Industrial Robotics M

Dante Piotto

spring semester 2023

Chapter 1

Elements of Kinematics and Dynamics of Industrial Robots

1.1 Introduction

A manipulator may be mechanically described as the interconnection of rigid bodies *(links)* through kinematic pairs *(joints)*. In this course we deal with open chains of links (serial configuration). The first link is the *base*, while the last link is the *end effector*. We will consider the links to be rigid bodies for the sake of simplicity.

A joint is an element that constrains one or more relative motion directions between two links. In robotics only two types of joints are used: rotoidal joints and prismatic joints.

1.1.1 Degrees of Freedom of a Manipulator

Each joint is characterised by the number of independent motion directions allowe between two consecutive links. This number defines the *degrees of freedom* of the joint. Rotoidal and prismatic joints have 1 degree of freedom (dof of the joint. Rotoidal and prismatic joints have 1 degree of freedom (dof). In place of spehrical joints, 3 rotoidal joints are used, forming a *spherical wrist*. The spherical wrist is typically at the end of the kinematic chain. If a joint has k dof, then the relative configuration between two rigid bodies may be expressed as a function of k variables $q_1, q_2, \ldots, q_k$, called *joint variables*.

1.1.2 Joint Space and Work Space

Joint Space: Let us stack all the joint variables in a vector

$$q = \begin{bmatrix} q_1 & q_2 & \dots & q_n \end{bmatrix}^T$$

with $q \in \mathcal{Q} \subset \mathbb{R}^{N_dof}$. The set $\mathcal{Q}$ is called *joint space*

Work Space: The work space is a subset of the Euclidean space $\mathbb{E}$ in which the robot executes its tasks. It is the set of all the points that the mechanical structure may assume, and in general is a 3D subset of $\mathbb{E}$. Each point of the work space is indicated by a vector x of proper dimension, $x \in \mathbb{R}^y$, where usually $y = \{3, 6\}$

configuration of a manipulator: It takes into account both the position and orientation of a reference frame fixed to the manipulator extremity (end effector). Then (locally):

- $x \in \mathbb{R}^3$ in a plane
- $x \in \mathbb{R}^6$ in space

Classification of manipulators: if $\mathbb{R}^n$ is the joint space and $\mathbb{R}^m$ is the work space:

- $n = m$: "normal" cases
- $n < m$: *defective* manipulators
- $n > m$: *redundant* manipualtors

1.1.3 Types of Robotic Manipulators

Common kinematic structures for robotic manipulators:

- Cartesian robots: 3 perpendicular prismatic joints. Simplest robot structure. Structure can be made to be very robust and therefore very large. Usually meant to move large payloads. Simple to control (1 motor for each direction of motion). Simple mapping between joint space and work space.
- Antropomorphic robots (majority): 3 rotational joints, 1 with a vertical axis and 2 with horizontal axes. More dexterous, harder to control, more complex coupling between joint space and work space. Errors on eaech motor stack in the final motion
- SCARA robots: Selective Compliance Assembly Robot Arm. Rigid along z, more compliant along x and y.
- Cylindrical and spherical robots: not used
- Spherical wrists: typically attached at the end of Cartesian, Antropomorphic or SCARA bots.

The first 3 joints define the basic kinematic structure

1.2 Kinematic model of a manipulator

x is made of 3 elements determining position an 3 determining orientation. These 3 represent angles, taht may be Roll-Pitch-yaw, or Euler angles (Z-Y-Z Euler angles are used in robotics)

1.2.1 Forward kinematics

mapping of joint space variables into work space variables

$$x = f(q) \quad q \in \mathbb{R}^n, x \in \mathbb{R}^m$$

It is possible to define different kinematic models for a robotic manipulator, i.e. different $f(q)$ functions. They are however mathematically equivalent

Homogenous trasformation matrix

the configuration of a rigid body b_i in a 3D space can be described by a *homogenous transformation matrix* betweeen a reference frame FF_i and the base frame FF_0 thus constructed:

$${}^0T_i = \begin{pmatrix} {}^0R_i & {}^0o_i \\ \mathbf{0} & 1 \end{pmatrix}$$

where ${}^0R_i$ is a rotation matrix belonging to $SO(3)$, and ${}^0o_i \in \mathbb{R}^3$ is a position vector. It can be noted that the determinant of the homogenous transformation matrix is also equal to 1.

For computational reasons, a new position vector $p = [p_x, p_y, p_z, 1]^T$ is defined. This way, the position ${}^0p$ of a generic point ${}^1p$ of the rigid body can be obtained as:

$${}^0p = {}^0\ T_1\ {}^1p$$

or, with some abuse of notation:

$$\begin{bmatrix} {}^0p \\ 1 \end{bmatrix} = {}^0\ T_1 \begin{bmatrix} {}^1p \\ 1 \end{bmatrix} = \begin{bmatrix} {}^0R_1\ {}^1p + {}^0\ o_1 \\ 1 \end{bmatrix}$$

in general

$${}^0T_n = {}^0\ T_1\ {}^1T_2 \ldots {}^{n-1}\ T_n$$

due to homogenous transformation being a linear operation.

Denavit-Hartenberg convention

- each link is numbered, from 0 to n, so that it is uniquely identified in the mechanical chain: $L_0, L_1, \ldots, L_n$. The base link is conventionally identified as L_0, the distal link as L_n. A manipulator with $n+1$ links has n joints.
- Joints are numbered from 1 to n. J_i connects links L_{i-1} and L_i

- Frames $\mathcal{F}_i$ are associated to each joint, according to the Denavit-Hartenberg procedure, and matrices ${}^{i-1}T_i$ are computed:
  - all the z axes are directed along the direction of motion
  - axis $x+1$ intersects and is perpendicular to axis z_i
- The Denavit-Hartenberg procedure requires 4 parameters for each link $(d_i, \theta_i, a_i, \alpha_i)$

This procedure provides a 4x4 matrix, not a vector, for x. The vector x can be computed as a function of the homogeneous trasformation matrix.

1.2.2 Inverse kinematics

mapping of work space variables into joint space variables

$$q = g(x) = f^{-1}(x) \quad q \in \mathbb{R}^n, x \in \mathbb{R}^m$$

A standard procedure to obtain the inverse kinematic model does not exist. Moreover, we may have

- no solution (if x does not belong to the workspace
- a finite set of solutions
- infinite solutions

A *closed form* solution is preferrable to numerical procedures (better computation, easier to select a particular solution in the set of solutions)
A closed form solution to the inverse kinematic problem may be obtained, if it exists, by trying one of these approaches:

- Algebraic approach: elaboration of the kinematic equatoins in order to obtain a set of 'simple' equations that can be solved in the uknown joint variables q
- Geometric approach: based on considerations dependent on the geometric structure of the manipulator

Algebraic approach

The values of the homogenoeus trasnformation matrix are known, and the homogeneous trasnformation matrix can be written as a function of joint variables. This yields 12 scalar equations (usually more than needed). Hopefully there are a sufficient number of equations that can be solved easily enough. As an additional resource, one can perform algebraic manipulations on the involved matrices to try and find a better set of equations. This procedure is in general not so simple

Geometric approach

Also called Pieper approach.
For several kinematic structures it is possible to apply the *kinematic decoupling* principle, that allows to decompose the overall 6Dof problem into two 3 Dof sub-problems:

- inverse kinematic problem for position
- inverse kinematic problem for orientation

Sufficient condition to solve the inverse kinematic problem for a 6DoF manipulator:

- 3 consecutive rotational joints whose axes intersect in a point (spherical wrist: notice that the position of the center of rotation of the spherical wrist does not depend on the joint variables of the spherical wrist but rather on the other joints)
- 3 consecutive rotation joints with parallel axes

1.2.3 Differential kinematics - the Jacobian

Ohter relations of intrest in robotics are those between:

- end-effector velocity and joint velocities

$$\begin{pmatrix} v \\ \omega \end{pmatrix} \iff \dot{q}$$

- force applied on the environment by the manipulator and corresponding joint torques:

$$\begin{pmatrix} f \\ n \end{pmatrix} \iff \tau$$

These relations are based on a matrix known as the *Jacobian of the manipulator*.
The Jacobian is also used for:

- studying the singularities of the mechanical structures
- defining numerical algorithms to solve the inverse kinematics
- studying manipulability properties

The six-dimensional velocity vector (twist) in the woek-space is defined with three linear and three rotational components

$$\dot{x} = \begin{pmatrix} v \\ \omega \end{pmatrix}$$

Two expressions of the Jacobian matrix can be defined:

- the analytic Jacobian, used when the rotational component in $\dot{x}$ is defined as $\dot{\gamma}$, the time-derivative of a triple of orientation variables
- the geometric Jacobian, used when the rotational component in $\dot{x}$ is defined as ω, the rotational velocity vector.

It is possible to relate the two expressions by means of a proper matrix $T(\gamma)$

$$J_G(q) = T(\gamma)J_A(q) = \begin{bmatrix} I_3 & 0 \\ 0 & \hat{T}(\gamma) \end{bmatrix} J_A(q)$$

Analytic Jacobian

The analytic expression of the Jacobian is obtained by differentiating the six-dimensional vector $x = f(q) = [p^T, \gamma^T]^T$ containing the position (p) and orientation (γ) of the end-effector in the work-space wrtnthe base frame FF0; usually γ are the Euler or RPY angles
By differentiating $f(q)$ we get:

$$dx = \frac{\partial f(q)}{\partial q} dq = J(q)dq \qquad (\dot{x} = J(q)\dot{q})$$

where the $m \times n$ matrix

$$J(q) = \begin{pmatrix} \frac{\partial f_1}{\partial q_1} & \frac{\partial f_1}{\partial q_2} & \cdots & \frac{\partial f_1}{\partial q_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial q_1} & \frac{\partial f_m}{\partial q_2} & \cdots & \frac{\partial f_m}{\partial q_n} \end{pmatrix}, \qquad J(q) \in \mathbb{R}^{m \times n}$$

Is called the *Jacobian matrix*
We have that the relationship between $\dot{x}$ and $\dot{q}$ is linear. The Jacobian can be conceptually split in two blocks, the first concerning the linear part of the velocity, the second concerning the angular part:

$$\dot{x}_A = J_A(q)\dot{q} = \begin{bmatrix} \dot{p} \\ \dot{\gamma} \end{bmatrix} = \begin{bmatrix} J_p \\ J_\gamma \end{bmatrix} \dot{q}$$

Geometric expression

In practice, the Jacobian is always computed in its geometric expression as it is much simpler. The geometric expression of the Jacobian is obtained considering as rotational components of the velocity vector $\dot{x}$ the rotational vector ω and not the derivative of a triple of angles γ

$$\dot{x} = \begin{bmatrix} \dot{p} \\ \dot{\gamma} \end{bmatrix} \quad \Longleftrightarrow \quad \dot{x} = \begin{bmatrix} \dot{p} \\ \omega \end{bmatrix}$$

$$\dot{x}_G = J_G(1)\dot{q} = \dot{x} = \begin{bmatrix} \dot{p} \\ \omega \end{bmatrix} = \begin{bmatrix} J_v \\ J_w \end{bmatrix} \dot{q} = \begin{bmatrix} J_{v_1} & J_{v_2} & \dots J_{v_n} \\ J_{\omega_1} & J_{\omega_2} & \dots J_{\omega_n} \end{bmatrix}$$

this leads to

$$v = J_{v_1}\dot{q}_1 + J_{v_2}\dot{q}_2 + \cdots + J_{v_n}\dot{q}_n$$
$$\omega = J_{\omega_1}\dot{q}_1 + J_{\omega_2}\dot{q}_2 + \cdots + J_{\omega_n}\dot{q}_n$$

it is possible to show, applying the superimposition principle by considering only the $i-th$ joint to be moving, that the generic $i-th$ column of the geometric Jacobian is defined as

$$\begin{bmatrix} J_{vi} \\ J_{\omega i} \end{bmatrix} = \begin{bmatrix} {}^0z_{i-1} \times ({}^0p_n - {}^0p_{i-1}) \\ {}^0z_{i-1} \end{bmatrix} \qquad \text{rotational joint}$$

$$\begin{bmatrix} J_{vi} \\ J_{\omega i} \end{bmatrix} = \begin{bmatrix} {}^0z_{i-1} \\ 0 \end{bmatrix} \qquad \text{rotational joint}$$

Once the forward kinematics of the manipulater is know these expressions are simple to obtain as the vectors ${}^0p_i$ and ${}^0z_i$ are obtained from the matrices ${}^0T_i$. This does not represent a computational burden with respect to the computation of the forward kinematic model.

Force domain

By exploiting the *virtual work principle*, a mapping similar to that in the velocity domain can be obtained for the force domain.
Because work is invariant wrt the frame in which it is computed:

$$w^T dx = \tau^T dq$$

where $w = (f^T n^T)^T$ is a 6-dimensional vector (*wrench*) collecting the linear forces f and the torques n applied to the manipulator. τ is the n-dimensional vector of the *joint forces/torques* since

$$dx = J(q)dq$$

we obtain by substituting

$$\tau = J^T(q)w$$

Singularities

When the Jacobian is rank deficient, there are directions in $\mathbb{R}^6$ that the robot cannot move in. Furthermore, limited velocities of the end-effector may correspond to infinite veolocities in the joint space. There is no well defined solution to the inverse kinematic problem: either no solution or infinite solutions. Singularities typically correspond to positions at the border of the workspace.

Inverse differential kinematics

The forward mapping betweeen joint and work-space is given by

$$\dot{x} = J(q)\dot{q}$$

Then, the inverse mapping is given as the solution of a linear algebraic equation

- $m = n$: if the manipulator is not in a singular configuration, it is possible to use the inverse of the jacobian:

$$\dot{q} = J^{-1}(q)\dot{x}$$

- $m \neq n$: the Moore-Penrose pseudoinverse of the Jacobian is used:

$$\dot{q} = J^{\dagger}(q)\dot{x}$$

if $m < n$ (right pseudoinverse):

$$J^{\dagger} = J^T(JJ^T)^{-1}$$
$$J^{\dagger} = I_m$$

if $m > n$ (left pseudoinverse):

$$J^{\dagger} = (J^T J)^{-1}J^T$$
$$J^{\dagger} = I_n$$

When the joint space is larger than the work space ($n > m$), the solution of $\dot{x} = J\dot{q}$:

- $rank(J) = min(m, n) = m \quad \implies \mathcal{R}(J) = \mathbb{R}^m$
- $\forall \dot{x}, \exists \dot{q}$ such that $\dot{x} = J\dot{q}$ more than one exists
- $\dot{q} = J^{\dagger}\dot{x} + \dot{q}_N \implies \dot{x} = J(J^{\dagger}\dot{x} + \dot{q}_n = \dot{x} \quad \forall \dot{q}_N in \mathcal{N}(J)$
  $\implies \dot{q} = J^{\dagger}\dot{x} + (I - J^{\dagger}J)y$ is a general expression of the solution
  $(I - J^{\dagger}J)$ is a base of $\mathcal{N}(J)$
- $\dot{q} = J^{\dagger}\dot{x}$ has *minimum norm*

Inverse kinematics with the Jacobian

The Jacobian may be used to solve the inverse kinematic problem if a closed form solution is not available. A method could be to compute

$$q_{k+1} = q_k + J^{-1}(q_k)v_k dt$$

where a numerical integration of the position is performed. Unfortunately, this method is subject to numerical drifts and initialization problems, and it is therefore difficult to obtain the desired solution.

An alternative method is based on a feedback loop, by defining an error in the work-space. Given

$$e = x_d - x$$

by differentiating, we obtain

$$dote = \dot{x}_d - J(q)\dot{q}$$

it is necessary to define $\dot{q}$ in such a way that the error e converges to 0

Algorithm 1

if

$$\dot{q} = J^{-1}(q)(\dot{x}_d + Ke)$$

with $K = K^T > 0$, then

$$\dot{e} = \dot{x}_d - JJ^{-1}(\dot{x}_d + Ke)$$
$$\dot{e} = -Ke$$

Typically K is chosen as a diagonal matrix, leading to

$$\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \vdots \\ \dot{e}_n \end{bmatrix} = \begin{bmatrix} K_1 & \cdots & & \\ \vdots & K_2 & & \\ & & \ddots & \\ & & & K_n \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}$$

Which is completely decoupled and converging to 0.

Algorithm 2

We use a Lyapunov technique:
Given $\dot{x}_d = 0$, if

$$\dot{q} = J^T(q)Ke$$

with $K = K^T > 0$, then

$$V(e) = \frac{1}{2}e^T Ke \implies \dot{V}(e) = e^T K\dot{e} = e^T K(\dot{x}_d - J\dot{q})$$
$$= -e^T KJ(q)\dot{q}$$

we choose $\dot{q} = J^T Ke$, leading to:

$$\dot{V}(e) = -e^T KJJ^T Ke \leq 0$$

1.2.4 Manipulability Measures

The Jacobian may be used to evaluate the achievable performaces of a manipulator, in terms of velocities and applicable forces, through the *manipulability ellipsoids*

Velocity manipulability ellipsoids

Given a sphere with unit radius radius in the joint velocity space, $\dot{q}^T\dot{q} = 1$, that may be considered as a"cost" (energy input to the manipulator), we want to obtain its equivalent expression in the work-space. From $\dot{x} = J(q)\dot{q}$ we obtain:

$$\begin{aligned} \dot{q} &= J^{\dagger}\dot{x} \\ \implies \dot{x}J^{\dagger T}J^{\dagger}\dot{x} &= 1 \\ \implies \dot{x}^T(JJ^T)^{\dagger}\dot{x} &= 1 \end{aligned}$$

which is an ellipsoid in the work-space $\mathbb{R}^m$

- the principal axes are directed along the eigenvectors of JJ^T
- the length of the principal axes are given by the singular values of J, $\sigma_i = \sqrt{\lambda_i(JJ^T)}$

Force manipulability ellipsoids

Let us now consider the equation $\tau^T\tau = 1$ from which we obtain:

$$w^T JJ^T w = 1$$

This equation describes an ellipsoid in the $\mathbb{R}^m$ force space

- the principal axes are directed along the eigenvectors of JJ^T
- the length of the principal axes are given by the inverse of the singular values of J, $\sigma_i = \sqrt{\lambda_i(JJ^T)}$

This result confirms the duality of the velocity/force spaces: along the directions in which high velocity performances are possible, low force performances are achievable and viceversa

Chapter 2

Dynamic Model of Robotic Manipulators

Dymanics of a robot manipulator: analysis of the relationship between the applied forces/torques and the resulting motion of a robot. For the dynamics, it is possible to define two "models":

- Direct model: goven the forces/torques applied at joints, the hoint position and velocity, compute the acceleration

  $$\ddot{q} = f(q, \dot{q}, \tau)$$

  and then

  $$\dot{q} = int\ddot{q}dy \qquad q = \int qdotqdt$$

- Inverse model: given the desired joint acceleration, velocity and position, compute the corresponding force/torque to be applied

  $$\tau = f^{-1}(\ddot{q}, \dot{q}, q) = g(\ddot{q}, \dot{q}, q)$$

In deriving the dynamic model of a robot, we refer to a chain of rigid bodies mutually, and ideally, connected among them The dynamic model of a robot is used for:

- simulation: testing motion behaviour without resorting to physical experiments
- analysis and synthesis of control laws
- alaysis of the structural properties of a manipulator in the design phase

2.1 Euler-Lagrange model

The dynamic model obtained through this method is simpler and more intuitive, and also better suited to understanding the effects of changes in the mechanical parameters. The links are considered altogether, and the model is obtained analytically. Drawbacks: the model is obtained starting from the kinetic and potential energies (non intuitive) and the model is not computationally efficient.

Lagrange coordinates: independet variables used to describe the position of rigid bodies in space. For the same physical system, several choices of Lagrangian coordinates are usually possible. In robotics, we use *joint variables* $q_1, q_2, \ldots, q_n$. Input torques should be defined according to the chosen Lagrangian coordinates $\implies$ joint torques $\tau_1, \tau_2, \ldots, \tau_n$

From physics, we know it is possible to define:

- the Kinetic Energy function $K(q\dot{q})$
- the Potential Energy function $P(q)$

And therefore the Lagrangian function

$$\mathcal{L}(q\dot{q}) = K(q, \dot{q}) - P(q)$$

The EEuler-Lagrange equations for a system described by n Lagrange coordinates q_i are

$$\psi_i = \frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right) - \frac{\partial \mathcal{L}}{\partial q_i} \qquad i = 1, \ldots, n$$

Being ψ_i the non-conservative (external or dissipative) generalized forces performing work on q_i. In robotics, we consider:

- τ_i joint actuator torque
- $[J^T F_c]_i$ term due to external (contact) forces
- $d_{ii}\ \dot{q}_i$ joint friction torque

Therefore:

$$\psi_i = \tau_i + [J^T F_c]_i - d_{ii}\dot{q}_i$$

It can be noted that

$$K = \sum_{i=1}^{n} K_i \qquad P = \sum_{i=1} nP_i$$

2.1.1 obtained model

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + D\dot{q} + g(q) = \tau$$

Non-linear wrt the state. It can be made linear wrt a set of parameters α

2.1.2 Identification of the dynamic parameters

From:

$$Y(q, \dot{q}, \ddot{q})\alpha = \tau$$

by measuring along a proper trajectory the quantities $q, \dot{q}, \ddot{q}, \tau$ ($N > p$ samples) one obtains:

$$\tau_0 = Y_0\alpha$$
$$Y_0^T\tau_0 = Y_0^T Y_0\alpha$$
$$\alpha = (Y_0^T Y_0)^{-1} Y_0^T \tau_0 = Y_0^\dagger \tau_0$$

2.1.3 Dynamic model in simulink

When only one of the two of the joints has friction energy still leaks out of the system because the system is coupled so movement of each link affects the other link causing the energy to be lost.

2.1.4 Linearization of the Euler-Lagrange model

By neglecting friction effects ($D = 0$), and considering the vector $C(q, \dot{q})\dot{q} = c(q, \dot{q})$, the non linear dynamic model of a robot manipulator is:

$$M(q)\ddot{q} + c(q, \dot{q}) + g(q) = u$$

It is possible to obtain a linear dynamic model which is valid only locally around a ginven operative condition (not used, very bad, uselss, stupid, why even bother):

- linearization around a (constant) equilibrium state
- linearization along a (nominal, time-varying) equilibrium trajectory

skipped because excessively useless and pointless

2.1.5 Symbolic computation of the dynamic model

The Euler-Lagrange formulation of the dynamic model of a robot arm can be expressed in a form suitable to be implemented in a symbolic package (Mathematica or Matlab Symbolic toolbox) in order to obtain, given the kinematic and mechanical parameters of each link/joint, the model expressed in a symbolic form. Parameters needed: DH parameters, masses, inertia matrices, position of centers of mass.

2.2 Newton-Euler model

Based on a computationally efficient recursive technique that expoits the serial structure of an industrial manipulator. On the other hand, the model is not expressed in closed form. The model is based on the balance of forces and torques acting on each link.

The dynamic equations are written separately for each link and a *recursive formulation* is obtained: forward recursion (from Link 0 to Link n) for the computation of the "propagation" of accelerations and velocities, backward recursion (from Link n to Link 0) for the computation of the "propagation" of forces and torques. The basic equations are:

- Newton's second law of motion: (conservation of momentum mv)

$$f = \frac{d(mv)}{dt}$$

  where m is mass, v is the linear velocity, f the sum of all the applied external forces and mv the momentum. Since in robotics the masses of the links are constant (and considered as concentrated in the center of mass) we have the well known Newton equation $f = ma = m\ddot{p}_C$

- Euler's law of motion(conservation of the angular momentum $L = I\omega$)

$$\ni = \frac{d({}^0I^0\omega}{dt}$$

  where ${}^0I$ is the moment of inertia (rotational inertia) expressed in an inertial frame FF0 placed in the center of mass, ${}^0\omega$ the rotational veolcity, τ the sum of the torques applied to the rigid body.

Let us consider the generic $i-th$ link of the manipulator. Define, in an inertial frame, the following parameters: NOTE: write them at some point please (slide 102)

m_i	mass of the link	$\ddot{p}_{C_i}$	acceleration of C_i
I_i	rotational inertia	$\ddot{\omega}_i$	acceleration of C_i
R_{i-1,C_i}	vector from FFi-1 to the CoM C_i	g_i	gravity acceleration
r_{i,C_i}	vector from FFi tot he Com C_i	f_i	force applied on L_i by L_{i-1}
$r_{i-1,i}$	vector from FFi-1 to FFi	$-f_{i+1}$	force applied on L_i by L_{i+1}
$\dot{p}_{C_i}$	linear velocity of C_i	ν_i	torque applied on L_i by L_{i-1} wrt FFi-1
$\dot{p}_i$	linear velocity of FFi	$-\nu_{i+1}$	torque applied on L_i by L_{i+1} wrt FFi-1
ω_i	rotational velocity of the link		

Therefore, for each link we have:
Newton law:

$$f_i - f_{i+1} + m_i g = m_i \ddot{p}c_i$$

Euler law:

$$\nu_i + f_i \times r_{i-1,C_i} - \nu_{i+1} - fi + 1 \times r_{i,C_i} = \frac{d}{dt}(I_i\omega_i)$$

The Euler law results:

$$\begin{aligned}\nu_i + f_i \times r_{i-1,C_i} - \nu_{i+1} - fi + 1 \times r_{i,C_i} = \frac{d}{dt}(I_i\omega_i) = I_i\dot{\omega}_i + \frac{d}{dt}(R_i^i I_i R_i^T)\omega_i \\ = I_i\dot{\omega}_i + (\dot{R}_i^i I_i R_i^T + R_i^i I_i \dot{R}_i^T)\omega_i \\ = I_i\dot{\omega}_i + S(\omega_i)(R_i^i I_i R_i^T(\omega_i) + R_i^i I_i R_i^T S(\omega_i)\omega_i \\ = I_i\dot{\omega}_i + \omega_i \times (I_i\omega_i) \qquad [S(\omega_i)\omega_i = \omega_i \times \omega_i = 0]\end{aligned}$$

slide 104-105-106-107 couldn't keep up
After forward and backward recursion, we obtain forces/torques applied on the links. We require torques to be applied by the motors. The generalized force τ_i applied on th $i-th$ joint by the link L_{i-1} can be computed as the projection of f_i or ν_i along the z_{i-1} axis. We have:

$$\tau_i = \nu_i^T z_{i-1}$$
$$\tau_i = f_i^T z_{i-1}$$

Friction may be considered at this point, for example

$$\tau_i = \nu_i^T z_{i-1} + \mu_i \dot{q}_i$$
$$\tau_i = f_i^T z_{i-1} + \mu_i \dot{q}_i$$

Overall algorithm

1. given joint poistions, velocities and accelerations link velocities and accelerations are computed through forward recursion
2. forces and torques are computed through backward recursion
3. joint torques are computed through force/torque projection

This algorithm allows for a variety of useful outputs: slide 113

2.3 Dynamic model in the work-space

It may be of interest to obtain the dynamic model in the work-space and not in the joint-space. For the sake of simplicity, let us consider a 6 dof manipulator, not in a singualr configuration. Since $M(q)\ddot{q}+C(q,\dot{q})\dot{q}+D\dot{q}+g(q) = \tau+J(q)^T F_a$, then

$$\ddot{q} = M^{-1}(q)\left[\tau + J^T(q)F_a - C(q,\dot{q})\dot{q} - D\dot{q} - g(q)\right]$$

Moreover, by deriving $\dot{x} = J(q)\dot{q}$ with respect to time, we have:

$$\ddot{x} = J(q)\ddot{q} + \dot{J}(q)\dot{q}$$

Therefore

$$\hat{M}(x)\ddot{x} + \hat{C}(c,\dot{x})\dot{x} + \hat{D}\dot{x} + \hat{g}(x) = F + F_a \quad \tau = J^T F$$

with

$$\hat{M} = (JM^{-1}J^T)^{-1}$$
$$\hat{C} = \hat{M}$$

Chapter 3

Control of robot manipulators

3.1 Nominal case

All the parameters are known exactly and they are assumed to be constant in time.
Control problem: definition of the input signals for the joints (e.g. torques or actuator input voltages) in order to achieve a predefined behaviour for the manipulator. The achievable performances can vary due to:

- the many control techniques available to solve such a problem
- the hardware used to implement the control algorithms
- the mechanical configuration of the robot (anthropomorphic, cartesia, ...)

The robot performances are mainly influeneced by the mechanical design and by the actuation system. For example, gearboxes "decouple" the load from the motor (disturbances on the load are diminished towards the motor through the gearbox), however they usually introduce non linear effects such as dead-zones, friction, elasticity,...; Direct Drive motors (can provide high torque with low speed without reduction gears) ensure better performances and do not introduce non-linearities in the transmission chain; however a more relevant dynamic coupling between joints is present.

Control problems:

- Control of the robot's motion (position control schemes)
  - joint-space control
  - workspace control
- Control of the interaction with the workspace (force control schemes)

Control schemes:

- Decentralized (or independent) control schemes (SISO)
- Centralized control schemes (MIMO)

Dynamic model of a manipualator:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + D(\dot{q} + g(q) = \tau + J^T(q)F_a$$

Control problem: define the generalized forces τ to be applied to the joints in order to obtain a desired trajectory $q_d(t)$
In non-linear control schemes it is best to choose a sampling frequency as large as possible

3.2 Decentralized position control

Actuators: velocities $\omega_m(t)$, positions q_m, torques $\tau_m(t)$
Gearboxes: reduction ratio K_r
Joints: velocities $\omega(t)$, positions q, generalized forces $\tau(t)$

$$K_r\omega = \omega_m \quad K_r q = q_m \quad \tau_m = K_r^{-1}\tau$$

K_r is a diagonal matrix with elements $\gg 1$ we can observe that:

$$q0K_r^{-1}q_m$$
$$\dot{q} = K_r^{-1}\dot{q}_m$$
$$\ddot{q} = K_r^{-1}\ddot{q}_m$$
$$\tau = K_r\tau_m$$

we get

$$M(q)K_r^{-1}\ddot{q}_m + CK_r^{-1}\dot{q}_m + DK_r^{-1}\dot{q}_m + g(q_m) = K_r\tau_m$$

and therefore

$$K_r^{-1}M(q)K_r^{-1}\ddot{q}_m + K_r^{-1}CK_r^{-1}\dot{q}_m + K_r^{-1}DK_r^{-1}\dot{q}_m + K_r^{-1}g(q_m) = K_r\tau_m$$

The diagonal of the matrix $M(q)$ is composed by two kinds of elements

- inertia terms that do not depend on the robot's configuration
- terms that depend on the robot's configuration

Therefore:

$$M(q) = \bar{M} + \Delta M(q)$$

where $\bar{M}$ is a diagonal matrix with constant elements (i.e. the mean values of the joints inertia). From the robot dynammic model it follows that

$$\tau_m = (K_r^{-1}\bar{M}K_r^{-1})\ddot{q}_m + D_m\dot{q}_m + d$$

having renamed

$$D_m = K_r^{-1} D K_r^{-1}$$
$$d = (K_r^{-1}\Delta M(q)K_r^{-1})qddotq_m + (K_r^{-1}C(q,\dot{q})K_r^{-1})\dot{q}_m + K_r^{-1}g(q_m)$$

D_m is a matrix collecting the motors friction coefficients, d is a term that can be considered as a disturbance that collects nonlinearities and coupling effects. The disturbance terms are divided by K_r^2 and therefore greatly reduced (the gravity is only divided by K_r) Based on this we can design a linear controller, with the hope that d may be small enough. d grows with the speed and acceleration of the robot.

How to reduce the effects of the disturbance d:

- high gain
- integral action to annhiliate the steady state error (e.g. the effect of the gravitational term)

Possible solution: PI controller

$$C(s)0K_c\frac{1+sT_c}{s}$$

3.2.1 possible control scheme

A cascade configuration can be employed If inertia is constant, current control implies acceleration control as torque is directly proportional to acceleration.
In robotics (and in general in motion control systems), feedwarward control actions are usually adopted with very positive results:

- velocity ff action
- acceleration (torque) ff action

With ff actions, it is possible to compensate (partially) also some parts of the "distrubance" term $d(t)$. Since $d(t)$ is known from the dynamic model, given the desired position, velocity and acceleration signals it is possible to compute a ff control action to compensate for $d(t)$:

$$d_d = (K_r^{-1}\Delta M(q_d)K_r^{-1})qddotq_{md+}(K_r^{-1}C(q_d,\dot{q}_d)K_r^{-1})\dot{q}_{md} + K_r^{-1}g(q_{md})$$

It is to be noted that in any case $d \neq d_d$

3.3 Centralized control

We introduce two centralized control schemes:

3.3.1 PD + gravity compensation

Given a desired reference configuration q_d, the goal is to define a controller ensuring the global asympototic stability of the nonlienar dynamical system described by:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + D(\dot{q} + g(q) = \tau + J^T(q)F_a$$

For this purpose, let us define the error as

$$\tilde{q} = q_d - q$$

and consider a dynamic system with state x given by

$$x = \begin{bmatrix} \tilde{q} \\ \dot{q} \end{bmatrix}$$

The *direct Lyapunov method* is exploited for the control law definition. Let us consider the following candidate Lyapunov function:

$$V(x) = V(\tilde{q},\dot{q}) = \frac{1}{2}\dot{q}^T M(q)\dot{q} + \frac{1}{2}\tilde{q}^T K_P \tilde{q} > 0 \quad \forall \dot{q}, \tilde{q} \neq 0$$

Where K_P is a square $(n \times n)$ positive definite matrix. Usually, it is chosen as a diagonal matrix. Function $V(\tilde{q},\dot{q})$ is composed by two terms, the first expressing the kinetic energy of the system, the second cane be interpreted as eleastic energy stored by springs with stiffness K_P. Tehse springs are a physical interpretation of the proportional control.

We compute the time derivative of the candidate Lyapunov function:

$$\begin{aligned} \dot{V}(x) &= \dot{q}^T M\ddot{q} + \frac{1}{2}\dot{q}^T\dot{M}\dot{q} - \dot{q}^T K_P\tilde{q} \\ &= \dot{q}^T(u - C\dot{q} - D\dot{q} - g(q)) + \frac{1}{2}\dot{q}^T\dot{M}\dot{q} - \dot{q}^T K_P\tilde{q} \\ &= \frac{1}{2}\dot{q}^T(\dot{M} - 2C)\dot{q} - \dot{q}^T D\dot{q} + \dot{q}^T(u - g(q) - K_P\tilde{q}) \end{aligned}$$

note that $\frac{1}{2}\dot{q}^T(\dot{M} - 2C)\dot{q} = 0$ due to the choice of C(Christoffel symbols. Also, by choosing

$$u = g(q) + K_P\tilde{q} + K_D\dot{q}$$

where K_D is a positive definite matrix, we obtain a proportional and derivate control with gravity compensation, and we have:

$$\dot{V}(x) = -\dot{q}^T D\dot{q} - \dot{q}K_D\dot{q}$$

note that the term introduced by the derivative control acts like a friction, bringing the energy of the system towards 0 faster.

The PD control scheme can be interpreted as a spring and a damper.

$$\dot{V}(x) = -\dot{q}^T(D + K_D)\dot{q}$$

So far, we have guaranteed that the system stops, but we do not know where.

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + D(\dot{q} + g(q) = K_P\tilde{q} - K_D\dot{q} + g(q)$$

we observe that all the terms other than the one with $\tilde{q}$, in steady state cancel out:

$$K_P\tilde{q} = 0$$

That is, as K_p is positive definite

$$q = q_d$$