

اصول علم ربات – اسلاید هفتم

Fundamentals of Robotics – Slide 07

Kinematics 1

دکتر مهدی جوانمردی

زمستان ۱۴۰۰

[slides adapted from Gianni Di Caro, @CMU with permission]

Locomotion → Kinematics vs. Dynamics

Locomotion:

Displacement of body / body parts from their original position to a state of *rest*, or of motion that causes **moving** from one point to another. In physical world, it requires application of **forces**

Dynamics:

The study of motion (of a **mass**) through the direct modeling of the **forces** that cause it

Kinematics:

The study of motion **without taking into consideration the forces** that cause it. It is based on *geometric relations, positions, velocities, and accelerations*.

Forward Kinematics:

Use of kinematic equations to **determine / predict** the final configuration/pose of a robot based on the *specification* of the values for the control variables (e.g., $\mathbf{v}, \boldsymbol{\omega}$)

Inverse Kinematics:

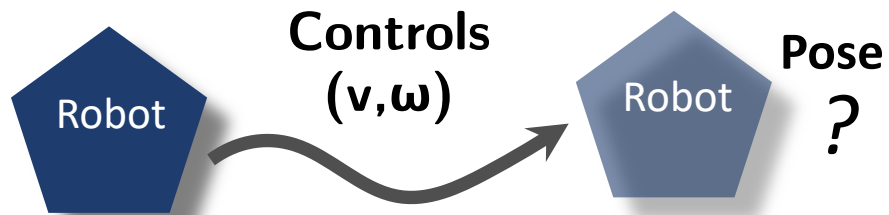
Given the **desired final configuration** (of the effectors/pose), make use of the kinematic equations to *determine the values* of the control variables that allow to achieve it.

Motion planning

The specification of the **entire movement of the robot** in terms of its control variables to achieve the desired configurations in (s, t) .

Motion *control* and Motion *prediction*

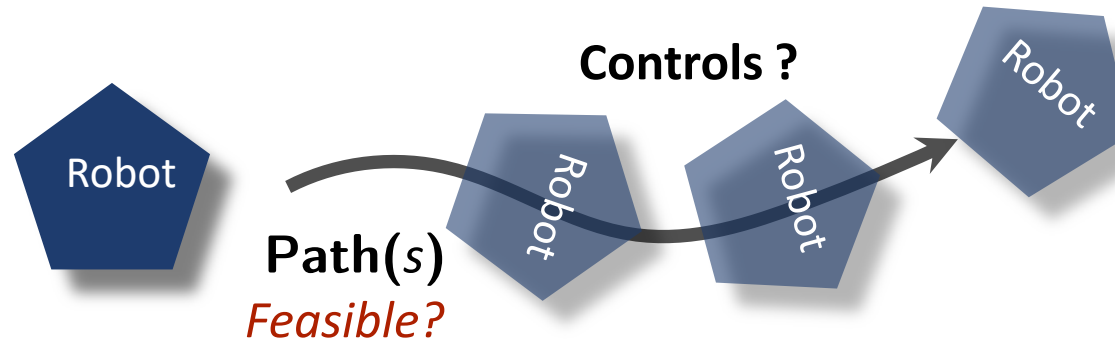
Posture prediction: *Forward Kinematics*



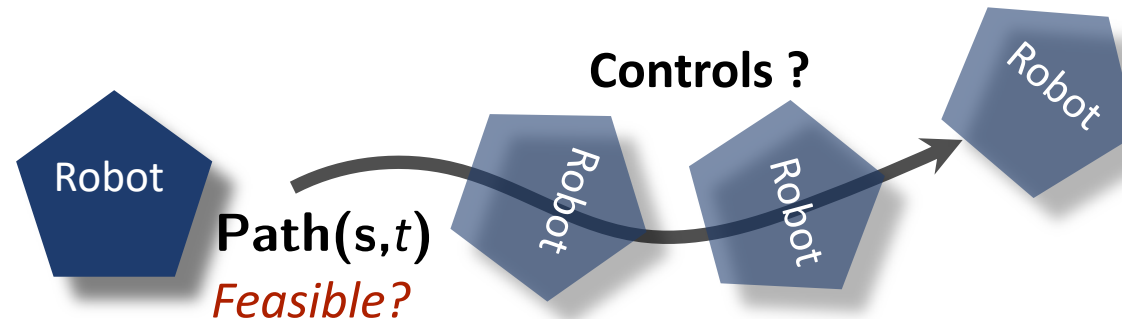
Posture regulation: *Inverse Kinematics*



Path following
(geometry)

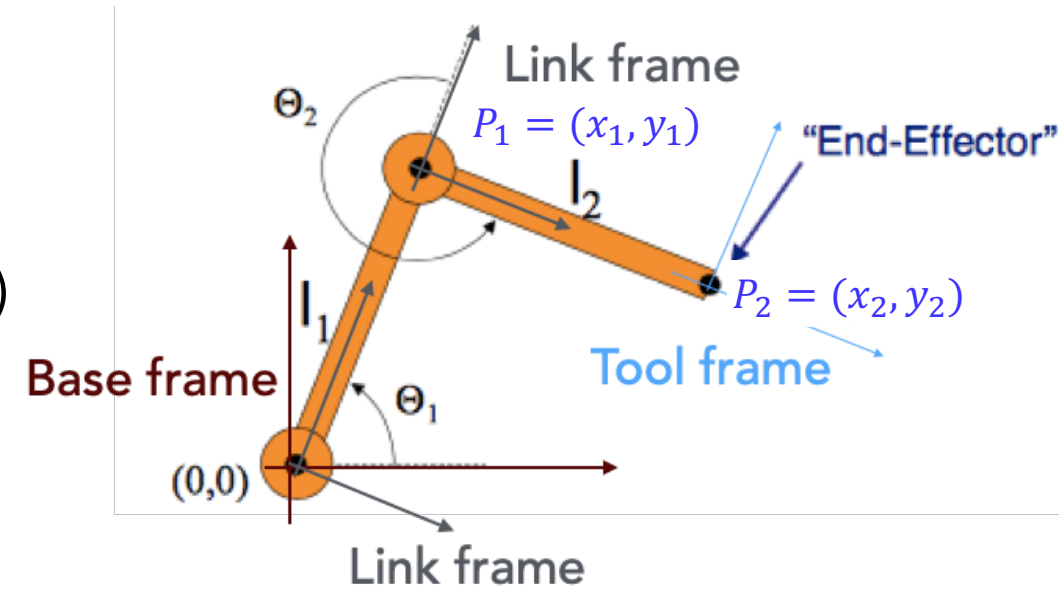


Trajectory following
(physics, time)



Forward kinematics for a (simple) arm robot

- We can control the configuration of the robot using the *actuators*, the **motors that drive the angles of the joints**
- We want to use the robot in its *workspace* (or in the *task space*)
→ Control the pose of the **end-effector**, the *tool*

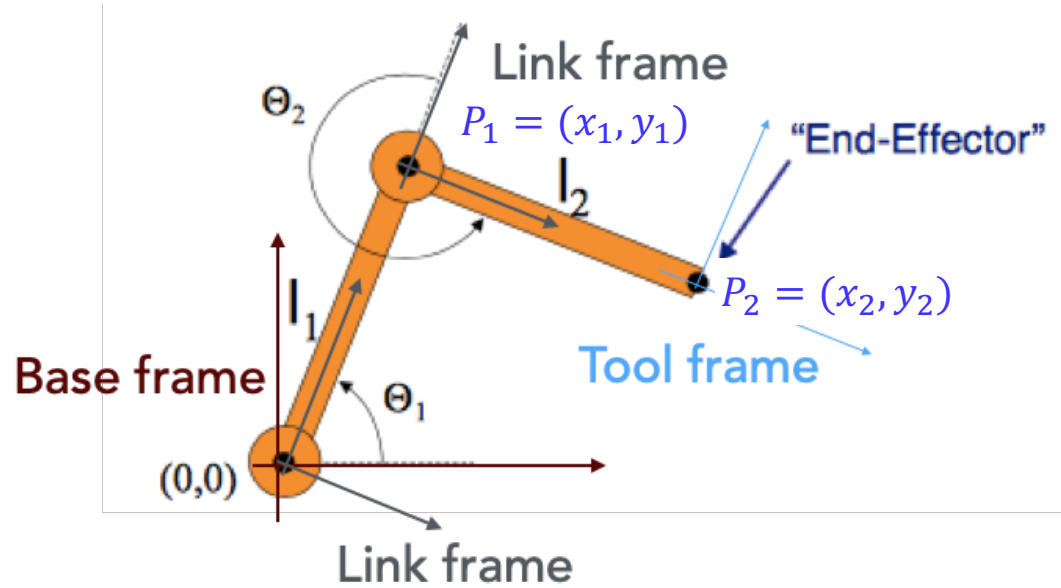


- ❖ **Fundamental problem:** given the the **configuration of the robot**, i.e., the value of the joint angles, determine the **end-effector position, X** , or more in general, the **pose of the end-effector** (the pose of the Tool frame in the Base frame or, in the World frame representing the robot's workspace)

Goal: find pose r of end-effector as a function f of the configuration variables

$$r = f(q) \quad f: \mathbb{R}^n \rightarrow \mathbb{R}^m \quad n = 2, m = 3$$

Forward kinematics for a (simple) arm robot: Find pose



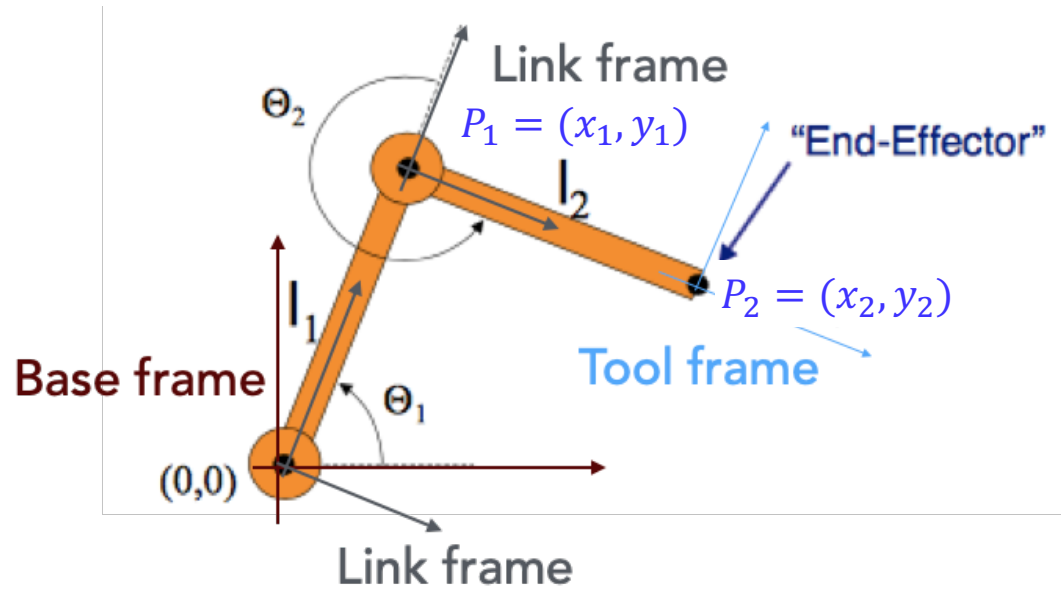
$q = [\theta_1 \ \theta_2]^T$ ✓ Configuration variable, **degrees of freedom that we can control**

$r = [x \ y \ \theta]^T$ ➤ Pose of end effector

$$f: [\theta_1 \ \theta_2] \rightarrow [x \ y \ \theta]$$

- Position $P_1 = (x_1, y_1)$ of first joint between links l_1 and l_2 :
$$\begin{aligned} x_1 &= l_1 \cos \theta_1 \\ y_1 &= l_1 \sin \theta_1 \end{aligned}$$
- Position $P_2 = (x_2, y_2)$ of end-effector:
$$\begin{aligned} x_2 &= x_1 + l_2 \cos(\theta_1 + \theta_2) \\ y_2 &= y_1 + l_2 \sin(\theta_1 + \theta_2) \end{aligned}$$
- Orientation of end-effector is $\theta_1 + \theta_2$

Forward kinematics for a (simple) arm robot: Find pose



$$\mathbf{r} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} x_1 + l_2 \cos(\theta_1 + \theta_2) \\ y_1 + l_2 \sin(\theta_1 + \theta_2) \\ \theta_1 + \theta_2 \end{bmatrix}$$

Forward kinematic equations of the robot relating its control parameters θ_1, θ_2 (joint configuration) to the pose of its end-effector in the local coordinate system defined by the **Base frame**.

Configuration space: $\theta_1 \in [0, \pi] \times \theta_2 \in [-\pi, \pi]$

Configuration space

→ Forward kinematics equations

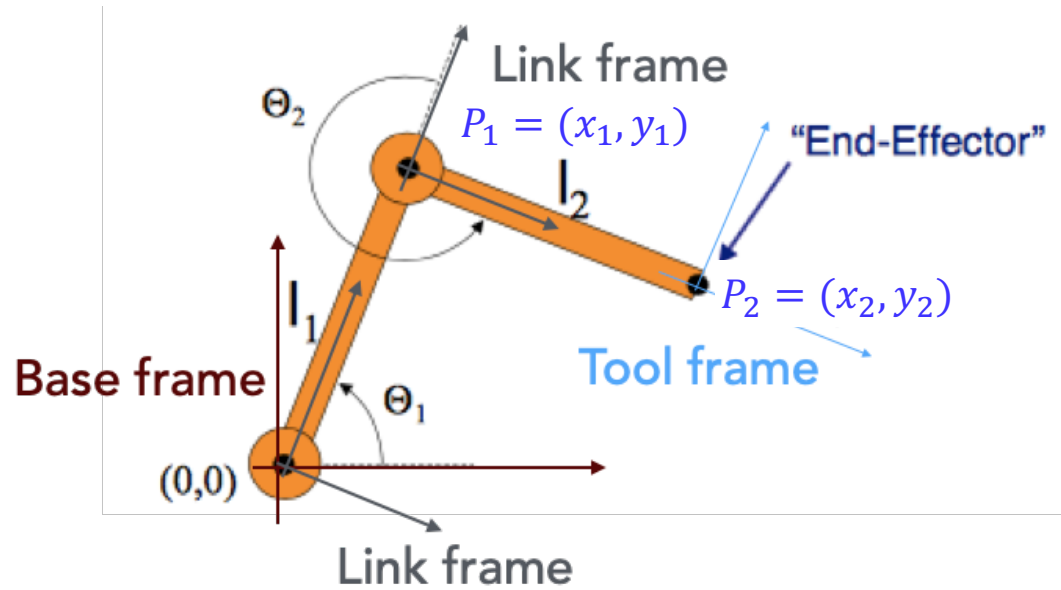
→ Workspace

$$f: [\theta_1 \ \theta_2] \rightarrow [x \ y \ \theta]$$

The Tool frame has a pose which is the result of a rotation $\theta_1 + \theta_2$ and a translation x, y . As a **3D transform**:

$$f(\theta_1, \theta_2) = \begin{bmatrix} c_{\theta_1\theta_2} & -s_{\theta_1\theta_2} & 0 & c_{\theta_1\theta_2}l_2 + c_{\theta_1}l_1 \\ s_{\theta_1\theta_2} & c_{\theta_1\theta_2} & 0 & s_{\theta_1\theta_2}l_2 + s_{\theta_1}l_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Forward kinematics for a (simple) arm robot: Find pose



$$\mathbf{r} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} x_1 + l_2 \cos(\theta_1 + \theta_2) \\ y_1 + l_2 \sin(\theta_1 + \theta_2) \\ \theta_1 + \theta_2 \end{bmatrix}$$

Forward kinematic equations of the robot relating its control parameters θ_1, θ_2 (joint configuration) to the pose of its end-effector in the local coordinate system defined by the **Base frame**.

Configuration space: $\theta_1 \in [0, \pi] \times \theta_2 \in [-\pi, \pi]$

The Tool frame has a pose which is the result of a rotation $\theta_1 + \theta_2$ and a translation x, y . As a **3D transform**:

Configuration space

→ Forward kinematics equations

→ Workspace

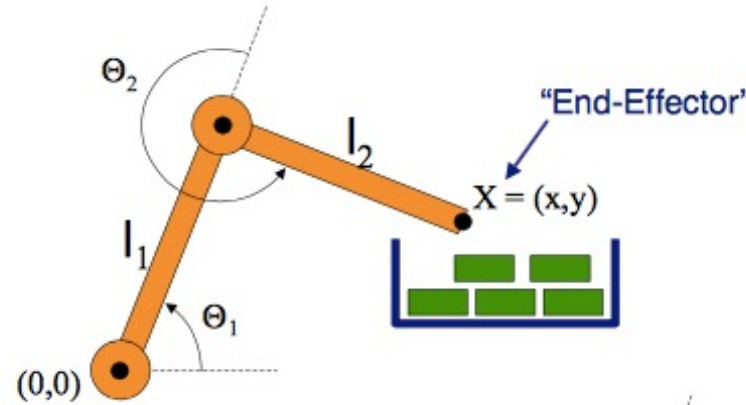
$$f: [\theta_1 \ \theta_2] \rightarrow [x \ y \ \theta]$$

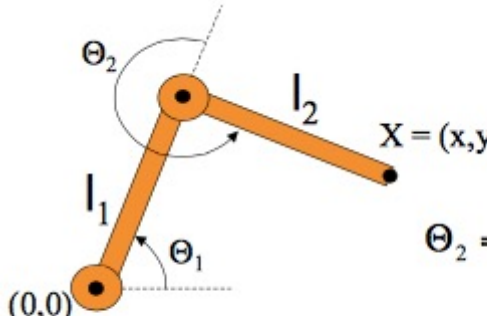
$$f(\theta_1, \theta_2) = \begin{bmatrix} c_{\theta_1\theta_2} & -s_{\theta_1\theta_2} & 0 & c_{\theta_1\theta_2}l_2 + c_{\theta_1}l_1 \\ s_{\theta_1\theta_2} & c_{\theta_1\theta_2} & 0 & s_{\theta_1\theta_2}l_2 + s_{\theta_1}l_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

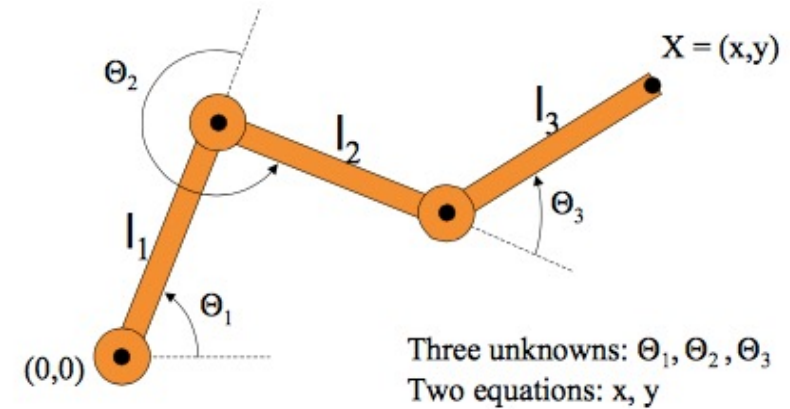
Inverse kinematics for an arm robot

Given the **desired final pose** of the end-effector,
determine the **values for the joint variables** that allow to achieve it

$$q = f^{-1}(r), \quad f^{-1}: \mathbb{R}^m \rightarrow \mathbb{R}^n$$




$$\Theta_2 = \cos^{-1} \left(\frac{x^2 + y^2 - l_1^2 - l_2^2}{2l_1l_2} \right)$$
$$\Theta_1 = \frac{-(l_2 \sin(\Theta_2)x + (l_1 + l_2 \cos(\Theta_2))y)}{(l_2 \sin(\Theta_2))y + (l_1 + l_2 \cos(\Theta_2))x}$$



Solve the kinematic equations w.r.t. the final pose
What if the desired configuration must be achieved
from an initial pose (and constraints are present)?

Usually the problem admits
multiple solutions

Arm robots: more to be done ...

- ❖ Our simple derivation of the forward kinematics equations **do not scale to complex robot chains!**
 - Denavit-Hartenberg notation
- ❖ Inverse kinematics: equations are heavily non-linear and admit multiple solutions → **A difficult problem of mathematical optimization!**

To be continued ...

Multivariate vector maps

$$\mathbf{r} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} x_1 + l_2 \cos(\theta_1 + \theta_2) \\ y_1 + l_2 \sin(\theta_1 + \theta_2) \\ \theta_1 + \theta_2 \end{bmatrix} \quad \mathbf{r} = f(q) \quad f: [\theta_1 \ \theta_2] \rightarrow [x \ y \ \theta]$$

$$f: \mathbb{R}^n \rightarrow \mathbb{R}^m \quad n = 2, m = 3$$

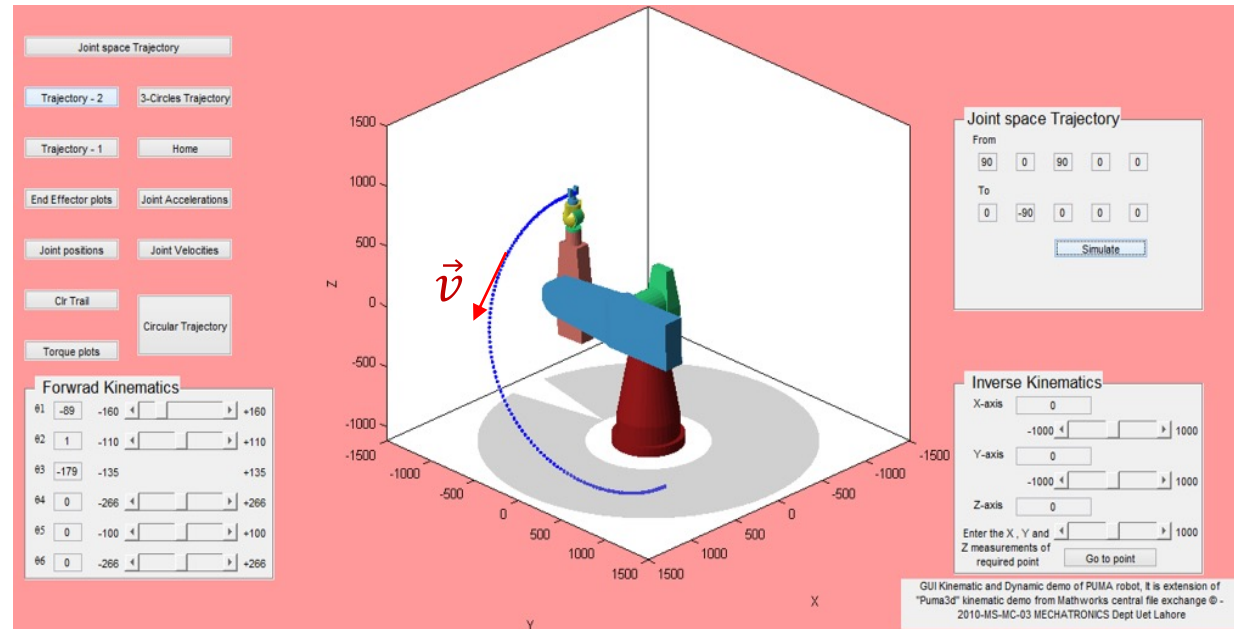
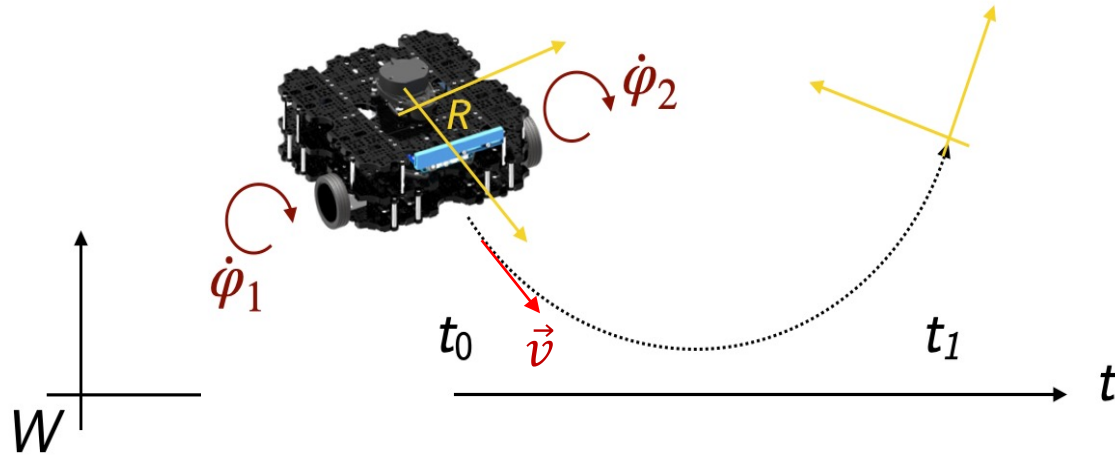
f is a **multivariate, vector function**:

- **Multivariate**: a function of **multiple variables** ($n = 2$)
- **Vector**: the codomain is a vector space, the output value is not a scalar, it consists instead of 3 values (x, y, θ) , such that we can write f as a vector of **three function components**: $f \equiv [f_x \ f_y \ f_\theta]$, where each function component is a multivariate, **scalar** function

$$f \equiv \begin{cases} f_x(\theta_1, \theta_2) = x_1 + l_2 \cos(\theta_1 + \theta_2) = l_1 \cos \theta_1 + l_2 \cos(\theta_1 + \theta_2) \\ f_y(\theta_1, \theta_2) = y_1 + l_2 \sin(\theta_1 + \theta_2) = l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2) \\ f_\theta(\theta_1, \theta_2) = \theta_1 + \theta_2 \end{cases} \quad \begin{cases} f_x: [\theta_1 \ \theta_2] \rightarrow \mathbb{R} \\ f_y: [\theta_1 \ \theta_2] \rightarrow \mathbb{R} \\ f_\theta: [\theta_1 \ \theta_2] \rightarrow \mathbb{R} \end{cases}$$

Differential kinematics

- So far, only the **geometry of motion** has been considered, in the **space of positions**
 - What about the **temporal evolution** of the robot's configuration?
- ❖ **Forward differential kinematics:** compute the relationship between the generalized velocities at the joints (i.e., the speed/rotation of our motors or wheels) and the **velocity \vec{v} of the robot's end-effector (arm) or reference point (mobile chassis) in the workspace**



Forward differential kinematics

- ❖ **Forward differential kinematics**: compute the relationship between the generalized velocities at the joints (i.e., the speed/rotation of our motors or wheels) and the **velocity \vec{v} of the robot's end-effector (arm) or reference point (mobile chassis) in the workspace**

Generalize coordinates and velocities, in configuration space: our **controls**

$$q = [q_1, \dots, q_n]^T$$

$$\dot{q} = [\dot{q}_1, \dots, \dot{q}_n]^T$$

Velocity in the 3D space, in the Inertial World frame

Translational (linear)
velocity

$$v = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}$$

Angular (rotational)
velocity

$$\omega = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

Twist vector

$$\nu = [v \quad \omega]^T$$

Differential forward kinematics: the scenario

Forward kinematics:

$$\mathbf{r} = f(\mathbf{q})$$

$$\begin{bmatrix} x \\ y \\ z \\ \varphi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} f_x(\mathbf{q}) \\ f_y(\mathbf{q}) \\ f_z(\mathbf{q}) \\ f_\varphi(\mathbf{q}) \\ f_\theta(\mathbf{q}) \\ f_\psi(\mathbf{q}) \end{bmatrix}$$

Generalized coordinates become **function of time**: based on our controls, they change over time, $\mathbf{q} \equiv \mathbf{q}(t) \rightarrow \mathbf{f} \equiv \mathbf{f}(t)$

Differential Forward kinematics:

→ **Derivatives** w.r.t. time → $\dot{\mathbf{r}} = \dot{\mathbf{f}}(\mathbf{q}(t))$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \boldsymbol{\nu} = \begin{bmatrix} v \\ \omega \end{bmatrix} = \dot{\mathbf{f}}(\mathbf{q})$$

How do we compute the time derivative of $\mathbf{f}(\mathbf{q})$?

Time derivative of mapping $f(q(t))$

$$f: q = [q_1, q_2, \dots, q_n] \rightarrow \begin{bmatrix} f_x(q) \\ f_y(q) \\ f_z(q) \\ f_\varphi(q) \\ f_\theta(q) \\ f_\psi(q) \end{bmatrix} \quad \begin{matrix} f_i \equiv f_i(q_1, q_2, \dots, q_n) \\ f_i: [q_1, q_2, \dots, q_n] \rightarrow \mathbb{R} \end{matrix} \quad \boxed{q = q(t)} \quad \dot{f}(q) = \frac{df(q(t))}{dt} = \begin{bmatrix} \dot{f}_x(q(t)) \\ \dot{f}_y(q(t)) \\ \dot{f}_z(q(t)) \\ \dot{f}_\varphi(q(t)) \\ \dot{f}_\theta(q(t)) \\ \dot{f}_\psi(q(t)) \end{bmatrix} \quad ??$$

- Let's consider the simple, specific case scenario that we have addressed before, and make θ_1 and θ_2 functions of time (subject to robot's controls) $\rightarrow f$ becomes a composite function of time, t :

$$f \equiv \begin{cases} f_x(\theta_1(t), \theta_2(t)) = l_1 \cos \theta_1(t) + l_2 \cos(\theta_1(t) + \theta_2(t)) \\ f_y(\theta_1(t), \theta_2(t)) = l_1 \sin(\theta_1(t)) + l_2 \sin(\theta_1(t) + \theta_2(t)) \\ f_\theta(\theta_1(t), \theta_2(t)) = \theta_1(t) + \theta_2(t) \end{cases}$$

- ✓ Each f_i component is a:
 - **composite** function
 - **multivariate** function
- ✓ f is a **vector** function

Calculus issues

$$f \equiv \begin{cases} f_x(\theta_1(t), \theta_2(t)) = l_1 \cos \theta_1(t) + l_2 \cos(\theta_1(t) + \theta_2(t)) \\ f_y(\theta_1(t), \theta_2(t)) = l_1 \sin(\theta_1(t)) + l_2 \sin(\theta_1(t) + \theta_2(t)) \\ f_\theta(\theta_1(t), \theta_2(t)) = \theta_1(t) + \theta_2(t) \end{cases}$$

General case

$$\dot{f}(q) = \frac{df(q(t))}{dt} = \begin{bmatrix} \dot{f}_x(q(t)) \\ \dot{f}_y(q(t)) \\ \dot{f}_z(q(t)) \\ \dot{f}_\phi(q(t)) \\ \dot{f}_\theta(q(t)) \\ \dot{f}_\psi(q(t)) \end{bmatrix}$$

✓ Each f 's component, f_i , is a:

- **composite** function
- **multivariate** function

✓ f is a **vector** function

- What is a composite function?
- How do we compute the derivative of a composite function?
- How do we compute the derivative of a multivariate function?
- How do we compute the derivative of a vector function?

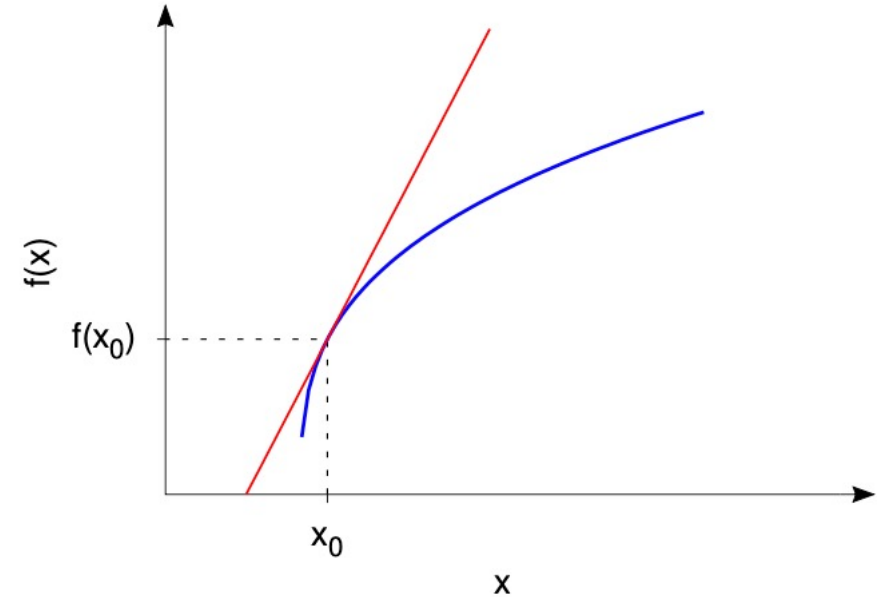
Let's make a short calculus detour

Derivatives: rate of change for scalar functions of ONE variable

$$f: \mathbb{R} \rightarrow \mathbb{R}$$

Derivative: Instantaneous rate of change of $f(x)$ along x axis
(the only coordinate axis!)

$$f'(x)|_{x_0} = \lim_{h \rightarrow 0} \frac{f(x_0 + h) - f(x_0)}{h}$$

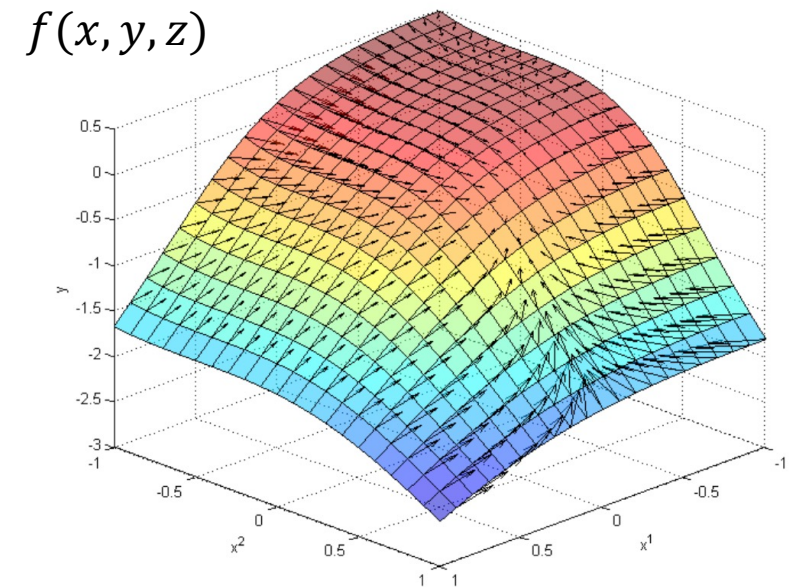


- $f'(x_0) \in \mathbb{R} \rightarrow$ It's a scalar, a number \rightarrow Slope of the tangent line in x_0
- Slope / derivative is zero when rate of change is locally zero
- A scalar is a **vector** of dimension 1 (we only have ONE coordinate axis to consider for rate of change of the function $f(x)$!)

Derivatives for multivariate functions: Gradients

- $f: \mathbb{R}^n \rightarrow \mathbb{R}$ → Need to compute the **rate of change** of the function long each coordinate axis, i.e., for each one of the n **variables**
- → *Derivative* is expressed by a **Gradient vector** $\nabla f(\mathbf{x}): \mathbb{R}^n \rightarrow \mathbb{R}^n$ where each component of the vector represents the 'derivative' of f w.r.t. ONE specific variable, keeping all the other variables constant (i.e., *freezing* them)

$$f(\mathbf{x}) : [\mathbf{x}_1 \ \mathbf{x}_2 \ \dots \ \mathbf{x}_n]^T \in \mathbb{R}^n \rightarrow \mathbb{R} \quad \nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial \mathbf{x}_1} \\ \vdots \\ \frac{\partial f}{\partial \mathbf{x}_n} \end{bmatrix}$$



Gradient vector: **rate of change** of a **multivariate** function $f(x_1, x_2, \dots, x_n)$ **along ALL the n coordinate axes \hat{x}_i** . It is an n -dimensional vector of the **partial derivatives** of the function long all axes

Partial derivatives for multivariate functions

- **Partial derivative:** Rate of change of a generic multivariate function $f(x_1, x_2, \dots, x_n)$ along one specific coordinate axis. The rate of change along the x_i axis, It is indicated by:

$$\frac{\partial f(x_1, x_2, \dots, x_i, \dots, x_n)}{\partial x_i}$$

E.g., $\frac{\partial f}{\partial x_2}$ indicates the rate of change along the x_2 axis.

- ✓ **In practice**, a partial derivative is a (regular) derivative computed along the specified axis keeping the values along the other axes constant

Partial derivatives: how to compute

$$f(x_1, x_2) = x_1^2 + 3x_2^3 \quad \frac{\partial f}{\partial x_1} = 2x_1 \quad \frac{\partial f}{\partial x_2} = 9x_2^2$$

$$f(x_1, x_2, x_3) = x_1^2 x_2 x_3 + 3x_1 x_2 \quad \frac{\partial f}{\partial x_1} = 2x_1 x_2 x_3 + 3x_2 \quad \frac{\partial f}{\partial x_2} = x_1^2 x_3 + 3x_1 \quad \frac{\partial f}{\partial x_3} = x_1^2 x_2$$

$$f(x_1, x_2, x_3, x_4) = 3 \frac{\cos(x_1 x_4) \sin(x_2^5)}{e^{x_2} + (1 + x_2^2)/(x_1 x_2 x_4)} + 5x_1 x_3 x_4$$

$$\frac{\partial f}{\partial x_3}(a, b, c, d) \quad ? \quad \text{The value of the partial derivative in one specific point}$$

$$\frac{\partial f}{\partial x_3}(x_1, x_2, x_3, x_4) = 5x_1 x_4$$

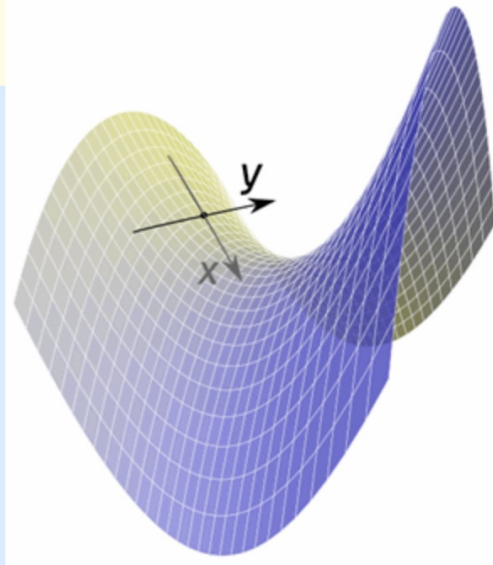
$$\frac{\partial f}{\partial x_3}(a, b, c, d) = 5ad.$$

Gradients vectors

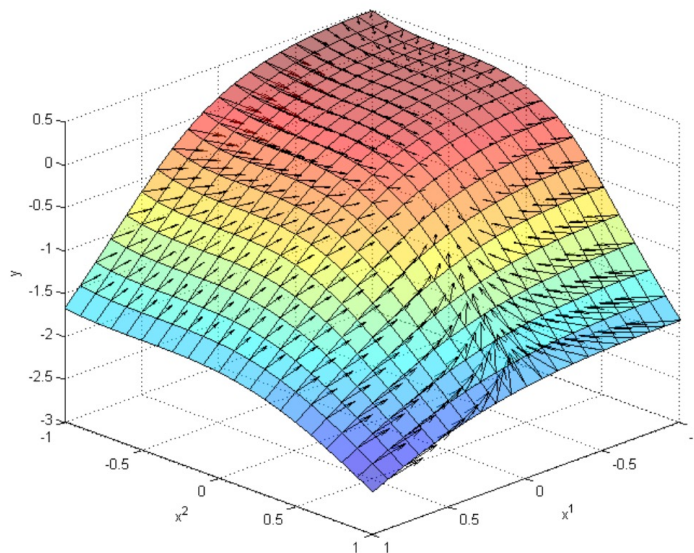
Example: a function for a surface that depends on two variables \mathbf{x} and \mathbf{y}

When we find the slope in the \mathbf{x} direction (while keeping \mathbf{y} fixed) we have found a partial derivative.

Or we can find the slope in the \mathbf{y} direction (while keeping \mathbf{x} fixed).



$$\nabla f(x_1, x_2) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{bmatrix}$$



- The gradient vectors at each point define a **vector field**
- At each point, the Gradient vector is **orthogonal** to the plane tangent to the surface

Derivative for (univariate) composite functions and chain rule

- A **composite function** is the combination of two functions: a function that takes as input the output of another function



E.g., $f(x) = 2x + 1$, $g(z) = z^4$ $h(x) = (2x + 1)^4$

Let's call $u = f(x)$ the **output of the inner function** $\rightarrow h(x) = g(u)$

$$h' = \frac{dh}{dx} = \frac{dh}{du} \frac{du}{dx}$$

- **Chain rule of derivation**: compute the variation/derivative of the **composite** function h w.r.t. the independent variable x as the product of the variation of h w.r.t. u , the intermediate output, and the variation of u w.r.t. to x
- We can also say that the chain rule relates the output h to the input via an intermediate variable u representing the output of the inner function

$$\frac{dh}{du} = 4(2x + 1)^3$$

Derivative of outer part of $h(x)$

$$\frac{du}{dx} = 2$$

Derivative of inner part of $h(x)$

$$h' = 8(2x + 1)^3$$

Time derivative for multivariate composite functions $f_i(q_1(t), q_2(t), \dots, q_n(t))$

- Previous example is for *univariate scalar functions*
- ✓ Can generalize to **multivariate functions**, such as ours $f_i(q(t)) = f_i(q_1(t), q_2(t), \dots, q_n(t))$
- Need to use **partial derivatives** of f_i w.r.t. each generalized variable $q_i(t)$ and use **linearity** of derivation

$$\dot{f}_i = \frac{df_i}{dt} = \frac{\partial f_i}{\partial q_1} \frac{dq_1}{dt} + \frac{\partial f_i}{\partial q_2} \frac{dq_2}{dt} + \dots + \frac{\partial f_i}{\partial q_n} \frac{dq_n}{dt} = \sum_{j=1}^n \frac{\partial f_i}{\partial q_j} \frac{dq_j}{dt}$$

- We can also write it in a more compact vector-product form:

$$\dot{f}_i = \frac{df_i}{dt} = \begin{bmatrix} \frac{\partial f_i}{\partial q_1} & \frac{\partial f_i}{\partial q_2} & \dots & \frac{\partial f_i}{\partial q_n} \end{bmatrix} \begin{bmatrix} \frac{dq_1}{dt} \\ \frac{dq_2}{dt} \\ \vdots \\ \frac{dq_n}{dt} \end{bmatrix}$$

Differential kinematics, Jacobian matrix

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} \dot{f}_x(q(t)) \\ \dot{f}_y(q(t)) \\ \dot{f}_z(q(t)) \\ \dot{f}_\varphi(q(t)) \\ \dot{f}_\theta(q(t)) \\ \dot{f}_\psi(q(t)) \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{\partial x}{\partial q_1} & \frac{\partial x}{\partial q_2} & \dots & \frac{\partial x}{\partial q_n} \\ \frac{\partial y}{\partial q_1} & \frac{\partial y}{\partial q_2} & \dots & \frac{\partial y}{\partial q_n} \\ \frac{\partial z}{\partial q_1} & \frac{\partial z}{\partial q_2} & \dots & \frac{\partial z}{\partial q_n} \\ \frac{\partial \omega_x}{\partial q_1} & \frac{\partial \omega_x}{\partial q_2} & \dots & \frac{\partial \omega_x}{\partial q_n} \\ \frac{\partial \omega_y}{\partial q_1} & \frac{\partial \omega_y}{\partial q_2} & \dots & \frac{\partial \omega_y}{\partial q_n} \\ \frac{\partial \omega_z}{\partial q_1} & \frac{\partial \omega_z}{\partial q_2} & \dots & \frac{\partial \omega_z}{\partial q_n} \end{bmatrix}}_{J(q)} \underbrace{\begin{bmatrix} \frac{dq_1}{dt} \\ \frac{dq_2}{dt} \\ \vdots \\ \frac{dq_n}{dt} \end{bmatrix}}_{\dot{q}} = J(q)\dot{q}$$

Jacobian matrix
(of forward kinematic map)

Differential kinematics, Jacobian matrix

Jacobian matrix

$$J(q) = \begin{bmatrix} \frac{\partial x}{\partial q_1} & \frac{\partial x}{\partial q_2} & \cdots & \frac{\partial x}{\partial q_n} \\ \frac{\partial y}{\partial q_1} & \frac{\partial y}{\partial q_2} & \cdots & \frac{\partial y}{\partial q_n} \\ \frac{\partial z}{\partial q_1} & \frac{\partial z}{\partial q_2} & \cdots & \frac{\partial z}{\partial q_n} \\ \frac{\partial \omega_x}{\partial q_1} & \frac{\partial \omega_x}{\partial q_2} & \cdots & \frac{\partial \omega_x}{\partial q_n} \\ \frac{\partial \omega_y}{\partial q_1} & \frac{\partial \omega_y}{\partial q_2} & \cdots & \frac{\partial \omega_y}{\partial q_n} \\ \frac{\partial \omega_z}{\partial q_1} & \frac{\partial \omega_z}{\partial q_2} & \cdots & \frac{\partial \omega_z}{\partial q_n} \end{bmatrix}$$

- Each **column** quantifies how each component of velocity instantaneously changes when one specific configuration variable (e.g., an angle) is infinitesimally changed
- Each **row** quantifies how the change in each configuration variable affects one particular component of velocity.

Differential forward kinematics: velocity vs. changes in controls q

$$\nu = [v \quad \omega]^T = J(q) \cdot [\dot{q}_1, \dots, \dot{q}_n]^T = J(q) \cdot \dot{q}$$

$$\nu = \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial q_1} & \frac{\partial x}{\partial q_2} & \cdots & \frac{\partial x}{\partial q_n} \\ \frac{\partial y}{\partial q_1} & \frac{\partial y}{\partial q_2} & \cdots & \frac{\partial y}{\partial q_n} \\ \frac{\partial z}{\partial q_1} & \frac{\partial z}{\partial q_2} & \cdots & \frac{\partial z}{\partial q_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial \omega_x}{\partial q_1} & \frac{\partial \omega_x}{\partial q_2} & \cdots & \frac{\partial \omega_x}{\partial q_n} \\ \frac{\partial \omega_y}{\partial q_1} & \frac{\partial \omega_y}{\partial q_2} & \cdots & \frac{\partial \omega_y}{\partial q_n} \\ \frac{\partial \omega_z}{\partial q_1} & \frac{\partial \omega_z}{\partial q_2} & \cdots & \frac{\partial \omega_z}{\partial q_n} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{bmatrix} = J(q) \cdot \dot{q}$$

- ❖ The Jacobian matrix expresses the relationship between **velocities in the joint (configuration) space** and **velocities in the end-effector space (workspace)**