Robot Mechanics — Lecture Notes

Sommersemester 2026

Jun.-Prof. Dr. Edoardo Milana
Laboratory for Soft Machines, IMTEK, Universität Freiburg

2026-06-10

Table of contents

1 1. Introduction

1.1 What is a Robot?

The word *robota* (meaning "work" or "forced labour" in Slavic languages) entered engineering vocabulary through Karel Čapek's 1920 play *Rossum's Universal Robots* (R.U.R.), in which artificial human-like creatures are built as cheap workers. Isaac Asimov later formalised behavioural rules in *I, Robot* (1950) — the **Three Laws of Robotics**:

1. A robot may not injure a human being or, through inaction, allow a human being to come to harm.
2. A robot must obey orders given to it by human beings, except where such orders would conflict with the First Law.
3. A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.

A working engineering definition: **Robotics is the science studying the intelligent connection between perception and action** (Siciliano et al.). More broadly, robotics is concerned with machines that can replace human beings in the execution of a task, as regards both physical activity and decision-making.

1.2 Robot Systems

Every robot, regardless of application, is built from four interacting subsystems.

1.2.1 Mechanical system

The essential component is the **mechanical system**, which provides the physical structure: a locomotion apparatus (wheels, crawlers, legs) and/or a manipulation apparatus (arms, end-effectors, hands). Its design draws on articulated and continuum mechanics and materials engineering.

1.2.2 Actuation system

The **actuation system** animates the mechanical structure. It converts stored energy into motion and force. The two dominant technologies are:

- **Electric actuators** (DC motors, brushless motors): compact, precise, easily controlled; the standard choice for manipulation.
- **Hydraulic actuators**: very high force and power density; used when large forces are needed (e.g. heavy-duty industrial arms, legged robots).

The design of an actuation system falls within motion control: servomotors, drives, and transmissions.

1.2.3 Sensory system

The **sensory system** gives the robot perception. Sensors fall into two classes:

- **Proprioceptive sensors** measure the robot's own internal state — joint encoders, IMUs, motor current.
- **Exteroceptive sensors** measure the environment — force/torque transducers, lidar, stereo cameras, tactile arrays.

Realising a sensory system involves signal conditioning, data processing, and information retrieval.

1.2.4 Control system

The **control system** connects perception to action. It executes a task plan, closes feedback loops around the sensors, and compensates for modelling uncertainties:

> *"The realisation of such a system follows the same feedback principle devoted to control of human body functions, possibly exploiting the description of the robotic system's components (physical model)."* — Siciliano et al.

The control problem draws on cybernetics, artificial intelligence, and computational architecture.

The diagram below summarises the relationship between the four subsystems and the course scope:

Table 1.1: Robot subsystems and course scope

Subsystem	Key discipline	This course?
Mechanical	Articulated mechanics	
Actuation	Motion control	(modelling)
Sensory	Signal processing	(strain sensing)
Control	Feedback control	—

1.3 Robot Mechanics

Kinematics describes the motion of a robot with respect to a fixed reference frame, ignoring the forces and moments that cause the motion. Two problems are central:

- **Direct kinematics:** given joint positions, determine the end-effector position and orientation.
- **Inverse kinematics:** given a desired end-effector pose, find the joint positions that achieve it.

Differential kinematics extends this to velocities: the **Jacobian** matrix maps joint velocities to end-effector velocities. The Jacobian also connects end-effector forces to joint torques (statics) and is the foundation for dynamics and control.

1.4 Robot Planning

Planning specifies what motion the robot should execute. Two levels are distinguished:

- **Trajectory planning:** generate time-parameterised joint or end-effector trajectories from a concise task description (e.g. pick-up and release points, or a continuous path).
- **Motion planning:** find collision-free paths in a workspace with obstacles, formulated in configuration space. Solution methods include exact, probabilistic (e.g. RRT, PRM), and heuristic algorithms.

1.5 Robot Control

The trajectories generated by planning are reference inputs to the **motion control system**. The control problem is to compute the forces and torques that the joint actuators must deliver so that the robot tracks those references.

Because a manipulator is an articulated system, the motion of one link influences all others — the equations of motion contain **coupling dynamic effects** among the joints. These cannot be compensated from the model alone; feedback loops (using proprioceptive sensors) are essential to meet accuracy requirements.

1.6 Industrial Robots

1.6.1 Structured environments

Industrial robotics operates in **structured environments** whose geometric and physical characteristics are mostly known in advance. This limits the autonomy required and enables the design of highly accurate, repeatable machines.

Industrial robot manipulators are valued for their **versatility, adaptability, accuracy, and repeatability**. The first commercially deployed robot was the **Unimate** (1961), a hydraulic arm installed on a General Motors assembly line for die-casting operations.

1.6.2 Industrial applications

Typical industrial uses include manipulation (pick-and-place, machine feeding), assembly and packaging, spray painting and coating, arc and spot welding, laser cutting and welding, gluing and sealing, and mechanical machining (milling, drilling, deburring, grinding).

1.6.3 End-effectors

The **end-effector** is the part of the robot that interacts with the environment, typically mounted at the end of the kinematic chain. Its nature depends entirely on the task. For manipulation the most common end-effectors are **grippers** — fingered mechanical grippers or vacuum (suction-cup) devices. Specialised end-effectors include welding torches, spray nozzles, and force-controlled machining spindles.

1.6.4 Market trends

Global yearly installations of industrial robots have grown steadily, with the electronics and automotive sectors historically dominant. Recent growth is being driven by new sectors such as food handling, logistics, and e-commerce fulfilment.

1.7 Service Robots

Industrial robots perform specific, repetitive tasks with extreme precision. They are typically fixed to the ground, operate at high power, are isolated from human workers by safety cages, and require low autonomy.

Service robots, by contrast, perform professional or personal tasks in collaboration with humans and in unstructured environments. They must move (mobile robots), manipulate gently, be energy-efficient, and operate with high autonomy.

1.7.1 Application domains

Service robotics spans a wide range of domains:

- **Space robotics:** rovers (e.g. Perseverance on Mars) combine autonomy with remote control and must withstand extreme temperatures and radiation. Sending robots is far cheaper than sending humans and they can work continuously without life-support.
- **Underwater robotics:** ROVs and AUVs map ocean floors, monitor marine ecosystems, protect subsea infrastructure, and support naval operations. Soft robotic fish exploit passive compliance to swim efficiently using body deformation.
- **Medical robotics:** includes bionic limbs, rehabilitation exoskeletons, nursing assistants, and surgical robots. The **da Vinci Surgical System** is the benchmark for minimally invasive surgery — surgeons operate through a console, with the robot providing tremor filtering, motion scaling, and improved dexterity.
- **Field robotics:** mine exploration, de-mining, civil and naval construction, inspection and surveillance, fire-fighting, emergency rescue.
- **Home and service:** domestic cleaning (Roomba), lawn mowing, museum guides, agriculture, food industry.

1.7.2 Collaborative robots (cobots)

Cobots are designed for direct human-robot interaction, unlike traditional industrial robots. Key requirements:

- **Safety and flexibility** over raw speed and force.
- **Mechanical compliance** — more important than motion accuracy. Compliance can be physical (springs, compliant materials) or algorithmic (torque/force control via joint sensors).
- Lightweight materials, rounded edges, force-limited drives, and low operating speeds reduce inertial hazards.

Commercial examples include Universal Robots (UR series), ABB GoFa, and **Franka Robotics** — a 7-DOF arm with torque sensors at every joint, enabling force-sensitive manipulation and safe physical interaction.

1.7.3 Robotic hands

Anthropomorphic end-effectors for dexterous manipulation or active prostheses. Robotic hands are equipped with tactile sensors to improve human-like grasping and are typically tendon-driven (motors placed in the forearm or palm to reduce distal inertia). The **Shadow Hand** has 20 DOF driven by 20 motors.

Underactuated hands use fewer motors than DOF by harnessing passive mechanical compliance. The **Pisa/IIT SoftHand** actuates 19 DOF with a single motor, relying on adaptive synergies to conform to object shapes during grasping.

1.7.4 Legged robots

Legged robots use articulated limbs for locomotion, inspired by animal biomechanics (biomimetics). Advantages over wheeled platforms: greater versatility across uneven terrain, stairs, and obstacles. The cost is complex balance control and higher energy consumption.

- **Hexapod robots:** six legs; statically stable (centre of mass always supported by a triangle of feet), simpler balance control.
- **Quadruped robots:** four legs; dynamically stable gaits (e.g. Boston Dynamics Spot); widely used for inspection and surveillance.

1.7.5 Humanoids

The humanoid form factor is motivated by the vision of **general-purpose robots** operating in a world designed for humans — using the same tools, doors, staircases, and workspaces. However, bipedal locomotion is dynamically unstable and demands sophisticated balance and whole-body control.

Building a humanoid requires integrating all robot subsystems at high performance:

Table 1.2: Components of a humanoid robot

Subsystem	Key technologies
Actuation	Electric motors, gears, springs (series elastic actuators)
Sensing	Encoders, IMUs, force/torque sensors, cameras, lidar

Subsystem	Key technologies
Mechanics	Lightweight links, compliant joints, dexterous hands
Control	Whole-body control, balance, planning algorithms
Power	High-energy-density battery packs

1.8 Scope of These Notes

This course focuses on **Robot Mechanics** — the kinematic and static analysis of manipulator structures. The thread running through the material is the **direct kinematics map**

$$x = h(q),$$

which relates the **joint variables** q (the configuration in *joint space*) to the **operational variables** x (the end-effector pose in *task space*). The notes cover how to build this map, how to differentiate and invert it, and how to ground it in physical actuators.

Table 1.3: Course roadmap

Topic	Question answered
Rigid-body pose & rotation matrices	How is orientation represented?
Euler angles, axis/angle, quaternions	What are alternative orientation representations?
Homogeneous transformations	How are position and orientation combined?
Denavit–Hartenberg convention	How is a chain of links parameterised?
Direct kinematics	Where is the end-effector given the joints?
Inverse kinematics	What joints place the end-effector at a target?
Differential kinematics (Jacobian)	How do joint *rates* map to task *rates*?
Statics	How do end-effector wrenches map to joint torques?
Mobility (Grübler)	How many degrees of freedom does a mechanism have?
Actuators & gears	How is motion produced and transmitted?
Strain sensing	How is force/torque measured in practice?

2 2. Robot Components

Before developing the mathematical theory of kinematics, it is useful to survey the physical building blocks of a manipulator. Every robot integrates four classes of components:

- **Mechanical units** — rigid links connected via joints; supporting structure (mobility), wrist (dexterity), end-effector (task execution).
- **Actuation units** — motors (electrical, hydraulic, pneumatic) and transmissions; motion control algorithms.
- **Sensor units** — proprioceptive (joint position and velocity) and exteroceptive (force, proximity, vision).
- **Supervision units** — task planning and control; artificial intelligence and reasoning.

2.1 Mechanical Structure

2.1.1 Links and joints

The individual bodies making up a mechanism are called **links**. Links are treated as rigid bodies to simplify modelling. At high speeds or heavy loads, material elasticity may become significant; in that case flexibility must be accounted for. Chains, cables, and belts may also be considered links.

The connection between two links is a **joint**, which constrains the relative motion between the connected members. A joint with f_i degrees of freedom imposes $c_i = m - f_i$ constraints, where m is the DOF of a free rigid body ($m = 3$ in 2D, $m = 6$ in 3D).

Joint types and their DOF:

Table 2.1: Joint types and degrees of freedom

Joint	Symbol	DOF f_i	Motion allowed
Revolute (hinge)	R	1	Rotation about a fixed axis
Prismatic (sliding)	P	1	Translation along a fixed axis

Joint	Symbol	DOF f_i	Motion allowed
Helical (screw)	H	1	Coupled rotation + translation
Cylindrical	C	2	Independent rotation + translation on one axis
Universal	U	2	Two orthogonal revolute axes
Spherical (ball-and-socket)	S	3	Rotation about any axis through the centre

2.1.2 Kinematic chains

A **kinematic chain** is a sequence of links connected by joints. Two topologies:

- **Open chain (serial):** a single sequence of links from base to end-effector. Each link connects to at most two neighbours.
- **Closed chain (parallel):** a loop of links; the end-effector is connected to the base by more than one path.

A kinematic chain becomes a **mechanism** when one link is fixed to the ground. A mechanism becomes a **machine** when an energy source (motor, hydraulic circuit) is added so that it can perform useful work.

2.1.3 Configuration space and degrees of freedom

The **configuration** of a robot is a complete specification of the position of every point of the robot. For a rigid-link robot, a finite number of coordinates suffices. The minimum such number is the **degrees of freedom (DOF)**:

> *The DOF is the dimension of the configuration space (C-space) — the set of all possible configurations of the robot.*

Examples: a door has DOF = 1 (the hinge angle θ); a rigid body on a plane has DOF = 3 (x, y, θ).

2.1.3.1 Grübler's formula

For a mechanism with N links (including the base), J joints, and joint freedoms f_i:

$$\boxed{\text{dof} = m(N - 1 - J) + \sum_{i=1}^{J} f_i}$$

with $m = 3$ (planar) or $m = 6$ (spatial). The formula can yield zero (structure), negative (over-constrained), or positive (mechanism) values.

Table 2.2: Grübler formula worked examples

Mechanism	m	N	J	$\sum f_i$	DOF
Four-bar linkage	3	4	4	4	1
Five-link chain	3	5	5	5	2
Gough–Stewart	6	8	9	15	6
Delta robot	6	17	21	45	3
Jansen linkage	3	12	16	16	1

2.1.4 Task space and workspace

Task space is the space in which the robot's task is naturally expressed — $\mathbb{R}^2$ for pen-plotting, or the 6-DOF rigid-body pose space for general manipulation. The choice of task space is driven by the task, not by the robot.

Workspace is the subset of the environment the end-effector can reach. Its shape and volume depend on the manipulator geometry and on joint limits. A point in the workspace may be reachable by more than one robot configuration — the workspace does not fully specify the configuration.

2.2 Manipulator Geometries

Industrial manipulators are classified by the geometry of their first three joints, which determines the workspace shape.

2.2.1 Cartesian (Gantry) — PPP

Three mutually orthogonal prismatic joints. Each DOF corresponds directly to a Cartesian coordinate. High mechanical stiffness; uniform positioning accuracy throughout the workspace (a rectangular parallelepiped). Low dexterity since all joints are prismatic. Objects are approached from the side (or from above in a gantry arrangement). Applications: material handling, heavy-load assembly, machine tools.

2.2.2 Cylindrical — RPP

One revolute followed by two prismatic joints. High stiffness from the prismatic joints; accuracy decreases with increasing horizontal stroke. Workspace: portion of a hollow cylinder. The horizontal prismatic joint gives good access to cavities. Applications: large-object handling (hydraulic motors preferred).

2.2.3 Spherical — RRP

Two revolute joints followed by one prismatic. Lower stiffness than the above two. Workspace: portion of a hollow sphere (can include the floor). Applications: machining with electric motors.

2.2.4 SCARA — RRP (parallel axes)

Selective Compliant Arm for Robotic Assembly. Two revolute joints plus one vertical prismatic, all axes parallel (z-axes). High stiffness to vertical loads; compliant horizontally (hence *selective* compliance). 4 DOF total. Accuracy decreases with distance from the first joint axis. Applications: micro-assembly, pick-and-place of small objects.

2.2.5 Articulated (Anthropomorphic) — RRR

Three revolute joints with the first axis orthogonal to the other two (which are parallel). The second joint is the *shoulder*, the third the *elbow*. Most dexterous geometry; all joints revolute. Workspace: approximately a large spherical volume. Positioning accuracy varies across the workspace. The dominant industrial layout. Applications: welding, painting, assembly, machining.

2.2.6 Delta (Parallel)

A parallel robot with three kinematic chains connecting the base to a moving platform. High speed and acceleration (only the platform and end-effector move — motors stay on the base). High rigidity; errors from each chain do not accumulate. Smaller, less dexterous workspace than serial robots. 3 translational DOF (no platform rotation). Applications: high-speed pick-and-place (food, pharma, electronics).

2.2.7 Jansen mechanism

A single-DOF planar linkage ($N = 12$, $J = 16$, DOF$= 1$) that converts crank rotation into a realistic walking gait. Used as a leg module in legged robots.

2.3 Actuation Units

2.3.1 Servomotor characteristics

A **servomotor** is a motor with a closed-loop position or velocity controller. Key characteristics for robotic use: low rotor inertia, high power-to-weight ratio, wide velocity range (up to ~2000 rpm), high position accuracy (~1/1000 turn), and power from 10 W to 10 kW.

2.3.2 Pneumatic actuators

Pneumatic actuators convert pressurised air into mechanical work via pistons. Simple and low-cost, but difficult to control precisely due to air compressibility. Main uses: binary gripper actuation, McKibben artificial muscles, soft-robotic applications.

2.3.3 Hydraulic actuators

Hydraulic actuators pump an incompressible fluid into a piston or rotary motor.

- **Pros:** self-lubricated, spark-free, very large torques, high power density, no overheating under static load.
- **Cons:** require reservoir and pump, large dimensions, high cost, oil-leak risk, lower efficiency than electric.

Force multiplication across pistons of areas A_1, A_2:

$$F_2 = \frac{A_2}{A_1} F_1.$$

Piston dynamics (area A, spring k_s, damping b, pressure P_m, load F_L):

$$P_m A = m_p \ddot{d} + b\dot{d} + F_L + k_s d.$$

2.3.4 Electrical actuators

Electric motors convert electrical energy into mechanical work. Pros: widely available power supply, low cost, high efficiency, no leaks, easy maintenance. Cons: static overheating; dangerous in flammable environments.

2.3.4.1 DC motor model

A brushed DC motor has a rotor (armature), stator (permanent magnets), and a commutator ring. The coupled electrical–mechanical equations are:

$$V_a = L_a \frac{di}{dt} + R_a\, i + K_v\, \omega, \qquad K_T\, i = I_m \frac{d\omega}{dt} + b\, \omega + \tau_L,$$

where K_v is the back-EMF constant and K_T is the torque constant. Power balance gives:

$$\boxed{K_v = K_T} \quad \text{(in SI units)}.$$

2.3.4.2 Motor characteristic curves

The steady-state speed–torque curve is linear:

$$\omega = \omega_0 \left(1 - \frac{\tau}{\tau_{\text{stall}}}\right).$$

Peak mechanical power $P_{\max} = \frac{1}{4}\omega_0 \tau_{\text{stall}}$ occurs at the midpoint. Current is proportional to torque. Best practice: operate near peak efficiency, which occurs just below the power peak.

2.3.4.3 Brushless motors

Brushless motors move the magnets to the rotor and windings to the stator, eliminating the commutator. Advantages: reduced losses, less maintenance, better heat dissipation, more compact rotor, lower inertia — at higher cost due to electronic commutation.

2.4 Motion Transmissions

Transmissions serve three roles: (1) quantitative transformation (speed reduction / torque multiplication), (2) qualitative transformation (rotation to linear motion), and (3) structural benefit (locating motors near the base to reduce moving mass).

2.4.1 Gear ratio

$$n = \frac{\dot{\theta}_1}{\dot{\theta}_2} = \frac{u_2}{u_1}, \qquad \tau_2 = n\,\eta_g\,\tau_1,$$

where u_i are pitch radii and η_g is gearbox efficiency.

2.4.2 Transmission types

Table 2.3: Transmission types

Type	Purpose	Key limitation
Spur / worm gears	Change direction or axis of motion	Deformation, backlash
Lead screw, rack and pinion	Rotation → translation	Friction, elasticity, backlash
Toothed belts and chains	Displace motor from joint axis	Belt compliance; chain vibration
Harmonic drive	In-line, compact, high ratio (up to 320:1)	Elasticity
Transmission shafts	Long in-link shafts	Alignment; flexible couplings needed

Rack and pinion converts rotation to linear motion. **Epicyclic (planetary) gears** offer three gear ratios by fixing one of three components (ring, sun, or planet carrier). **Sun-and-planet** mechanisms convert reciprocating to rotary motion.

2.4.3 Backlash

Backlash is the angular gap between mating gear teeth, needed to accommodate manufacturing tolerances, thermal expansion, and lubrication. Excessive backlash causes positioning error and vibration under reversing loads; insufficient backlash causes overloading, heat, and accelerated wear. Backlash is eliminated in harmonic drives.

2.4.4 Harmonic drive

A harmonic drive achieves up to 320:1 reduction in the same volume a planetary gear yields ~10:1, through elastic deformation of a thin flexspline. Key features: **no backlash**, compact, lightweight, high torque, same-axis input/output. Widely used in robot joints, satellite mechanisms, and surgical robots.

2.4.5 Optimal gear ratio

Motor inertia J_M, load inertia J_L, gear ratio n. The motor torque required for load acceleration a is:

$$\tau_M = \left(J_M n + \frac{J_L}{n} \right) a.$$

Minimising over n:

$$\boxed{n_{\text{opt}} = \sqrt{J_L / J_M}}$$

This **inertia-matching** condition balances reflected load inertia against motor inertia. Increasing n beyond n_{opt} helps load inertia but penalises the motor; there is a clear trade-off.

2.5 Sensing Units

Proprioceptive sensors measure the robot's internal state: joint position, velocity, torque, link acceleration. **Exteroceptive sensors** measure interaction with the environment: external force/torque, proximity, vision, and other physical quantities (gas, sound, humidity, temperature).

Key measurement-system properties: **resolution** (smallest detectable change), **accuracy** (closeness to true value), **repeatability**, **linearity**, **bandwidth**, and **noise floor**.

2.5.1 Position sensors

Linear displacement: potentiometers, LVDTs, inductosyns, Hall sensors. Angular displacement: potentiometers, resolvers, Hall sensors, digital encoders.

2.5.1.1 Digital encoder

A code disk rotates with the joint, alternating opaque and transparent sectors on concentric tracks. Infrared LEDs shine through the disk; photodetectors convert light pulses to electrical pulses. Resolution: $360°/2^N$ per revolution for an N-track disk (e.g. $N = 12$ gives $\approx 0.088°$). **Incremental** encoders count from a reference; **absolute** encoders give a unique code at every position.

2.5.2 Accelerometers

MEMS accelerometers integrate a tiny proof mass on silicon. Two sensing principles:

- **Capacitive:** mass displacement changes the gap between mass and fixed plates, altering capacitance. Measured and converted to a voltage proportional to acceleration.
- **Piezoresistive:** mass displacement strains embedded piezoresistors, changing their resistance. Resistance change is converted to a voltage.

Compact, low-cost, low-power — enabling IMU integration with gyroscopes.

2.5.3 Force and torque sensors

2.5.3.1 Strain gauge and Wheatstone bridge

A strain gauge changes resistance with strain:

$$\frac{\Delta R}{R} = \mathrm{GF} \cdot \varepsilon, \qquad \frac{V_o}{V_i} \approx \frac{\mathrm{GF} \cdot \varepsilon}{4}.$$

where GF is the gauge factor and the voltage ratio is from a balanced Wheatstone bridge linearised for small strains.

2.5.3.2 Load cell

Strain gauges bonded to a metal elastic element (beam or column). Applied force causes a known elastic deformation; strain is measured and converted to force using the element's known geometry and material modulus.

2.5.3.3 Torque cell

Gauges mounted at ±45° on a low-torsional-stiffness element (hollow shaft or flange). Torsional deformation generates shear strains detected by the gauges, connected to a Wheatstone bridge via a slip ring.

2.5.3.4 Six-axis force/torque sensor

Measures all six wrench components ($F_x, F_y, F_z, M_x, M_y, M_z$) simultaneously. The **Maltese cross** design connects a central hub to an outer ring via four thin spokes; gauges on each spoke respond to different load combinations. The full wrench is recovered via a calibration matrix:

$$\gamma = \mathbf{C}\,\mathbf{v},$$

where $\mathbf{v}$ is the vector of bridge voltages and $\mathbf{C}$ is determined during factory calibration. Used at robot wrists for force-controlled assembly and safe human-robot interaction.

2.6 Control Hierarchy

Robot control spans multiple abstraction levels:

Table 2.4: Robot control hierarchy

Level	Function	Methods
High-level	Decision making, behaviour	Behaviour trees, AI
	Localisation & mapping (mobile)	SLAM
	Task-space trajectory generation	Model Predictive Control
Mid-level	Coordinate transformation	Inverse kinematics
Low-level	Joint servo control	PID, force/impedance control

The low-level inner loop runs at ~1 kHz, controlling each servomotor from joint encoder feedback. The mid-level converts task-space references to joint-space commands via inverse kinematics. The high-level plans tasks, handles decisions, and — for mobile robots — builds and queries a map.

3 3. Rigid Body Pose

The first step in describing any robot is to say *where* each link is and *which way* it is pointing. This chapter builds the mathematical vocabulary: reference frames, rotation matrices, and their alternative parameterisations. We will see that the same 3×3 matrix $\mathbf{R}$ serves three distinct roles — as a description of orientation, as a change-of-coordinates map, and as an active rotation operator. Keeping those roles separate avoids confusion when composing transforms along a kinematic chain.

3.1 Frames and position

Consider a fixed **reference frame** O-xyz and a **body frame** O'-$x'y'z'$ rigidly attached to a moving body. The **pose** of the body is fully described by:

1. the **position** of the body-frame origin O', expressed as the vector $\overrightarrow{OO'}$ on the reference axes, and
2. the **orientation** of the body frame relative to the reference frame.

Expressing the origin offset on the reference axes,

$$\overrightarrow{OO'} = \begin{bmatrix} o'_x \\ o'_y \\ o'_z \end{bmatrix}, \qquad \overrightarrow{O'} = o'_x\,\hat{x} + o'_y\,\hat{y} + o'_z\,\hat{z}.$$

The reference unit vectors are $\hat{x} = [1,0,0]^{\mathsf{T}}$, $\hat{y} = [0,1,0]^{\mathsf{T}}$, $\hat{z} = [0,0,1]^{\mathsf{T}}$.

3.2 The rotation matrix

Let $\hat{x}', \hat{y}', \hat{z}'$ be the unit vectors of the body frame O'-$x'y'z'$, written with respect to O-xyz. Each is a linear combination of the reference axes:

$$\begin{aligned} \hat{x}' &= x'_x\,\hat{x} + x'_y\,\hat{y} + x'_z\,\hat{z}, \\ \hat{y}' &= y'_x\,\hat{x} + y'_y\,\hat{y} + y'_z\,\hat{z}, \\ \hat{z}' &= z'_x\,\hat{x} + z'_y\,\hat{y} + z'_z\,\hat{z}. \end{aligned}$$

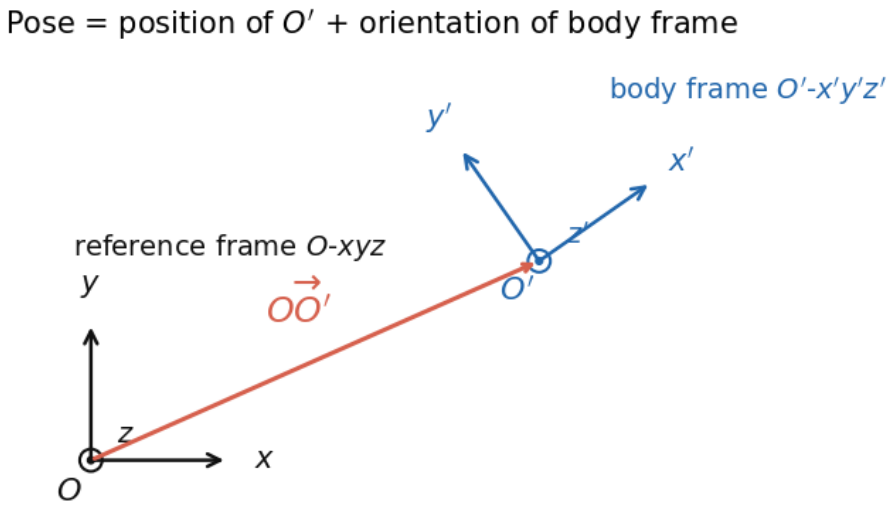

Figure 3.1: Reference frame O-xyz (black) and body frame O'-$x'y'z'$ (blue). The pose of the body is given by the position vector $\overrightarrow{OO'}$ together with the orientation of the body frame.

Stacking the body axes as **columns** gives the **rotation matrix**

$$\mathbf{R} = \begin{bmatrix} \hat{x}' & \hat{y}' & \hat{z}' \end{bmatrix} = \begin{bmatrix} x'_x & y'_x & z'_x \\ x'_y & y'_y & z'_y \\ x'_z & y'_z & z'_z \end{bmatrix}.$$

Each entry is a **direction cosine** — the dot product between a body axis and a reference axis. For example, the first column collects $\hat{x}'^{\mathsf{T}}\hat{x}$, $\hat{x}'^{\mathsf{T}}\hat{y}$, $\hat{x}'^{\mathsf{T}}\hat{z}$.

3.3 Orthonormality

Because the body axes are mutually orthogonal unit vectors,

$$\begin{aligned} \hat{x}'^{\mathsf{T}}\hat{y}' = 0, \quad \hat{y}'^{\mathsf{T}}\hat{z}' = 0, \quad \hat{z}'^{\mathsf{T}}\hat{x}' = 0 \qquad &\text{(orthogonality)}, \\ \hat{x}'^{\mathsf{T}}\hat{x}' = 1, \quad \hat{y}'^{\mathsf{T}}\hat{y}' = 1, \quad \hat{z}'^{\mathsf{T}}\hat{z}' = 1 \qquad &\text{(unit length)}. \end{aligned}$$

These six constraints mean that $\mathbf{R}$ belongs to the **special orthogonal group** SO(3):

$$\mathbf{R}^{\mathsf{T}}\mathbf{R} = \mathbf{I}, \qquad \det \mathbf{R} = +1, \qquad \Longrightarrow \qquad \boxed{\mathbf{R}^{\mathsf{T}} = \mathbf{R}^{-1}}$$

The transpose equals the inverse — a fact used constantly when reversing a transformation. The 9 elements of $\mathbf{R}$ carry only 3 independent degrees of freedom (the remaining 6 are fixed by the orthonormality constraints).

3.4 Elementary rotations

A rotation by angle α about the z-axis ($\hat{z} = \hat{z}'$) sends the reference axes to

$$\hat{x}' = \cos\alpha\,\hat{x} + \sin\alpha\,\hat{y}, \qquad \hat{y}' = -\sin\alpha\,\hat{x} + \cos\alpha\,\hat{y},$$

giving the three **elementary rotation matrices**:

$$\mathbf{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}, \qquad \mathbf{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}.$$

A negative rotation is simply the transpose,

$$\mathbf{R}_k(-\theta) = \mathbf{R}_k{}^{\mathsf{T}}(\theta), \qquad k \in \{x, y, z\},$$

consistent with orthogonality.

3.5 Vector representation across frames

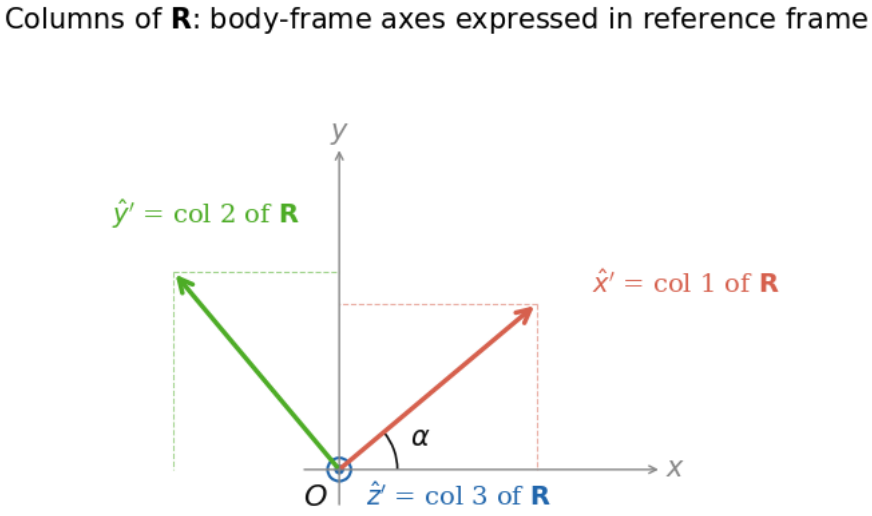

Figure 3.2: The body-frame axes $\hat{x}', \hat{y}', \hat{z}'$ expressed in the reference frame are the three columns of $\mathbf{R}$. Dashed lines project each tip onto the reference axes, showing the direction-cosine entries.

The rotation matrix serves three equivalent roles simultaneously:

1. **Orientation description. R** describes the orientation of frame O' relative to frame O by storing the body-frame axes as its columns.

2. **Change of coordinates (transformation matrix).** A point P has coordinates $p' = [p'_x, p'_y, p'_z]^\mathsf{T}$ in the body frame. Expanding the same physical vector on the reference axes gives the **change of representation**:

$$\boxed{p = \mathbf{R}\, p'}$$

In 2D this reads $\begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} p'_x \\ p'_y \end{bmatrix}$.

In this role, **R** is the **transformation matrix** that maps coordinates from the body frame to the reference frame (*passive* interpretation: the point does not move; only its description changes).

3. **Active rotation.** The same matrix rotates a vector *within* a fixed frame by angle α. Using the angle-sum identities, a point at angle β with magnitude $|p|$ moves to angle $\alpha + \beta$, which is exactly the action of $\mathbf{R}_z(\alpha)$.

3.6 Composition of rotations

Given three frames 0, 1, 2, a vector in frame 2 is brought into frame 0 by composing the intermediate rotations:

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

This follows directly from $p^0 = \mathbf{R}_1^0 p^1 = \mathbf{R}_1^0(\mathbf{R}_2^1 p^2)$.

Pre- vs. post-multiplication. When rotations are applied *about the axes of the current (rotated) frame*, they are **post-multiplied**:

$$\mathbf{R} = \mathbf{R}_1 \cdot \mathbf{R}_2 \cdot \mathbf{R}_3 \quad \text{(body-fixed axes)}.$$

When rotations are applied *about fixed reference axes*, they are **pre-multiplied** in the opposite order.

Two useful identities follow from orthogonality:

$$(\mathbf{R}_1^0)^{-1} = \mathbf{R}_0^1 = (\mathbf{R}_1^0)^\mathsf{T},$$
$$\mathbf{R}_1^0\, \mathbf{R}_2^1\, \mathbf{R}_1^2\, \mathbf{R}_0^1 = \mathbf{I}.$$

3.7 Euler angles

Rotation matrices have 9 elements and 6 constraints, so only 3 degrees of freedom are needed to describe orientation. **Euler angles** parameterise orientation by three successive rotations. There are $3^3 = 27$ possible axis combinations. However, two consecutive rotations must not be made around the same axis (the second would merely add to the first, using up a degree of freedom for nothing). This constraint eliminates 15 sets and leaves exactly **12 valid Euler angle conventions**.

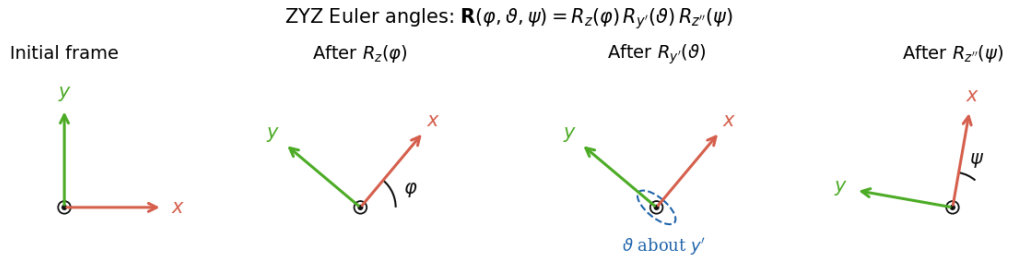

Figure 3.3: ZYZ Euler angle sequence. Each step rotates the frame about the current axis; the fourth panel shows the composed orientation. The y' tilt in step 3 (dashed ellipse) is the out-of-plane rotation $R_{y'}(\vartheta)$.

3.7.1 ZYZ convention

The most common set in robotics is the **ZYZ** convention, with $\phi = [\varphi, \vartheta, \psi]^\mathsf{T}$:

1. Rotate by φ about $z \to R_z(\varphi)$
2. Rotate by ϑ about the *current* $y' \to R_{y'}(\vartheta)$
3. Rotate by ψ about the *current* $z'' \to R_{z''}(\psi)$

$$\mathbf{R}(\phi) = \mathbf{R}_z(\varphi)\,\mathbf{R}_y(\vartheta)\,\mathbf{R}_z(\psi).$$

Carrying out the product (writing $c_\bullet = \cos$, $s_\bullet = \sin$):

$$\mathbf{R} = \begin{bmatrix} c_\varphi c_\vartheta c_\psi - s_\varphi s_\psi & -c_\varphi c_\vartheta s_\psi - s_\varphi c_\psi & c_\varphi s_\vartheta \\ s_\varphi c_\vartheta c_\psi + c_\varphi s_\psi & -s_\varphi c_\vartheta s_\psi + c_\varphi c_\psi & s_\varphi s_\vartheta \\ -s_\vartheta c_\psi & s_\vartheta s_\psi & c_\vartheta \end{bmatrix}.$$

Inverse problem (extracting the angles). Given $\mathbf{R}$, read off the angles as:

$$\vartheta = \operatorname{atan2}\left(\sqrt{r_{13}^2 + r_{23}^2},\; r_{33}\right),$$

with φ and ψ recovered from the remaining entries. The two-argument arctangent is used throughout to resolve the correct quadrant. Note that two solutions exist (corresponding to $+\vartheta$ and $-\vartheta$), analogous to the elbow-up/elbow-down ambiguity in inverse kinematics.

3.7.2 Roll–Pitch–Yaw (RPY)

The **RPY** convention uses rotations about *fixed* (world) axes, with $\phi = [\varphi, \vartheta, \psi]^\mathsf{T}$ denoting roll, pitch, and yaw:

$$\mathbf{R}(\phi) = \mathbf{R}_z(\psi)\,\mathbf{R}_y(\vartheta)\,\mathbf{R}_x(\varphi).$$

Because the axes are fixed, later rotations are pre-multiplied on the left — note the reversed order compared to body-fixed ZYZ.

3.8 Axis and angle

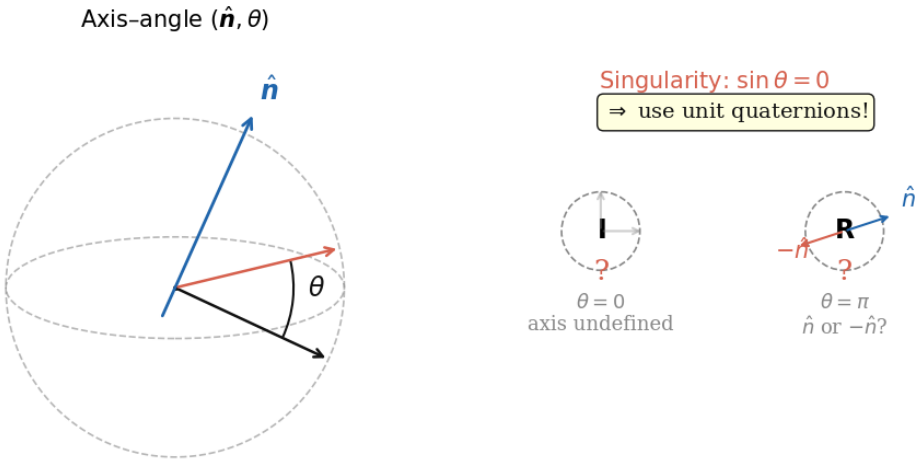

Figure 3.4: Left: axis–angle representation $(\hat{n}, \theta)$. The rotation axis $\hat{n}$ passes through the origin; θ is measured in the plane perpendicular to it. Right: the two singularity cases $\theta = 0$ (axis undefined) and $\theta = \pi$ (axis direction ambiguous), resolved by unit quaternions.

Any rotation can be specified by a unit axis $\hat{n} = [n_x, n_y, n_z]^\mathsf{T}$ and an angle θ. The corresponding rotation matrix is obtained by aligning $\hat{n}$ with $\hat{z}$, rotating by θ, then undoing the alignment:

$$\mathbf{R}(\hat{n}, \theta) = \mathbf{R}_z(\alpha)\, \mathbf{R}_y(\beta)\, \mathbf{R}_z(\theta)\, \mathbf{R}_y(-\beta)\, \mathbf{R}_z(-\alpha),$$

where the alignment angles satisfy

$$\sin\alpha = \frac{n_y}{\sqrt{n_x^2 + n_y^2}}, \quad \cos\alpha = \frac{n_x}{\sqrt{n_x^2 + n_y^2}}, \quad \sin\beta = \sqrt{n_x^2 + n_y^2}, \quad \cos\beta = n_z.$$

A useful symmetry: reversing both the axis and the angle gives the same rotation,

$$\mathbf{R}(-\theta, -\hat{n}) = \mathbf{R}(\theta, \hat{n}).$$

> ⚠ Warning
>
> **Singularity.** The inverse problem (extracting θ and $\hat{n}$ from $\mathbf{R}$) breaks down when $\sin\theta = 0$, i.e. at $\theta = 0$ or $\theta = \pi$. At $\theta = 0$ the rotation is the identity and the axis is undefined; at $\theta = \pi$ the direction of the axis becomes ambiguous. **If $\sin\theta = 0$, there are problems!** Unit quaternions (below) resolve both cases without singularity.

3.9 Unit quaternions

A **unit quaternion** $\mathcal{Q} = \{\eta,\ \varepsilon\}$ encodes the axis/angle pair without the singularities:

$$\eta = \cos\frac{\theta}{2} \text{ (scalar part)}, \qquad \varepsilon = \sin\frac{\theta}{2}\,\hat{n} = \begin{bmatrix}\varepsilon_x \\ \varepsilon_y \\ \varepsilon_z\end{bmatrix} \text{ (vector part)}.$$

The unit-norm constraint follows directly from $\cos^2 + \sin^2 = 1$:

$$\eta^2 + \varepsilon_x^2 + \varepsilon_y^2 + \varepsilon_z^2 = 1.$$

Advantages over Euler angles: no representation singularities (gimbal lock), numerically stable interpolation (spherical linear interpolation, SLERP), and efficient composition (quaternion multiplication replaces matrix multiplication).

3.10 Homogeneous transformations

3.10.1 Combining rotation and translation

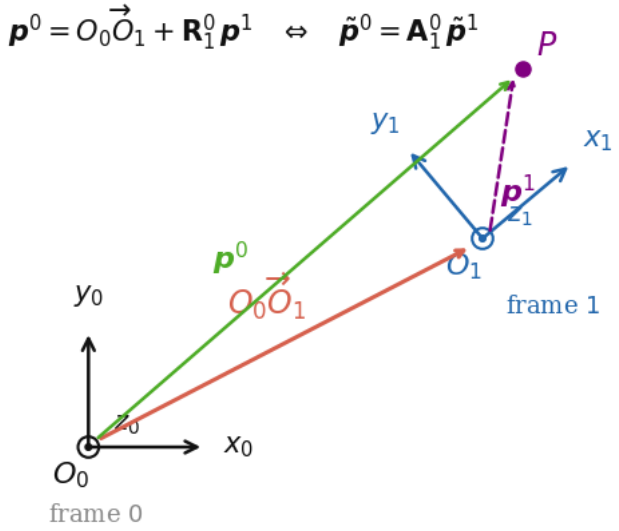

Figure 3.5: Homogeneous transformation $\mathbf{A}_1^0$. The position of point P expressed in frame 0 (green p^0) equals the translation $\overrightarrow{O_0O_1}$ plus the rotated local coordinates $\mathbf{R}_1^0 p^1$ (purple). The 4×4 matrix $\mathbf{A}_1^0$ packages both in a single operation.

Position changes by both a rotation and a translation. For a point known in frame 1,

$$p^0 = \overrightarrow{O_1'^0} + \mathbf{R}_1^0\, p^1.$$

Augmenting the coordinate vector with a trailing 1, $\tilde{p} = [p^\mathsf{T}, 1]^\mathsf{T} \in \mathbb{R}^4$, lets this be written as a single 4×4 **homogeneous transformation**:

$$\mathbf{A}_1^0 = \begin{bmatrix} \mathbf{R}_1^0 & \overrightarrow{O_1'^0} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}, \qquad \tilde{p}^0 = \mathbf{A}_1^0\,\tilde{p}^1.$$

The bottom row enforces $1 = 1$, ensuring the augmented form is consistent.

3.10.2 Inverse transformation

Unlike rotation matrices, $\mathbf{A}^{-1} \neq \mathbf{A}^{\mathsf{T}}$ in general. Inverting $p^0 = \overrightarrow{O_1'^0} + \mathbf{R}_1^0 p^1$ gives

$$\mathbf{A}_0^1 = (\mathbf{A}_1^0)^{-1} = \begin{bmatrix} \mathbf{R}_0^1 & -\mathbf{R}_0^1 \overrightarrow{O_1'^0} \\ \mathbf{0}^{\mathsf{T}} & 1 \end{bmatrix}.$$

3.10.3 Composition along a chain

Transformations compose by matrix product, exactly as rotation matrices do:

$$\tilde{p}^0 = \mathbf{A}_1^0 \, \mathbf{A}_2^1 \, \mathbf{A}_3^2 \cdots \mathbf{A}_n^{n-1} \, \tilde{p}^n.$$

This is the algebraic backbone of open-chain kinematics and motivates the Denavit–Hartenberg convention in the next chapter.

4 4. Direct and Inverse Kinematics

Given the pose representation developed in the previous chapter, this chapter addresses the **forward problem** (computing end-effector pose from joint variables) and the **inverse problem** (finding joint variables that realise a desired end-effector pose). Both problems are central to robot programming: the controller works in joint space, but tasks are naturally specified in task space, so we need reliable maps in both directions.

4.1 The direct kinematics map

For a manipulator with joint variables q, the pose of the end-effector frame relative to the base is

$$\mathbf{T}_e^b(q) = \begin{bmatrix} \hat{n}_e^b & \hat{s}_e^b & \hat{a}_e^b & p_e^b \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & p_e^b \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix},$$

where $(\hat{n}, \hat{s}, \hat{a})$ are the end-effector frame unit vectors and p_e^b is its origin. An open chain with n joints decomposes this into a product of link transforms:

$$\mathbf{T}_n^0(q) = \mathbf{A}_1^0(q_1)\, \mathbf{A}_2^1(q_2) \cdots \mathbf{A}_n^{n-1}(q_n),$$

and including fixed base and tool offsets,

$$\mathbf{T}_e^b = \mathbf{T}_0^b\, \mathbf{T}_n^0(q)\, \mathbf{T}_e^n.$$

Joint space and task space. The joint configuration $q \in \mathbb{R}^n$ lives in **joint space**. The end-effector pose $x \in \mathbb{R}^m$ (position + orientation) lives in **task space**. The map $x = h(q)$ is in general nonlinear, which makes the inverse problem non-trivial.

4.1.1 Worked example: two-link planar arm

For a planar arm with link lengths a_1, a_2 and joint angles θ_1, θ_2, write $c_1 = \cos\theta_1$, $c_{12} = \cos(\theta_1 + \theta_2)$, etc. The end-effector transform is

$$\mathbf{T}_e^b(\theta_1, \theta_2) = \begin{bmatrix} 0 & s_{12} & c_{12} & a_1 c_1 + a_2 c_{12} \\ 0 & -c_{12} & s_{12} & a_1 s_1 + a_2 s_{12} \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

The position entries make geometric sense: the end-effector reach is the vector sum of two link vectors projected onto x and y.

For longer kinematic chains a systematic, rule-based approach is essential.

4.2 The Denavit–Hartenberg convention

The **Denavit–Hartenberg (DH) convention** attaches a frame to each link using exactly **four parameters per joint**. It provides the systematic, rule-based approach that makes analysis of longer kinematic chains tractable. A worked animation of the frame-assignment procedure is available on ILIAS.

Table 4.1: Denavit–Hartenberg parameters

Parameter	Symbol	Description
Link length	a_i	distance between z_{i-1} and z_i along x_i
Link twist	α_i	angle between z_{i-1} and z_i about x_i
Link offset	d_i	distance between x_{i-1} and x_i along z_{i-1}
Joint angle	θ_i	angle between x_{i-1} and x_i about z_{i-1}

Frame assignment rules:

1. Choose axis z_i along the axis of joint $i + 1$.
2. Locate origin O_i at the intersection of z_i with the common normal to z_{i-1} and z_i.
3. Choose x_i along the common normal from joint i to joint $i + 1$.
4. Choose y_i to complete a right-handed frame.

For a **revolute joint**, θ_i is the variable and d_i is fixed. For a **prismatic joint**, d_i is the variable and θ_i is fixed.

4.2.1 DH link transform

The transform from frame $i-1$ to frame i factors into a screw along z and a screw along x:

$$\mathbf{A}_{i'}^{i-1} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad \mathbf{A}_{i}^{i'} = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Their product is the standard DH link matrix:

$$\mathbf{A}_{i}^{i-1} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

4.3 DH examples

4.3.1 Three-link planar arm (3R)

All joints revolute, all axes parallel to z, all twists $\alpha_i = 0$.

Table 4.2: DH parameters for the planar 3R arm ($*$ = variable)

Link	a_i	α_i	d_i	θ_i
1	a_1	0	0	θ_1^*
2	a_2	0	0	θ_2^*
3	a_3	0	0	θ_3^*

With all twists zero the chain stays planar. Using $c_{123} = \cos(\theta_1+\theta_2+\theta_3)$,

$$\mathbf{T}_3^0 = \mathbf{A}_1^0\mathbf{A}_2^1\mathbf{A}_3^2 = \begin{bmatrix} c_{123} & -s_{123} & 0 & a_1c_1 + a_2c_{12} + a_3c_{123} \\ s_{123} & c_{123} & 0 & a_1s_1 + a_2s_{12} + a_3s_{123} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

For this arm the end-effector frame does **not** coincide with frame 3: the approach unit vector of the tool must be aligned with z_3, which requires a fixed rotation of 90° about y_3. A constant tool transform $\mathbf{T}_e^3$ is appended to account for this offset:

$$\mathbf{T}_e^b = \mathbf{T}_0^b\,\mathbf{T}_3^0\,\mathbf{T}_e^3, \qquad \mathbf{T}_e^3 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

4.3.2 Spherical arm (RRP)

Two revolute joints plus one prismatic (θ_1, θ_2 variable, d_3 variable).

Table 4.3: DH parameters for the spherical (RRP) arm

Link	a_i	α_i	d_i	θ_i
1	0	$-\pi/2$	0	θ_1^*
2	0	$\pi/2$	d_2	θ_2^*
3	0	0	d_3^*	0

Note that the rotation matrix of the end-effector depends only on the two revolute joints. The individual link matrices are

$$\mathbf{A}_1^0 = \begin{bmatrix} c_1 & 0 & -s_1 & 0 \\ s_1 & 0 & c_1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{A}_2^1 = \begin{bmatrix} c_2 & 0 & s_2 & 0 \\ s_2 & 0 & -c_2 & 0 \\ 0 & 1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{A}_3^2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Their product gives the direct kinematics

$$\mathbf{T}_3^0(q) = \begin{bmatrix} c_1c_2 & -s_1 & c_1s_2 & c_1s_2d_3 - s_1d_2 \\ s_1c_2 & c_1 & s_1s_2 & s_1s_2d_3 + c_1d_2 \\ -s_2 & 0 & c_2 & c_2d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad q = \begin{bmatrix} \theta_1 \\ \theta_2 \\ d_3 \end{bmatrix}.$$

4.3.3 Anthropomorphic arm (3R elbow)

Three revolute joints in the human-arm configuration: the first axis is perpendicular to the plane of the arm, the second and third axes are parallel to each other (the shoulder and elbow).

Table 4.4: DH parameters for the anthropomorphic arm

Link	a_i	α_i	d_i	θ_i
1	0	$\pi/2$	0	θ_1^*
2	a_2	0	0	θ_2^*
3	a_3	0	0	θ_3^*

The end-effector frame does **not** coincide with frame 3 in general. A constant tool transform $\mathbf{T}_e^3$ is appended to express the desired approach direction at the end-effector.

4.3.4 Spherical wrist (3R)

A **spherical wrist** is a three-revolute sub-chain whose three joint axes intersect at a common **wrist centre point**. Links are numbered from 4, since this sub-chain is typically mounted at the end of a 3R positioning arm (making a 6-DOF manipulator in total).

The key property of the spherical wrist is that it produces a **pure rotation**: the wrist centre does not move as the wrist joints change, so position and orientation can be controlled independently. The resulting rotation matrix has exactly the form of the **ZYZ Euler angle matrix** $\mathbf{R}_z(\varphi)\mathbf{R}_{y'}(\vartheta)\mathbf{R}_{z''}(\psi)$, where φ, ϑ, ψ map to joints 4, 5, 6 respectively.

This decoupling is the basis of the **arm-wrist decomposition** for 6-DOF manipulators: the first three joints position the wrist centre (solved by inverse kinematics of the arm), and the last three joints orient the end-effector (solved by the ZYZ inverse formula). The end-effector frame coincides with frame 6 by convention for the spherical wrist.

More complex manipulators — such as a SCARA arm with a spherical wrist, or an anthropomorphic arm with an added wrist — can be assembled by concatenating their DH chains. The arm-wrist decomposition applies as long as the last three joint axes intersect at a single point.

4.4 Inverse kinematics

The **inverse kinematics** (IK) problem seeks joint variables that realise a desired pose:

$$q = h^{-1}(x).$$

This is fundamentally harder than direct kinematics for several reasons:

- The equations are in general **nonlinear**, so closed-form solutions do not always exist.
- **Multiple solutions** typically exist (e.g. elbow-up vs. elbow-down).
- There may be **infinitely many** solutions for a kinematically redundant manipulator ($n > m$).
- There may be **no solution** if the desired pose lies outside the manipulator's workspace.

4.4.1 Algebraic solution: planar 3R arm

For the planar 3R arm the task vector is

$$x = \begin{bmatrix} p_x \\ p_y \\ \varphi \end{bmatrix} = \begin{bmatrix} a_1 c_1 + a_2 c_{12} + a_3 c_{123} \\ a_1 s_1 + a_2 s_{12} + a_3 s_{123} \\ \theta_1 + \theta_2 + \theta_3 \end{bmatrix}.$$

Step 1. Remove the third link by computing the **wrist point**:

$$p_{Wx} = p_x - a_3 c_\varphi, \qquad p_{Wy} = p_y - a_3 s_\varphi.$$

Step 2. Square and add to isolate θ_2:

$$c_2 = \frac{p_{Wx}^2 + p_{Wy}^2 - a_1^2 - a_2^2}{2 a_1 a_2}, \quad -1 \le c_2 \le 1,$$
$$s_2 = \pm\sqrt{1 - c_2^2}, \qquad \boxed{\theta_2 = \operatorname{atan2}(s_2, c_2)} \tag{IK.1}$$

The two signs give elbow-up ($s_2 > 0$) and elbow-down ($s_2 < 0$) configurations.

Step 3. With θ_2 known, solve a linear system for θ_1:

$$s_1 = \frac{(a_1 + a_2 c_2)\, p_{Wy} - a_2 s_2\, p_{Wx}}{p_{Wx}^2 + p_{Wy}^2}, \qquad c_1 = \frac{(a_1 + a_2 c_2)\, p_{Wx} + a_2 s_2\, p_{Wy}}{p_{Wx}^2 + p_{Wy}^2},$$
$$\boxed{\theta_1 = \operatorname{atan2}(s_1, c_1)} \tag{IK.2}$$

Step 4. The orientation equation closes the chain:

$$\boxed{\theta_3 = \varphi - \theta_1 - \theta_2} \tag{IK.3}$$

4.4.2 Geometric solution: planar 3R arm

The same result follows from triangle geometry. The law of cosines on the triangle formed by the two links and the wrist point gives:

$$c_2 = \frac{p_{Wx}^2 + p_{Wy}^2 - a_1^2 - a_2^2}{2a_1 a_2}, \qquad \theta_2 = \pm \cos^{-1}(c_2). \tag{IK.1}$$

Introducing the auxiliary angles

$$\alpha = \operatorname{atan2}(p_{Wy}, p_{Wx}), \qquad \beta = \cos^{-1}\left(\frac{p_{Wx}^2 + p_{Wy}^2 + a_1^2 - a_2^2}{2a_1\sqrt{p_{Wx}^2 + p_{Wy}^2}}\right),$$

the base joint is

$$\boxed{\theta_1 = \alpha \mp \beta} \tag{IK.2}$$

with the sign matched to the elbow choice, and

$$\boxed{\theta_3 = \varphi - \theta_2 - \theta_1} \tag{IK.3}$$

4.4.3 Inverse kinematics of the spherical wrist

For a manipulator with a spherical wrist, the IK decouples:

1. **Arm IK:** compute the wrist centre position from the desired end-effector pose and tool length; solve the arm's IK for the first three joints.
2. **Wrist IK:** extract the desired wrist rotation $\mathbf{R}_{\text{wrist}}$ and solve the ZYZ inverse problem for joints 4, 5, 6.

This decoupling is possible only because the spherical wrist axes intersect, so changing wrist joints does not move the wrist centre.

5 5. Differential Kinematics and Statics

Whereas direct kinematics is a *static* map (joints → pose), differential kinematics is its *instantaneous* derivative: how do joint *rates* map to end-effector *velocities*? The central tool is the **Jacobian matrix**. The same Jacobian, transposed, maps end-effector forces and moments back to joint torques — a duality that connects velocity control to force control and underpins much of robot statics.

> **Note**
>
> Two different Jacobians appear in the literature. The **analytical Jacobian** $\mathbf{J}_A = \partial h/\partial q$ arises from differentiating the pose map directly. The **geometric Jacobian** $\mathbf{J}$ maps joint rates to the physical twist (linear velocity + angular velocity). In general $\mathbf{J} \neq \mathbf{J}_A$; they agree only when the chosen orientation parameterisation has the property $\dot{\phi} = \omega_e$, which is not the case for Euler angles. The distinction matters: using $\mathbf{J}_A$ where $\mathbf{J}$ is required leads to errors in velocity and force calculations.

5.1 From pose to velocity

Differentiating the direct kinematics map $x = h(q)$ gives

$$\dot{x} = \frac{\partial h}{\partial q}\,\dot{q} = \underbrace{\mathbf{J}_A(q)}_{\text{analytical Jacobian}}\,\dot{q}.$$

However, $\dot{\phi}$ (the time derivative of an orientation parameterisation) is generally *not* the angular velocity vector ω. It is therefore more natural to work with the **geometric Jacobian** that maps joint rates directly to the end-effector twist.

5.2 Derivative of a rotation matrix

Differentiating the orthogonality identity $\mathbf{R}(t)\mathbf{R}(t)^\mathsf{T} = \mathbf{I}$ gives

$$\dot{\mathbf{R}}(t)\mathbf{R}(t)^\mathsf{T} + \mathbf{R}(t)\dot{\mathbf{R}}(t)^\mathsf{T} = \mathbf{0}.$$

Define $\mathbf{S}(t) = \dot{\mathbf{R}}(t)\mathbf{R}(t)^{\mathsf{T}}$. Since $\mathbf{S} + \mathbf{S}^{\mathsf{T}} = \mathbf{0}$, $\mathbf{S}$ is **skew-symmetric** and corresponds to a cross product:

$$\mathbf{S}(\omega) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}, \qquad \omega \times b = \mathbf{S}(\omega)\, b.$$

Rearranging gives the **matrix differential equation for a rotating body**:

$$\boxed{\dot{\mathbf{R}}(t) = \mathbf{S}(\omega)\, \mathbf{R}(t)}$$

For a point fixed in the body, $p = \mathbf{R}(t)p'$, so

$$\dot{p} = \dot{\mathbf{R}}p' = \mathbf{S}(\omega)\mathbf{R}p' = \omega \times \mathbf{R}p' = \omega \times p,$$

recovering the familiar formula $v = \omega \times r$. A useful conjugation identity is $\mathbf{R}\,\mathbf{S}(\omega)\,\mathbf{R}^{\mathsf{T}} = \mathbf{S}(\mathbf{R}\omega)$.

5.3 Link velocity propagation

The angular and linear velocities propagate outward from the base along the chain, one joint at a time. The derivation uses a single rule from rigid-body mechanics: **the velocity of any point on a rigid body equals the velocity of a reference point on that body plus the cross product of the angular velocity with the relative position vector.**

For joint i:

Revolute joint: adds a rotation about $\hat{z}_{i-1}$ at rate $\dot{\theta}_i$:

$$\omega_i = \omega_{i-1} + \dot{\theta}_i\, \hat{z}_{i-1}.$$

The linear velocity of the next frame origin gains a cross-product term: $\dot{p}_i = \dot{p}_{i-1} + \omega_i \times (p_i - p_{i-1})$.

Prismatic joint: translates along $\hat{z}_{i-1}$ at rate $\dot{d}_i$ but adds no rotation:

$$\omega_i = \omega_{i-1}, \qquad \dot{p}_i = \dot{p}_{i-1} + \dot{d}_i\, \hat{z}_{i-1}.$$

Applying the rigid-body rule recursively from the fixed base ($\omega_0 = 0$, $\dot{p}_0 = 0$) accumulates contributions from every joint in the chain up to the end-effector.

5.4 The geometric Jacobian

Collecting the velocity contributions, the end-effector twist is

$$v_e = \begin{bmatrix} \dot{p}_e \\ \omega_e \end{bmatrix} = \begin{bmatrix} \mathbf{J}_P \\ \mathbf{J}_O \end{bmatrix} \dot{q} = \mathbf{J}(q)\,\dot{q}.$$

Each column of $\mathbf{J}$ is the velocity contribution of one joint:

$$\begin{bmatrix} \mathbf{J}_{P_i} \\ \mathbf{J}_{O_i} \end{bmatrix} = \underbrace{\begin{bmatrix} \hat{z}_{i-1} \\ \mathbf{0} \end{bmatrix}}_{\text{prismatic}} \qquad \text{or} \qquad \underbrace{\begin{bmatrix} \hat{z}_{i-1} \times (p_e - p_{i-1}) \\ \hat{z}_{i-1} \end{bmatrix}}_{\text{revolute}}.$$

The geometric quantities $\hat{z}_{i-1}$, p_e, p_{i-1} are read from the direct-kinematics transforms $\mathbf{A}_1^0 \cdots \mathbf{A}_*^*$:

$$\hat{z}_{i-1} = \mathbf{R}_1^0(q_1) \cdots \mathbf{R}_{i-1}^{i-2}(q_{i-1})\,\hat{z}_0, \qquad \hat{z}_0 = [0,0,1]^{\mathsf{T}}.$$

5.4.1 Example: 3-link planar arm

All joint axes are $\hat{z}_0 = \hat{z}_1 = \hat{z}_2 = [0,0,1]^{\mathsf{T}}$. The Jacobian is

$$\mathbf{J} = \begin{bmatrix} -a_1 s_1 - a_2 s_{12} - a_3 s_{123} & -a_2 s_{12} - a_3 s_{123} & -a_3 s_{123} \\ a_1 c_1 + a_2 c_{12} + a_3 c_{123} & a_2 c_{12} + a_3 c_{123} & a_3 c_{123} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}.$$

The bottom three rows confirm that the angular velocity is purely about $\hat{z}$, consistent with the planar motion constraint.

5.4.2 Example: anthropomorphic arm

With DH parameters as in Table 4.4, the relevant axes are $\hat{z}_0 = [0,0,1]^\mathsf{T}$, $\hat{z}_1 = \hat{z}_2 = [s_1, -c_1, 0]^\mathsf{T}$, $p_0 = p_1 = [0,0,0]^\mathsf{T}$, and

$$p_2 = \begin{bmatrix} a_2 c_1 c_2 \\ a_2 s_1 c_2 \\ a_2 s_2 \end{bmatrix}, \qquad p_3 = \begin{bmatrix} c_1(a_2 c_2 + a_3 c_{23}) \\ s_1(a_2 c_2 + a_3 c_{23}) \\ a_2 s_2 + a_3 s_{23} \end{bmatrix}.$$

Carrying out the cross products yields

$$\mathbf{J} = \begin{bmatrix} -s_1(a_2 c_2 + a_3 c_{23}) & -c_1(a_2 s_2 + a_3 s_{23}) & -c_1 a_3 s_{23} \\ c_1(a_2 c_2 + a_3 c_{23}) & -s_1(a_2 s_2 + a_3 s_{23}) & -s_1 a_3 s_{23} \\ 0 & a_2 c_2 + a_3 c_{23} & a_3 c_{23} \\ 0 & s_1 & s_1 \\ 0 & -c_1 & -c_1 \\ 1 & 0 & 0 \end{bmatrix}.$$

5.5 Singularities

A **kinematic singularity** occurs where $\mathbf{J}$ loses rank — one or more directions of end-effector motion become unattainable regardless of the joint rates. For the planar 2R arm,

$$\mathbf{J} = \begin{bmatrix} -a_1 s_1 - a_2 s_{12} & -a_2 s_{12} \\ a_1 c_1 + a_2 c_{12} & a_2 c_{12} \end{bmatrix}, \qquad \det \mathbf{J} = a_1 a_2 s_2.$$

Hence $\det \mathbf{J} = 0$ exactly when $\theta_2 = 0$ or $\theta_2 = \pi$, corresponding to the arm fully stretched (**boundary singularity**) or fully folded (**internal singularity**). Near singularities small end-effector velocities require very large joint rates, and inverse kinematics solutions become numerically ill-conditioned.

5.6 Analytical Jacobian and representation singularities

> **!** Important
>
> $\mathbf{J} \neq \mathbf{J}_A$ in general. The **analytical Jacobian** $\mathbf{J}_A = \partial h / \partial q$ is the partial derivative of the pose map, while the geometric Jacobian $\mathbf{J}$ maps joint rates to the physical twist. They differ because $\dot{\phi} \neq \omega_e$ in general — the time derivative of an orientation parameterisation is not the same as the angular velocity vector.

For a ZYZ parameterisation $\phi = [\varphi, \vartheta, \psi]^{\mathsf{T}}$, the angular velocity is

$$\omega_e = \underbrace{\begin{bmatrix} 0 & -s_\varphi & c_\varphi s_\vartheta \\ 0 & c_\varphi & s_\varphi s_\vartheta \\ 1 & 0 & c_\vartheta \end{bmatrix}}_{\mathbf{T}(\phi_e)} \dot{\phi},$$

so the full velocity transform is

$$v_e = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{T}(\phi_e) \end{bmatrix} \dot{x}_e = \mathbf{T}_A \, \dot{x}_e, \qquad \mathbf{J} = \mathbf{T}_A \, \mathbf{J}_A.$$

When $\mathbf{T}(\phi_e)$ is singular (at $s_\vartheta = 0$), the orientation parameterisation hits a **representation singularity** (also called gimbal lock). This is distinct from a kinematic singularity of the arm itself and can be avoided by switching orientation representation mid-trajectory.

5.7 Statics

5.7.1 Wrench and joint torques

At static equilibrium the end-effector exerts a **wrench**

$$\gamma_e = \begin{bmatrix} F_e \\ \mu_e \end{bmatrix}$$

(force F_e and moment μ_e), and the joints carry generalised torques $\tau = [\tau_1, \dots, \tau_n]^{\mathsf{T}}$.

5.7.2 Virtual work duality

Apply virtual joint displacements δq. The virtual work of the joint torques is $\tau^{\mathsf{T}} \delta q$. The virtual work of the wrench is

$$F_e{}^{\mathsf{T}} \, \delta p_e + \mu_e{}^{\mathsf{T}} \, \omega_e \, \delta t = \gamma_e{}^{\mathsf{T}} \, \mathbf{J} \, \delta q.$$

Setting internal and external virtual work equal for arbitrary δq:

$$\boxed{\tau = \mathbf{J}^{\mathsf{T}}(q) \, \gamma_e}$$

The **transpose of the same Jacobian** that maps joint rates to end-effector velocity maps end-effector wrenches to joint torques. This kinematic–static duality is a fundamental result: it means that a manipulator's force capability in task space is directly determined by its velocity mapping in joint space.

In particular, near a singularity the Jacobian loses rank, so the manipulator can exert large forces in certain directions while losing the ability to generate forces in others.