



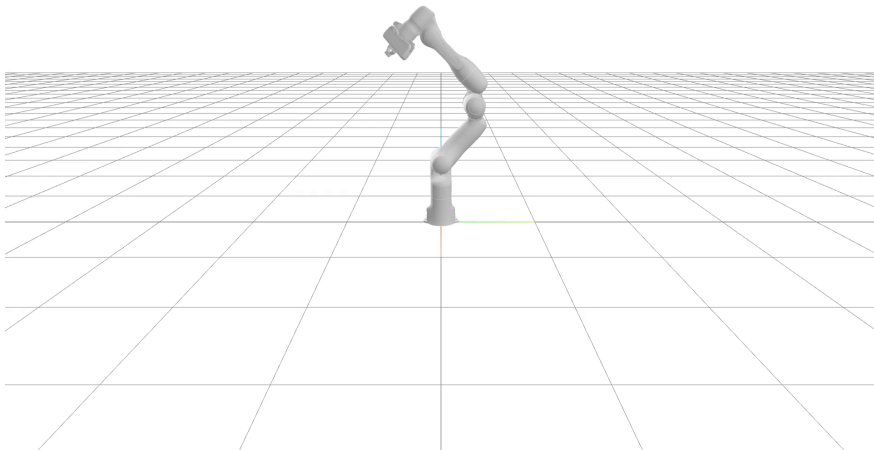
Robotics: Differential Kinematics and Statics

Vladimír Petřík

vladimir.petrik@cvut.cz

09.10.2023

Motivation



Differential kinematics

- ▶ We know how to compute end-effector pose from the configuration
 - ▶ forward kinematics
 - ▶ $\mathbf{x}(t) = f_{fk}(\mathbf{q}(t))$
 - ▶ $\mathbf{x}(t)$ is expressed in task-space, *i.e.* $SE(2)$, $SE(3)$, or \mathbb{R}^2 , \mathbb{R}^3 for position only
 - ▶ $\mathbf{q}(t) \in \mathbb{R}^N$ is configuration (joint space)
 - ▶ t represents time
- ▶ Differential kinematics
 - ▶ relates end-effector velocity to joint velocities
 - ▶ $\dot{\mathbf{x}} = \frac{d\mathbf{x}(t)}{dt} \in \mathbb{R}^M$
 - ▶ Jacobian of the manipulator is core structure in the analysis



Jacobian

Forward kinematics:

$$\mathbf{x}(t) = f_{\text{fk}}(\mathbf{q}(t))$$

Jacobian:

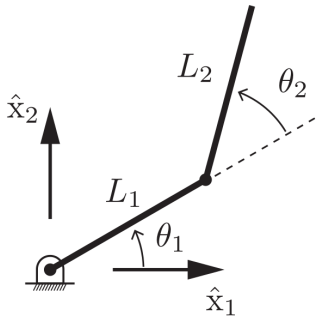
$$\begin{aligned}\dot{\mathbf{x}} &= \frac{d\mathbf{x}(t)}{dt} \\ &= \frac{\partial f_{\text{fk}}(\mathbf{q})}{\partial \mathbf{q}} \frac{d\mathbf{q}(t)}{dt} \\ &= \frac{\partial f_{\text{fk}}(\mathbf{q})}{\partial \mathbf{q}} \dot{\mathbf{q}} \\ &= J(\mathbf{q})\dot{\mathbf{q}}\end{aligned}$$

$$J(\mathbf{q}) = \frac{\partial f_{\text{fk}}(\mathbf{q})}{\partial \mathbf{q}} \in \mathbb{R}^{M \times N}$$



Planar robot example

- ▶ FK: $\mathbf{q} = (\theta_1, \theta_2)^\top \rightarrow (x, y)^\top$
 - ▶ $x = L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2)$
 - ▶ $y = L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2)$
- ▶ $\dot{\mathbf{x}} = ?$
 - ▶ $\dot{x}_1 = -L_1 \dot{\theta}_1 \sin \theta_1 - L_2(\dot{\theta}_1 + \dot{\theta}_2) \sin(\theta_1 + \theta_2)$
 - ▶ $\dot{y}_1 = L_1 \dot{\theta}_1 \cos \theta_1 + L_2(\dot{\theta}_1 + \dot{\theta}_2) \cos(\theta_1 + \theta_2)$
 - ▶ $J(\mathbf{q}) = \begin{pmatrix} -L_1 \sin \theta_1 - L_2 \sin(\theta_1 + \theta_2) & -L_2 \sin(\theta_1 + \theta_2) \\ L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) & L_2 \cos(\theta_1 + \theta_2) \end{pmatrix}$
 - ▶ Jacobian depends on the configuration \mathbf{q}

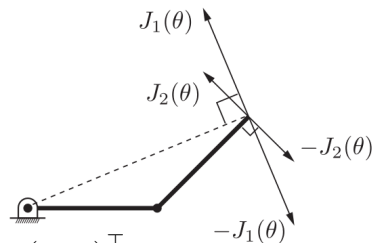


Jacobian dimension

- ▶ $J(\mathbf{q}) = \frac{\partial f_{fk}(\mathbf{q})}{\partial \mathbf{q}} \in \mathbb{R}^{M \times N}$
- ▶ M task-space DoF
- ▶ N joint-space DoF
- ▶ Redundant robots: $N > M$
- ▶ Under-actuated robots: $N < M$
- ▶ 2 DoF robot with translation task space: 2×2
- ▶ 2 DoF robot with $SE(2)$ task space: 3×2
- ▶ 5 DoF robot with $SE(2)$ task space: 3×5
- ▶ 6 DoF robot with $SE(3)$ task space: 6×6
- ▶ 7 DoF robot with $SE(3)$ task space: 6×7

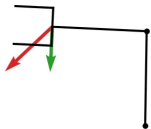


Jacobian properties



- ▶ $J(\mathbf{q}) = (J_1(\mathbf{q}) \quad J_2(\mathbf{q}))$
- ▶ First column corresponds to the end-point velocity for $\dot{\mathbf{q}} = (1 \quad 0)^\top$
- ▶ Second column corresponds to the end-point velocity for $\dot{\mathbf{q}} = (0 \quad 1)^\top$
- ▶ $\dot{\mathbf{x}} = \mathbf{v}_{\text{tip}} = J_1(\mathbf{q})\dot{\theta}_1 + J_2(\mathbf{q})\dot{\theta}_2$
- ▶ We can generate tip velocity in any direction if $J_1(\mathbf{q})$ and $J_2(\mathbf{q})$ are not collinear
 - ▶ when they are collinear? e.g. $\theta_2 = 0$
 - ▶ Jacobian is singular matrix \rightarrow configurations are called **singularities**
 - ▶ rank of Jacobian is not maximal
 - ▶ end-effector is unable to generate velocity in a certain direction

Jacobian columns visualization



How to compute jacobian numerically

- ▶ Finite difference method

- ▶ $f'(x_0) \approx \frac{f(x_0+\delta)-f(x_0)}{\delta}, \quad \delta \rightarrow 0$

- ▶ $J = \begin{pmatrix} \frac{\partial x}{\partial q_0} & \frac{\partial x}{\partial q_1} & \dots \\ \frac{\partial y}{\partial q_0} & \frac{\partial y}{\partial q_1} & \dots \\ \frac{\partial \theta}{\partial q_0} & \frac{\partial \theta}{\partial q_1} & \dots \end{pmatrix}$

- ▶ $\frac{\partial x}{\partial q_0}(\mathbf{q}) \approx \frac{f_{fk,x}(\mathbf{q}+\boldsymbol{\delta})-f_{fk,x}(\mathbf{q})}{\delta}, \quad \boldsymbol{\delta} = (\delta \quad 0 \quad \dots)^\top$

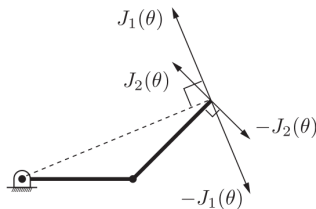
- ▶ Repeat for every element of J

- ▶ Slow to compute, easy to implement → used in testing



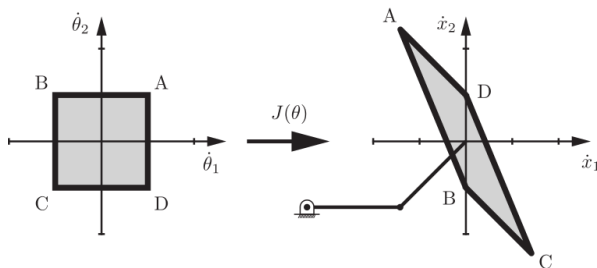
How to compute jacobian analytically

- ▶ $J = (J_v \ J_w)^\top$ i.e. translation and rotation part
- ▶ Translation part:
 - ▶ i -th column (\mathbf{n}_S) is perpendicular to vector \mathbf{t} , connecting i -th joint to end-effector
 - ▶ S - reference frame, J - frame attached to i -th joint, E end-effector frame
 - ▶ \mathbf{t}_{JE} - translation part of $T_{JE} \in SE(2)$
 - ▶ $\mathbf{n} = R(90)\mathbf{t}_{JE}$ - perpendicular vector
 - ▶ $\mathbf{n}_S = R_{SJ}\mathbf{n}$ - change of reference frame
 - ▶ For prismatic joints: $\mathbf{n}_S = R_{SJ}\mathbf{a}$
 - ▶ \mathbf{a} is axis of translation
- ▶ Rotation part
 - ▶ 1 for revolute joints
 - ▶ 0 for prismatic joints



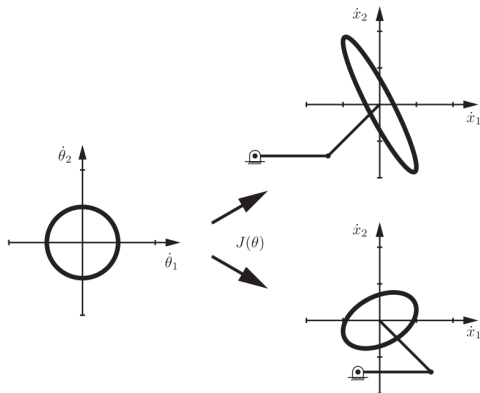
Jacobian application - velocity limits

- ▶ $\dot{\mathbf{x}} = J(\mathbf{q})\dot{\mathbf{q}}$
- ▶ Velocity limits are given for each joint
 - ▶ configuration independent
- ▶ What are the velocity we can achieve with end-effector?
 - ▶ depends on configuration
 - ▶ use jacobian to map joint-space velocity to task-space velocity



Manipulability ellipsoid

- ▶ Unit circle in joint velocity space, *i.e.* $\|\dot{\mathbf{q}}\| = 1$
- ▶ Mapping through Jacobian to ellipsoid in end-effector space
- ▶ Closer the ellipsoid is to sphere, more easily can end-effector move in arbitrary direction



How to compute manipulability ellipsoid

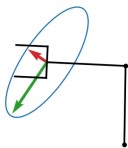
- ▶ If $J(\mathbf{q})$ is non-singular
- ▶ Solution to $\mathbf{u}^\top A^{-1} \mathbf{u} = 1$ is ellipsoid
 - ▶ eigen vectors of A show directions of principal axes of the ellipsoid
 - ▶ square roots of eigen values are lengths of the principal axis

$$\begin{aligned} 1 &= \|\dot{\mathbf{q}}\| \\ &= \dot{\mathbf{q}}^\top \dot{\mathbf{q}} \\ &= (J(\mathbf{q})^{-1} \dot{\mathbf{x}})^\top (J(\mathbf{q})^{-1} \dot{\mathbf{x}}) \\ &= \dot{\mathbf{x}}^\top J(\mathbf{q})^{-\top} J(\mathbf{q})^{-1} \dot{\mathbf{x}} \\ &= \dot{\mathbf{x}}^\top \left(J(\mathbf{q}) J(\mathbf{q})^\top \right)^{-1} \dot{\mathbf{x}} \end{aligned}$$



Manipulability ellipsoid example

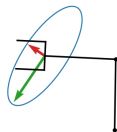
- ▶ 2 DoF robot, translation only, $\text{eig}(JJ^T)$



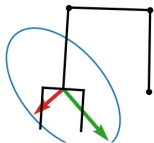
How close we are to singularity?

$$\begin{aligned}\mu_1 &= 7.2522 \\ \mu_2 &= 0.2499\end{aligned}$$

- ▶ Condition number of JJ^T
 - ▶ $\mu_1 = \frac{\lambda_{\max}(JJ^T)}{\lambda_{\min}(JJ^T)} \geq 1$
 - ▶ λ is eigen value of a given matrix
 - ▶ the larger μ_1 is, the closer to singularity we are
 - ▶ **Small μ_1 is preferred**
- ▶ Volume of manipulability ellipsoid
 - ▶ the smaller volume is, the closer to singularity we are
 - ▶ $\mu_2 = \sqrt{\lambda_1 \lambda_2 \cdots} = \det(JJ^T)$
 - ▶ **Large μ_2 is preferred**

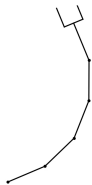
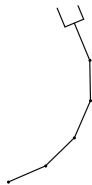
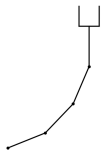


Redundant robots and singularities



Null-space of jacobian

- ▶ $\text{Null}(A) = \ker(A) = \{\mathbf{x} \mid A\mathbf{x} = \mathbf{0}\}$
- ▶ Find $\dot{\mathbf{q}}$ s.t. $\dot{\mathbf{x}} = \mathbf{0}$
 - ▶ $\dot{\mathbf{q}}_{\text{null}} \in \ker(J)$
 - ▶ there are multiple solutions if we have more DoF



Statics analysis

- ▶ Conservation of power: (power at the joints) = (power to move the robot) + (power at the end-effector)
- ▶ Static equilibrium: no power is used to move the robot, *i.e.* no motion
 - ▶ (power at the joints) = (power at the end-effector)
 - ▶ $\boldsymbol{\tau}^\top \dot{\boldsymbol{q}} = F^\top \dot{\boldsymbol{x}}$
 - ▶ $\boldsymbol{\tau}$ joint torques
 - ▶ $\dot{\boldsymbol{q}}$ joint velocities
 - ▶ F end-effector force
 - ▶ $\dot{\boldsymbol{x}}$ end-effector velocity
 - ▶ $\dot{\boldsymbol{x}} = J(\boldsymbol{q})\dot{\boldsymbol{q}}$
 - ▶ $\boldsymbol{\tau}^\top = F^\top J(\boldsymbol{q})$
 - ▶ $\boldsymbol{\tau} = J(\boldsymbol{q})^\top F$



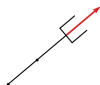
Statics - compensating external force

- ▶ Consider external force applied to the end-effector is $-F$.
- ▶ How to compute joint torques s.t. robot is static?
 - ▶ $\tau_{\text{ext}} = J(\mathbf{q})^T F$
 - ▶ end-effector needs to generate force F to compensate external $-F$
 - ▶ this equation assumes gravity does not act on a robot
 - ▶ $\tau = \tau_{\text{ext}} + \tau_g$
 - ▶ τ_g compensates gravity acting on a robot
 - ▶ For Panda robot, you can directly command τ_{ext}



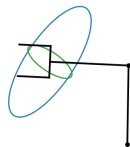
Force caused by given torques

- ▶ If J is invertible (when it is invertible?)
 - ▶ $F = J(\mathbf{q})^{-\top} \boldsymbol{\tau}$
- ▶ Redundant robots
 - ▶ even for fixed end-effector we can have internal motion
 - ▶ static equilibrium assumption is not valid \rightarrow dynamics needed
- ▶ Under-actuated robots
 - ▶ fixed end-effector will immobilize the robot
 - ▶ robot cannot actively generate forces in null-space of J^\top : $\ker(J^\top) = \{F \mid J^\top F = \mathbf{0}\}$
 - ▶ however, robot can resist external force in the null-space without moving
 - ▶ red arrow shows null-space
- ▶ Singularities (square J , but non-invertible)
 - ▶ non-zero null-space



Force ellipsoid

- ▶ How easy is to generate force in a given direction.
- ▶ Eigen analysis of $(JJ^T)^{-1}$
 - ▶ Blue - manipulability ellipsoid (i.e. JJ^T)
 - ▶ Green - force ellipsoid (i.e. $(JJ^T)^{-1}$)
- ▶ Easy motion in a direction \rightarrow difficult to compensate force in that direction
- ▶ Close to singularity:
 - ▶ area of manipulability ellipsoid $\rightarrow 0$
 - ▶ area of force ellipsoid $\rightarrow \infty$



Summary

- ▶ Differential kinematics
 - ▶ Jacobian and its properties
 - ▶ How to compute Jacobian
 - ▶ Manipulability ellipsoids
 - ▶ How to measure distance to singularity
- ▶ Statics
 - ▶ Static equilibrium relation of joint torques and task-space forces
 - ▶ Force ellipsoids



Laboratory

- ▶ Implementation of jacobian computation for planar manipulator
 - ▶ Finite difference method
 - ▶ Analytical method
- ▶ Generování of movement in null-space

