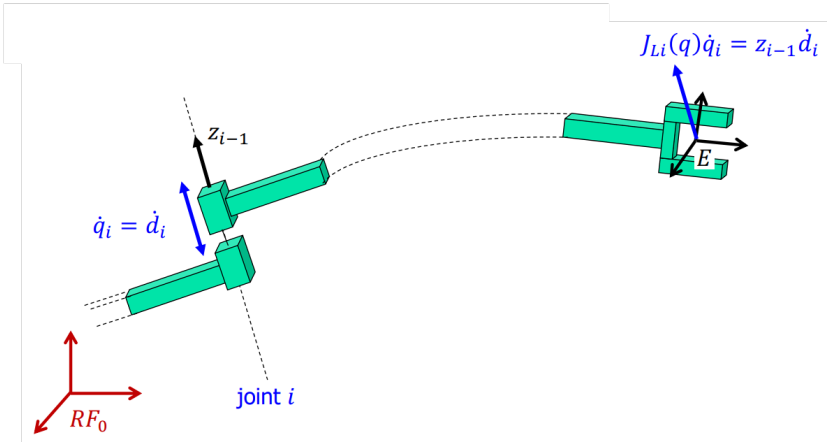
$\mathbf{v}_E$ and $\boldsymbol{\omega}_E$ are the linear and angular velocities of the end effector. For the principle of super position we have that

$$\mathbf{v}_E = J_{L1}(\mathbf{q})\dot{q}_1 + \cdots + J_{Ln}(\mathbf{q})\dot{q}_n \tag{5.54}$$
$$\boldsymbol{\omega}_E = J_{A1}(\mathbf{q})\dot{q}_1 + \cdots + J_{An}(\mathbf{q})\dot{q}_n \tag{5.55}$$

where $\dot{q}_i$ is the i-the component of the vector $\dot{\mathbf{q}}$. In this way we can split the contribution to the linear and the angular velocity. This contributions are the 3 dimensional vectors J_{Li}, J_{Ai} (the columns of the matrix) and can be derived by geometric inspection.

Prismatic Joint

We discuss how a prismatic joint q_i that is changing $\dot{q}_i \neq 0$ is related to the final linear and angular velocity of the end effector. We can imagine all the other joint's as freezed:



Let $\dot{d}_i$ to be the velocity of the joint i, the end effector will move at the same speed along the i-th joint axes, that we denote $\mathbf{z}_{i-1}$.

- $\mathbf{z}_0$ is the z axis of the first joint, in the reference frame RF_0, is just $(0\ 0\ 1)^T$.
- $\mathbf{z}_i$ is the z axis of the second joint
- ...
- $\mathbf{z}_{n-1}$ is the z axis of the n-th (last) joint.

In general, if ${}^{i-1}T_i$ is the homogeneous transformation derived from the DH assignment of frames, and ${}^{i-1}R_i$ is the related rotation matrix (the 3×3 sub matrix), then, the z axis of the i-th joint is given by

$$\mathbf{z}_{i-1} = {}^0R_{i-1}\mathbf{z}_0 = {}^0R_1\,{}^1R_2 \ldots {}^{i-2}R_{i-1}\begin{pmatrix}0\\0\\1\end{pmatrix} \tag{5.56}$$

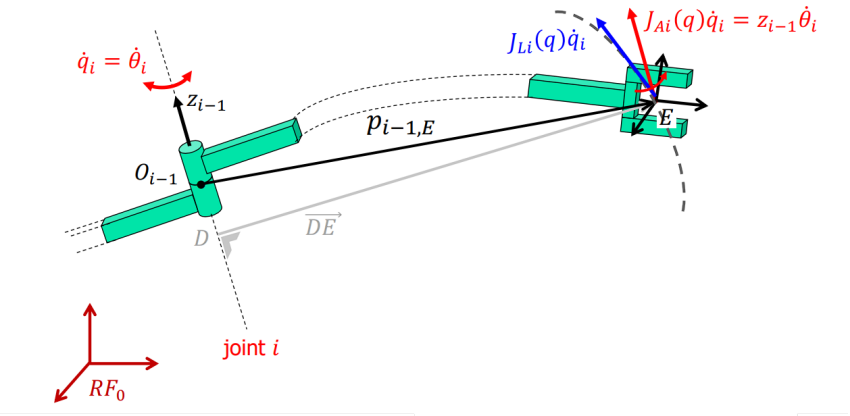
Clearly each rotation matrix ${}^{i-1}R_i$ depends on the current configuration $\mathbf{q}$. In the end, if the i-th joint is moving with a speed of $\dot{d}_i$, in the reference frame RF_0 the end effector will have a linear velocity of $\mathbf{z}_{i-1}\dot{d}_i$. The contribution on the angular velocity from a prismatic joint is null.

$$\text{linear contribution: } J_{Li}(\mathbf{q}) = \mathbf{z}_{i-1} = {}^0R_{i-1}\begin{pmatrix}0\\0\\1\end{pmatrix} \tag{5.57}$$

$$\text{angular contribution: } J_{Ai}(\mathbf{q}) = \mathbf{0} \tag{5.58}$$

Revolut Joint

Let's see the contribution of a revolut joint, this one will influence either the angular and the linear velocity.

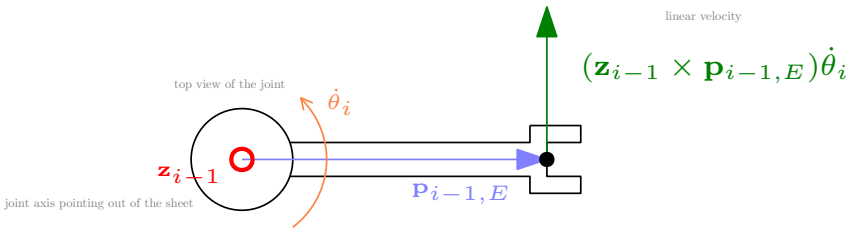


The angular contribution is straightforward, if a revolut joint i have a speed of $\dot{\theta}_i$, then the end effector will have an angular velocity of $\mathbf{z}_{i-1}\dot{\theta}_i$.

We denote $\mathbf{p}_{i-1,E}$ the position of the end effector expressed in the frame of the i-th joint.

- $\mathbf{p}_{0,E}$ is the end effector position expressed in the first frame, so is given by the complete direct kinematics of the manipulator
- $\mathbf{p}_{n-1,E}$ is the position of the end effector expressed in the last joint frame, that corresponds to the origin of the end effector, so $\mathbf{p}_{n-1,E} = \mathbf{0}$.

It's easy to see how the linear velocity given from the revolut joint is orthogonal to the z axis of the frame.

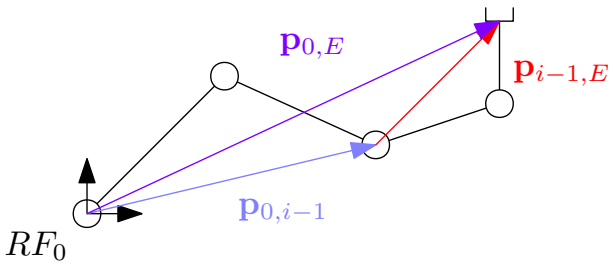


So, the i-th row of the linear part of the geometric Jacobian is $\mathbf{z}_{i-1} \times \mathbf{p}_{i-1,E}$. It's not hard to find the expression of the vector $\mathbf{p}_{i-1,E}$. We consider the matrix

$$^0T_{i-1} = {}^0T_1\,{}^1T_2 \dots {}^{i-2}T_{i-1} \tag{5.59}$$

We denote $\mathbf{p}_{0,i-1}$ the fourth columns of the matrix $^0T_{i-1}$, this is the center of the i-th joint's frame expressed in RF_0:

$$\mathbf{p}_{0,i-1} = {}^0T_{i-1} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{5.60}$$



Then, $\mathbf{p}_{i-1,E}$ is the difference between the complete direct kinematics and $\mathbf{p}_{0,i-1}$

$$\mathbf{p}_{i-1,E} = \mathbf{p}_{0,E} - \mathbf{p}_{0,i-1} \tag{5.61}$$

Using the homogeneous matrices:

$$\mathbf{p}_{i-1,E} = {}^0T_n \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} - {}^0T_{i-1} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \implies \tag{5.62}$$

$$\mathbf{p}_{i-1,E} = ({}^0T_n - {}^0T_{i-1}) \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{5.63}$$

In this discussion, we consider $\mathbf{p}_{i-1,E}$ with the 3 component vector, removing the last component (useful for the homogeneous transformations). The i-th row for the linear and angular part given by a revolut joint i are:

$$\text{linear contribution: } J_{Li}(\mathbf{q}) = \mathbf{z}_{i-1} \times \mathbf{p}_{i-1,E} \tag{5.64}$$

$$\text{angular contribution: } J_{Ai}(\mathbf{q}) = \mathbf{z}_{i-1} = {}^0R_{i-1} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{5.65}$$

$$\tag{5.66}$$

5.1.2 Mobility Analysis

We will now consider the space of the torques of the joint's (a more detailed discussion will be given in the following sections). Let's say that τ_i is the scalar torque of the i-th joint, the vector $\boldsymbol{\tau}$ is in the torque space (a subset of $\mathbb{R}^n$, this depends on the capabilities of the motors).

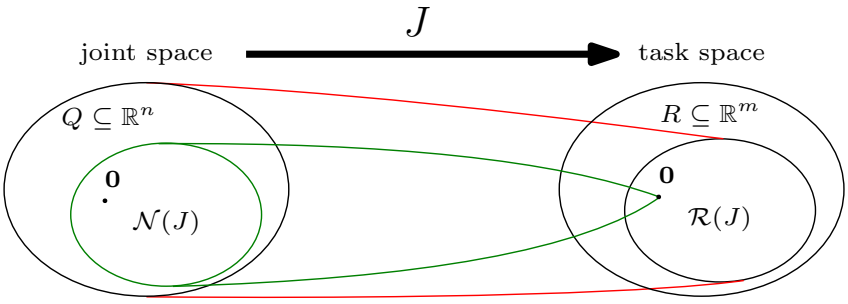
Definition 9 *given two subspaces of V: W_1, W_2, such that*

$$W_1 \cap W_2 = \{\mathbf{0}\} \tag{5.67}$$

*the **direct sum** $W_1 \oplus W_2$ is the subspace of V such that*

$$\mathbf{w} \in W_1 \oplus W_2 \implies \exists \mathbf{w}_1 \in W_2,\ \mathbf{w}_2 \in W_2 \text{ such that } \mathbf{w}_1 + \mathbf{w}_2 = \mathbf{w} \tag{5.68}$$

The following diagram resumes the space where the Jacobian matrix operates.



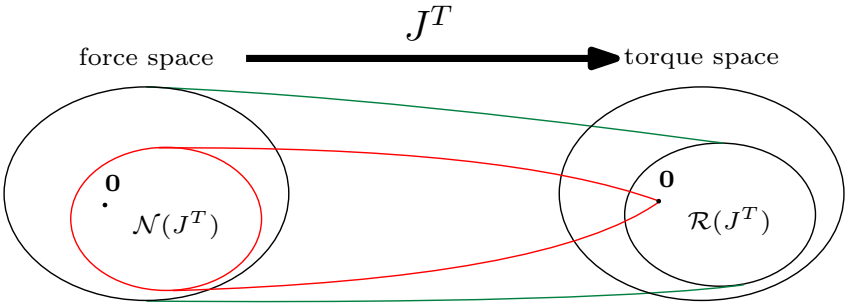
With $\mathcal{N}(J)$ is denoted the kernel (null space) of J (also denote $\ker(J)$), when with $\mathcal{R}(J)$ is denoted the range of J.

$$\mathcal{N}(J) = \{\mathbf{q} \in Q \ :\ J\mathbf{q} = \mathbf{0}\} \tag{5.69}$$

$$\mathcal{R}(J) = \{\mathbf{r} \in \mathbb{R}^m \ :\ \exists \mathbf{q} \in \mathbb{R}^n \text{ such that } J\mathbf{q} = \mathbf{r}\} \tag{5.70}$$

The transpose of the Jacobian matrix is a map between the torque space and the force space for the task vector. Given $\boldsymbol{\tau}$, the resulting force applied on the task vector is given by $\mathbf{F}$ that satisfies:

$$\boldsymbol{\tau} = J^T\mathbf{F} \tag{5.71}$$



The null space of the Jacobian is the **dual space** of the range of the Jacobian transpose, and the null space of the Jacobian transpose is the dual space of the range of the Jacobian, this means that

$$\mathcal{R}(J^T) \oplus \mathcal{N}(J) = \mathbb{R}^n \tag{5.72}$$

$$\mathcal{R}(J) \oplus \mathcal{N}(J^T) = \mathbb{R}^m \tag{5.73}$$

The range and the null space of the Jacobian matrix is directly related to the mobility of the end effector for a manipulator.

- $\mathcal{R}(J(\mathbf{q}))$ is the subspace og the generalized velocities of the task vector that can be instantaneously realized in the current configuration $\mathbf{q}$.
- if $\mathbf{v}_d \notin \mathcal{R}(J(\mathbf{q}))$, there is no joint's velocities $\dot{\mathbf{q}}$ that can realize the desired task velocity $\mathbf{v}_d$. This happens if the rank $\rho(J)$ of the Jacobian *is not maximal*.
- if $\rho(J) = m$, the Jacobian have max rank, so any task velocity can be realized (the end effector can be moved in any direction of the task space $\mathbb{R}^m$).
- if $\rho(J) < m$, there are directions in which the end effector cannot move (instantaneously). The forbidden velocities is the ones contained in $\mathcal{N}(J^T(\mathbf{q}))$, since is the complement of $R(J(\mathbf{q}))$ that has dimension $m - \rho(J)$.
- if $\mathcal{N}(J(\mathbf{q})) \neq \{\mathbf{0}\}$ there are joint velocities $\dot{\mathbf{q}} \neq \mathbf{0}$ that produces zero motion of the end effector (or in general, task vector). This always happens if the robot is redundant: $m < n$.

Some examples will clarify the relations between the Jacobian matrix and the mobility of the end effector. Let's consider a planar 3R robot ($m = 2$, $n = 3$) with unitary lengths for the links. The Jacobian matrix is the 2×3 matrix:

$$J(\mathbf{q}) = \begin{pmatrix} -s_1 - s_{12} - s_{123} & -s_{12} - s_{123} & -s_{123} \\ c_1 + c_{12} + c_{123} & c_{12} + c_{123} & c_{123} \end{pmatrix} \tag{5.74}$$

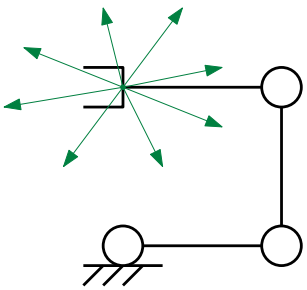
where $c_1 = \cos(q_1), c_{123} = \cos(q_1 + q_2 + q_3)$ and so on. The end effector velocity is given by

$$\dot{\mathbf{p}} = J(\mathbf{q})\dot{\mathbf{q}} \tag{5.75}$$

In the configuration $\mathbf{q}^* = (0, \frac{\pi}{2}, \frac{\pi}{2})$, the Jacobian evaluates in

$$J(\mathbf{q}^*) = \begin{pmatrix} -1 & -1 & 0 \\ 0 & -1 & -1 \end{pmatrix} \tag{5.76}$$

the rank is maximal: $\rho(J) = 2$, so the end effector can move in any direction in $\mathbb{R}^2$.



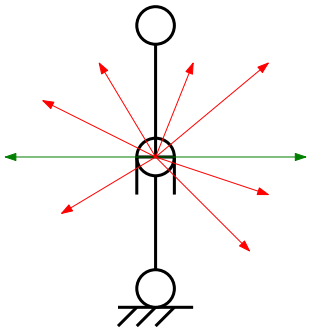
Now, let $\mathbf{q}^* = (\frac{\pi}{2}, 0, \pi)$the Jacobian evaluates in

$$J(\mathbf{q}^*) = \begin{pmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \tag{5.77}$$

the matrix doesn't have full rank since $J_1 + J_3 = \mathbf{0}$, we have that $\rho(J) = 1$. The only directions which the end effector can move are the ones in the range:

$$\mathcal{R}(J) = \text{span}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \{\begin{pmatrix} \alpha \\ 0 \end{pmatrix} \in \mathbb{R}^2 \ : \ \alpha \in \mathbb{R}\} \tag{5.78}$$

The end effector cannot move along the y axis.



5.1.3 Kinematic Singularities

We already called a "singularity" a case where the number of solutions for the inverse kinematics of a robot differs from the regular case, we say that a configuration $\mathbf{q}$ for the joint's variables is **singular** if we cannot find a vector for the joint's velocities $\dot{\mathbf{q}}$ that realizes a desired end effector velocity $\dot{\mathbf{r}}$ in some directions of the task space. This happens when the rank of the Jacobian matrix is not maximal, hence, the determinant is zero.

A robot is "close" to a singularity if the determinant is really small and close to zero (the Jacobian matrix is almost singular), in that case, to realize some small end effector velocities we may need really large joint velocities (not feasible for the motors).

It is required to analyze the mobility of a robot to avoid singularities and help during trajectory planning and motion control.

- when $m = n$, a configuration $\mathbf{q}$ si singular if $\det J(\mathbf{q}) = 0$
- when $m < n$, a configuration $\mathbf{q}$ si singular if $\det(J(\mathbf{q})J^T(\mathbf{q})) = 0$

Let's consider a 2R planar robot, tha Jacobian matrix is

$$J(\mathbf{q}) = \begin{pmatrix} -l_1 s_1 - l_2 s_{12} & -l_2 s_{12} \\ l_1 c_1 + l_2 c_{12} & l_2 c_{12} \end{pmatrix} \tag{5.79}$$

the determinant is

$$\det(J) = (-l_1 s_1 - l_2 s_{12})(l_2 c_{12}) - (-l_2 s_{12})(l_1 c_1 + l_2 c_{12}) = l_1 l_2 \sin q_2 \tag{5.80}$$

so we have a singularity configuration when $\sin q_2 = 0 \implies q_2 = 0$ or $q_2 = \pi$. These configurations identifies the points on the boundaries of the primary workspace (or at the center if $l_1 = l_2$). For a polar RRP robot, the direct kinematics is

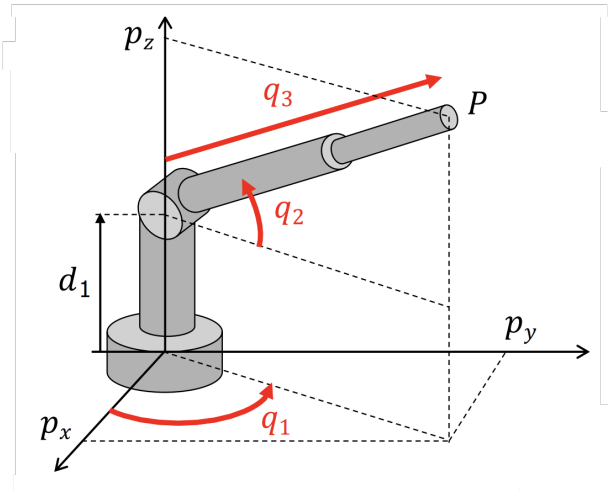
$$\begin{cases} p_x = q_3 \cos q_2 \cos q_1 \\ p_y = q_3 \cos q_2 \sin q_1 \\ p_z = d_1 + q_3 \sin q_2 \end{cases} \tag{5.81}$$

so the Jacobian matrix is

$$J(\mathbf{q}) = \begin{pmatrix} -q_3 s_1 c_2 & -q_3 c_1 s_2 & c_1 c_2 \\ q_3 c_1 c_2 & -q_3 s_1 s_2 & s_1 c_2 \\ 0 & q_3 c_2 & s_2 \end{pmatrix} \tag{5.82}$$

the determinant is

$$\det(J) = q_3^2 \cos q_2 \tag{5.83}$$



We have a singular configuration if

- $q_3 = 0$, in that case, the end effector is along the z axis with the prismatic joint fully retracted, in that case the Jacobian matrix becomes:

$$J(\mathbf{q}) = \begin{pmatrix} 0 & 0 & c_1 c_2 \\ 0 & 0 & s_1 c_2 \\ 0 & 0 & s_2 \end{pmatrix} \tag{5.84}$$

  the rank is 1 and the and effector can reach (instantaneously) velocities along only one direction (depending on q_1 and q_2).

- $q_2 = \pm\frac{\pi}{2}$, this is a "better" singularity than the previous one since the rank of J becomes 2, the end effector is along the z axis. In that situation, all the EE velocities that can be realized lies on a plane in the cartesian space that is parallel to the z axis and contains the origin. The Jacobian matrix becomes

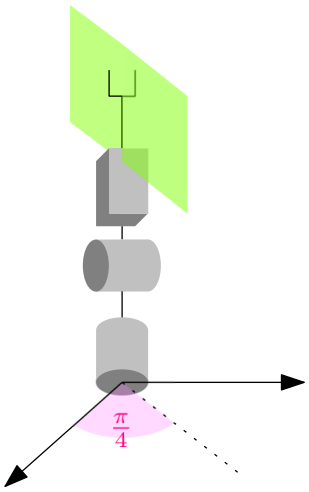
$$J(\mathbf{q}) = \begin{pmatrix} 0 & -q_3 c_1 s_2 & 0 \\ 0 & -q_3 s_1 s_2 & 0 \\ 0 & 0 & s_2 \end{pmatrix} = \begin{cases} \begin{pmatrix} 0 & -q_3 c_1 & 0 \\ 0 & -q_3 s_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ \begin{pmatrix} 0 & q_3 c_1 & 0 \\ 0 & q_3 s_1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \end{cases} \tag{5.85}$$

if $\mathbf{q} = (\frac{\pi}{4}, \frac{\pi}{2}, 1)$, the Jacobian matrix is

$$J(\mathbf{q}) = \begin{pmatrix} 0 & -0.7071 & 0 \\ 0 & -0.7071 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The velocities that can be realized are

$$\begin{pmatrix} 0 & -0.7071 & 0 \\ 0 & -0.7071 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{pmatrix} = \begin{pmatrix} -0.7071\dot{q}_2 \\ -0.7071\dot{q}_2 \\ \dot{q}_3 \end{pmatrix} \tag{5.86}$$



the green plane on the image shows the realizable velocities.

5.2 Inverse Differential Kinematics

We want to realize a desired end effector velocity $\mathbf{v}$, so, we need the corresponding joint's velocities $\dot{\mathbf{q}}$, if the Jacobian J is squared and non singular:

$$\dot{\mathbf{q}} = J^{-1}\mathbf{v} \tag{5.87}$$

this is a straightforward inversion method, but it has problems when

- we are near a singularity (the resulting $\dot{\mathbf{q}}$ will have a big norm).
- we are in the case $n > m$, so we can't invert a non square matrix.

It is possible to consider an incremental solution that solves the IDK problem *online*. Let's consider the direct kinematics function

$$\mathbf{r} = f_r(\mathbf{q}) \tag{5.88}$$

a small increment $d\mathbf{r}$ in function of a small increment $d\mathbf{q}$ is

$$d\mathbf{r} = J_r(\mathbf{q})d\mathbf{q} \tag{5.89}$$

if we are in the position $\mathbf{r}$, we want to perform a small displacement to reach $\hat{\mathbf{r}}$:

$$\mathbf{r} \longrightarrow \hat{\mathbf{r}} = \mathbf{r} + d\mathbf{r} \tag{5.90}$$

so we find the next configuration $\hat{\mathbf{q}}$ by solving

$$\hat{\mathbf{q}} = f_r^{-1}(\mathbf{r} + d\mathbf{r}) \tag{5.91}$$

so the inverse differential kinematics can be solved online by solving the inverse kinematics for $\mathbf{r} + d\mathbf{r}$, possibly with a numerical methods where we use the previous configuration $\mathbf{q}$ as the initial guess, in this case, the Newton method should converge in a few iterations. Clearly, if J is square and invertible, the increment $d\mathbf{q}$ can be found easily:

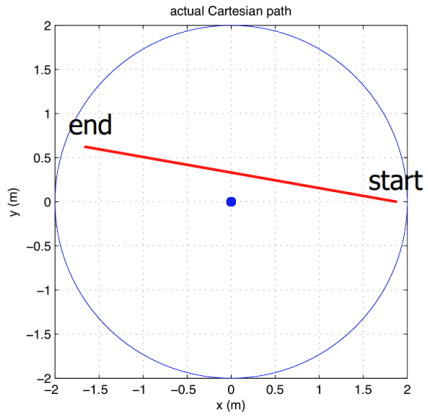
$$d\mathbf{q} = J^{-1}d\mathbf{r} \tag{5.92}$$

and the next configuration will be

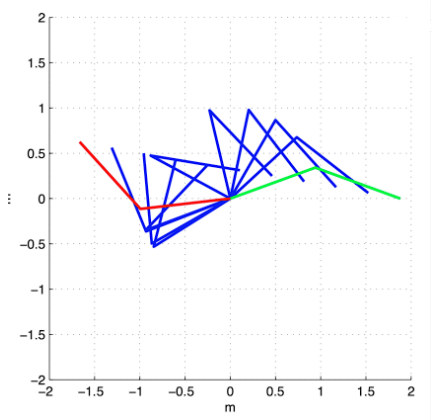
$$\hat{\mathbf{q}} = \mathbf{q} + d\mathbf{q} \tag{5.93}$$

When we compute the desired joint velocity, problems arises near the singularities, the required small displacements $d\mathbf{r}$ may lead to sudden increase of the velocity of one or more joint.

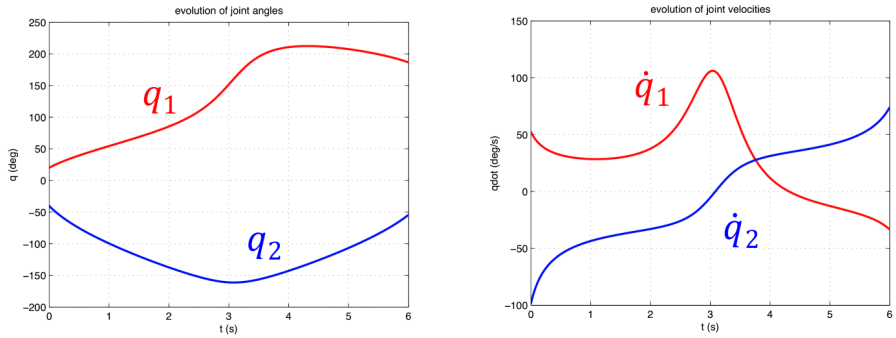
The online method works well when we are far from singularities, clearly, if we want to go from point A to point B along a straight line, we can perform the online method by imposing a constant velocity $\mathbf{v}$, due to the numerical integration error, the final position will be near B, but not exactly there. Let's consider an example, we have a 2R planar robot that have to follow the path shown in the picture:



This is a path (from left to the right) that have an angle of 170° respect to the x-axis. The desired velocity $\mathbf{v}$ will be constant and will have a norm of $\|\mathbf{v}\| = 0.6\frac{m}{s}$, this velocity will be executed for $T = 6$ seconds. The stroboscopic view of the manipulator from the starting position to the end is shown in the following picture.

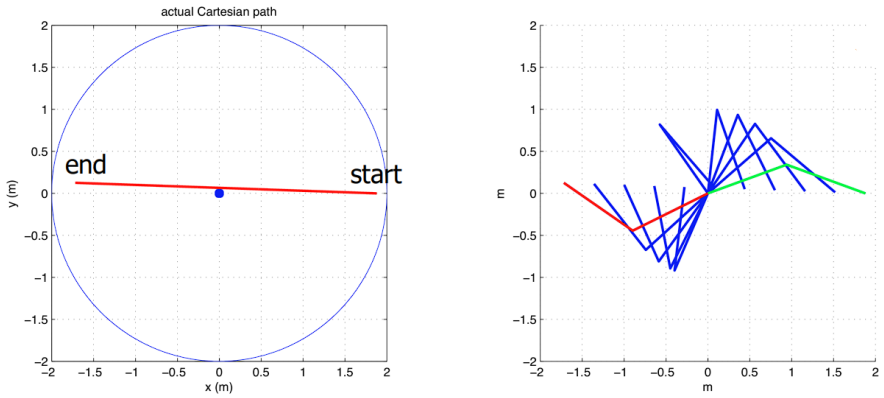


The following plots shows the configuration and the joint's velocities over time:



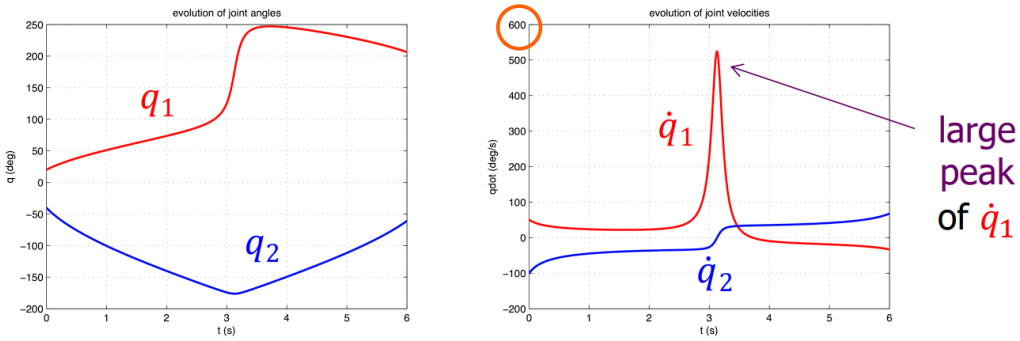
We can observe a small peak in the joint's velocities at $T = 3$, since at that time step the end effector is near (but still far enough) to the origin. At the end, the norm between the end effector position and the initial desired point is around $0.65 \cdot 10^{-10}$, this small error is due to the numerical integration.

We consider now a more "degenerate" case, where the line pass near the origin:



This is a path (from left to the right) that have an angle of 178° respect to the x-axis. The desired velocity $\mathbf{v}$ will be constant and will have a norm of $\|\mathbf{v}\| = 0.6\frac{m}{s}$, this velocity will be executed for $T = 6$ seconds.

In that case, around $T = 3.2$ seconds, the end effector will be really close to the origin (singular configuration), and we observe a peak for the velocity of the first joint $\dot{q}_1$, that arise in a small time window from 100 degree per second to 500 degree per second. The following plots shows the configuration and the joint's velocities over time:



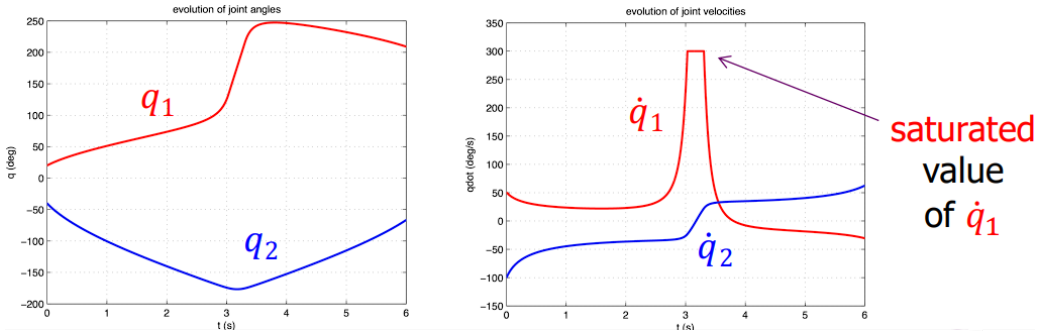
At the end, the distance between the end effector position and the initial desired point is around $1.8 \cdot 10^{-9}$, this small error (like the previous case) is due to the numerical integration.

In this simulation, we are assuming that a joint velocity of 500 degree per second is feasible for the motors, let's add in the simulation a saturation limit for the joint's velocities, in particular, we impose that

$$|q_i| \leq \frac{300^\circ}{\text{second}} \tag{5.94}$$

In that case, when the required velocity (given from the inversion) will be higher than $\frac{300^\circ}{\text{second}}$, the motor will actuate the joint's without exceeding that value.

The following plots shows the configuration and the joint's velocities over time:



For a small time interval around $T = 3$s, the velocity of q_1 remain fixed at 300 degrees per second, such a saturation problem ultimately leads to a large error in the final position of the end effector (of about 6 centimeters) upon completion of the execution.

5.2.1 Damped Least Squares Method

The following method is a robust live algorithm that is used to solve the inverse differential kinematics problem. Given a desired end effector velocity $\mathbf{v}$, we want to find the joint's velocity $\dot{\mathbf{q}}$ that

- have a small norm
- realize (as close as possible) the EE desired velocity

given that, we set up an unconstrained optimization problem:

$$\dot{\mathbf{q}}^* = \min_{\dot{\mathbf{q}}} \frac{\lambda}{2}\|\dot{\mathbf{q}}\|^2 + \frac{1}{2}\|J\dot{\mathbf{q}} - \mathbf{v}\|^2 \tag{5.95}$$

$\lambda \geq 0$ is a real number called **damping factor** that weights the problem (if λ is big, a small norm solution will be more pursued than a solution that realize the desired velocity). This is a quadratic problem and can be solved analytically by setting the gradient to zero. Let's denote the objective function $H(\dot{\mathbf{q}})$.

$$\nabla_{\dot{\mathbf{q}}} H = \left(\frac{\partial H}{\partial \dot{q}_i}\right) = \lambda\dot{\mathbf{q}} + J^T(J\dot{\mathbf{q}} - \mathbf{v}) \tag{5.96}$$

we set

$$\nabla_{\dot{\mathbf{q}}} H = \mathbf{0} \implies \tag{5.97}$$

$$\lambda\dot{\mathbf{q}} + J^T(J\dot{\mathbf{q}} - \mathbf{v}) = \mathbf{0} \implies \tag{5.98}$$

$$(\lambda I_n + J^T J)\dot{\mathbf{q}} = J^T\mathbf{v} \implies \tag{5.99}$$

$$\dot{\mathbf{q}} = (\lambda I_n + J^T J)^{-1} J^T \mathbf{v} \tag{5.100}$$

where I_n denotes the $n \times n$ identity matrix. It can be proved that

$$(\lambda I_n + J^T J)^{-1} J^T = J^T(\lambda I_n + JJ^T)^{-1} \tag{5.101}$$

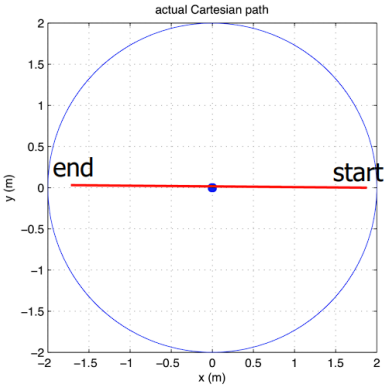
So, the solution of the DLS problem is

$$(\lambda I_n + J^T J)^{-1} J^T \mathbf{v} = J_{DLS}\mathbf{v} \tag{5.102}$$

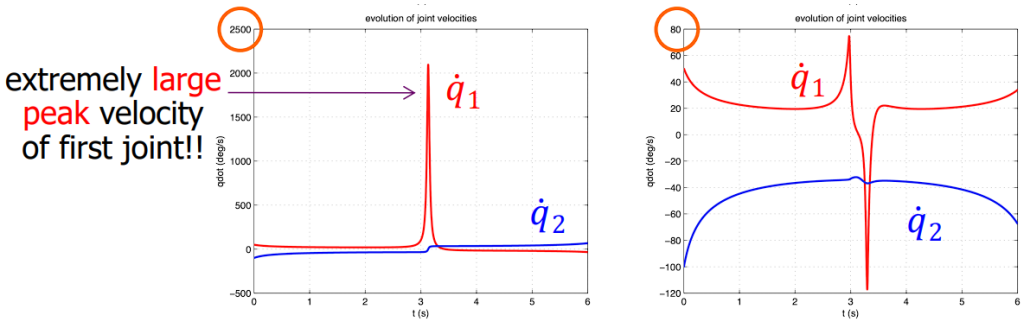
we denote J_{DLS} the matrix that maps the desired end effector velocity to the solution joint's velocities

$$J_{DLS} = (\lambda I_n + J^T J)^{-1} J^T. \tag{5.103}$$

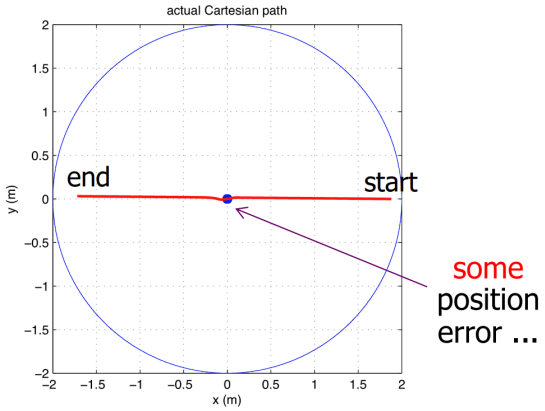
The damping factor λ can be chosen non zero only when we are near singularities. To prove that this method results in better performance, we considered a path that passes really close to the origin:



The plots of the joint's velocities over time:



in the left, we have the classical inverse iterative method, in the right the DLS method, that produces joint's velocities always limited : $|\dot{q}_i| < 120$ degrees per second. The path performed by the method is the following:



We have a cartesian error of about 25 millimeters near the singularity, that are recovered by a feedback action (see Chapter 7).

5.2.2 Pseudo-inverse method

The pseudo inverse method consider an optimization problem similar to the DLS. We consider a constrained optimization problem:

$$\dot{\mathbf{q}}^* = \min_{\dot{\mathbf{q}}} \frac{1}{2}\|\dot{\mathbf{q}}\|^2 \tag{5.104}$$

$$\text{such that } J\dot{\mathbf{q}} = \mathbf{v} \tag{5.105}$$

this problem is equivalent to the following:

$$\mathbf{q} \dot{\in} S^* = \min_{\dot{\mathbf{q}}} \frac{1}{2}\|\dot{\mathbf{q}}\|^2 \tag{5.106}$$

$$S = \{\dot{\mathbf{q}} \in \mathbb{R}^n \ : \ \|J\dot{\mathbf{q}} - \mathbf{v}\| \text{ is minimum } \} \tag{5.107}$$

This problem admits an analytical solution, we define the Lagrangian function:

$$L = \frac{1}{2}\dot{\mathbf{q}}^T\dot{\mathbf{q}} + \boldsymbol{\lambda}^T(J\dot{\mathbf{q}} + \mathbf{v}) \tag{5.108}$$

where $\boldsymbol{\lambda}$ are the Lagrangian multipliers. We can find a solution by setting the gradient (respect to $\dot{\mathbf{q}}$ and to $\boldsymbol{\lambda}$) to zero:

$$\begin{cases} \nabla_{\dot{\mathbf{q}}} L = \dot{\mathbf{q}} + J^T\boldsymbol{\lambda} = \mathbf{0} \\ \nabla_{\boldsymbol{\lambda}} L = J\dot{\mathbf{q}} - \mathbf{v} = \mathbf{0} \end{cases} \implies \begin{cases} \dot{\mathbf{q}} = -J^T\boldsymbol{\lambda} \\ -JJ^T\boldsymbol{\lambda} = \mathbf{v} \end{cases} \tag{5.109}$$

by solving for $\boldsymbol{\lambda}$;

$$\boldsymbol{\lambda} = -(JJ^T)^{-1}\mathbf{v} \tag{5.110}$$

since $\dot{\mathbf{q}} = -J^T\boldsymbol{\lambda}$, the solution for $\dot{\mathbf{q}}$ is

$$\dot{\mathbf{q}} = J^T(JJ^T)^{-1}\mathbf{v} = J^{\#}\mathbf{v} \tag{5.111}$$

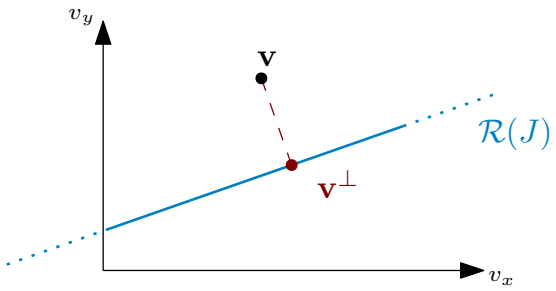
the matrix $J^{\#} = J^T(JJ^T)^{-1}$ is the **pseudo inverse** of the Jacobian matrix. The following properties are satisfied:

- if $\mathbf{v} \in \mathcal{R}(J)$, then the constrain is satisfied, and the solution $\dot{\mathbf{q}} = J^{\#}\mathbf{v}$ will realize $\mathbf{v}$ and minimize the norm $\|\dot{\mathbf{q}}\|$.
- else, if $\mathbf{v} \notin \mathcal{R}(J)$, the given solution will minimize the error $\|J\dot{\mathbf{q}} - \mathbf{v}\|$.

Note how the solution of the optimization problem is the velocity

$$J\dot{\mathbf{q}} = JJ^{\#}\mathbf{v} = \mathbf{v}^{\perp} \tag{5.112}$$

where $\mathbf{v}^{\perp}$ is the orthogonal projection of $\mathbf{v}$ on $\mathcal{R}(J)$.



The general definition of the pseudo inverse of a matrix J is the following.

Definition 10 *Given a matrix J of m rows and n columns, the pseudo inverse $J^{\#}$ is the matrix that satisfies the following relation:*

- $JJ^{\#}J = J$
- $(JJ^{\#})^T = JJ^{\#}$
- $J^{\#}JJ^{\#} = J^{\#}$
- $(J^{\#}J)^T = J^{\#}J$

the pseudo inverse $J^{\#}$ always exists and is unique.

If the rank of J is full, the pseudo inverse can be computed with simple algebraic forms:

- $\rho(J) = m = n \implies J^{\#} = J^{-1}$
- $\rho(J) = m < n \implies J^{\#} = J^T(JJ^T)^{-1}$
- $\rho(J) = n < m \implies J^{\#} = (J^TJ)^{-1}J^T$

In general, $J^{\#}$ can be computed numerically with the singular value decomposition of J.

5.2.3 Singular Value Decomposition

This section is a linear algebra recap about singular value decomposition for a matrix.

Definition 11 *Given a matrix A of m rows and n columns, there always exists a triplets of matrices U, Σ, V such that:*

- $A = U\Sigma V^T$
- $U \in Mat(n \times n)$
- $\Sigma \in Mat(n \times m)$
- $V \in Mat(m \times m)$

The matrices U, V are unitary matrices (the conjugate transpose is equal to the inverse) and are not uniques. Σ is unique and is a diagonal matrices for which the diagonal contains all the singular values of A.

$$\Sigma = \begin{pmatrix} \sigma_1 & 0 & 0 & \dots & 0 & \dots & 0 & 0 \\ 0 & \sigma_2 & 0 & \dots & 0 & \dots & 0 & 0 \\ 0 & 0 & \sigma_3 & \dots & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \sigma_r & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & \dots & 0 & 0 \end{pmatrix} \tag{5.113}$$

where

$$\sigma_1 \geq \sigma_2 \cdots \geq \sigma_r \geq 0 \tag{5.114}$$

the rank of A is r. We can express the pseudo inverse of A in terms of U, V, Σ as follows:

$$A^{\#} = V\Sigma^{\#}U^T \tag{5.115}$$

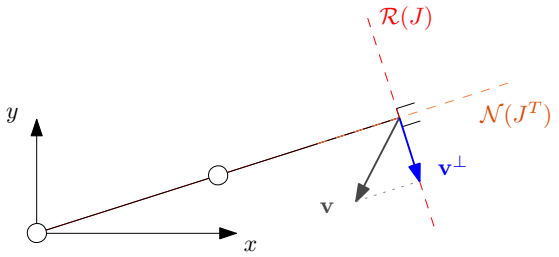
so, we just have to compute the pseudo inverse of Σ, that is a $n \times m$ matrix, computed as follows

- we consider Σ
- we substitute each value σ_i with the inverse $\frac{1}{\sigma_i}$

$$\Sigma^{\#} = \begin{pmatrix} \frac{1}{\sigma_1} & 0 & 0 & \dots & 0 & \dots & 0 & 0 \\ 0 & \frac{1}{\sigma_2} & 0 & \dots & 0 & \dots & 0 & 0 \\ 0 & 0 & \frac{1}{\sigma_3} & \dots & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \frac{1}{\sigma_r} & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & \dots & 0 & 0 \end{pmatrix} \tag{5.116}$$

In many programming languages and libraries, algorithms that computes the singular value decomposition for a matrix are provided.

Let's consider an example with the planar 2R robot, with $l_1 = l_2 = 1$, in a singular configuration with $q_2 = 0$.



The Jacobian and the pseudo inverse are:

$$J = \begin{pmatrix} -2s_1 & -s_1 \\ 2c_1 & c_1 \end{pmatrix} \qquad J^{\#} = \frac{1}{5}\begin{pmatrix} -2s_1 & 2c_1 \\ -s_1 & c_1 \end{pmatrix} \tag{5.117}$$

if $q_2 = 0$, then $\rho(J) = 1$. The vector $\dot{\mathbf{q}} = J^{\#}\mathbf{v}$ is the minimum norm joint velocity that realize exactly $\mathbf{v}^\perp$.

- at $q_1 = \frac{\pi}{6}$, for $\mathbf{v} = (-0.5, 0)$, we get

$$J^{\#}\mathbf{v} = \begin{pmatrix} 0.1 \\ 0.05 \end{pmatrix} \implies \mathbf{v}^\perp = JJ^{\#}\mathbf{v} = \begin{pmatrix} -1/8 \\ \sqrt{3}/8 \end{pmatrix} \tag{5.118}$$

- if $q_1 = \frac{\pi}{2}$, $\mathbf{v} \in \mathcal{R}(J)$, so with $\dot{\mathbf{q}} = J^{\#}\mathbf{v}$ we get

$$JJ^{\#}\mathbf{v} = \mathbf{v}^\perp = \mathbf{v} \tag{5.119}$$

In general, any solution for the inverse differential kinematics problem

$$J\dot{\mathbf{q}} = \mathbf{v} \tag{5.120}$$

can be written in the following form:

$$\dot{\mathbf{q}} = J^{\#}\mathbf{v} + (I - J^{\#}J)\boldsymbol{\xi} \tag{5.121}$$

where $(I - J^{\#}J)$ is the orthogonal projection on $\mathcal{N}(J)$ and $\boldsymbol{\xi}$ is a joint velocity vector. This solution is not the minimum norm one, but is the optimal solution for a slightly modified constrained optimization problem:

$$\min_{\dot{\mathbf{q}}} \frac{1}{2}\|\dot{\mathbf{q}} - \boldsymbol{\xi}\|^2 \tag{5.122}$$

$$\text{such that } J\dot{\mathbf{q}} = \mathbf{v} \tag{5.123}$$

with that solution, the actual task velocity obtained is

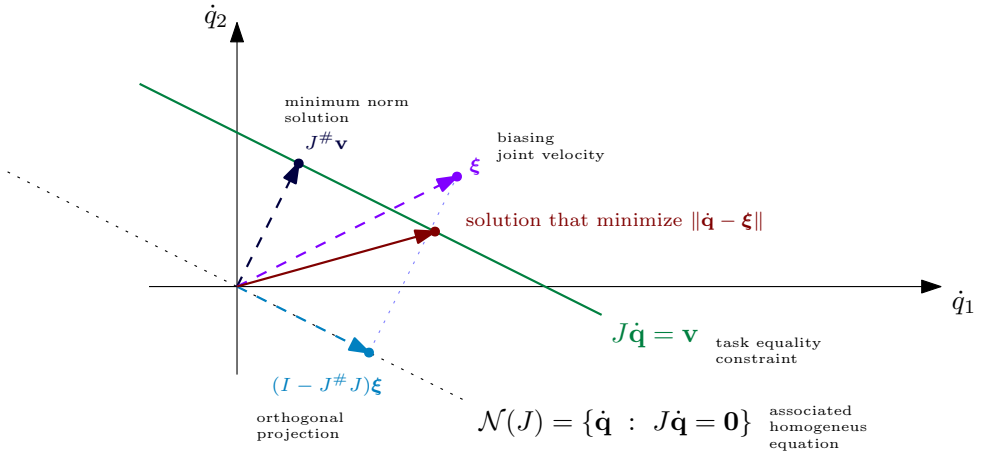
$$J\dot{\mathbf{q}} = J(J^{\#}\mathbf{v} + (I - J^{\#}J)\boldsymbol{\xi}) = \tag{5.124}$$

$$JJ^{\#}\mathbf{v} + J(I - J^{\#}J)\boldsymbol{\xi} = JJ^{\#}\mathbf{v} \tag{5.125}$$

if $\mathbf{v} \in \mathcal{R}(J)$, it exists $\mathbf{w}$ such that $\mathbf{v} = J\mathbf{w}$:

$$JJ^{\#}\mathbf{v} = JJ^{\#}(J\mathbf{w}) = J\mathbf{w} = \mathbf{v} \tag{5.126}$$

In the equation (5.121) the solution si given by the sum of the orthogonal projection of $\boldsymbol{\xi}$ on $\mathcal{N}(J)$ and the minimum norm solution $J^{\#}\mathbf{v}$:



5.3 Velocity Manipulability

This section is about the manipulability of the end effector:

how easy is to move an end effector towards a desired velocity by imposing joint's velocity?

In details, we say that is "hard" to move the task vector in a specific direction if the required joint velocity is high, in particular, we want to measure how close to a singularity we are.

Definition 12 *An ellipsoid in* $\mathbb{R}^n$ *is the set of points* $\mathbf{p}$ *that satisfies*

$$\mathbf{p}^T A^{-1}\mathbf{p} \leq 1 \tag{5.127}$$

where A *is symmetric, and positive definite.*

Given the joint space Q, we consider the unitary sphere:

$$\{\dot{\mathbf{q}} \ : \ \dot{\mathbf{q}}^T\dot{\mathbf{q}} = 1\} \tag{5.128}$$

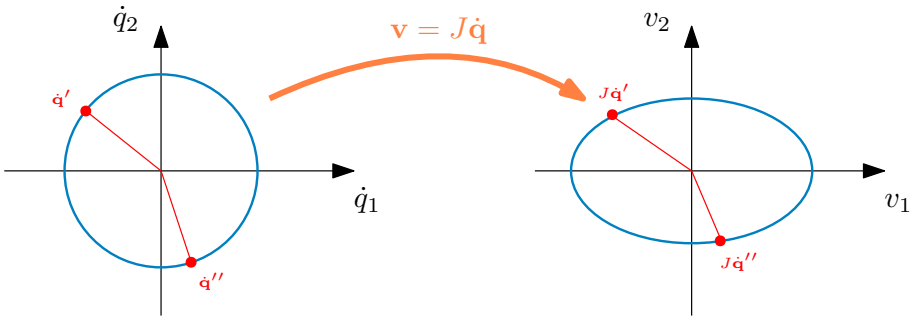
we know that, to obtain a velocity $\mathbf{v}$, we have $\dot{\mathbf{q}} = J^{\#}\mathbf{v}$, so

$$\dot{\mathbf{q}}^T\dot{\mathbf{q}} = 1 \implies (J^{\#}\mathbf{v})^T(J^{\#}\mathbf{v}) = 1 \implies \mathbf{v}^T {J^{\#}}^T J^{\#}\mathbf{v} = 1 \tag{5.129}$$

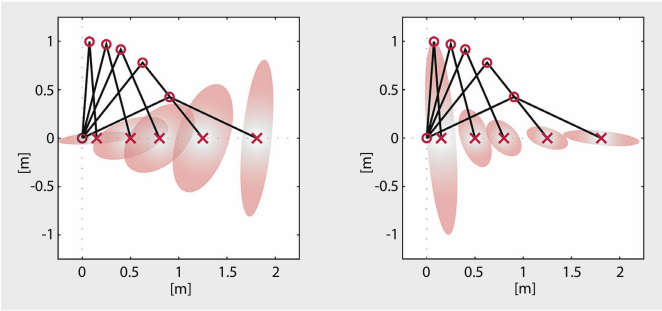
if $\rho(J) = m$ (we are not in a singular configuration), ${J^{\#}}^T J^{\#} = (JJ^T)^{-1}$:

$$\mathbf{v}^T(JJ^T)^{-1}\mathbf{v} = 1 \tag{5.130}$$

We note how all the unitary norm joint velocity $\dot{\mathbf{q}}$ identifies the surface of an ellipsoid $\mathbf{v}^T(JJ^T)^{-1}\mathbf{v} = 1$ in the task space, the ellipsoid is described by the matrix JJ^T. Geometrically, the unit sphere in the joint space is stretched in the ellipsoid in the task space.



This ellipsoid can be drawn in the task space attached to the end effector, this will be easily steerable along the ellipsoid's major axes, and will be harder to manipulate along the minor axes.

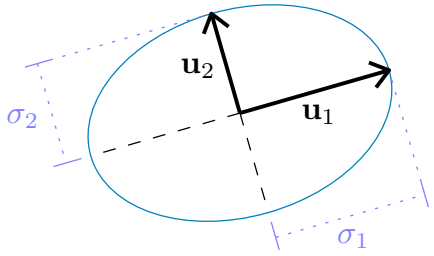


The fact is that, the more the ellipsoid is flattened, the more it is hard to manipulate the end effector to the minor axes, and the closer we are to a singular configuration. In fact, in a singular configuration, the ellipsoid is totally flatten, and becomes a plane in the space (the end effector lose manipulability).

We will describe more formally this relation between the ellipsoid and the singular configurations. The i-th singular value of J is the root of the i-th eigen value of JJ^T

$$\sigma_i(J) = \sqrt{\lambda_i(JJ^T)} \tag{5.131}$$

this is the length of the principal axes, and the direction are the one related to the i-th eigen vector of JJ^T.



We define the **manipulability measure** the quantity

$$w = \sqrt{\det(JJ^T)} = \prod_{i=1}^{m} \sigma_i \geq 0 \tag{5.132}$$

the quantity w is proportional to the volume of the ellipsoid:

- a small w means that we are near a singular case (the ellipsoid is flat)

With the singular values decomposition:

$$JJ^T = U\Sigma V^T V \Sigma^T U^T = U\Sigma\Sigma^T U^T \tag{5.133}$$

the matrix $\Sigma\Sigma^T$ looks like that:

$$\Sigma\Sigma^T = \begin{pmatrix} \sigma_1^2 & & & & \\ & \ddots & & & \\ & & \sigma_r^2 & & \\ & & & 0 & \\ & & & & \ddots \\ & & & & & 0 \end{pmatrix} \tag{5.134}$$

Let's consider a column vector $\mathbf{u}_i$ of U:

$$JJ^T\mathbf{u}_i = \tag{5.135}$$

$$U\Sigma\Sigma^T U^T\mathbf{u}_i = \tag{5.136}$$

$$U\Sigma\Sigma^T \begin{pmatrix} \mathbf{u}_1^T\mathbf{u}_i \\ \vdots \\ \mathbf{u}_n^T\mathbf{u}_i \end{pmatrix} \tag{5.137}$$

but U is a unitary matrix, so the dot product of each distinct column vector is zero, except for the dot product of a column with himself that is 1.

$$U\Sigma\Sigma^T \begin{pmatrix} \mathbf{u}_1^T\mathbf{u}_i \\ \vdots \\ \mathbf{u}_n^T\mathbf{u}_i \end{pmatrix} = U\Sigma\Sigma^T\mathbf{e}_i \tag{5.138}$$

where $\mathbf{e}_i$ is the all zero vector except for the i-th coordinate that is 1. Since $\Sigma\Sigma^T$ is a diagonal matrix:

$$U\Sigma\Sigma^T\mathbf{e}_i = U\sigma_i^2\mathbf{e}_i = \sigma_i^2 U\mathbf{e}_i = \sigma_i^2\mathbf{u}_i \tag{5.139}$$

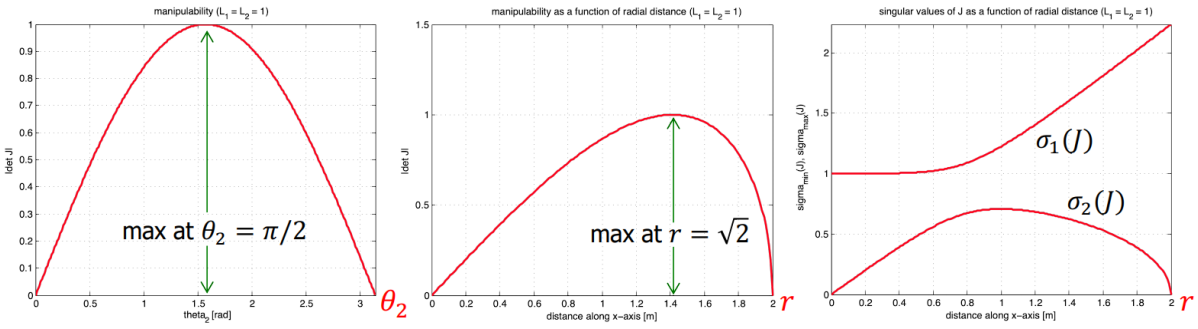
In the end we have that:

$$JJ^T\mathbf{u}_i = \sigma_i^2\mathbf{u}_i \tag{5.140}$$

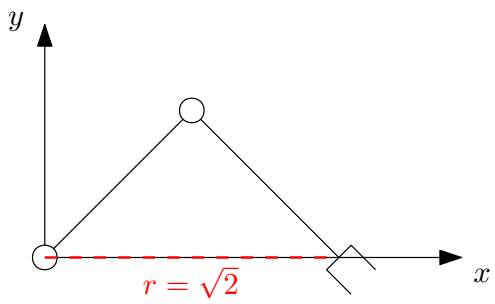
so σ_i is the eigen value of JJ^T associated with the eigen vector $\mathbf{u}_i$.

Let's consider the planar 2R robot, with $l_1 = l_2 = 1$, the following plots shows (in order)

1. the manipulability in function of q_2
2. the manipulability in function of the x axis distance of the end effector
3. the singular values of J in function of the x axis distance of the end effector

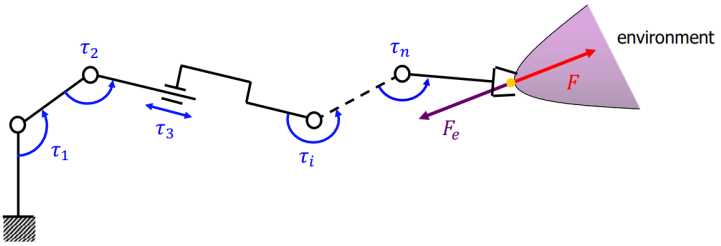


the maximum value is obtained at $r = \sqrt{2}$, this is the best configuration for manipulability (where the ellipsoid have the maximal volume).



5.4 Static Forces Transformations

Let's consider a robot manipulator, let $\boldsymbol{\tau} = (\tau_i)$ to be the vector of the torques applied to each joint.



We assume that the end effector is in contact with a rigid body of the environment, and is applying to it a force $\mathbf{F}$. For the principle of action and reaction, on the end effector is applied a force $\mathbf{F}_e = -\mathbf{F}$, so we are in a static condition where the robot doesn't move. With $\mathbf{F}$ we denote the generalized forces vector, that contains linear and angular components, since on the end effector might be applied a linear force and/or a torque, we call **wrench** this generalized forces vector $\mathbf{F} \in \mathbb{R}^6$.

Our goal is to describe the relation between the wrench $\mathbf{F}$ and the torque vector $\boldsymbol{\tau}$. The robotic manipulator is a constrained system of rigid bodies, in particular, the bodies (the links) can move in the space with holonomic constraints, let's say that we have n rigid bodies, each described by a position vector $\mathbf{p}_i$ (their frame origin). The constraints are relative to the position, so there exists k constraints $f_1 \dots f_k$ such that

$$f_1(\mathbf{p}_1 \dots \mathbf{p}_n) = 0 \tag{5.141}$$

$$\vdots \tag{5.142}$$

$$f_k(\mathbf{p}_1 \dots \mathbf{p}_n) = 0 \tag{5.143}$$

even if there are n bodies (described by three coordinates each), the system doesn't have $3n$ degrees of freedom, we already know that the system have n degrees of freedom (one for each joint). The choice of the DH coordinates $\mathbf{q} = q_1 \dots q_n$ makes the system free of constraints. Let's remind that these are a set of Lagrangian coordinates for our robotic system with n joints.

A **virtual displacement** for the system is a displacement $\delta\mathbf{q}$ that is compatible with the constraints, since with the choice of the DH variables we don't have any constraint, any infinitesimal displacement $d\mathbf{q}$ is a virtual displacement.

Let's consider an infinitesimal (a virtual) displacement $d\mathbf{q}$ of the joint's variables. The corresponding linear and angular displacement of the end effector will be

$$\begin{pmatrix} d\mathbf{p} \\ \boldsymbol{\omega} dt \end{pmatrix} = J(\mathbf{q}) d\mathbf{q} \tag{5.144}$$

the virtual work dW_e (equivalent to the infinitesimal work) of the end effector is given by the dot product of the wrench applied on it and the end effector "velocity" (in this case, the infinitesimal displacement):

$$dW_e = -F^T \begin{pmatrix} d\mathbf{p} \\ \boldsymbol{\omega} dt \end{pmatrix} = -F^T J(\mathbf{q}) d\mathbf{q} \tag{5.145}$$

the virtual work dW_q on the joints is given by the dot product of the torque vector and the infinitesimal displacement $d\mathbf{q}$:

$$dW_q = \boldsymbol{\tau}^T d\mathbf{q} \tag{5.146}$$

The total virtual work of the system is given by $dW = dW_e + dW_q$. For the principle of the **virtual work**, since the system is in a static equilibrium, the work associated to this infinitesimal variation respect to the equilibrium configuration is null:

$$dW = dW_e + dW_q = 0 \tag{5.147}$$

hence

$$\boldsymbol{\tau}^T d\mathbf{q} - F^T \begin{pmatrix} d\mathbf{p} \\ \boldsymbol{\omega} dt \end{pmatrix} = 0 \implies \tag{5.148}$$

$$\boldsymbol{\tau}^T d\mathbf{q} - F^T J(\mathbf{q}) d\mathbf{q} = 0 \implies \tag{5.149}$$

$$\boldsymbol{\tau}^T d\mathbf{q} = F^T J(\mathbf{q}) d\mathbf{q} \implies \tag{5.150}$$

$$\boldsymbol{\tau} = J^T(\mathbf{q})\mathbf{F} \tag{5.151}$$

Therefore we derived the relation between $\boldsymbol{\tau}$ and $\mathbf{F}$, and is given by the transpose of the Jacobian matrix. Let's keep in mind that in our case we considered equivalent virtual displacements and infinitesimal displacements due to the absence of constraints for the Lagrangian variables $\mathbf{q}$. Since $\rho(J) = \rho(J^T)$, a singular configuration for the velocity map is also singular for the wrench map. As already discussed, the following relations holds:

$$\mathcal{R}(J) = \{\mathbf{v} \in \mathbb{R}^m : \exists \dot{\mathbf{q}} \in \mathbb{R}^n, J\dot{\mathbf{q}} = \mathbf{v}\}$$
$$\mathcal{N}(J^T) = \left\{\mathbf{F} \in \mathbb{R}^m : J^T\mathbf{F} = \mathbf{0}\right\}$$
$$\mathcal{R}(J) \oplus \mathcal{N}(J^T) = \mathbb{R}^m$$

$$\mathcal{R}(J^T) = \left\{\boldsymbol{\tau} \in \mathbb{R}^n : \exists \mathbf{F} \in \mathbb{R}^m, J^T\mathbf{F} = \boldsymbol{\tau}\right\}$$
$$\mathcal{N}(J) = \{\dot{\mathbf{q}} \in \mathbb{R}^n : J\dot{\mathbf{q}} = \mathbf{0}\}$$
$$\mathcal{R}(J^T) \oplus \mathcal{N}(J) = \mathbb{R}^n$$

5.4.1 Force Manipulability

The discussion about the velocity ellipsoid for a manipulator can be done the forces too. We consider all the unitary norm torques $\boldsymbol{\tau}$:

$$\boldsymbol{\tau}^T\boldsymbol{\tau} = 1$$

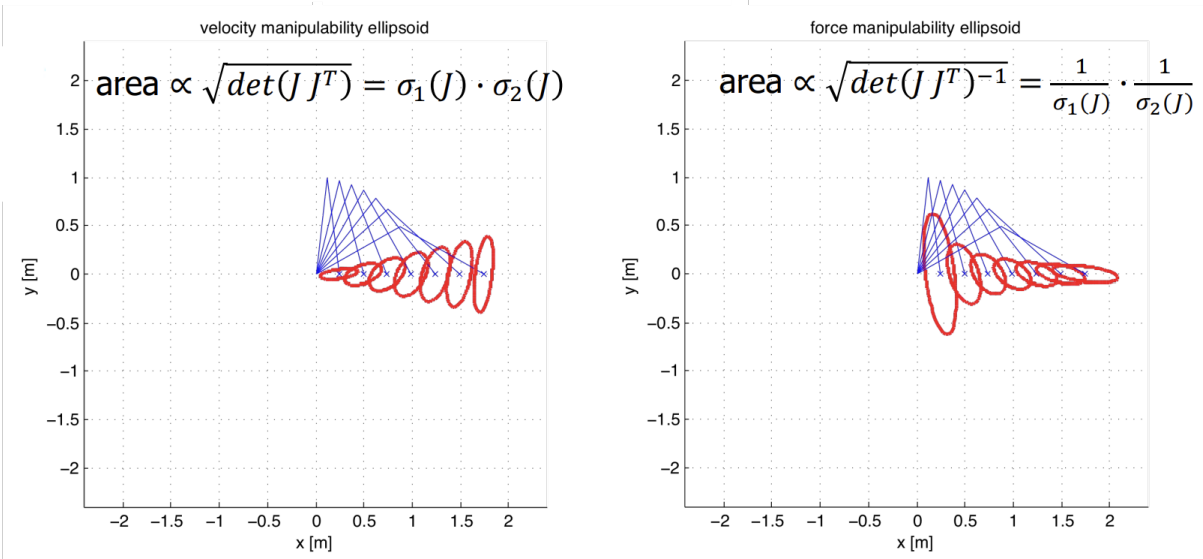
since $\boldsymbol{\tau} = J^T\mathbf{F}$, this identifies an ellipsoid in the force space:

$$\mathbf{F}^T J J^T \mathbf{F} = 1$$

the area of the ellipsoid is proportional to

$$\sqrt{\det(JJ^T)^{-1}} \tag{5.152}$$

so this one can be used as a measure of singularity (such as the velocity case). The two ellipsoids (velocity and force) are aligned along the same axes, but they are oriented in the opposite way in terms of elongation.



The interpretation of the force ellipsoid is the following:

> Along the direction of the longest axes of the ellipsoid, it is easier to impose a wrench $\mathbf{F}$ by applying small torques $\boldsymbol{\tau}$. Along the shortest one, even to impose small wrenches, it may be required a huge torque.

CHAPTER

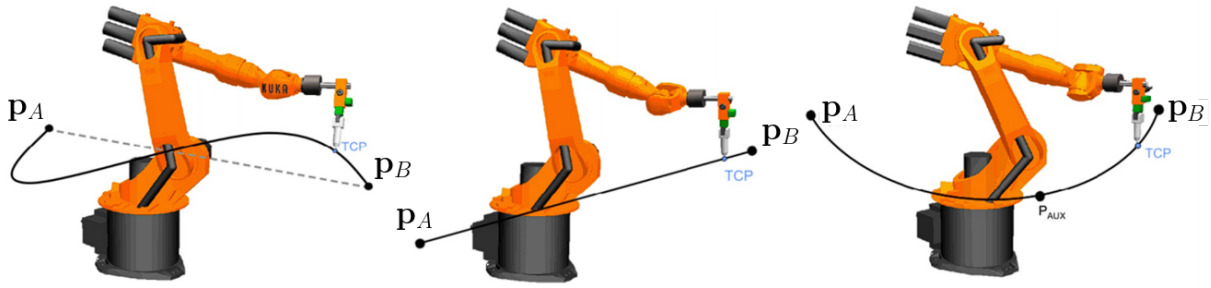
6

TRAJECTORY PLANNING

We described how to impose task positions by controlling the joint's angle value, we now describe how to make the end effector to follow a general trajectory in the cartesian space, by changing it's pose over time. We recall that

- Motion planning: find the path (without collisions) that the robot have to follow (find the correct trajectory)
- Trajectory planning: control the joint's such that the robot end effector follow a given trajectory.

In this section *we do not cover* motion planning, only the trajectory planning. The aim of a trajectory planner, is to, given a sequence of poses over time, to produce the profiles over time of the joint's angle for the robot controller. Let's consider a simple problem: We have two points in the space: $\mathbf{p}_A$ and $\mathbf{p}_B$ (expressed in the robot reference frame), and we want the robot end effector to pass from $\mathbf{p}_A$ to $\mathbf{p}_B$.



There are many ways, we can go from the start to the end trough a spline, a straight line, or a circular line. To make the end effector to go from $\mathbf{p}_A$ to $\mathbf{p}_B$ trough a straight line, we consider the following trajectory:

$$\mathbf{p}(s) = \mathbf{p}_A + s(\mathbf{p}_B - \mathbf{p}_A) \tag{6.1}$$

where $s \in [0, 1]$ is the parameter that define the trajectory. We can assign a timing law by making s a scalar function in the variable $t \in [0, T]$

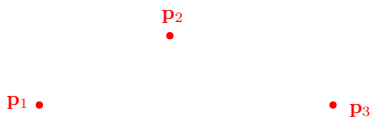
$$s = s(t) \in [0, 1] \tag{6.2}$$
$$s(0) = 0 \implies \mathbf{p} = \mathbf{p}_A \tag{6.3}$$
$$s(T) = 1 \implies \mathbf{p} = \mathbf{p}_B \tag{6.4}$$

Given that, we define a **trajectory** as the combination of a *geometric path* parametrized by $s \in [0, 1]$, and a *timing law* parametrized by $t \in [0, T]$ that describe the time evolution. There are two ways to

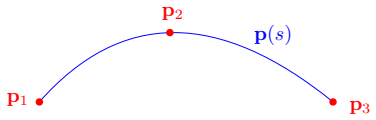
make the robot follow a path, we can plan the trajectory in the *joint space* or in the *cartesian space* (the difference will be clarified later).

In general, we can resume the trajectory planning operative sequence in the following steps:

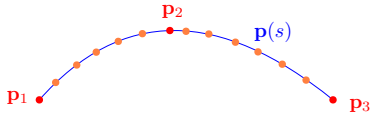
1. We get as an input a sequence of points (named **knots**) in the cartesian space.



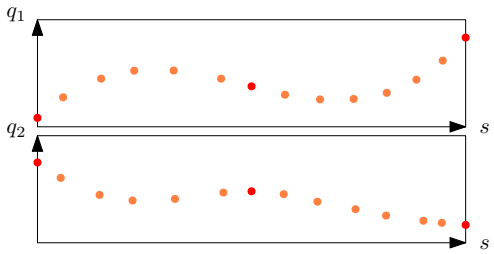
2. We consider an interpolation of the knots to define a trajectory $\mathbf{p}(s)$



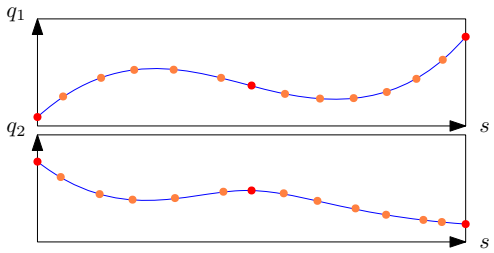
3. We sample $\mathbf{p}(s)$ to get a dense but finite sequence of knots on the trajectory



4. We compute the inverse kinematics for each sampled knots to get a sequence of points in the joint space



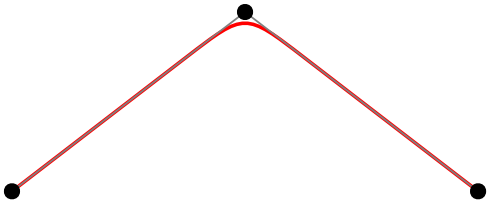
5. We interpolate these points to obtain a trajectory $\mathbf{q}(s)$ in the joint space.



We may skip from the step 2 directly to get the resulting trajectory $\mathbf{q}(s)$ in the joint space, but this analytic inversion is a complex process, usually unfeasible (this problem can be seen as the task to find the solution of a continuously infinite number of non linear systems of equations).

The timing law is usually chosen by consider some optimal criteria (minimum transfer time) to be satisfied under some constraints imposed by the actuator capabilities. Some trajectory are at priori unfeasible for the manipulator. A trajectory with a straight edge, may require a discontinuous jump for the velocities of the joint's, that is unfeasible for the motor, in fact, if a manipulator have to perform a path like that (such in a square trajectory), there are two options:

- the end effector must stops on the edge, and then continue
- the end effector will follow a rounded curve that approximate the edge (over-fly)



over-fly trajectory

Example: We want to define a trajectory for $q(s)$, $s \in [0,1]$ from $q(0) = q_i = 0$ to $q(1) = q_f = \pi$, with zero velocity in the start and in the end

$$\dot{q}(0) = \dot{q}(1) = 0 \tag{6.5}$$

to do so, we need a cubic path, for example:

$$q(s) = s + 3(\pi - 1)s^2 - 2(\pi - 1)s^3 \tag{6.6}$$

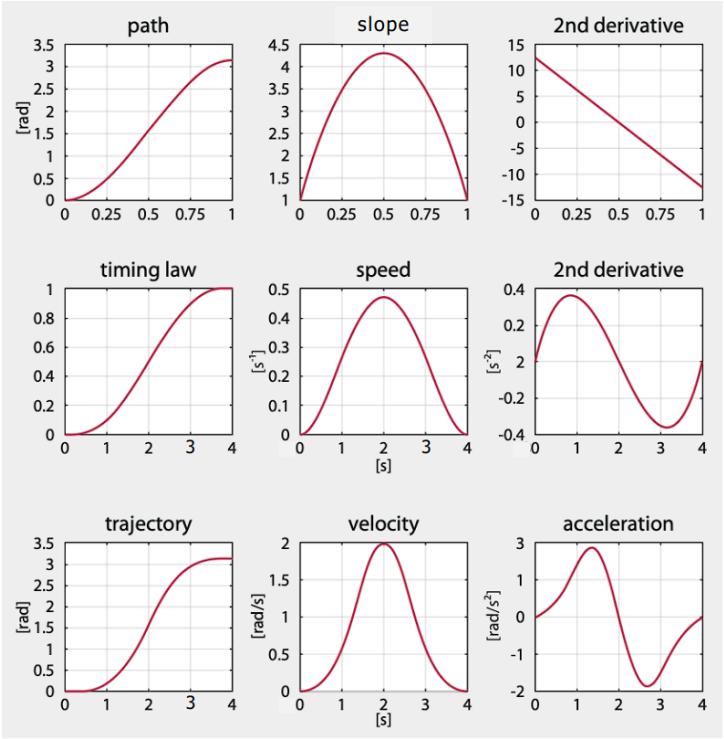
we consider a timing law from $t_i = 0$ sec to $t_f = 4$ sec ($T = t_f - t_i = 4$), with the conditions of zero velocity and zero acceleration in the start and in the end:

$$\dot{s}(t_i) = \ddot{s}(t_i) = \dot{s}(t_f) = \ddot{s}(t_f) = 0 \tag{6.7}$$

we need a quintic timing law:

$$s(t) = 10\tau^3 - 15\tau^4 + 6\tau^5, \quad \tau = \frac{t}{T} = \frac{t}{4} \tag{6.8}$$

in the end, the trajectory in function of time $q(t)$, will be a polynomial of degree 15.



When we define a trajectory, we have to take in account different things:

- We are defining the path in the cartesian space or in the joint space?
- Wat type of task we have to perform? It can be point to point (PTP), multi point (MP) with knots, or concatenated
- what geometric path we have to follow? It can be rectilinear, polynomial, harmonic, exponential, cycloid ecc...

- we have to define the timing law: bang-bang in acceleration, trapezoidal in velocity, polynomial ecc...
- the motion of all the joints have to be coordinated (starts and ends at the same instant) or they can move independently?

Planning a trajectory in the Cartesian space offers a more direct visualization of the generated path. However, if the task involves avoiding non-static obstacles, online planning is required. While planning in the joint space lets us bypass online kinematic inversion, it doesn't allow for obstacle avoidance and the resulting path might cross manipulator singularities.

The offline planning in the joint space is easier but not always possible, the online (re-)planning is needed when environment interaction occurs or when sensor-based motion is required.

6.1 Path Planning

Relevant characteristics for a geometric path are the following:

- predictability and accuracy of the path: We would like to have predictable motion outside the knots.
- flexibility: allow concatenation of primitive segments, over-fly of knots
- computational efficiency and memory space for MP tasks: store only the coefficients of polynomial functions
- smoothness: in the space and in time, we would like to have at least C^1 function for the timing law, possibly, we would like to have continuity up to the jerk.

Let's consider the general case where we have a generic parametrization for the geometric path $\mathbf{q}$:

$$\mathbf{q}(s) = \begin{pmatrix} q_1(s) \\ \vdots \\ q_n(s) \end{pmatrix}, \quad s \in [s_i, s_f] \tag{6.9}$$

We consider the path derivatives

$$\mathbf{q}'(s) = \frac{d\mathbf{q}}{ds} \quad \mathbf{q}''(s) = \frac{d^2\mathbf{q}}{ds^2} \tag{6.10}$$

these quantities are related to the **curvature** of the path, this is a scalar quantity κ defined for each s that describe how curved is the path in that point:

$$\kappa(s) = \frac{\|\mathbf{q}'(s) \times \mathbf{q}''(s)\|}{\|\mathbf{q}'(s)\|^3} \tag{6.11}$$

We say that a path satisfies the **regularity condition** if the curvature κ is bounded, hence, the first derivative of $\mathbf{q}$ is never zero:

$$\mathbf{q}'(s) \neq 0, \quad s \in [s_i, s_f] \tag{6.12}$$

How can we choose a parametrization s for the path? Given $\mathbf{q}$, we have a special parametrization, given by the **arc length** σ. The length of the path is

$$L = \int_{s_i}^{s_f} \|\mathbf{q}'(r)\| dr \tag{6.13}$$

the parametrization by the arc length is given by $\sigma = \sigma(s)$:

$$\sigma(s) = \int_{s_i}^{s} \|\mathbf{q}'(r)\| dr \tag{6.14}$$

this is independent with respect to the choice of s. By considering this parametrization:

$$\mathbf{q}(\sigma), \quad \sigma \in [\sigma(s_i), \sigma(s_f)] = [0, L] \tag{6.15}$$

with this special parametrization we have always a unit tangent vector to the path:

$$\|\mathbf{q}'(\sigma)\| = 1 \tag{6.16}$$

and the curvature is given by the norm of the second derivative:

$$\kappa(\sigma) = \|\mathbf{q}''(\sigma)\| \tag{6.17}$$

After we define a parametrization for $\mathbf{q}(s)$ and a timing law $s(t)$ we can consider the derivative of $\mathbf{q}$ in time by the space-time decomposition:

$$\dot{\mathbf{q}}(t) = \frac{d\mathbf{q}}{ds}\dot{s}(t) = \mathbf{q}'\dot{s} \tag{6.18}$$

$$\ddot{\mathbf{q}}(t) = \frac{d\mathbf{q}}{ds}\ddot{s}(t) + \frac{d^2\mathbf{q}}{ds}\dot{s}^2(t) = \mathbf{q}'\ddot{s} + \mathbf{q}''\dot{s}^2 \tag{6.19}$$

6.1.1 Trajectory Bounds

In real case situations, the actuators can't provide an unbounded amount of joint velocity, we model this limit as bounds for each joint:

$$|\dot{q}_j(t)| \leq v_{j,max} \quad j \in 1, \dots n \quad t \in [0, T] \tag{6.20}$$

this limit have to be represented for q_j with the parametrization s, since

$$\dot{q_j}(t) = q_j'(s)\dot{s}(t) \implies \tag{6.21}$$

$$|\dot{q_j}(t)| = |q_j'(s)| \cdot |\dot{s}(t)| \tag{6.22}$$

the condition becomes:

$$\max_{s\in[s_i,s_f],t\in[0,T]} |q_j'(s)| \cdot |\dot{s}(t)| \leq v_{j,max}, \quad j \in 1, \dots, n \tag{6.23}$$

by doing some algebraic manipulation, we get:

$$\max_{t\in[0,T]} |\dot{s}(t)| \leq \min_{j=1,\dots,n} \frac{v_{j,\max}}{\max_{s\in[s_i,s_f]} |q_j'(s)|} \tag{6.24}$$

by denoting

$$V = \min_{j=1,\dots,n} \frac{v_{j,\max}}{\max_{s\in[s_i,s_f]} |q_j'(s)|} \tag{6.25}$$

we have

$$|\dot{s}(t)| \leq V \tag{6.26}$$

If this condition is always satisfied, then we will never outrange the actuator limits.
TODO:Acceleration limit slide 28

6.2 Trajectory Planning in the Joint Space

When we define a trajectory $\mathbf{q}(t)$ in time or $\mathbf{q}(s)$ with the timing law $s = s(t)$ we can work component wise for each q_j. In the common case, we define implicitly a trajectory by interpolating a set of points in the space with some boundary conditions, with a fixed class of template functions.

Typical classes of functions are

- polynomials
- trigonometric
- clothoids
- exponentials

the imposed conditions are given in time and/or space such as:

- pass trough points (interpolate the knots)
- initial, final, intermediate velocity
- initial, final, intermediate acceleration
- continuity for the time or space derivative up to the k-th order.

Clearly, the more conditions we have, the more the function will be "complex" (for example, the order of the polynomial will be higher).

6.2.1 PTP Planning

We want to define a trajectory from $\mathbf{q}_i$ to $\mathbf{q}_f$, with the condition to have a desired velocity at the start and at the end of the trajectory. The four conditions are

$$\mathbf{q}(0) = \mathbf{q}_i \tag{6.27}$$
$$\mathbf{q}(1) = \mathbf{q}_f \tag{6.28}$$
$$\mathbf{q}'(0) = \mathbf{v}_i \tag{6.29}$$
$$\mathbf{q}'(1) = \mathbf{v}_f \tag{6.30}$$
$$\tag{6.31}$$

To satisfy these four conditions with a polynomial function, is strictly necessary that the order is at least 3. We consider a cubic polynomial defined in $s \in [0, 1]$:

$$\mathbf{q}(s) = \mathbf{q}_i + \Delta\mathbf{q}(as^3 + bs^2 + cs + d) \tag{6.32}$$

where $\Delta\mathbf{q} = \mathbf{q}_f - \mathbf{q}_i$ and $a, b, c, d \in \mathbb{R}$ are the coefficient that we have to tune to satisfy the conditions. For the first condition

$$\mathbf{q}(0) = \mathbf{q}_i + \Delta\mathbf{q}d = \mathbf{q}_i \iff \tag{6.33}$$
$$d = 0 \tag{6.34}$$

For the other conditions, we have to solve the following system of equation

$$\begin{cases} \mathbf{q}(1) = \mathbf{q}_i + (\mathbf{q}_f - \mathbf{q}_i)(a + b + c) = \mathbf{q}_f \\ \mathbf{q}'(0) = (\mathbf{q}_f - \mathbf{q}_i)c = \mathbf{v}_i \\ \mathbf{q}'(1) = (\mathbf{q}_f - \mathbf{q}_i)(3a + 2b + c) = \mathbf{v}_f \end{cases} \implies \tag{6.35}$$

$$\begin{cases} a + b + c = 1 \\ c = \frac{\mathbf{v}_i}{(\mathbf{q}_f - \mathbf{q}_i)} \\ 3a + 2b + c = \frac{\mathbf{v}_i}{(\mathbf{q}_f - \mathbf{q}_i)} \end{cases} \tag{6.36}$$

this is a system of three equations in three unknowns. If $\mathbf{v}_i = \mathbf{v}_f = \mathbf{0}$ we have

$$\begin{cases} a + b = 1 \\ c = 0 \\ 3a + 2b = 0 \end{cases} \implies \tag{6.37}$$

$$\begin{cases} a = -2 \\ b = 3 \\ c = 0 \end{cases} \tag{6.38}$$

Let's see an alternative trajectory that considers trigonometric functions, we have the four boundary constraints for a rest-to-rest motion:

$$\mathbf{q}(0) = \mathbf{q}_i \tag{6.39}$$
$$\mathbf{q}(1) = \mathbf{q}_f \tag{6.40}$$
$$\mathbf{q}'(0) = \mathbf{0} \tag{6.41}$$
$$\mathbf{q}'(T) = \mathbf{0} \tag{6.42}$$
$$\tag{6.43}$$

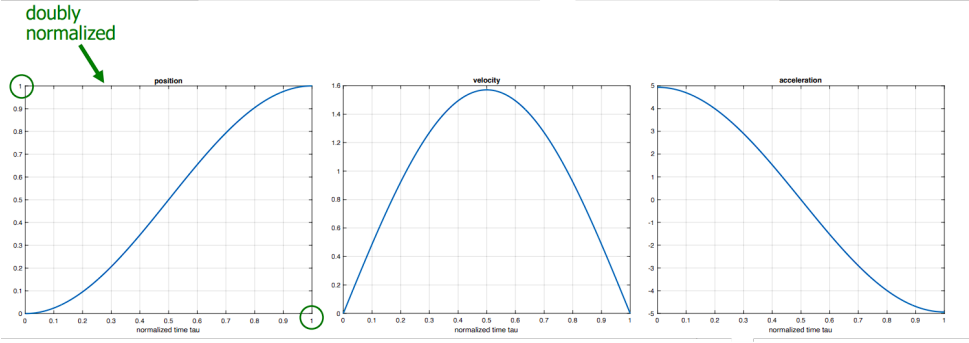
denoting $\Delta\mathbf{q} = \mathbf{q}_f - \mathbf{q}_i$, and $\tau = \frac{t}{T} \in [0, 1]$, we consider the following trajectory

$$\mathbf{q}(\tau) = \mathbf{q}_i + \Delta\mathbf{q}\frac{1 - \cos(\pi\tau)}{2} \tag{6.44}$$

We have the following profiles for the velocity and the acceleration:

$$\dot{\mathbf{q}}(\tau) = \frac{\Delta\mathbf{q}}{T}\frac{\pi}{2}\sin(\pi\tau) \tag{6.45}$$

$$\ddot{\mathbf{q}}(\tau) = \frac{\Delta\mathbf{q}}{T^2}\frac{\pi^2}{2}\cos(\pi\tau) \tag{6.46}$$



To satisfy more conditions, we need more sofisticate templates, for example, to satisfy the following 6 conditions (in normalized time $\tau = \frac{t}{T}$)

$$\mathbf{q}(0) = \mathbf{q}_i \tag{6.47}$$

$$\mathbf{q}(1) = \mathbf{q}_f \tag{6.48}$$

$$\mathbf{q}'(0) = \mathbf{v}_i T \tag{6.49}$$

$$\mathbf{q}'(1) = \mathbf{v}_f T \tag{6.50}$$

$$\mathbf{q}''(0) = \mathbf{a}_i T^2 \tag{6.51}$$

$$\mathbf{q}''(1) = \mathbf{a}_f T^2 \tag{6.52}$$

$$\tag{6.53}$$

is required a 6-degree polynomial:

$$\mathbf{q}(\tau) = (1-\tau)^3\left(\mathbf{q}_i + (3\mathbf{q}_i + \mathbf{v}_i T)\tau + (\mathbf{a}_i T^2 + 6\mathbf{v}_i T + 12\mathbf{q}_i)\frac{\tau^2}{2}\right) \tag{6.54}$$

$$+ \tau^3\left(\mathbf{q}_f + (3\mathbf{q}_f - \mathbf{v}_f T)(1-\tau) + (\mathbf{a}_f T^2 - 6\mathbf{v}_f T + 12\mathbf{q}_f)(1-\tau)^2\frac{1}{2}\right) \tag{6.55}$$

in a rest-to-rest trajectory with initial and final velocity and acceleration the form is simpler:

$$\mathbf{q}(\tau) = \mathbf{q}_i + \Delta\mathbf{q}(6\tau^5 - 5\tau^4 + 10\tau^3) \tag{6.56}$$

In general, for symmetric boundary conditions (that imposes zero values for higher order derivatives) the interpolating polynomial is always of odd degree, the coefficient of the doubly normalized polynomial are always

- integers
- alternate in sign
- sum up to 1
- all zero for the terms of degree less or equals to $\frac{\text{degree}-1}{2}$

For the MP tasks (interpolates with a function a large number of knots) polynomials are not recommended, since, for interpolate a function of N points, is required a $N-1$-degree polynomial, and this will have $N-2$ maximum and minimum points, so there will be a condition of oscillation outside the interpolating points (*wandering*).

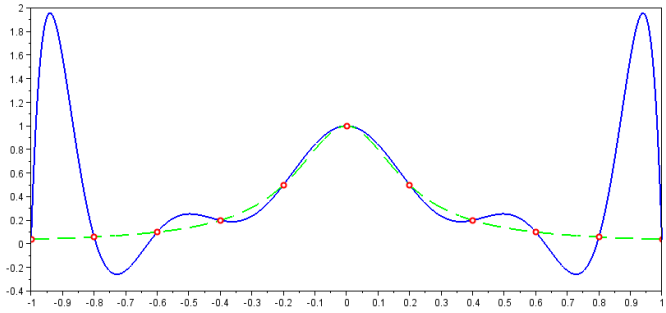


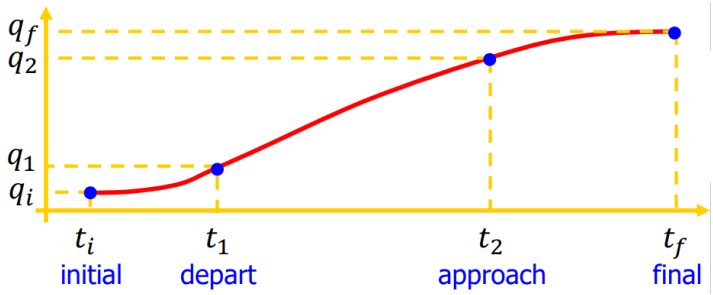
Figure 6.1: Note how the blue line (the interpolating polynomial for the 11 points) is subject to uncontrolled oscillation outside the points.

We call **wandering** the general situation where we don't know how a trajectory will behave outside the interpolation points

There is another interesting case, about pick and place operations, where we consider a three phases trajectory known as **LTS** (Lift off, Travel, Set down)

- the robot should move slowly when picking the object
- accelerate when traveling
- decelerate to place the object

We can use three functions, one for each phase, two polynomial of degree 4 for the lift off and set off operations, and one 3-degree polynomial for the travel operation.



This graph concerns a generic joint of the **q** vector. The boundary conditions for this trajectory are (for a single joint q)

$$q(t_i) = q_0 \tag{6.57}$$
$$q(t_1^-) = q(t_1^+) = q_1 \tag{6.58}$$
$$q(t_2^-) = q(t_2^+) = q_2 \tag{6.59}$$
$$q(t_f) = q_f \tag{6.60}$$

for the initial and final velocity acceleration:

$$\dot{q}(t_i) = \dot{q}(t_f) = 0 \tag{6.61}$$
$$\ddot{q}(t_i) = \ddot{q}(t_f) = 0 \tag{6.62}$$

the conditions for the continuity (up to acceleration) are

$$\dot{q}(t_1^-) = \dot{q}(t_1^+) \tag{6.63}$$
$$\dot{q}(t_2^-) = \dot{q}(t_2^+) \tag{6.64}$$
$$\ddot{q}(t_1^-) = \ddot{q}(t_1^+) \tag{6.65}$$
$$\ddot{q}(t_2^-) = \ddot{q}(t_2^+) \tag{6.66}$$

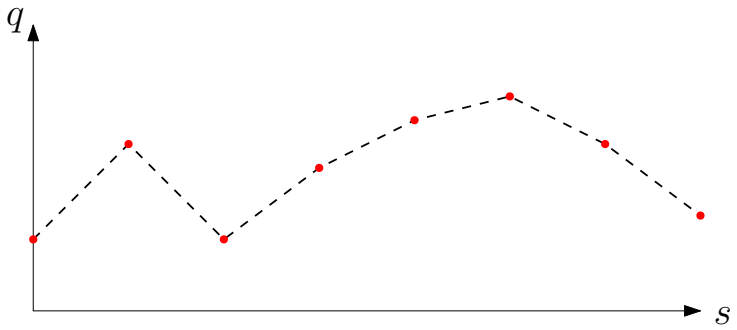
this is a 14 equations linear system and a solution can be found in symbolic form. In particular, the solutions consists in these 3 polynomial function q_L, q_T, q_S where

- q_L is a 4-degree polynomial and is defined in $[t_i, t_1]$
- q_T is a 3-degree polynomial and is defined in $[t_1, t_2]$
- q_S is a 4-degree polynomial and is defined in $[t_2, t_f]$

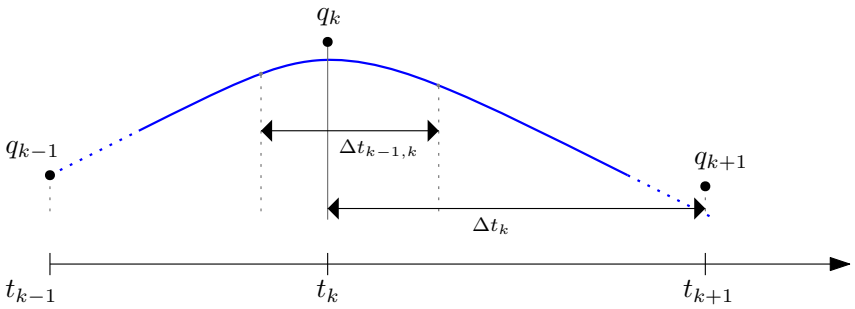
and the final trajectory for the joint will be

$$q(t) = \begin{cases} q_L(t) \text{ if } t \in [t_i, t_1] \\ q_T(t) \text{ if } t \in [t_1, t_2] \\ q_S(t) \text{ if } t \in [t_2, t_f] \end{cases} \tag{6.67}$$

This concept can be generalized: Instead of using an high order polynomial to interpolates N points, is better to use more polynomials of a lower degree, in order to reduce the wandering. Emphasizing that concept to the fullest, we might interpolates N points with $N-1$ linear functions.



By doing so, we would loose continuity on the tangent curve on the knots, a smart idea is to use quadratic polynomials to *blend the linear segments* and overfly the knots.



We denote Δt_k the time interval $t_{k+1} - t_k$. We define the **blending interval** $\Delta t_{k-1,k}$ the time where we include the quadratic polynomial instead of the linear function. The original linear segment from q_k to q_{k+1} is

$$\theta_k(t) = q_k + (q_{k+1} - q_k)\frac{s - s_k}{\Delta s_k} \tag{6.68}$$

The derivative is

$$\theta'_k = \frac{q_{k+1} - q_k}{\Delta s_k} \tag{6.69}$$

on the blending interval, the acceleration of the joint will be constant:

$$\theta''_k = \frac{\theta'_k - \theta'_{k-1}}{\Delta t_{k-1,k}} \tag{6.70}$$

by integrating two times this we will have a quadratic function with constant second derivative.

6.2.2 Splines Interpolation

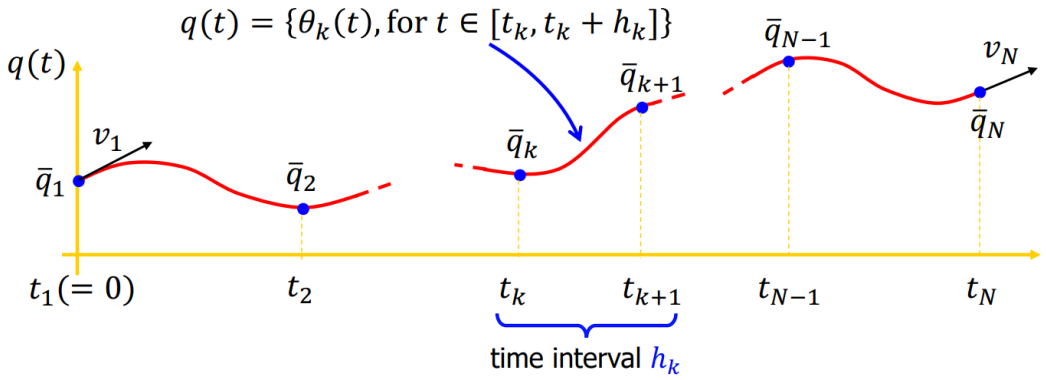
To plan a trajectory in the joint space, we have to solve the following problem

interpolate N knots, with continuity up to the second derivative

A solution consists in the **spline curve**, a series of $N-1$ cubic polynomial concatenated to pass through the N knots, continuous up to the second derivative at the $N-2$ internal knots. Since each cubic polynomial have 4 coefficients, and we have $N-1$ functions, the total number of coefficients of a spline is $4(N-1)$. The conditions to satisfy are

- $2(N-1)$: passage on the knots for each cubic
- $N-2$: continuity of first derivative on the internal knots
- $N-2$: continuity of second derivative on the internal knots

The total number of conditions are $4(N-1)-2$, so there will be 2 free parameters after fixing the other coefficients to satisfy the constraints. We describe the method for a single joint q.



We split the total time of the trajectory $[t_1, t_N]$ in time intervals $[t_k, t_{k+1}]$, that are the interval to pass from q_k to q_{k+1}. Each θ_k are a cubic polynomial in function of $\tau = t - t_k \in [0, h_k]$

$$\theta_k(\tau) = \sum_{i=0}^{3} a_{ki}\tau^i \tag{6.71}$$

The continuity conditions for the velocity and the acceleration are the following:

$$\begin{aligned} \dot{\theta}_k(h_k) &= \dot{\theta}_{k+1}(0) \\ \ddot{\theta}_k(h_k) &= \ddot{\theta}_{k+1}(0) \end{aligned} \qquad k = 1\ldots, N-2 \tag{6.72}$$

Notice how, in each interval $[t_k, t_{k+1}]$, the cubic polynomial θ_k have to satisfy exactly 4 boundary constraints.

We now describe an algorithm to compute a spline, we assume that the velocities on the internal knots (the derivative of the cubic function θ_k) are known, in that case, there exists a unique solution for the spline.

- known data:

$$\begin{aligned} \theta_k(0) = q_k = a_{k0} &\text{ knots position} \\ \dot{\theta}_k(0) = v_k = a_{k1} &\text{ derivative on the knots} \end{aligned} \qquad k = 1\ldots, N$$

Notice how we denoted a_{k0}, a_{k1} the value of θ_k and $\dot{\theta}_k$ on 0. Remember that θ_k is in function of $\tau = t - t_k$, so $\theta_k(\tau = 0) = \theta_k(t = t_k)$

- a_{k0} is the value of the spline (the total trajectory) at $t = t_k$
- a_{k1} is the value of the spline's derivative at $t = t_k$

a_{k0} and a_{k1} are the first two coefficients of θ_k, we compute a_{k2} and a_{k3} by solving the following linear system:

$$\begin{pmatrix} h_k^2 & h_k^3 \\ 2h_k & 3h_k^3 \end{pmatrix} \begin{pmatrix} a_{k2} \\ a_{k3} \end{pmatrix} = \begin{pmatrix} q_{k+1} - q_k - v_k h_k \\ v_{k+1} - v_k \end{pmatrix} \tag{6.73}$$

And we impose the continuity on the cubic θ_k by letting the second derivative to be

$$\ddot{\theta}_k(h_k) = 2a_{k2} + 6a_{k3}h_k = 2a_{k+1,2} = \ddot{\theta}_{k+1}(0) \tag{6.74}$$

We can express $a_{k2}, a_{k3}, a_{k+1,2}$ in terms of the still unknown velocities such as in the system in equation (6.73), this yields to a linear system of the following form:

$$A(h_1 \dots h_{N-1}) \begin{pmatrix} v_2 \\ \vdots \\ v_{N-1} \end{pmatrix} = \mathbf{b}(h_1 \dots h_{N-1}, q_1 \dots q_N, v_1, v_N) \tag{6.75}$$

Where $A(h_1 \dots h_{N-1})$ is a tridiagonal matrix defined as follows:

$$\begin{pmatrix} 2(h_1+h_2) & h_1 & & & & & \\ h_3 & 2(h_2+h_3) & h_2 & & & & \\ & h_4 & 2(h_3+h_4) & h_3 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & & h_{N-2} & 2(h_{N-3}+h_{N-2}) & h_{N-3} \\ & & & & & h_{N-1} & 2(h_{N-2}+h_{N-1}) \end{pmatrix} \tag{6.76}$$

and $\mathbf{b}(h_1 \dots h_{N-1}, q_1 \dots q_N, v_1, v_N)$ is the following vector:

$$\begin{pmatrix} \frac{3}{h_1 h_2}\left(h_2^2(q_3-q_2)+h_1^2(q_2-q_1)\right) - h_2 v_1 \\ \frac{3}{h_2 h_3}\left(h_3^2(q_4-q_3)+h_2^2(q_3-q_2)\right) \\ \vdots \\ \frac{3}{h_{N-3} h_{N-2}}\left(h_{N-2}^2(q_{N-1}-q_{N-2})+h_{N-3}^2(q_{N-2}-q_{N-3})\right) \\ \frac{3}{h_{N-2} h_{N-1}}\left(h_{N-1}^2(q_N-q_{N-1})+h_{N-2}^2(q_{N-1}-q_{N-2})\right) - h_{N-2} v_N \end{pmatrix} \tag{6.77}$$

To resume the algorithm:

- We have as an input, the knots q_k for each $k = 1 \dots N$ and the velocities on the boundary : v_1 and v_N.
- We compute the velocities on the internal knots $v_2 \dots v_{N-1}$ by solving the linear system in equation (6.75).
- We compute the accelerations of the $N-2$ internal knots by solving the system (6.73) for each k.
- In the end, we have $a_{k0}, a_{k1}, a_{k2}, a_{k3}$ for each k, each cubic is

  $$\theta_k(\tau) = a_{k0} + a_{k1}\tau + a_{k2}\tau^2 + a_{k3}\tau^3 \qquad \tau = t - t_k \in [0, h_k] \tag{6.78}$$

  each cubic can be also expressed in the global time t:

  $$\theta_k(t) = a_{k0} + a_{k1}(t-t_k) + a_{k2}(t-t_k)^2 + a_{k3}(t-t_k)^3 \tag{6.79}$$

  when t_k is the moment when q have to be on q_k.

A spline curve is a good choice since it satisfy some good properties:

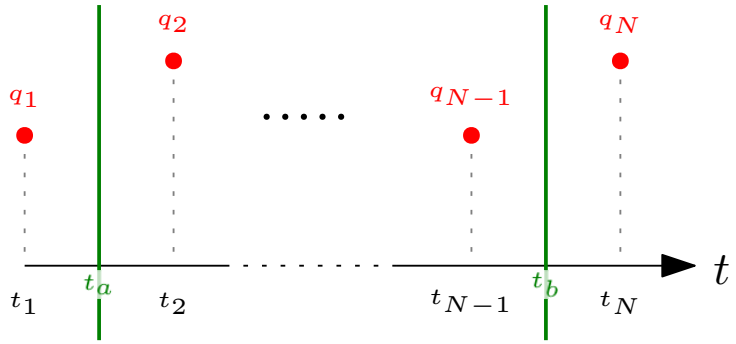
- a spline curve has the minimum curvature among all interpolating functions with continuous second derivative. For that reason, it's a good choice to use a spline for a trajectory in the space.
- a spline is uniquely determined by the set of data
  - $q_1 \dots q_N$ knots
  - $h_1 \dots h_N$ time intervals
  - v_1, v_N boundary velocities
- in time, the total motion occurs in $T = \sum_k h_k = t_n - t_1$
- the time intervals h_k can be chosen to minimize T under some bounds on velocities and accelerations in $[0, T]$

- we can construct a spline even if we want to impose a given acceleration for the initial and last knots : a_1, a_N.

By imposing zero acceleration:

$$q_1''(0) = q_N''(h_N) = 0 \tag{6.80}$$

We add two more constraints, the total number of conditions are $4N-2$ but we have only $4(N-1)$ available parameters, so we have to introduce 2 **virtual knots**. The first one will be defined in $t_a \in [t_1, t_2]$, the second one is defined in $t_b \in [t_{N-1}, t_N]$.



These two knots defined in t_a, t_b are virtual since the value for q in that instants is not defined

$$q(t_a), q(t_b) \text{ undefined} \tag{6.81}$$

These two knots creates two new sub intervals since instead of $[t_1, t_2]$ we will have $[t_1, t_a]$ and $[t_a, t_2]$ (analogous for t_b), and we will have two additional cubics θ_a and θ_b, bringing the number of free parameters to a total of $4(N+1)$. On the other hand, the total number of conditions is also increased to $4(N+1)$.

6.3 Trajectory Planning in the Cartesian Space

There are two ways to impose a trajectory for a manipulator:

- By defining a series of knots in the joint space, and making the manipulator to pass trough a configuration to an other by imposing different positions
- By defining a parametrized curve γ in the space and by controlling the end effector velocity with the inverse Jacobian matrix, considering the derivative of γ over time.

We usually consider a series of knots and interpolate the paths with simple curves, such as arc of circles or directly straight line.

6.3.1 Planning a Linear Path

Let's consider two points in the space $\mathbf{p}_i, \mathbf{p}_f \in \mathbb{R}^3$, and initial and final velocities $v_i, v_f \in \mathbb{R}$, these are just real numbers since the direction are implicitly given by $\mathbf{p}_f - \mathbf{p}_i$ (since we are planning a straight line). A simple path parametrization is given by

$$\mathbf{p}(s) = \mathbf{p}_i + s(\mathbf{p}_f - \mathbf{p}_i) \qquad s \in [0, 1] \tag{6.82}$$

Let L to be the length of the path: $L = \|\mathbf{p}_f - \mathbf{p}_i\|$, we can consider a different parametrization $s = \frac{\sigma}{L}$:

$$\mathbf{p}(\sigma) = \mathbf{p}_i + \frac{\sigma}{L}(\mathbf{p}_f - \mathbf{p}_i) \qquad \sigma \in [0, L] \tag{6.83}$$

with this parametrization σ represents the current length of the path. The velocity is given by

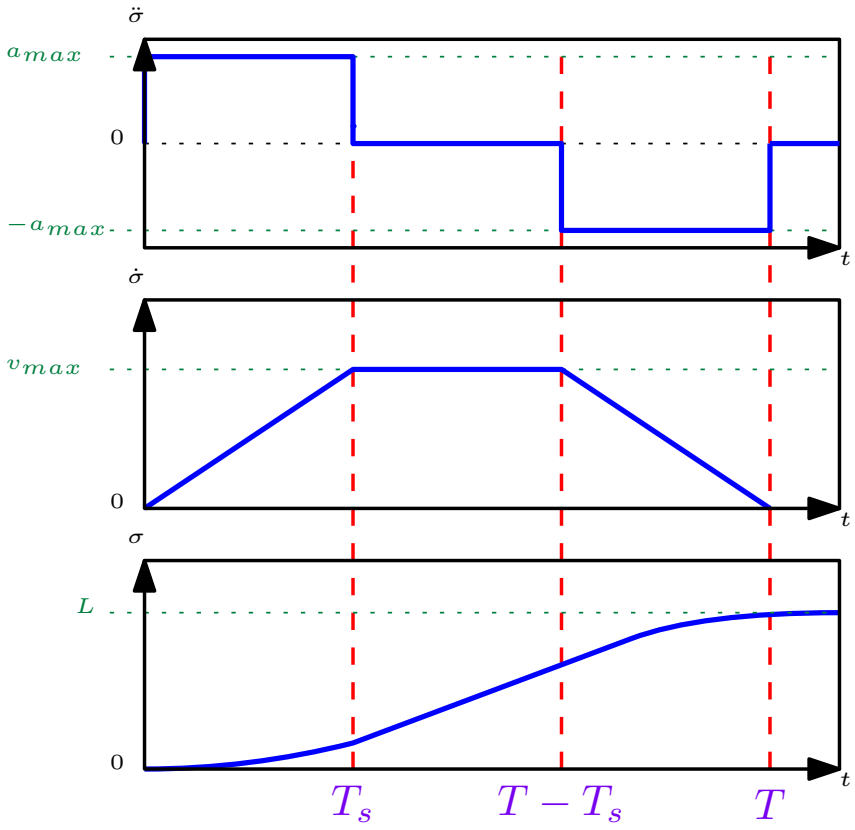
$$\dot{\mathbf{p}}(s) = \frac{d\mathbf{p}}{ds}\dot{s} = (\mathbf{p}_f - \mathbf{p}_i)\dot{s} = \frac{1}{L}(\mathbf{p}_f - \mathbf{p}_i)\dot{\sigma} \tag{6.84}$$

We recall that s is a parameter that depends on time t when we assign a timing law. The acceleration is given by

$$\ddot{\mathbf{p}}(s) = \frac{d^2\mathbf{p}}{ds^2}\dot{s}^2 + \frac{d\mathbf{p}}{ds}\ddot{s} = (\mathbf{p}_f - \mathbf{p}_i)\ddot{s} = \frac{1}{L}(\mathbf{p}_f - \mathbf{p}_i)\ddot{\sigma} \tag{6.85}$$

since $\frac{d^2\mathbf{p}}{ds^2} = \mathbf{0}$. We have to assign a timing law to satisfy the constraints on final and initial velocities v_i, v_f. For simplicity, we let $v_i = v_f = 0$ meters per seconds.

We consider a **trapezoidal timing law** $\sigma(t)$ named *bang-coast-bang*. The point will accelerate until it reaches the max velocity v_{max} given by the limit of the actuators, in particular, will accelerate at the max acceleration a_{max}, to minimize the time to complete the path.



We define T the total time to complete the path and T_s the time to reach the max speed v_{max}. We note how the area under the speed profile is given by

$$\int_0^T \dot{\sigma}(t)dt = \tag{6.86}$$

$$(T_s)v_{max}\frac{1}{2} + \tag{6.87}$$

$$(T - 2T_s)v_{max} + \tag{6.88}$$

$$(T - (T - T_s))v_{max}\frac{1}{2} = \tag{6.89}$$

$$v_{max}(T - T_s) \tag{6.90}$$

for the fundamental theorem of calculus, this area is the total variation of $\sigma(t)$, that is L, so we get the equation

$$L = v_{max}(T - T_s) \tag{6.91}$$

it follows:

$$T_s = \frac{v_{max}}{a_{max}} \tag{6.92}$$

$$T = \frac{La_{max} + v_{max}^2}{a_{max}v_{max}} \tag{6.93}$$

If the path is to short, or, if the max acceleration is not strong enough, it might be impossible to reach the max velocity, in that case, the "coast" phase is absent and the velocity profile is just a triangle, this happens if

$$L \leq \frac{v_{max}^2}{a_{max}} \tag{6.94}$$

The explicit form of $\sigma(t)$ is

$$\sigma(t) = \begin{cases} \frac{a_{max}t^2}{2} \text{ if } t \in [0, T_s] \\ v_{max}t - \frac{v_{max}^2}{2a_{max}} \text{ if } t \in [T_s, T - T_s] \\ -\frac{a_{max}(t-T)^2}{2} + L \text{ if } t \in [T - T_s, T] \end{cases} \tag{6.95}$$

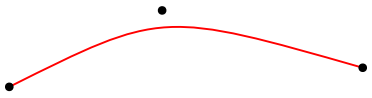
After defining a path over time for the point $\mathbf{p}(s) = \mathbf{p}(\sigma(t))$, we consider the velocity $\dot{\mathbf{p}}$, and at each sampling time we impose for the end effector this velocity by solving the inverse differential kinematics $\dot{\mathbf{q}} = J^{\#}(\mathbf{q})\dot{\mathbf{p}}$.

> In practice (especially in industrial robotics), simply solving $\dot{\mathbf{q}} = J^{\#}(\mathbf{q})\dot{\mathbf{p}}$ can lead to numerical integration errors or drift, causing the actual position to diverge from the desired one. To prevent this, a closed feedback loop is typically used by adding an error compensation term.

6.3.2 Concatenation of Linear Paths

We have to plan a trajectory for the end effector that pass trough N points in the cartesian space. A way to define a cartesian trajectory is to let the end effector to follow $N-1$ linear paths between the points.

Clearly, there is a discontinuity between two paths, that forces the end effector to stop. The way to account this is to *overfly* the points, as we already seen.



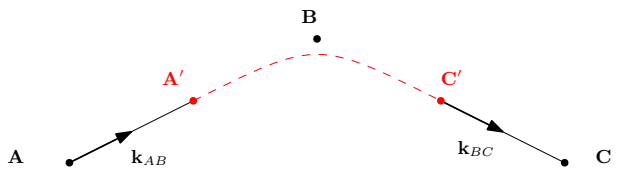
Let's model this problem.

- We have three points in the space **A**, **B** and **C**
- **A** and **C** are the initial and final points, **B** is the *via point*, that we have to overfly.
- The directions of the linear paths **AB** and **BC** are given by

$$\mathbf{k}_{AB} = \frac{\mathbf{B} - \mathbf{A}}{\|\mathbf{B} - \mathbf{A}\|} \tag{6.96}$$

$$\mathbf{k}_{BC} = \frac{\mathbf{C} - \mathbf{B}}{\|\mathbf{C} - \mathbf{B}\|} \tag{6.97}$$

- Given v_1 and v_2 (constant speeds on the linear paths **AB** and **BC**), we want to find the time ΔT needed for the transition at constant acceleration.



We want to define the profile of the point $\mathbf{p}(t)$ on the overfly trajectory, in $t \in [0, \Delta T]$ (the transition starts at $t = 0$). We have some constraints

- the initial velocity of $\mathbf{p}(t)$ is $\mathbf{k}_{AB}v_1$

- the final velocities of $\mathbf{p}(t)$ is $\mathbf{k}_{BC}v_2$

So

$$\dot{\mathbf{p}}(0) = \mathbf{k}_{AB}v_1 \tag{6.98}$$
$$\dot{\mathbf{p}}(\Delta T) = \mathbf{k}_{BC}v_2 \tag{6.99}$$

Since on the linear paths the velocity is constant, the acceleration of $\mathbf{p}$ should be zero on $t = 0, \Delta T$

$$\ddot{\mathbf{p}}(0) = \mathbf{0} \tag{6.100}$$
$$\ddot{\mathbf{p}}(\Delta T) = \mathbf{0} \tag{6.101}$$

The constant acceleration needed to go from a velocity of $\mathbf{k}_{AB}v_1$ to $\mathbf{k}_{BC}v_2$ in a total time of ΔT is

$$\ddot{\mathbf{p}}(t) = \frac{1}{\Delta T}(\mathbf{k}_{BC}v_2 - \mathbf{k}_{AB}v_1) \tag{6.102}$$

The velocity profile is given by integrating that in time

$$\int_0^t \ddot{\mathbf{p}}(\tau)d\tau = \frac{1}{\Delta T}(\mathbf{k}_{BC}v_2 - \mathbf{k}_{AB}v_1)t + \mathbf{c} \tag{6.103}$$

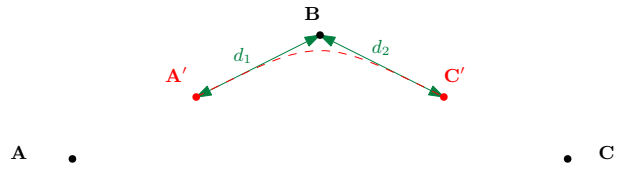
where $\mathbf{c}$ is a constant that is the initial velocity (that is $\mathbf{k}_{AB}v_1$), so:

$$\dot{\mathbf{p}}(t) = \frac{1}{\Delta T}(\mathbf{k}_{BC}v_2 - \mathbf{k}_{AB}v_1)t + \mathbf{k}_{AB}v_1 \tag{6.104}$$

By integrating again the velocity we get the position profile that is a parabolic curve (the constant added after the integration is $\mathbf{A}'$, the initial position where the overfly trajectory starts).

$$\mathbf{p}(t) = \mathbf{A}' + \frac{1}{2\Delta T}(\mathbf{k}_{BC}v_2 - \mathbf{k}_{AB}v_1)t^2 + \mathbf{k}_{AB}v_1 t \tag{6.105}$$

We now have to find the total transition time ΔT. We denote d_1 the distance between $\mathbf{A}'$ and $\mathbf{B}$, and d_2 the distance between $\mathbf{C}'$ and $\mathbf{B}$.



So we define the two vectors:

$$\mathbf{B} - \mathbf{A}' = d_1\mathbf{k}_{AB} \tag{6.106}$$
$$\mathbf{C}' - \mathbf{B} = d_2\mathbf{k}_{BC} \tag{6.107}$$

We know that $\mathbf{p}(\Delta T) = \mathbf{C}'$, so:

$$\mathbf{p}(\Delta T) = \mathbf{A}' + \frac{\Delta T}{2}(v_1\mathbf{k}_{AB} + v_2\mathbf{k}_{BC}) = \mathbf{C}' \implies \tag{6.108}$$
$$-\mathbf{B} + \mathbf{A}' + \frac{\Delta T}{2}(v_1\mathbf{k}_{AB} + v_2\mathbf{k}_{BC}) = \mathbf{C}' - \mathbf{B} \implies \tag{6.109}$$
$$\frac{\Delta T}{2}(v_1\mathbf{k}_{AB} + v_2\mathbf{k}_{BC}) = \mathbf{C}' - \mathbf{B} + \mathbf{B} - \mathbf{A}' \implies \tag{6.110}$$
$$\frac{\Delta T}{2}(v_1\mathbf{k}_{AB} + v_2\mathbf{k}_{BC}) = d_1\mathbf{k}_{AB} + d_2\mathbf{k}_{BC} \tag{6.111}$$

So we can express d_1 and d_2 in terms of ΔT

$$d_1 = v_1\frac{\Delta T}{2} \tag{6.112}$$
$$d_2 = v_2\frac{\Delta T}{2} \tag{6.113}$$

In the end:

$$\Delta T = 2\frac{d_1}{v_1} \tag{6.114}$$

$$d_2 = d_1\frac{v_2}{v_1} \tag{6.115}$$

So, when we have to design an overfly parabolic trajectory, we have to decide one parameter between d_1, d_2 or ΔT (the other two follows from the relations (6.114) and (6.115)).

There exists other ways to interpolate points in the cartesian space, usually the manipulators have the built-in feature for a *circular path* through 3 points. In robots with spherical wrist, the planning may be decomposed between the orientation and the position:

- We plan the position for the center of the wrist (first $n-3$ joints)
- we plan separately the orientation with the last 3 coincident joints
- we assign a common timing law.

In general, it is convenient to parametrize the cartesian geometric path $\mathbf{p}(s)$ with its arc length, if so, the following conditions holds:

- The velocity can be expressed as follows

$$\dot{\mathbf{p}} = \frac{d\mathbf{p}}{dt} = \frac{d\mathbf{p}}{ds}\frac{ds}{dt} = \mathbf{p}'\dot{s} \tag{6.116}$$

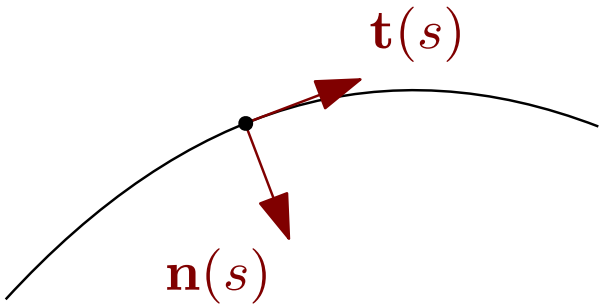
  where $\mathbf{p}'$ is the unit vector always tangent to $\mathbf{p}(s)$, and $\dot{s}$ is the absolute value of the tangential velocity. We denote $\mathbf{t}(s)$ the tangent direction. It is trivial that $\dot{s} \geq 0$ since the timing law is always designed to be non decreasing.

- the acceleration is

$$\ddot{\mathbf{p}} = \frac{d^2\mathbf{p}}{ds^2}\cdot\left(\frac{ds}{dt}\right)^2 + \frac{d\mathbf{p}}{ds}\cdot\frac{d^2s}{dt^2} = \mathbf{p}''\dot{s}^2 + \mathbf{p}'\ddot{s} \tag{6.117}$$

  where $\|\mathbf{p}''(s)\|$ is the *curvature* $\kappa(s)$ of the path $\mathbf{p}$ in s. $\mathbf{p}''\dot{s}^2$ is the *centripetal* acceleration that is orthogonal to the direction of the path, we denote $\mathbf{n}(s)$ the normal acceleration. $\ddot{s}$ is the tangential acceleration.

We defined the tangent direction $\mathbf{t}(s)$ and the normal direction $\mathbf{n}(s)$.



We define the *binormal direction*:

$$\mathbf{b}(s) = \mathbf{t}(s) \times \mathbf{n}(s) \tag{6.118}$$

In general, for a smooth and non degenerate curve $\mathbf{p}(s) \in \mathbb{R}^3$ parametrized by s that is not necessarily its arc length, we can define a reference frame for the path as follows:

- $\mathbf{t}(s)$ is the unit tangent vector:

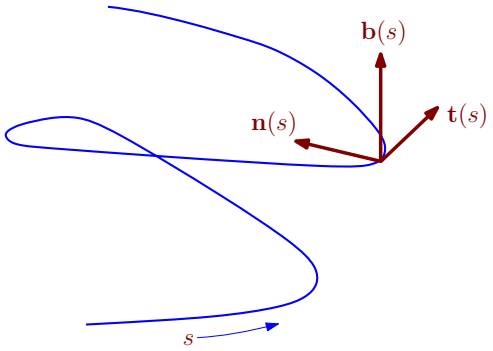
$$\mathbf{t}(s) = \frac{\mathbf{p}'(s)}{\|\mathbf{p}'(s)\|} \tag{6.119}$$

- $\mathbf{n}(s)$ is the normal unit vector that is directed toward the center of the osculating circle of the trajectory at point s.

$$\mathbf{n}(s) = \frac{\mathbf{t}'(s)}{\|\mathbf{t}'(s)\|} \tag{6.120}$$

- $\mathbf{b}(s)$ is the binormal vector that is orthogonal to the previous two to complete an orthonormal reference frame.

$$\mathbf{b}(s) = \mathbf{t}(s) \times \mathbf{n}(s) \tag{6.121}$$



This is known as the **Frenet Frame**. We can express the curvature and torsion of the path $\mathbf{p}$ at point s in the following general way:

$$\kappa(s) = \frac{\|\mathbf{p}'(s) \times \mathbf{p}''(s)\|}{\|\mathbf{p}'(s)\|^3} \tag{6.122}$$

$$\tau(s) = [\mathbf{p}'(s) \cdot (\mathbf{p}''(s) \times \mathbf{p}'''(s))] \frac{1}{\|\mathbf{p}'(s) \times \mathbf{p}''(s)\|^2} \tag{6.123}$$

Example

Let's consider the path

$$\mathbf{p}(s) = \begin{pmatrix} 6s+2 \\ 5s^2 \\ -8s \end{pmatrix}, \quad s \in [0,1] \tag{6.124}$$

We want to define the Frenet frame. We compute $\mathbf{p}'(s)$:

$$\frac{d\mathbf{p}}{ds} = \begin{pmatrix} 6 \\ 10s \\ -8 \end{pmatrix} \tag{6.125}$$

the norm is

$$\|\mathbf{p}'(s)\| = \sqrt{6^2 + (10s)^2 + 8^2} = \sqrt{100 + 100s^2} = 10\sqrt{1+s^2} \tag{6.126}$$

the tangent vector is

$$\mathbf{t}(s) = \begin{pmatrix} 10\sqrt{1+s^2}(6s+2) \\ 10\sqrt{1+s^2}5s^2 \\ -10\sqrt{1+s^2}8s \end{pmatrix} = \begin{pmatrix} 60s\sqrt{1+s^2} + 20\sqrt{1+s^2} \\ 50s^2\sqrt{1+s^2} \\ -80s\sqrt{1+s^2}s \end{pmatrix} \tag{6.127}$$

Subsequently, the remaining frame vectors can be calculated. For the curvature, we compute:

$$\mathbf{p}'(s) \times \mathbf{p}''(s) = \tag{6.128}$$

$$\begin{pmatrix} 6 \\ 10s \\ -8 \end{pmatrix} \times \begin{pmatrix} 0 \\ 10 \\ 0 \end{pmatrix} = \begin{pmatrix} 80 \\ 0 \\ 60 \end{pmatrix} \tag{6.129}$$

the norm is

$$\|\mathbf{p}'(s) \times \mathbf{p}''(s)\| = \sqrt{80^2 + 60^2} = 100 \tag{6.130}$$

the curvature is

$$\kappa(s) = \frac{100}{(10\sqrt{1+s^2})^3} = \frac{1}{10 \cdot (s^2+1)^{3/2}} \tag{6.131}$$

We can talk about the *optimality* of a path, we say that a trajectory is optimal if can be executed in minimum time under velocity and acceleration constraints. The optimal timing law is the bang-coast-bang profile type in acceleration, in general, the optimal trajectory for a PTP motion is a *straight line* in the *joint space*

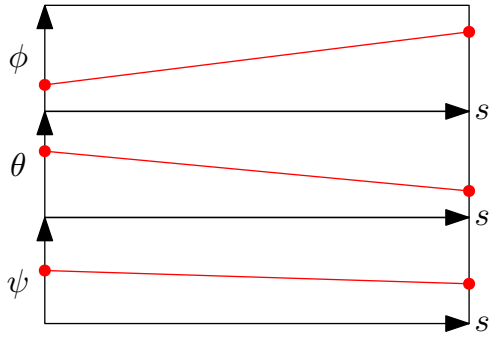
- for a 3P spatial robot, a straight line in the joint space corresponds to a straight line in the cartesian space
- this does not holds for an articulated robot with revolut joints.

6.3.3 Planning Orientation

We want to plan a change of orientation in time for the end effector of a manipulator. A naive approach is the following: We use the minimal representations of orientation with the Euler angles ϕ, θ, ψ, and we plan a trajectory for each component independently.

$$\begin{cases} \phi(s) = \phi_i + s(\phi_f - \phi_i) \\ \theta(s) = \theta_i + s(\theta_f - \theta_i) \\ \psi(s) = \psi_i + s(\psi_f - \psi_i) \end{cases} \tag{6.132}$$

For example, we plan a straight line in the angles ϕ, θ, ψ space with a cubic timing law:



The problem is that we loos understanding of the intermediate orientations between the starting and the final one, leading to wandering. An alternative method is based on the axis angle representation.

Let R_A to be the matrix that describe the initial orientation and R_B the one that describe the final orientation. We consider

$$R_A^T R_B \tag{6.133}$$

the matrix that describe the changing of orientation from the initial one to the desired orientation. We express this matrix in axis angle orientation

$$R(\mathbf{r}, \theta_{AB}) = R_A^T R_B \tag{6.134}$$

by solving the inverse problem and finding $\mathbf{r}$ and θ_{AB}. We plan a timing law $\theta(t)$ that starts from $\theta(t) = 0$ and ends at $\theta(t) = \theta_{AB}$, interpolating them in time $t \in [0, T]$. We impose the actual end effector orientation at time t as

$$R_A R(\mathbf{r}, \theta(t)) \tag{6.135}$$

In that case:

$$R_A R(\mathbf{r}, \theta(0)) = R_A R(\mathbf{r}, 0) = R_A \tag{6.136}$$

$$R_A R(\mathbf{r}, \theta(T)) = R_A R(\mathbf{r}, \theta_{AB}) = R_A R_A^T R_B = R_B \tag{6.137}$$

6.3.4 Uniform Time Scaling

When a robot's planned motion ignores physical velocity or acceleration limits, it may become unfeasible or inefficient. To fix this without changing the path's shape, you can apply a time scaling procedure. This adjusts the timing of the movement to ensure it stays within the robot's actual capabilities while optimizing performance. We describe this procedure in the joint space $\mathbf{q}$. Given a trajectory $\mathbf{q}(t)$ with $t \in [0, T]$, we define a new timing law by scaling t with $k > 0$ in the following way

$$t \in [0, T] \implies r = kt \in [0, T_s] \tag{6.138}$$

with $T_s = kT$. If $k < 1$, the motion are speeded up, if $k > 1$ the motion are slowed down. So we have defined this new trajectory

$$\mathbf{q}_s(r), \quad r \in [0, T_s] \tag{6.139}$$

the geometric path is the same of $\mathbf{q}$, and is no affected by the scaling procedure:

$$\mathbf{q}_s(r) = \mathbf{q}_s(kt) = \mathbf{q}(t) \tag{6.140}$$

The joint's velocity and acceleration are scaled for this new timing law:

$$\dot{\mathbf{q}}_s(r) = \frac{d\mathbf{q}_r}{dr} = \frac{d\mathbf{q}}{dt}\frac{dt}{dr} = \frac{1}{k}\dot{\mathbf{q}}(t) \tag{6.141}$$

$$\ddot{\mathbf{q}}_s(r) = \frac{d\dot{\mathbf{q}}_r}{dr} = \frac{d\dot{\mathbf{q}}}{dt}\frac{dt}{dr} = \frac{1}{k^2}\dot{\mathbf{q}}(t) \tag{6.142}$$

so the velocity is scaled by $\frac{1}{k}$ and the acceleration by $\frac{1}{k^2}$. We should apply the time scaling procedure if

- one of the velocity or acceleration constraint are violated (by slowing down the trajectory)
- none of the velocity or acceleration constraint are saturated (by speeding up the trajectory as much as we can)

let $v_{j,max}$ and $a_{j,max}$ to be the bounds for velocity and acceleration of joint j. Given the original trajectory $\mathbf{q}(t)$, we compute the maximum velocity and acceleration reached for each joint during the trajectory:

$$v_{j,peak} = \max_{t\in[0,T]} |\dot{q}_j(t)| \tag{6.143}$$

$$a_{j,peak} = \max_{t\in[0,T]} |\ddot{q}_j(t)| \tag{6.144}$$

We should choose as the time scaling factor k the following value:

$$k = \max_{j=1,\dots n} \left\{ \frac{v_{j,peak}}{v_{j,max}}, \sqrt{\frac{a_{j,peak}}{a_{j,max}}} \right\} \tag{6.145}$$

in that case, if some bounds are violated, the choice of equation (6.145) yields to a $k > 1$ that slow down the trajectory, by leading to a new $\mathbf{q}_s(r)$ that doesn't violate any constraint. Conversely, if all bounds are strictly satisfied, the choice of k yields to a speeded up trajectory $\mathbf{q}_s(r)$ with $k < 1$ such that at least one of the velocities or acceleration constraint are saturated in some instant (making the trajectory as fast as possible).

6.3.5 Doubly Normalized Polynomials

This small section is about a fast way to define a trajectory over some constraints. One of the most usual case, is to define a trajectory for the joints $\mathbf{q}$ (or analogously, for the task vector $\mathbf{r}$, or the position $\mathbf{p}$) that satisfies 4 boundary conditions:

$$\mathbf{q}(0) = \mathbf{q}_i \tag{6.146}$$

$$\mathbf{q}(1) = \mathbf{q}_f \tag{6.147}$$

$$\dot{\mathbf{q}}(0) = \mathbf{v}_i \tag{6.148}$$

$$\dot{\mathbf{q}}(1) = \mathbf{v}_f \tag{6.149}$$

If $\mathbf{q}$ should perform the trajectory in time from $t = 0$ to $t = T$, we can normalize the time variable by considering $\tau = \frac{t}{T}$, in such way, the trajectory $\mathbf{q}(\tau)$ is defined from 0 to 1. For these 4 conditions, a 3-degree polynomial is defined (we consider a single joint with the scalar variable $q(\tau)$):

$$q(\tau) = q_i + \Delta q(a\tau^3 + b\tau^2 + c\tau + d) \tag{6.150}$$

where $\Delta q = q_f - q_i$. The derivative of q respect to t is

$$\frac{d}{dt}\left(q_i + \Delta q(a\tau^3 + b\tau^2 + c\tau + d)\right) = \tag{6.151}$$

$$\frac{d}{dt}\left(q_i + \Delta q(a(t/T)^3 + b(t/T)^2 + c(t/T) + d)\right) = \tag{6.152}$$

$$\Delta q\left(\frac{3a}{T^3}t^2 + \frac{2b}{T^2}t + \frac{c}{T}\right) = \tag{6.153}$$

$$\frac{\Delta q}{T}\left(3a\tau^2 + 2b\tau + c\right) \tag{6.154}$$

We can solve the system of equations in a, b, c, d to find the numerical values of the coefficients.

$$\begin{cases} q_i + \Delta q d = q_i \\ q_i + \Delta q(a+b+c+d) = q_f \\ \frac{\Delta q}{T} c = v_i \\ \frac{\Delta q}{T}(3a+2b+c) = v_f \end{cases} \implies \begin{cases} d = 0 \\ c = \frac{Tv_i}{\Delta q} \end{cases} \tag{6.155}$$

then we have a system in two variables with two equations:

$$\begin{cases} 3a + 2b = T\dfrac{v_f - v_i}{\Delta q} \\ \\ a + b = \dfrac{q_f - q_i - Tv_i}{\Delta q} \end{cases} \tag{6.156}$$

the solution for the coefficients is

$$a = \frac{T(v_f + v_i) - 2\Delta q}{\Delta q} \tag{6.157}$$

$$b = \frac{3\Delta q - T(v_f + 2v_i)}{\Delta q} \tag{6.158}$$

$$c = \frac{Tv_i}{\Delta q} \tag{6.159}$$

$$d = 0 \tag{6.160}$$

In the rest-to-rest case where $v_i = v_f = 0$:

$$a = -2 \tag{6.161}$$

$$b = 3 \tag{6.162}$$

$$c = 0 \tag{6.163}$$

$$d = 0 \tag{6.164}$$

Example

We have to plan a rest-to-rest trajectory for q starting from $q_i = 1$ to $q_f = 4$ from $t = 0$ to $t = 2$ with continuity up to acceleration. We use the 3-degree normalized polynomial:

$$\Delta q = q_f - q_i = 3 \tag{6.165}$$

$$q(\tau) = 1 + 3(-2\tau^3 + 3\tau^2) \quad \tau \in [0, 1] \tag{6.166}$$

$$q(t) = 1 + 3\left(-\frac{1}{4}t^3 + \frac{3}{4}t^2\right) \quad t \in [0, 2] \tag{6.167}$$

the velocity is

$$\dot{q}(t) = -\frac{9}{4}t^2 + \frac{9}{2}t \tag{6.168}$$

CHAPTER

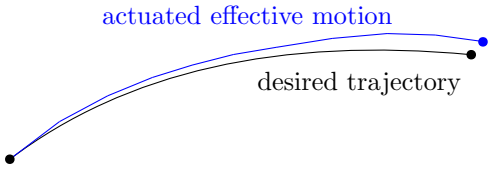
7

KINEMATIC CONTROL

This chapter requires some basic knowledge in control theory.

7.1 Feedback Loop

As we already explained, when we plan a trajectory in the cartesian space $\mathbf{p}(s)$, we impose a velocity for the end effector at each time step t by considering $\dot{\mathbf{p}}(s(t))$. If we want to realize a specific motion task without considerable errors, this is not sufficient: initial errors, discrete time implementation and uncertain robot parameters yields to an actual motion that differs from the desired trajectory.



For these reasons, our general control scheme should include

- a *feed forward action*, that commands the velocity of the end effector based on the planned trajectory $\mathbf{p}(s)$ (what we discussed in the previous chapter).
- a **feedback** action, that dynamically commands the robots and depends on the current error that came from the robot state measures, is necessary to impose *asymptotic stability* and to make the error converge to zero.

This control scheme is applied for the two general class of control problems:

- regulation: Make the system (our robot manipulator) to reach a constant pose/configuration
- tracking: Make the manipulator to follow a time-varying reference

About the notation: From now on, we denote

- $\mathbf{q}(t)$ the *current* configuration of the robot manipulator
- $\mathbf{p}(t)$ the *current* position of the task vector
- $\mathbf{q}_d(s(t))$ the *desired* configuration of the robot manipulator
- $\mathbf{p}_d(s(t))$ the *desired* position of the task vector

The error that we consider dynamically is the difference between the desired configuration and the actual configuration, the error can be considered in the joint space or in the task space. In the joint space:

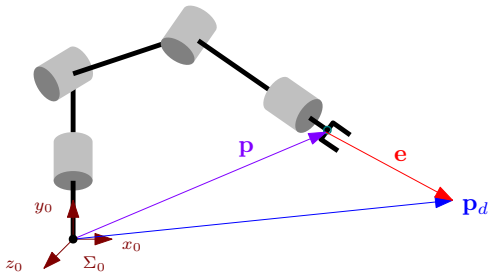
$$\mathbf{e}(t) = \mathbf{q}_d(s(t)) - \mathbf{q}(t) \tag{7.1}$$

- $\mathbf{q}_d(s(t))$: desired configuration at time t
- $\mathbf{q}(t)$: actual configuration at time t
- $s(t)$: timing law
- $\mathbf{e}(t) \in \mathbb{R}^n$ actual error in the joint space (n is the number of joints)

In the cartesian space:

$$\mathbf{e}(t) = \mathbf{p}_d(s(t)) - \mathbf{p}(t) \tag{7.2}$$

- $\mathbf{p}_d(s(t))$: desired position at time t
- $\mathbf{p}(t)$: actual position of the end effector at time t
- $\mathbf{e}(t) \in \mathbb{R}^3$ actual error in the task space (the segment between the desired position and the current one)



An example of a simple control low is the following, at each time step:

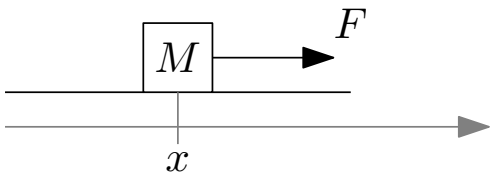
- We measure the current position $\mathbf{p}$ (for example, we get $\mathbf{q}$ from the absolute encoder mounted on the actuators and compute the direct kinematics $f_r(\mathbf{q})$)
- we consider the error such as the difference between $\mathbf{p}$ and the desired position $\mathbf{p}_d$
- we impose for the manipulator a velocity that is proportional to the error $\mathbf{e}$, by using the inverse differential kinematics $J^{\#}(\mathbf{q})\mathbf{e}$.

At low level, we can't control directly the velocity of the end effector, the robot is an electro-mechanical system driven by the motors that produces torques, there is a low-level control scheme that, given the desired velocity v_d, finds the correct input current i to realize the torque τ needed to make the motors to spin at the desired velocity v_d.

In this chapter, we assume that we can directly control the velocity of the motors. The low level control scheme for a single DC motor is presented in section 2.1.

7.1.1 Feedback Loop Example

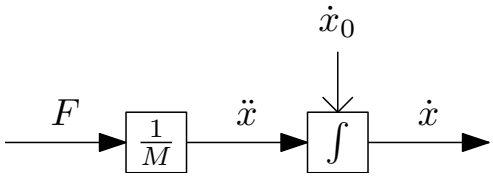
In this section we give some introductory example of how a feedback control scheme works for some simple linear systems. Let's consider the following toy example: There is a body of mass M kilograms, that lies on a plane, with zero friction, we can control the force F applied on the body in the axis parallel to the plane:



The configuration of the system is the x-axis position of the mass on the plane, that we denote x. The dynamic of the body is given by Newton's law:

$$M\ddot{x} = F \tag{7.3}$$

Since $\ddot{x} = \frac{F}{M}$, the dynamic of the system can be described by the following block diagram.

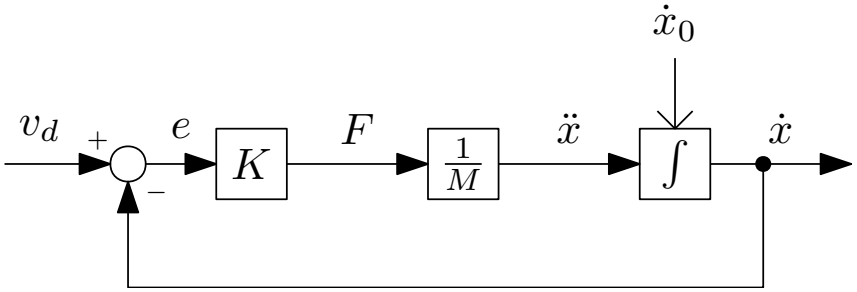


F is the input, the model of the system is given by a division by M and an integration, the output is the velocity $\dot{x}$, $\dot{x}_0$ is the initial velocity.

$$\dot{x}(t) = \frac{F}{M}t + \dot{x}_0 \tag{7.4}$$

We want to impose a desired velocity $v_d(t)$ for the body, we can add a feedback loop that computes the error and uses it as the input by scaling it for a factor $K > 0$:

$$F = K(v_d - \dot{x}) \tag{7.5}$$



The system dynamic is described by the following differential equation:

$$M\ddot{x} = K(v_d(t) - \dot{x}) \tag{7.6}$$

if the desired velocity $v_d(t) = \bar{v}_d$ is constant:

$$M\ddot{x} = K(\bar{v}_d - \dot{x}) \tag{7.7}$$

this first order differential equation in $\dot{x}$, for an initial velocity $\dot{x}_0$ admits an analytical solution:

$$\dot{x}(t) = \dot{x}_0 + ((\bar{v}_d - \dot{x}_0))\left(1 - \exp(-\frac{K}{M}t)\right) \tag{7.8}$$

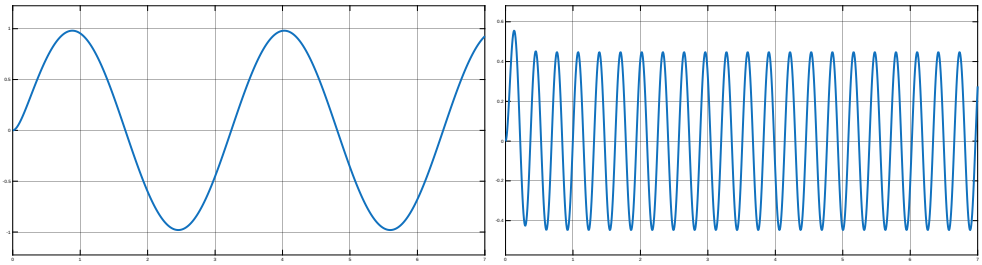
for t that goes to infinity, $\dot{x}$ converges to $\bar{v}_d$. the transfer function of the system is

$$H(s) = \frac{1}{Ms} \tag{7.9}$$

where s is the complex variable. By adding the feedback loop the global transfer function of the system with the controller became

$$W(s) = \frac{1}{\frac{M}{K}s + 1} \tag{7.10}$$

the cutoff frequency is $\frac{K}{M}$, so practically, we need to increase K if we want to impose a time varying desired velocity $v_d(t)$ at high frequency. For example, by letting $\frac{K}{M} = 10$, we have the two possible time responses to unit sinusoidal velocity reference $v_d(t) = \sin(\omega t)$, the left one with $\omega = 2$, the right one $\omega = 20$.



the controller realize perfectly the sinusoidal input with $\omega = 2$, but have an attenuation of the amplitude when tries to realize the input with $\omega = 20$.

7.2 Joint and Cartesian Control

We already explained how the low level control scheme of the robot is not treated in this context, in particular, we see the robot as a system that perfectly realizes the desired motion.

So, in this section we will just describe how to compute the input velocity in the joint space to realize a desired position $\mathbf{p}$.

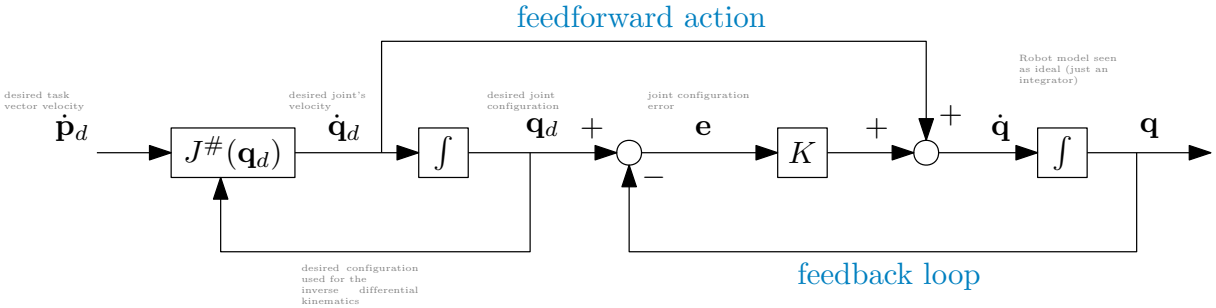
We can control the task vector by computing the error in the joint space or in the cartesian space. We consider now the kinematic control of joint motion, we have a reference position to follow in time:

$$\mathbf{p}_d \tag{7.11}$$

that is analytically defined by the trajectory planning module, so we have access to his derivative

$$\dot{\mathbf{p}}_d \tag{7.12}$$

this one will be the input of our control scheme.



The error is computed by considering the difference between the desired configuration and the actual configuration, K is a diagonal positive matrix.

$$\mathbf{e} = \mathbf{q}_d - \mathbf{q} \implies \tag{7.13}$$

$$\dot{\mathbf{e}} = \dot{\mathbf{q}}_d - \dot{\mathbf{q}} = \tag{7.14}$$

$$\dot{\mathbf{q}}_d - (\dot{\mathbf{q}}_d + K(\mathbf{q}_d - \mathbf{q})) = -K\mathbf{e} \tag{7.15}$$

So the system of differential equations that describe the evolution of the error is

$$\dot{\mathbf{e}} = -K\mathbf{e} \tag{7.16}$$

in scalar form:

$$\begin{cases} \dot{e}_1 = -k_1 e_1 \\ \vdots \\ \dot{e}_n = -k_n e_n \end{cases} \tag{7.17}$$

for each $j \in 1 \ldots, n$ the solution is given by

$$e_i(t) = e_{i,0} \exp(-kt) \tag{7.18}$$

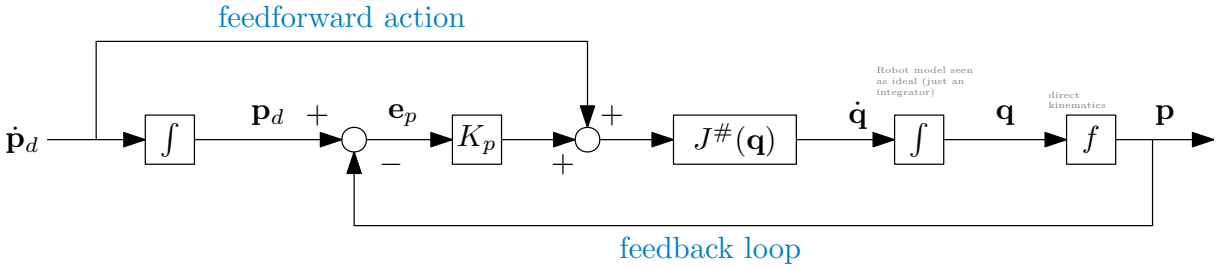
where $e_{i,0}$ is the initial error for the i-th joint. So the error in the joint space goes exponentially to zero. We can use the Jacobian matrix to explicit the error $\mathbf{e}_p$ in the cartesian space:

$$\mathbf{e}_p = \mathbf{p}_d - \mathbf{p} \implies \tag{7.19}$$
$$\dot{\mathbf{e}}_p = \dot{\mathbf{p}}_d - \dot{\mathbf{p}} = \tag{7.20}$$
$$\dot{\mathbf{e}}_p = J(\mathbf{q}_d)\dot{\mathbf{q}}_d - J(\mathbf{q})(\dot{\mathbf{q}}_d + K(\mathbf{q}_d - \mathbf{q})) = \tag{7.21}$$
$$\dot{\mathbf{e}}_p = J(\mathbf{q}_d)\dot{\mathbf{q}}_d - J(\mathbf{q})\dot{\mathbf{q}}_d - J(\mathbf{q})K(\mathbf{q}_d - \mathbf{q}) = \tag{7.22}$$
$$\dot{\mathbf{e}}_p = J(\mathbf{q}_d)\dot{\mathbf{q}}_d - J(\mathbf{q})\dot{\mathbf{q}}_d - J(\mathbf{q})\mathbf{e} \tag{7.23}$$

So when $\mathbf{q} \to \mathbf{q}_d$ we have that $\dot{\mathbf{e}}_p \to J(\mathbf{q})\mathbf{e}$, we can say that

$$\dot{\mathbf{e}}_p \simeq -J(\mathbf{q})KJ^{\#}(\mathbf{q})\mathbf{e}_p \tag{7.24}$$

We now consider a control scheme where the error is computed in the cartesian space.



Where K_p is a positive diagonal matrix. We want to derive the differential equation that describes the evolution of the error in time:

$$\mathbf{e}_p = \mathbf{p}_d - \mathbf{p} \implies \tag{7.25}$$
$$\dot{\mathbf{e}}_p = \dot{\mathbf{p}}_d - \dot{\mathbf{p}} = \tag{7.26}$$
$$\dot{\mathbf{p}}_d - J(\mathbf{q})J^{\#}(\mathbf{q})(\dot{\mathbf{p}}_d + K_p(\mathbf{p}_d - \mathbf{p})) = -K_p\mathbf{e}_p \tag{7.27}$$

so

$$\dot{\mathbf{e}}_p = -K_p\mathbf{e}_p \tag{7.28}$$

the cartesian error goes exponentially to zero.

APPENDIX

Useful Trigonometric Formulas

atan2 Definition

$$\operatorname{atan2}(y,x) = \begin{cases} \operatorname{atan}\left(\frac{y}{x}\right) & x > 0 \\ \pi + \operatorname{atan}\left(\frac{y}{x}\right) & y \geq 0, x < 0 \\ -\pi + \operatorname{atan}\left(\frac{y}{x}\right) & y < 0, x < 0 \\ \frac{\pi}{2} & y > 0, x = 0 \\ -\frac{\pi}{2} & y < 0, x = 0 \\ \text{undefined} & y = 0, x = 0 \end{cases}$$

Fundamental Identities

$$\begin{aligned} &\sin^2 x + \cos^2 x = 1 \\ &\tan^2 x + 1 = \sec^2 x \\ &1 + \cot^2 x = \csc^2 x \\ &\tan x = \frac{\sin x}{\cos x}, \quad \cot x = \frac{\cos x}{\sin x} \\ &\sec x = \frac{1}{\cos x}, \quad \csc x = \frac{1}{\sin x} \end{aligned}$$

Associated Angles

$$\begin{aligned} \sin(\pi - x) &= \sin x, & \cos(\pi - x) &= -\cos x \\ \sin(\pi + x) &= -\sin x, & \cos(\pi + x) &= -\cos x \\ \sin(2\pi - x) &= -\sin x, & \cos(2\pi - x) &= \cos x \\ \sin\left(\frac{\pi}{2} - x\right) &= \cos x, & \cos\left(\frac{\pi}{2} - x\right) &= \sin x \end{aligned}$$

Sum and Difference of Angles

$$\begin{aligned}
\sin(x+y) &= \sin x \cos y + \cos x \sin y \\
\cos(x+y) &= \cos x \cos y - \sin x \sin y \\
\tan(x+y) &= \frac{\tan x + \tan y}{1 - \tan x \tan y}
\end{aligned}$$

$$\begin{aligned}
\sin(x-y) &= \sin x \cos y - \cos x \sin y \\
\cos(x-y) &= \cos x \cos y + \sin x \sin y \\
\tan(x-y) &= \frac{\tan x - \tan y}{1 + \tan x \tan y}
\end{aligned}$$

Double and Triple Angle Formulas

$$\begin{aligned}
\sin(2x) &= 2 \sin x \cos x \\
\cos(2x) &= \cos^2 x - \sin^2 x = 2\cos^2 x - 1 = 1 - 2\sin^2 x \\
\tan(2x) &= \frac{2\tan x}{1 - \tan^2 x}
\end{aligned}$$

$$\begin{aligned}
\sin(3x) &= 3\sin x - 4\sin^3 x \\
\cos(3x) &= 4\cos^3 x - 3\cos x \\
\tan(3x) &= \frac{3\tan x - \tan^3 x}{1 - 3\tan^2 x}
\end{aligned}$$

Half-Angle Formulas

$$\begin{aligned}
\sin\frac{x}{2} &= \pm\sqrt{\frac{1-\cos x}{2}} \\
\cos\frac{x}{2} &= \pm\sqrt{\frac{1+\cos x}{2}} \\
\tan\frac{x}{2} &= \frac{\sin x}{1+\cos x} = \frac{1-\cos x}{\sin x}
\end{aligned}$$

Sum-to-Product Formulas

$$\begin{aligned}
\sin x + \sin y &= 2\sin\frac{x+y}{2}\cos\frac{x-y}{2} \\
\sin x - \sin y &= 2\cos\frac{x+y}{2}\sin\frac{x-y}{2} \\
\cos x + \cos y &= 2\cos\frac{x+y}{2}\cos\frac{x-y}{2} \\
\cos x - \cos y &= -2\sin\frac{x+y}{2}\sin\frac{x-y}{2}
\end{aligned}$$

Product-to-Sum Formulas

$$\begin{aligned}
\sin x \sin y &= \frac{1}{2}[\cos(x-y) - \cos(x+y)] \\
\cos x \cos y &= \frac{1}{2}[\cos(x-y) + \cos(x+y)] \\
\sin x \cos y &= \frac{1}{2}[\sin(x+y) + \sin(x-y)]
\end{aligned}$$

Exponentials and Complex Numbers

$$e^{ix} = \cos x + i \sin x$$
$$\cos x = \frac{e^{ix} + e^{-ix}}{2}, \quad \sin x = \frac{e^{ix} - e^{-ix}}{2i}$$
$$\tan x = \frac{e^{ix} - e^{-ix}}{i(e^{ix} + e^{-ix})}$$

$$(\cos x + i \sin x)^n = \cos(nx) + i \sin(nx)$$

Hyperbolic Functions and Exponentials

$$\sinh x = \frac{e^x - e^{-x}}{2}, \quad \cosh x = \frac{e^x + e^{-x}}{2}, \quad \tanh x = \frac{\sinh x}{\cosh x}$$
$$\sinh(ix) = i \sin x, \quad \cosh(ix) = \cos x$$

Complex Numbers in Trigonometric Form

$$z = r(\cos\theta + i \sin\theta) = re^{i\theta}$$
$$r = |z| = \sqrt{x^2 + y^2}, \quad \theta = \arg(z) = \tan^{-1} \frac{y}{x}$$

Periodicity and Symmetry

$$\sin(x + 2k\pi) = \sin x, \quad \cos(x + 2k\pi) = \cos x, \quad \tan(x + k\pi) = \tan x, \quad k \in \mathbb{Z}$$
$$\sin(-x) = -\sin x, \quad \cos(-x) = \cos x, \quad \tan(-x) = -\tan x$$

Products of Three Rotation Matrices

$$R_X(\alpha)R_Y(\beta)R_X(\gamma) = \begin{pmatrix} c\beta & s\beta s\gamma & s\beta c\gamma \\ s\alpha s\beta & -s\alpha s\gamma c\beta + c\alpha c\gamma & -s\alpha c\beta c\gamma - s\gamma c\alpha \\ -s\beta c\alpha & s\alpha c\gamma + s\gamma c\alpha c\beta & -s\alpha s\gamma + c\alpha c\beta c\gamma \end{pmatrix}$$

$$R_X(\alpha)R_Y(\beta)R_Z(\gamma) = \begin{pmatrix} c\beta c\gamma & -s\gamma c\beta & s\beta \\ s\alpha s\beta c\gamma + s\gamma c\alpha & -s\alpha s\beta s\gamma + c\alpha c\gamma & -s\alpha c\beta \\ s\alpha s\gamma - s\beta c\alpha c\gamma & s\alpha c\gamma + s\beta s\gamma c\alpha & c\alpha c\beta \end{pmatrix}$$

$$R_X(\alpha)R_Z(\beta)R_X(\gamma) = \begin{pmatrix} c\beta & -s\beta c\gamma & s\beta s\gamma \\ s\beta c\alpha & -s\alpha s\gamma + c\alpha c\beta c\gamma & -s\alpha c\gamma - s\gamma c\alpha c\beta \\ s\alpha s\beta & s\alpha c\beta c\gamma + s\gamma c\alpha & -s\alpha s\gamma c\beta + c\alpha c\gamma \end{pmatrix}$$

$$R_X(\alpha)R_Z(\beta)R_Y(\gamma) = \begin{pmatrix} c\beta c\gamma & -s\beta & s\gamma c\beta \\ s\alpha s\gamma + s\beta c\alpha c\gamma & c\alpha c\beta & -s\alpha c\gamma + s\beta s\gamma c\alpha \\ s\alpha s\beta c\gamma - s\gamma c\alpha & s\alpha c\beta & s\alpha s\beta s\gamma + c\alpha c\gamma \end{pmatrix}$$

$$R_Y(\alpha)R_X(\beta)R_Y(\gamma) = \begin{pmatrix} -s\alpha s\gamma c\beta + c\alpha c\gamma & s\alpha s\beta & s\alpha c\beta c\gamma + s\gamma c\alpha \\ s\beta s\gamma & c\beta & -s\beta c\gamma \\ -s\alpha c\gamma - s\gamma c\alpha c\beta & s\beta c\alpha & -s\alpha s\gamma + c\alpha c\beta c\gamma \end{pmatrix}$$

$$R_Y(\alpha)R_X(\beta)R_Z(\gamma) = \begin{pmatrix} s\alpha s\beta s\gamma + c\alpha c\gamma & s\alpha s\beta c\gamma - s\gamma c\alpha & s\alpha c\beta \\ s\gamma c\beta & c\beta c\gamma & -s\beta \\ -s\alpha c\gamma + s\beta s\gamma c\alpha & s\alpha s\gamma + s\beta c\alpha c\gamma & c\alpha c\beta \end{pmatrix}$$

$$R_Y(\alpha)R_Z(\beta)R_X(\gamma) = \begin{pmatrix} c\alpha c\beta & s\alpha s\gamma - s\beta c\alpha c\gamma & s\alpha c\gamma + s\beta s\gamma c\alpha \\ s\beta & c\beta c\gamma & -s\gamma c\beta \\ -s\alpha c\beta & s\alpha s\beta c\gamma + s\gamma c\alpha & -s\alpha s\beta s\gamma + c\alpha c\gamma \end{pmatrix}$$

$$R_Y(\alpha)R_Z(\beta)R_Y(\gamma) = \begin{pmatrix} -s\alpha s\gamma + c\alpha c\beta c\gamma & -s\beta c\alpha & s\alpha c\gamma + s\gamma c\alpha c\beta \\ s\beta c\gamma & c\beta & s\beta s\gamma \\ -s\alpha c\beta c\gamma - s\gamma c\alpha & s\alpha s\beta & -s\alpha s\gamma c\beta + c\alpha c\gamma \end{pmatrix}$$

$$R_Z(\alpha)R_X(\beta)R_Y(\gamma) = \begin{pmatrix} -s\alpha s\beta s\gamma + c\alpha c\gamma & -s\alpha c\beta & s\alpha s\beta c\gamma + s\gamma c\alpha \\ s\alpha c\gamma + s\beta s\gamma c\alpha & c\alpha c\beta & s\alpha s\gamma - s\beta c\alpha c\gamma \\ -s\gamma c\beta & s\beta & c\beta c\gamma \end{pmatrix}$$

$$R_Z(\alpha)R_X(\beta)R_Z(\gamma) = \begin{pmatrix} -s\alpha s\gamma c\beta + c\alpha c\gamma & -s\alpha c\beta c\gamma - s\gamma c\alpha & s\alpha s\beta \\ s\alpha c\gamma + s\gamma c\alpha c\beta & -s\alpha s\gamma + c\alpha c\beta c\gamma & -s\beta c\alpha \\ s\beta s\gamma & s\beta c\gamma & c\beta \end{pmatrix}$$

$$R_Z(\alpha)R_Y(\beta)R_X(\gamma) = \begin{pmatrix} c\alpha c\beta & -s\alpha c\gamma + s\beta s\gamma c\alpha & s\alpha s\gamma + s\beta c\alpha c\gamma \\ s\alpha c\beta & s\alpha s\beta s\gamma + c\alpha c\gamma & s\alpha s\beta c\gamma - s\gamma c\alpha \\ -s\beta & s\gamma c\beta & c\beta c\gamma \end{pmatrix}$$

$$R_Z(\alpha)R_Y(\beta)R_Z(\gamma) = \begin{pmatrix} -s\alpha s\gamma + c\alpha c\beta c\gamma & -s\alpha c\gamma - s\gamma c\alpha c\beta & s\beta c\alpha \\ s\alpha c\beta c\gamma + s\gamma c\alpha & -s\alpha s\gamma c\beta + c\alpha c\gamma & s\alpha s\beta \\ -s\beta c\gamma & s\beta s\gamma & c\beta \end{pmatrix}$$

Products of Two Rotation Matrices

$$R_X(\alpha)R_Y(\beta) == \begin{pmatrix} c\beta & 0 & s\beta \\ s\alpha s\beta & c\alpha & -s\alpha c\beta \\ -s\beta c\alpha & s\alpha & c\alpha c\beta \end{pmatrix}$$

$$R_X(\alpha)R_Z(\beta) == \begin{pmatrix} c\beta & -s\beta & 0 \\ s\beta c\alpha & c\alpha c\beta & -s\alpha \\ s\alpha s\beta & s\alpha c\beta & c\alpha \end{pmatrix}$$

$$R_Y(\alpha)R_X(\beta) == \begin{pmatrix} c\alpha & s\alpha s\beta & s\alpha c\beta \\ 0 & c\beta & -s\beta \\ -s\alpha & s\beta c\alpha & c\alpha c\beta \end{pmatrix}$$

$$R_Y(\alpha)R_Z(\beta) == \begin{pmatrix} c\alpha c\beta & -s\beta c\alpha & s\alpha \\ s\beta & c\beta & 0 \\ -s\alpha c\beta & s\alpha s\beta & c\alpha \end{pmatrix}$$

$$R_Z(\alpha)R_X(\beta) == \begin{pmatrix} c\alpha & -s\alpha c\beta & s\alpha s\beta \\ s\alpha & c\alpha c\beta & -s\beta c\alpha \\ 0 & s\beta & c\beta \end{pmatrix}$$

$$R_Z(\alpha)R_Y(\beta) == \begin{pmatrix} c\alpha c\beta & -s\alpha & s\beta c\alpha \\ s\alpha c\beta & c\alpha & s\alpha s\beta \\ -s\beta & 0 & c\beta \end{pmatrix}$$