Robot Dynamics

Silvan Stadelmann - 30. Januar 2026 - v0.0.1

github.com/silvasta/summary-rodyn

Contents

Kinematics

1 Vectors and Positions

Position vectors: Parametrize P in frame $\mathcal{A}$ as ${}_{\mathcal{A}}\mathbf{r}_{AP} = \mathbf{r}(\chi)$, where χ are parameters (e.g., Cartesian coords)

1.1 Linear Velocity

$$r = r(\chi) \qquad \dot{r} = \frac{\partial r}{\partial \chi}\dot{\chi} = E_p(\chi)\dot{\chi}$$

Exam tip: Used in Jacobians for task-space velocities

1.2 Rotations

Rotation matrix $\mathbf{C}_{\mathcal{AB}}$ transforms vectors from frame $\mathcal{B}$ to $\mathcal{A}$:

$${}_{\mathcal{A}}\mathbf{r}_{AP} = \mathbf{C}_{\mathcal{AB}} \cdot {}_{\mathcal{B}}\mathbf{r}_{AP}$$

Properties Orthogonal ($\mathbf{C}^T = \mathbf{C}^{-1}$), det=1 for proper rotations

Elementary rotations (about x,y,z axes by angle θ):

$$\mathbf{R}_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$\mathbf{R}_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

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

Composition $\mathbf{C}_{\mathcal{AC}} = \mathbf{C}_{\mathcal{AB}} \cdot \mathbf{C}_{\mathcal{BC}}$.

Homogeneous transformations (4x4 for position + orientation)

$$\mathbf{T}_{\mathcal{AB}} = \begin{bmatrix} \mathbf{C}_{\mathcal{AB}} & {}_{\mathcal{A}}\mathbf{r}_{AB} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}$$

Passive Rotate frame **Active** Rotate vector

Exam pitfall: Distinguish for inverse problems

1.3 Angular Velocity

Angular velocity $\boldsymbol{\omega}$ satisfies $\dot{C} = \boldsymbol{\omega} \times \mathbf{C}$, $\dot{C} = \mathbf{S}(\boldsymbol{\omega})\mathbf{C}$

$$\mathbf{S}(\mathbf{a}) = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}$$

Composition: ${}_{\mathcal{A}}\boldsymbol{\omega}_{\mathcal{AC}} = {}_{\mathcal{A}}\boldsymbol{\omega}_{\mathcal{AB}} + \mathbf{C}_{\mathcal{AB}\,\mathcal{B}}\boldsymbol{\omega}_{\mathcal{BC}}$.

1.4 Parametrization of 3d Rotations

Minimal params: 3 (due to SO(3) manifold)
Common for avoiding singularities in kinematics/control.

- **Rotation matrix** 9 params, 6 orthonormality constraints. Direct but redundant.
- **Euler angles** (e.g., ZYZ): $\mathbf{C} = \mathbf{R}_z(\phi)\mathbf{R}_y(\theta)\mathbf{R}_z(\psi)$
  3 params, **singularities** at $\theta = 0, \pi$ (gimbal lock).
  Exam: Derive matrix; convert to/from
- **Angle-axis** No singularities but multi-valued.
  $\mathbf{C} = \exp(\mathbf{S}(\mathbf{k}\theta)) = \mathbf{I} + \sin\theta\mathbf{S}(\mathbf{k}) + (1 - \cos\theta)\mathbf{S}^2(\mathbf{k})$ (Rodrigues) $\mathbf{k}$ unit vector, θ angle.
- **Rotation vector** $\boldsymbol{\rho} = \mathbf{k}\theta$, Similar to angle-axis.
- **Unit quaternions** 4 params, 1 constraint. No singularities; efficient for interpolation/composition.
  $\mathbf{q} = (q_0, \mathbf{q}_v) = (\cos(\theta/2), \mathbf{k}\sin(\theta/2))$, $\|\mathbf{q}\| = 1$.

1.5 Unit Quaternions

To rotation matrix

$$\mathbf{C}(\mathbf{q}) = \mathbf{I} + 2q_0\mathbf{S}(\mathbf{q}_v) + 2\mathbf{S}^2(\mathbf{q}_v)$$

From matrix Extract $\theta = \arccos((\mathrm{trace}(\mathbf{C}) - 1)/2)$

$$\mathbf{q}_1 \circ \mathbf{q}_2 = (q_{10}q_{20} - \mathbf{q}_{1v} \cdot \mathbf{q}_{2v}, q_{10}\mathbf{q}_{2v} + q_{20}\mathbf{q}_{1v} + \mathbf{q}_{1v} \times \mathbf{q}_{2v})$$

Rotate vector $\mathbf{v}' = \mathbf{q} \circ (0, \mathbf{v}) \circ \mathbf{q}^{-1}$ (pure quaternion)

Time derivative $\dot{\mathbf{q}} = \frac{1}{2}\mathbf{q} \circ (0, \boldsymbol{\omega})$

Exam: Use for singularity-free velocity integration.

2 Multi Body Kinematics

Generalized coordinates Joint variables $\mathbf{q} = (q_1, \ldots, q_n)^T$ (e.g., angles for revolute, displacements for prismatic)

End-effector configuration $\boldsymbol{\chi}_e = (\boldsymbol{\chi}_{eP}, \boldsymbol{\chi}_{eR})^T$ (position + orientation params)

Operational/task space Subset χ_o for specific tasks (e.g., position only)

2.1 Forward Kinematics

End-effector configuration $\boldsymbol{\chi}_e = f(\mathbf{q})$. For serial chains: Product of homogeneous transforms $\mathbf{T}_{0n} = \mathbf{T}_{01}\mathbf{T}_{12}\cdots\mathbf{T}_{(n-1)n}$

Denavit-Hartenberg (DH) params
Standard for link modeling **(crucial for exams!)**
Transform $\mathbf{T}_{i-1,i} =$

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

Link length: a_i| Link twist: α_i| Link offset: d_i| Joint angle: θ_i

Exam: Assign DH table for given robot; compute forward map; analyze workspace

2.2 Jacobians

Jacobi-Box

Differential map: $\dot{\chi}_e = J_{eA}(q)\dot{q}$

Differential map: $\dot{\boldsymbol{\chi}}_e = \mathbf{J}_{eA}(\mathbf{q})\dot{\mathbf{q}}$

$$J_{eA} = \frac{\partial \chi_e}{\partial q} = \begin{bmatrix} \frac{\partial \chi_1}{\partial q_1} & \cdots & \frac{\partial \chi_1}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial \chi_m}{\partial q_1} & \cdots & \frac{\partial \chi_m}{\partial q_n} \end{bmatrix}$$

Analytical Jacobian For orientation params (Euler rates,..)

Geometric Jacobian Columns from velocity contributions (prismatic: linear velocity; revolute: $\boldsymbol{\omega}_i \times \mathbf{r}_{ie} + \mathbf{v}_i$).

Prismatic Position $\mathbf{J}_{P,i} = \mathbf{z}_{i-1}$, Rotation $\mathbf{J}_{O,i} = \mathbf{0}$
Revolute Pos. $\mathbf{J}_{P,i} = \mathbf{z}_{i-1} \times \mathbf{r}_{ie}$, Rotation $\mathbf{J}_{O,i} = \mathbf{z}_{i-1}$

Singularity $det(J) = 0 \rightarrow$ loss of DOF
Exam: Compute rank, manipulability $\mu = \sqrt{\det(\mathbf{J}\mathbf{J}^T)}$.

2.3 Velocity in Moving Bodies

Rigid Body Formulation Point P velocity in frame A:
${}_{\mathcal{A}}\mathbf{v}_P = {}_{\mathcal{A}}\mathbf{v}_B + {}_{\mathcal{A}}\boldsymbol{\omega}_{\mathcal{AB}} \times {}_{\mathcal{A}}\mathbf{r}_{BP} + \mathbf{C}_{\mathcal{AB}\mathcal{B}}\mathbf{v}^{\text{rel}}_{BP}$.

Twist vector t $= (\boldsymbol{\omega}, \mathbf{v})^T$, propagates via adjoint matrices.

Exam: Recursive computation for serial chains (forward for velocities).

3 Inverse Kinematics

$$w_e^\star = J_{e0}\dot{q}$$

Solve $\mathbf{q} = f^{-1}(\boldsymbol{\chi}_e^\star)$

Analytical for low DOF (e.g., 3R planar: geometric)

Numerical otherwise (e.g., Newton-Raphson)

Velocity level $\dot{\mathbf{q}} = \mathbf{J}^\dagger\dot{\boldsymbol{\chi}}_e + (\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\dot{\mathbf{q}}_0$
(pseudo-inverse for redundancy; nullspace for secondary tasks)

Singularities Use damped least-squares
$\mathbf{J}^\dagger = \mathbf{J}^T(\mathbf{J}\mathbf{J}^T + \lambda^2\mathbf{I})^{-1}$

Redundancy If $n > m$, infinite solutions; optimize (e.g., min norm velocity).

3.1 Multi-task control

Single task: $\dot{\mathbf{q}} = \mathbf{J}^\dagger\dot{\boldsymbol{\chi}}^\star$.

Stacked: Combine Jacobians $\mathbf{J}_s = \begin{bmatrix}\mathbf{J}_1 \\ \mathbf{J}_2\end{bmatrix}$, solve if consistent.

Prioritized: $\dot{\mathbf{q}} = \mathbf{J}_1^\dagger\dot{\boldsymbol{\chi}}_1^\star + (\mathbf{I} - \mathbf{J}_1^\dagger\mathbf{J}_1)\mathbf{J}_2^\dagger(\dot{\boldsymbol{\chi}}_2^\star - \mathbf{J}_2\mathbf{J}_1^\dagger\dot{\boldsymbol{\chi}}_1^\star)$.

Error analysis

Task error: $\mathbf{e} = \boldsymbol{\chi}^\star - \boldsymbol{\chi}$

Control: $\dot{\boldsymbol{\chi}}^\star = \dot{\boldsymbol{\chi}}_d + \mathbf{K}\mathbf{e}$ (resolved rate)

Joint trajectory Interpolate $\mathbf{q}(t)$ (e.g., cubic polynomial for vel/acc constraints)

Exam patterns: Compute inverse for 2-3 DOF arm; handle redundancy/singularities in control loops.

Dynamics

4 Equation of Motion

Equations

$$M(q)\ddot{q} + b(q,\dot{q}) + g(q) = \tau + J_c^T F_c$$

$M(q)$	Mass matrix
$\ddot{q}$	Generalized coordinates
$b(q,\dot{q})$	Centrifugal and Coriolis forces
$g(q)$	Gravity forces
τ	Generalized forces
F_c	External forces
J_c	Contact Jacobian

4.1 Principle of Virtual Work

- Principle of virtual work (D'Alembert's Principle)

- Dynamic equilibrium imposes zero virtual work (for all virtual displacements)

- Newton's law for every particle in direction it can move

4.2 Single Rigid Body

4.3 Newton-Euler Method

4.4 Projected Newton-Euler

Principle of virtual work for multi-body systems

4.5 Lagrange II

- Kinetic energy

- Potential energy

4.6 External Forces and Torques

Forces $\tau_{F_{ext}} = \sum_{j=1}^{n_{f,ext}} J_{P,j}^T T_{ext,k}$

Torques $\tau_{T_{ext}} = \sum_{k=1}^{n_{m,ext}} J_{R,k}^T T_{ext,k}$

Actuators $\tau_{a,k} = (J_{S_k} - J_{S_{k-1}})^T F_{a,k} + (J_{R_k} - J_{R_{k-1}})^T$

4.7 Velocity in Moving Bodies

- Definitions

- Moving Frame

4.8 Prismatic Joints

TODO: Jacobians

- Position Jacobian

- Rotation Jacobian

- Example

5 Dynamic Control

5.1 Joint Impedance Control

5.2 Inverse Dynamics Control

Compensate for system dynamics + PD law on acceleration

- every joint behaves like decoupled mass-spring damper

- Eigenfrequency

- Damping

5.3 Task-space dynamic control

- single task: just use pseudo-inverse

- multiple task: stack J_i, w_i, pseudo-inverse, done (equal priority)

5.4 End-effector dynamics

- end-effector motion control

- trajectory control (feedforwad term)

6 Interaction Control

6.1 Operational Space Control

Generalized framework to control motion and force

- F_c as contact force

6.2 Selection Matrix

- seperate motion and force directions

6.3 Inverse Dynamics as QP

- use: some sort of "mass-matrix weighted pseudo-inverse"

- quadratic optimization

- Solving set of QPs

7 Floating Base Dynamics

7.1 Generalized coordinates

$$q = \begin{pmatrix} q_b \\ q_j \end{pmatrix} \quad \text{with} \quad q_b = \begin{pmatrix} q_{b_P} \\ q_{b_R} \end{pmatrix} \in \mathbb{R}^3 \times SO(3)$$

7.2 Generalized velocities and accelerations

$$u = \begin{pmatrix} {}_I v_B \\ {}_B \omega_I B \\ \dot{\varphi}_1 \\ \vdots \\ \dot{\varphi}_{n_j} \end{pmatrix} \quad \dot{u} = \begin{pmatrix} {}_I a_B \\ {}_B \psi_I B \\ \ddot{\varphi}_1 \\ \vdots \\ \ddot{\varphi}_{n_j} \end{pmatrix} \in \mathbb{R}^{6+n_j} = \mathbb{R}^{n_u}$$

Very often, people write $\dot{q}$ but they mean u

$$u = E_{fb} \cdot \dot{q} \quad \text{with} \quad E_{fb} \cdot \dot{q} = \begin{bmatrix} 1_{3\times 3} & 0 & 0 \\ 0 & E_{\chi R} & 0 \\ 0 & 0 & 1_{n_j \times n_j} \end{bmatrix}$$

7.3 Differential kinematics

7.4 Contacts and Constraints

7.5 Properties of Contact Jacobian

8 Dynamics of Floating Base Systems

- External Forces

- Soft Contact

- Hard Contact

8.1 Constraint consistent dynamics

8.2 Contact Dynamics

8.3 Dynamic Control Methods

- Behavior as Multiple Tasks

- Internal Forces

8.4 Control using inverse dynamics

8.5 Task Space Control as Quadratic Program

Legged Robot History of Legged Robotics

9 Quadrupedal Robots

L8.37 dotlist

10 High-gear Systems

with torque sensor or elasticity (SEA)

10.1 Actuation principle

- geared motor

- biomechanic ideas

- Series elastic actuator

- Exploitation of Passive Dynamics

11 Low-gear Systems

with current control only

- Low geared high torque motor

L8.47 dotlist

11.1 Motor and output inertia

formulas images

12 Hydraulic

pressure and/or load cell

L8.53 dotlist

- Antagonistic actuation possible

12.1 Pneumatic

- difficul doing closed loop control

12.2 acuation principle, othes

- SMA

- EAP

- Piezo-electtric

Issues

L8.58

13 Conrol of Legged Robots

Static vs Dynamic Stability

13.1 Contol Concepts

L8.61

13.2 Motion planning, high-gain kinematic trajcectory following

FORMULAS L8.60/73

Rotorcraft

Nunc sed pede. Praesent vitae lectus. Praesent neque justo, vehicula eget, interdum id, facilisis et, nibh. Phasellus at purus et libero lacinia dictum. Fusce aliquet. Nulla eu ante placerat leo semper dictum. Mauris metus. Curabitur lobortis. Curabitur sollicitudin hendrerit nunc. Donec ultrices lacus id ipsum.

Fixed-Wing

Nunc sed pede. Praesent vitae lectus. Praesent neque justo, vehicula eget, interdum id, facilisis et, nibh. Phasellus at purus et libero lacinia dictum. Fusce aliquet. Nulla eu ante placerat leo semper dictum. Mauris metus. Curabitur lobortis. Curabitur sollicitudin hendrerit nunc. Donec ultrices lacus id ipsum.