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 $\boldsymbol{\chi}_o$ 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 $\mathbf{t} = (\boldsymbol{\omega}, \mathbf{v})^T$, 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 Rigid-body Manipulators - Fixed Base

Equation of Motion

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

$M(q)$: Mass/inertia matrix (symmetric, positive definite)
$b(q,\dot{q})$: Coriolis/centrifugal vector $= C(q,\dot{q})\dot{q}$
$g(q)$: Gravity vector
τ : Joint torques/forces (actuators)
$J_c^T F_c$: External contact forces mapped to joint space

Properties $\dot{M} - 2C$ is skew-symmetric (useful for stability proofs)

Exam tip: Derive for 2-3 DOF arms; compute components numerically.

4.1 Principle of Virtual Work (D'Alembert's Principle)

For Dynamic Equilibrium:
Virtual work $\delta W = 0$ for all virtual displacements δq.

$$\sum_i (F_i - m_i \ddot{r}_i) \cdot \delta r_i + \sum_j (T_j - I_j \dot{\omega}_j - \omega_j \times I_j \omega_j) \cdot \delta \theta_j = 0$$

Applies Newton's laws in directions of possible motion.

Exam: Use to derive EoM for constrained systems.

4.2 Single Rigid Body Dynamics

Translational $m\ddot{r} = F$ (Newton)

Rotational $I\dot{\omega} + \omega \times I\omega = T$ (Euler)

In moving frame Velocities/accelerations include Coriolis terms

Exam: Compute for link in chain; relate to kinematics (e.g., ${}^I v = {}^I \dot{r} + \omega \times r$)

4.3 Newton-Euler Method

Recursive computation for serial chains. Efficient ($O(n)$)

Forward Propagate velocities/accelerations, base to end-effectr

Backward Propagate forces/torques from end-effector to base.

Formulas for link i: (Force/torque balance yields τ_i)

$${}^i v_i = {}^i R_{i-1} ({}^{i-1} v_{i-1} + {}^{i-1} \omega_{i-1} \times {}^{i-1} p_i) + {}^i \dot{q}_i z_i$$
$${}^i \omega_i = {}^i R_{i-1}^{i-1} \omega_{i-1} + {}^i \dot{q}_i z_i \quad \text{(revolute)}$$

exam: Apply to 3R arm, compare with Lagrangian

4.4 Projected Newton-Euler

Principle of virtual work for multi-body systems

4.5 Projected Newton-Euler

Applies virtual work to multi-body systems.
Project dynamics into joint space.
Sum virtual works for EoM: $\tau = \sum$ projected inertias/forces

Key Terms Inertia projection, Coriolis, gravity via Jacobians.

Exam: Derive mass matrix $M_{ij} = \sum_k$ trace$(J_k^T I_k J_k)$

4.6 Lagrange Formulation

EoM from $\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = \tau$ where $L = T - V$.

Energy: **Kinetic** $T = \frac{1}{2}\dot{q}^T M(q)\dot{q}$ **Potential** $V = \sum m_i g^T r_i$

$$M_{kl} = \sum_{i=\max(k,l)}^{n} \text{trace}\left(\frac{\partial T_i}{\partial q_k} J_i \frac{\partial T_i^T}{\partial q_l}\right) + m_i \frac{\partial r_i^T}{\partial q_k}\frac{\partial r_i}{\partial q_l}$$

$b = C\dot{q}$ from Christoffel symbols

Exam: Full derivation for planar 2R; identify M, C, g.

4.7 External Forces and Torques

Map to joint torques:

Forces

$$\tau_{F_{ext}} = \sum_{j=1}^{n_f} J_{P,j}^T F_{ext,j}$$

Torques

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

Actuators

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

J_P, J_R: Position/rotation Jacobians.

Exam: Compute for end-effector force.

4.8 Velocity in Moving Bodies

Velocity in frame i: **Linear** ${}^i v = {}^i \dot{r} + {}^i \omega \times {}^i r$ **Angular** ${}^i \omega$.

Twist vector: $V = \begin{bmatrix} v \\ \omega \end{bmatrix}$.

Propagation ${}^i V_i = {}^i A_{i-1}^{i-1} V_{i-1} + {}^i \dot{q}_i e_i$ (A: adjoint).

Exam: Use in NE recursion

4.9 Jacobians for Prismatic/Revolute Joints

Jacobian $J = \begin{bmatrix} J_v & J_\omega \end{bmatrix}$ maps $\dot{q} \to$ task velocity

Prismatic for joint i: Col i of $J_v = z_{i-1}$, $J_\omega = 0$

Revolute: $J_v = z_{i-1} \times (p - p_{i-1})$, $J_\omega = z_{i-1}$

Exam example: 2R planar arm Jacobian, singularity when det(J)=0 (e.g., extended arm).

5 Dynamic Control

Control loops
Position (inner vel/torque), Torque (feedforward dynamics).

Exam: Block diagrams for PD + gravity comp: $\tau = g(q) + K_p e + K_d \dot{e}$

5.1 Joint Impedance Control

$\tau = g(q) + K_p(q_d - q) + K_d(\dot{q}_d - \dot{q}) + K_i \int e\, dt + J^T F_{ext}$

Tuned as mass-spring-damper:
Eigenfreq $\omega = \sqrt{K_p/m}$ **Damping** $\zeta = K_d/(2\sqrt{mK_p})$

Exam: Stability via Lyapunov.

5.2 Inverse Dynamics Control (Computed Torque)

$\tau = M(q)(\ddot{q}_d + K_d \dot{e} + K_p e) + b(q, \dot{q}) + g(q)$

Decouples to $\ddot{e} + K_d \dot{e} + K_p e = 0$

Exam: Derive error dynamics; choose gains for crit. damping.

5.3 Task-Space Dynamic Control

EoM in task space $\Lambda(x)\ddot{x} + \mu(x, \dot{x}) + p(x) = F + J^{-T}\tau_{ext}$

Control $F = \Lambda(\ddot{x}_d + K_d \dot{e}_x + K_p e_x) + \mu + p$

For redundancy Use $J^\dagger = W^{-1}J^T(JW^{-1}J^T)^{-1}$ (weighted pseudo-inv)

Multiple tasks Stack Jacobians, null-space projectors $N = I - J^\dagger J$

Exam: Prioritize tasks (e.g., motion > posture)

5.4 End-Effector Dynamics

$\Lambda = (JM^{-1}J^T)^{-1}$ control as above.

Feedforward for traj: $\ddot{x}_d$ from planning.

Exam: Compute Λ for 3R arm.

6 Interaction Control

6.1 Operational Space Control

Unified: Includes force F_c in dynamics.

$\tau = J^T \Lambda(\ddot{x}_d + K_d \dot{e}_x + K_p e_x - JM^{-1}(b + g)) + (I - J^T \bar{J}^T)\tau_0$

Exam: Hybrid force-motion via selection matrix S (e.g., $S = I$ motion, $S = 0$ force)

6.2 Selection Matrix

S: Diagonal, separates DOFs (e.g., force in z, motion in x-y)
Control: Blend impedances.

Exam: Formula for hybrid: $\tau = J^T(SF_m + (I - S)F_f)$.

6.3 Inverse Dynamics as QP

Formulation $\min_u \|Au - b\|_W^2$ s.t. constraints (torque limits,...)

For tasks Hierarchical QPs, solve sequentially

Exam: Least-squares for overconstrained systems, weighted pseudo-inv $J_W^\dagger = W^{-1}J^T(JW^{-1}J^T)^{-1}$

7 Floating Base Dynamics

For mobile/legged robots: Unactuated base.

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

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

$$u = E_{fb} \cdot \dot{q} \quad \text{with} \quad E_{fb} = \begin{bmatrix} 1_{3\times3} & 0 & 0 \\ 0 & E_{\chi R} & 0 \\ 0 & 0 & 1_{n_j \times n_j} \end{bmatrix}$$

E_{fb} maps quaternions/Euler to twists.

Exam: Note $\dot{q} \neq u$ due to SO(3).

7.3 Differential Kinematics

Floating: $J = \begin{bmatrix} J_b & J_j \end{bmatrix}$, $\dot{x} = J(q)u$ (task vel from gen. vel)

7.4 Contacts and Constraints

Hard $J_c u = 0$ (no slip), **Soft** Compliant models

Constraints $J_c \dot{u} + \dot{J}_c u = 0$

Exam: Enforce via Lagrange multipliers $F_c = -\lambda$.

8 Dynamics of Floating Base Systems

$$M(q)\dot{u} + h(q, u) = S^T \tau + J_c^T F_c$$

where $S = \begin{bmatrix} 0 & I \end{bmatrix}$ (underactuated base), $h = b + g$.

8.1 Constraint-Consistent Dynamics

Project to null-space of constraints: $\bar{M}\dot{u} + \bar{h} = \bar{S}^T \tau$.

Exam: For legged, balance via CoM control

8.2 Contact Dynamics

Impacts Instant velocitiy change:

$\Delta u = -(J_c M^{-1} J_c^T)^{-1} J_c u^-$ (pre-impact)

Soft Spring-damper $F_c = k\delta + d\dot{\delta}$

8.3 Dynamic Control Methods

Multi-task Motion as tasks (e.g., CoM, feet)

Internal forces Null-space torques for stability.

8.4 Control Using Inverse Dynamics

$\tau = S^+(M\dot{u}_d + h - J_c^T F_{c,d}) + N\tau_0$

Exam: Choose $\dot{u}_d$ for tasks

8.5 Task Space Control as Quadratic Program

Hierarchical QP: Min cost for primary task, then secondary in null-space.

Exam: Formulate for torque optimization with friction cone constraints