Robot Dynamics

Silvan Stadelmann - 7. Februar 2026 - v0.3.0

github.com/silvasta/summary-rodyn

Contents

Kinematics

1 Vectors and Positions

Position vectors: Parametrize point P in frame $\mathcal{A}$ as ${}_\mathcal{A}\mathbf{r}_{AP} = \mathbf{r}(\chi)$, where χ are parameters (e.g. $\chi = (x, y, z)^T$).

1.1 Linear Velocity

$$\mathbf{r} = \mathbf{r}(\chi), \quad \dot{\mathbf{r}} = \frac{\partial \mathbf{r}}{\partial \chi}\dot{\chi} = \mathbf{E}_p(\chi)\dot{\chi}$$

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):

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

To apply: For point $\mathbf{p}$ in frame n, in frame 0: $\begin{bmatrix} {}^0\mathbf{p} \\ 1 \end{bmatrix} = \mathbf{T}_{0n} \begin{bmatrix} {}^n\mathbf{p} \\ 1 \end{bmatrix}$

Passive: Rotate frame **Active**: Rotate vector

Exam pitfall: Confuse active/passive in inverse kinematics

1.3 Angular Velocity

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

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

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

Exam note: Use skew-symmetric for deriving velocity Jacobians

1.4 Parametrization of 3d Rotations

Minimal params 3 - due to SO(3) manifold
Common for avoiding singularities in kinematics/control.
All parameterizations map to rotation matrix but differ in singularities, redundancy, and computation.

- **Rotation matrix** 9 params, 6 orthonormality constraints. Direct but redundant. No singularities, but not minimal, used for storage/composition.
- **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)
  Variants: ZYX (roll-pitch-yaw), XYZ. Angular velocity: $\boldsymbol{\omega} = \mathbf{E}(\phi, \theta, \psi)\dot{\boldsymbol{\alpha}}$, where $\mathbf{E}$ is singular at gimbal lock.
- **Angle-axis** Rodrigues' formula:
  $\mathbf{C} = \exp(\mathbf{S}(\mathbf{k}\theta)) = \mathbf{I} + \sin\theta\mathbf{S}(\mathbf{k}) + (1 - \cos\theta)\mathbf{S}^2(\mathbf{k})$
  $\mathbf{k}$ unit vector, $\theta \in [0, \pi]$. No singularities but multi-valued ($\theta = 0$ ambiguous). Good for small rotations.
- **Rotation vector** $\boldsymbol{\rho} = \mathbf{k}\theta$
  Similar to angle-axis; exponential map from $\mathfrak{so}(3)$ to SO(3). Velocity: $\boldsymbol{\omega} = \mathbf{T}(\boldsymbol{\rho})\dot{\boldsymbol{\rho}}$, with $\mathbf{T}$ near-identity for small ρ.
- **Unit quaternions**: 4 params, 1 norm constraint ($\|\mathbf{q}\| = 1$). No singularities, efficient for interpolation/composition. See dedicated subsection below.

1.5 Unit Quaternions

Standard definition: $\mathbf{q} = (q_0, \mathbf{q}_v) = (\cos(\theta/2), \mathbf{k}\sin(\theta/2))$, where $\|\mathbf{q}\| = 1$, representing rotation by θ around unit axis $\mathbf{k}$.
Antipodal: $\mathbf{q} \equiv -\mathbf{q}$.

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 angle $\theta = \arccos((\text{trace}(\mathbf{C}) - 1)/2)$, then $\mathbf{k}$ from skew-symmetric part. Quaternion: $q_0 = \cos(\theta/2)$, $\mathbf{q}_v = \mathbf{k}\sin(\theta/2)$.

Composition (Hamilton product):

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

Inverse: $\mathbf{q}^{-1} = (q_0, -\mathbf{q}_v)$.

Rotate vector: Treat vector as pure quaternion $(0, \mathbf{v})$, then $\mathbf{v}' = \mathbf{q} \circ (0, \mathbf{v}) \circ \mathbf{q}^{-1}$ (extract vector part).

Time derivative: $\dot{\mathbf{q}} = \frac{1}{2}\mathbf{q} \circ (0, \boldsymbol{\omega}) = \frac{1}{2}\mathbf{E}(\mathbf{q})\boldsymbol{\omega}$, where $\mathbf{E}$ maps angular velocity.

Real-world tip: Use quaternions for SLERP interpolation in trajectories; normalize after integration to avoid drift.

2 Multi Body Kinematics

Generalized coordinates Joint variables $\mathbf{q} = (q_1, \ldots, q_n)^T$ (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

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 hint: For revolute, θ_i variable; prismatic, d_i variable. Assign frames with z along joint axis.

2.2 Workspace Analysis

Reachable: All positions end-effector can reach
Dexterous: All poses.

2.3 Jacobians

Jacobi-Box

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

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

Geometric Jacobian Maps to end-effector twist $\mathbf{t} = \begin{bmatrix} \boldsymbol{\omega} \\ \mathbf{v} \end{bmatrix} = \mathbf{J}_G \dot{\mathbf{q}}$. Independent of param; differs from analytic in orientation (angular velocity vs. param rates). Relation: $\mathbf{J}_{eA} = \mathbf{T}(\chi_{eR})\mathbf{J}_G$, where $\mathbf{T}$ maps velocity (e.g., for Euler: $\boldsymbol{\omega} = \mathbf{E}\dot{\alpha}$).

Derivation (serial chain): Assign DH frames. For joint i:
- **Revolute**: Pos. column $\mathbf{J}_{P,i} = {}^{i-1}\mathbf{z}_{i-1} \times ({}^0\mathbf{r}_e - {}^0\mathbf{r}_{i-1})$ Rot. $\mathbf{J}_{O,i} = {}^{i-1}\mathbf{z}_{i-1}$ (in base frame)
- **Prismatic**: Pos. $\mathbf{J}_{P,i} = {}^{i-1}\mathbf{z}_{i-1}$ Rot. $\mathbf{J}_{O,i} = \mathbf{0}$

Total $\mathbf{J}_G = [\mathbf{J}_P \quad \mathbf{J}_O]^T$ (6xn).

Singularity: $\det(\mathbf{J}\mathbf{J}^T) = 0$ (DOF loss). Types: Boundary (workspace edge), internal (e.g., aligned links).
Manipulability: $\mu = \sqrt{\det(\mathbf{J}\mathbf{J}^T)}$; ellipsoid for velocity/-force transmission.

(Example 3R planar Jacobian (pos only):
$$\begin{bmatrix} -l_1 s_1 - l_2 s_{12} - l_3 s_{123} & -l_2 s_{12} - l_3 s_{123} & -l_3 s_{123} \\ l_1 c_1 + l_2 c_{12} + l_3 c_{123} & l_2 c_{12} + l_3 c_{123} & l_3 c_{123} \end{bmatrix}$$

Real-world tip: Use manipulability for path planning, damp near singularities ($\lambda \propto 1/\mu$) to prevent instability

2.4 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:
$$\mathbf{t}_i = \text{Ad}_{\mathbf{T}_{i-1,i}}\mathbf{t}_{i-1} + \mathbf{e}_i\dot{q}_i,$$
where $\text{Ad}_{\mathbf{T}} = \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{S}(\mathbf{r})\mathbf{C} & \mathbf{C} \end{bmatrix}$ (adjoint map), and unit twist $\mathbf{e}_i = (\mathbf{z}_i, \mathbf{0})^T$ for revolute or $(\mathbf{0}, \mathbf{z}_i)^T$ for prismatic.

3 Inverse Kinematics

Main idea: Solve $\mathbf{q} = f^{-1}(\chi_e^\star)$.

Numerical Newton-Raphson: $\mathbf{q}_{k+1} = \mathbf{q}_k + \mathbf{J}^{-1}(\chi^\star - f(\mathbf{q}_k))$

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

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

Redundancy: $n > m$; min-norm or secondary tasks (e.g., joint limit avoidance)

3.1 Analytical Inverse Kinematics

For 2R planar arm (lengths l_1, l_2, target (x,y)):
$\theta_2 = \pm \arccos\left(\frac{x^2+y^2-l_1^2-l_2^2}{2l_1l_2}\right)$,
$\theta_1 = \arctan(y,x) - \arctan(l_2 \sin\theta_2, l_1 + l_2\cos\theta_2)$
For 3R: Solve for θ_3 via circle intersection, then reduce to 2R (multiple solutions; check workspace).

Solve inverse for 3R, handle multiple solutions/elbow configs

3.2 Multi-Task Control

Single task: $\dot{\mathbf{q}} = \mathbf{J}^\dagger \dot{\chi}^\star$. **Stacked**: $\mathbf{J}_s = \begin{bmatrix} \mathbf{J}_1 \\ \mathbf{J}_2 \end{bmatrix}$, $\dot{\mathbf{q}} = \mathbf{J}_s^\dagger \begin{bmatrix} \dot{\chi}_1^\star \\ \dot{\chi}_2^\star \end{bmatrix}$.

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

(Null-space projection ensures task 2 doesn't affect task 1.)
Find $\dot{q}$ for desired task velocity:
- Single task: $\dot{q} = J^\dagger v_{task}$.
- Hierarchical (null-space): Primary J_1, v_1: $\dot{q} = J_1^\dagger v_1 + (I - J_1^\dagger J_1)\dot{q}_{null}$, where $\dot{q}_{null} = J_2^\dagger(v_2 - J_2 J_1^\dagger v_1)$.
- Multi-task: $\dot{q} = \sum N_i \dot{q}_i$, $\dot{q}_i = (J_i N_i)^+(v_i^* - J_i \sum_{k<i} \dot{q}_k)$, with $N_i = I - \sum_{k<i} J_k^\dagger J_k$.
- Near singularity: Damped LS $J^* = J^T(JJ^T + \lambda I)^{-1}$.

Error Analysis and Trajectories
Task error $\mathbf{e} = \chi^\star - \chi$
Control $\dot{\chi}^\star = \dot{\chi}_d + \mathbf{K}\mathbf{e}$ (resolved rate)
Joint trajectory Interpolate $\mathbf{q}(t)$
(cubic poly: $q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$; match vel/acc)

Exam patterns: Design inverse control loop for redundant arm; avoid singularities via damping/nullspace.

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: $\dot{M} - 2C$ skew-symmetric
Passivity: $\dot{q}^T(\dot{M} - 2C)\dot{q} = 0$, M bounded/invertible, linear in parameters (for identification)

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

For dynamic equilibrium: Virtual work $\delta W = 0$ for all 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, extends to constraints via multipliers.

Exam: Derive EoM for constrained systems (e.g., closed-chain); relate to projected dynamics.

4.2 Single Rigid Body Dynamics

Translational $m\ddot{r} = F$ (Newton)
Rotational $I\dot{\omega} + \omega \times I\omega = T$ (Euler)
Moving frame: Include Coriolis/centrifugal vel. ${}^I v = {}^I \dot{r} + \omega \times r$

4.3 Newton-Euler Method

Recursive for serial chains ($O(n)$ efficiency)
Forward: Velocities/accelerations base→EE
Backward: Forces/torques EE→base, yields τ_i.
For link i (revolute):
$${}^i\omega_i = {}^i R_{i-1}^{i-1}\omega_{i-1} + \dot{q}_i^i z_i,$$
$${}^i v_i = {}^i R_{i-1}({}^{i-1}v_{i-1} + {}^{i-1}\omega_{i-1} \times {}^{i-1}p_i) + \dot{q}_i^i z_i,$$
$${}^i \dot{v}_i = {}^i R_{i-1}^{i-1}\dot{v}_{i-1} + {}^i\dot{\omega}_i \times {}^i p_{c,i} + \ldots \text{ (full acc inc. Coriolis)}$$
Force: $f_i = m_i\dot{v}_{c,i} + \omega_i \times (\omega_i \times m_i r_{c,i})$

4.4 Projected Newton-Euler

The Projected Newton-Euler method uses the Principle of Virtual Work to project the Cartesian Newton-Euler equations of individual links into the space of generalized coordinates $\mathbf{q}$. This automatically eliminates internal constraint forces.

Mass Matrix Projection
$$\mathbf{M}(\mathbf{q}) = \sum_{k=1}^{n_L} \left(\mathbf{J}_{v,k}^T m_k \mathbf{J}_{v,k} + \mathbf{J}_{\omega,k}^T \mathbf{I}_k \mathbf{J}_{\omega,k}\right)$$

Nonlinear Terms (Coriolis/Centrifugal/Gravity)
Let $\mathbf{h} = \mathbf{C}\dot{\mathbf{q}} + \mathbf{g}$. This vector is the sum of link wrenches mapped to joint space via the transpose of the Jacobians:
$$\mathbf{h} = \sum_{k=1}^{n_L} \left(\mathbf{J}_{v,k}^T \mathbf{F}_k + \mathbf{J}_{\omega,k}^T \mathbf{T}_k\right)$$
Link-wise Newton-Euler forces/torques (evaluated at $\ddot{\mathbf{q}} = 0$):
$$\mathbf{F}_k = m_k\dot{\mathbf{v}}_{k,\text{rem}} - m_k\mathbf{g}_0$$
$$\mathbf{T}_k = \mathbf{I}_k\dot{\boldsymbol{\omega}}_{k,\text{rem}} + \boldsymbol{\omega}_k \times \mathbf{I}_k\boldsymbol{\omega}_k$$
$\dot{\mathbf{v}}_{\text{rem}}$ and $\dot{\boldsymbol{\omega}}_{\text{rem}}$ refer to convective acceleration terms (i.e., $\dot{\mathbf{J}}\dot{\mathbf{q}}$)

Virtual Work Derivation: The projection is valid because $\delta\mathbf{q}^T(\mathbf{M}\ddot{\mathbf{q}} + \mathbf{h} - \boldsymbol{\tau}) = 0$ for all virtual displacements $\delta\mathbf{q}$ that comply with the constraints.

4.5 Projected Newton-Euler

Combines Newton-Euler with Lagrange: Uses virtual work in generalized coordinates to project Cartesian dynamics into joint space, automatically handling constraints. Most practical for robotics as it yields EoM in standard form efficiently.
EoM: $\tau = \sum$ projected inertias/forces (via Jacobians)
Mass: $M_{ij} = \sum_k \text{trace}(J_{v,k}^T m_k J_{v,k} + J_{\omega,k}^T I_k J_{\omega,k})$.
Coriolis/gravity similarly projected (e.g., $b_i = \sum_k J_{v,k}^T(m_k\dot{v}_k + \omega_k \times m_k v_k) + J_{\omega,k}^T(\Theta_k\dot{\omega}_k + \omega_k \times \Theta_k\omega_k)$).
Derivation from virtual work: $\delta q^T(M\ddot{q} + b + g - \tau) = 0$ for constraint-compliant δq.

4.6 Lagrange Formulation

EoM from Euler-Lagrange equation: $\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = \tau$ where $L = T - V$ (Lagrangian).
Kinetic energy $T = \frac{1}{2}\dot{q}^T M(q)\dot{q} = \sum_i \frac{1}{2}m_i v_i^2 + \frac{1}{2}\omega_i^T I_i \omega_i$.
Potential energy $V = \sum_i m_i g^T r_i$ (or $U = \sum m_i g h_i$).
Mass matrix:
$$M_{kl} = \sum_{i=\max(k,l)}^{n} \text{trace}\left(\frac{\partial T_i}{\partial q_k} J_i \frac{\partial T_i^T}{\partial q_l}\right) + m_i \frac{\partial r_i^T}{\partial q_k}\frac{\partial r_i}{\partial q_l}$$

Coriolis matrix via Christoffel symbols:

$$C_{kj} = \sum_i \Gamma_{kji}\dot q_i, \qquad \Gamma_{kji} = \frac{1}{2}(\partial_k M_{ji} + \partial_j M_{ki} - \partial_i M_{kj})$$

Gravity: $g_k = -\frac{\partial V}{\partial q_k}$.

Computation steps:

- Forward kinematics for positions r_i, velocities v_i, ω_i
- Form T and V in terms of $q, \dot q$
- Apply Euler-Lagrange to get M, C, g

4.7 External Forces and Torques

Map to joints: $\tau_{ext} = J_P^T F_{ext} + J_R^T T_{ext}$

J_P, J_R: Position/rotation Jacobians.

Forces **Torques**

$$\tau_{F_{ext}} = \sum_{j=1}^{n_f} J_{P,j}^T F_{ext,j} \qquad \tau_{T_{ext}} = \sum_{k=1}^{n_m} J_{R,k}^T T_{ext,k}$$

Actuators

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

To compute τ (full EoM params):

1. Use forward kinematics for Jacobians J.
2. Compute $M(q)$ via Lagrange or PNE projection.
3. Compute $b = C\dot q$ using Christoffel or recursive NE.
4. Compute $g(q)$ from potential or NE gravity terms.
5. Add external: $\tau = M\ddot q + b + g - J_c^T F_c$.

4.8 Velocity in Moving Bodies

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

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).

5 Dynamic Control

Control loops

Position (inner velocity/torque), Torque (feedforward dynamics).

Exam: Block diagrams for PD + gravity comp:

$\tau = g(q) + K_p e + K_d\dot e$, error $\ddot e + K_d\dot e + K_p e = M^{-1}\delta\tau$

Synchronize: Outer position loop with inner torque, feedforward g for decoupling.

5.1 Joint Impedance Control

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

Mass-spring-damper: $\omega_n = \sqrt{K_p/m}, \zeta = K_d/(2\sqrt{mK_p})$

Prove stability for PD control using Lyapunov Stability

$V = \frac{1}{2}\dot q^T M\dot q + \frac{1}{2}e^T K_p e, \dot V = -\dot q^T K_d\dot q \le 0$

(LaSalle for convergence)

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: $\ddot e + K_d\dot e + K_p e = 0$, crit damp $K_d = 2\sqrt{K_p}$

5.3 Task-Space Dynamic Control

EoM

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

with $\Lambda = (JM^{-1}J^T)^{-1}$

Control $F = \Lambda(\ddot x_d + K_d\dot e_x + K_p e_x) + \mu + p$

Redundancy weighted psd-inv: $J^\dagger = W^{-1}J^T(JW^{-1}J^T)^{-1}$

Null-space projector $N = I - J^\dagger J$

Multiple tasks Stack Jacobians, project secondary to N

- Joint space: $\tau = M(q)(\ddot q^* + K_p e + K_d\dot e) + h(q, \dot q)$ $(h = C\dot q + g)$.
- With external force: $\tau_{\text{total}} = \tau_{\text{motion}} + J^T F_{ext}$.
- Task space: Solve $\ddot q_{des} = J^\dagger(\ddot x^* - \dot J\dot q)$, then plug into EoM.

5.4 End-Effector Dynamics

As above and with feedforward $\ddot x_d$ from trajectory planning.

Exam: Hybrid with selection S (diag, 0=force,1=motion)

6 Interaction Control

6.1 Operational Space Control

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

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

6.2 Selection Matrix

S: Diagonal matrix for hybrid control, e.g., $S_{ii} = 1$ for motion-controlled DOFs (position/velocity), $S_{ii} = 0$ for force-controlled DOFs. Allows blending impedance/force in task space (e.g., force in z for peg-in-hole, position in x-y). Control law: $F = SF_{\text{motion}} + (I - S)F_{\text{force}}$, where F_{motion} uses PD, F_{force} is desired force.

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

6.3 Inverse Dynamics as QP

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

Hierarchical Solve primary, project secondary to null

Hierarchical Optimization (QP) Formulation

$\min_{\ddot q, \tau, f_c} \|\tau\|^2 + \epsilon\|\ddot q\|^2 + \epsilon\|f_c\|^2$ s.t.

- Dynamics: $M\ddot q + h = S^T\tau + J_c^T f_c$.
- Contact: $J_c\ddot q + \dot J_c\dot q = 0$.
- Task (e.g., CoM): $J_{com}\ddot q + \dot J_{com}\dot q = \ddot x^*_{com}$.
- Bounds: $|\tau| \le \tau_{max}$, friction cone $\|f_{c,\perp}\| \le \mu f_{c,z}$.

Matrix form: Stack into $Ax = b$ (e.g., $x = [\dot u^T, \tau^T, F_c^T]^T$).

Exampel QP for 7DOF arm: min $\|\dot q\|$ s.t. $J\dot q = \dot x_d$, torque bounds; null for secondary (e.g., obstacle avoid)

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/Accelerations

Twist-based: $u = [{}^I v_b^T\ {}^b\omega_b^T\ \dot q_j^T]^T \in \mathbb{R}^{6+n_j}$; $\dot u$ similar. Map: $u = E_{fb}\dot q$ (E_{fb} handles rot param, e.g., quats to ang vel).

Note $\dot q \neq u$ due to SO(3) use for non-holonomic systems

7.3 Generalized velocities and accelerations

$$u = \begin{pmatrix} {}_I v_B \\ {}_B\omega_{IB} \\ \dot\varphi_1 \\ \vdots \\ \dot\varphi_{n_j} \end{pmatrix} \quad \dot u = \begin{pmatrix} {}_I a_B \\ {}_B\psi_{IB} \\ \ddot\varphi_1 \\ \vdots \\ \ddot\varphi_{n_j} \end{pmatrix} \in \mathbb{R}^{6+n_j} = \mathbb{R}^{n_u}$$

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

E_{fb} maps quaternions/Euler to twists.

Note: Slides often simplify to $\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{S}^\top\boldsymbol{\tau} + \mathbf{J}_c^\top\mathbf{F}_c$ for $\mathbf{h} = \mathbf{b} + \mathbf{g}$. Important property: $\dot{\mathbf{M}} - 2\mathbf{C}$ is skew-symmetric (passivity for control proofs).

7.4 Differential Kinematics

Floating: $J = [J_b\ J_j]$, $\dot x = J(q)u$ (task vel from gen. vel)

7.5 Contacts and Constraints

Hard $J_c u = 0$ (no-slip) **Const. acc.** $J_c\dot u + \dot J_c u = 0$

Soft $F_c = k\delta + d\dot\delta$ **Friction** Cone $|F_t| \le \mu F_n$

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

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

8 Dynamics of Floating Base Systems

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

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

Centroidal: $A_G\dot u + \dot A_G u = \sum F_{ext} + g_G$

(CMM A_G for momentum).

Exam: Derive centroidal momentum for quadruped, control CoM vel via $A_G u$ for balance under disturbances.

8.1 Constraint-Consistent Dynamics

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

Full EoM: $M\dot u + h = S^T\tau + J_c^T F_c$.

Constraint: $J_c\dot u = -\dot J_c u$ (no slip).

Projected dynamics: Use projector P ($PJ_c^T = 0$): $P(M\dot u + h) = PS^T\tau$. (Eliminates F_c for feasibility checks.)

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 CoM, feet as priorities.

Inv dyn: $\tau = S^+(M\dot u_d + h - J_c^T F_{c,d}) + N\tau_0$

Exam: $\dot u_d$ from tasks; QP for torque opt w/ cones.

8.4 Lyapunov Stability for PD Control

For $\tau = -K_p\tilde q - K_d\dot q + g(q)$:

Candidate $V = \frac{1}{2}\dot q^T M\dot q + U_g + \frac{1}{2}\tilde q^T K_p\tilde q$,

$\dot V = -\dot q^T K_d\dot q \le 0$ (asymptotically stable if $K_p, K_d > 0$)

Application in Robotics

9 Legged Robotics

9.1 System Analysis and DoF

Generalized Coordinates

(Separated into unactuated base $\mathbf{q}_b$ and actuated joints $\mathbf{q}_j$)

$$\mathbf{q} = \begin{bmatrix} \mathbf{q}_b \\ \mathbf{q}_j \end{bmatrix} \quad \text{dims:} \begin{cases} \text{Fixed Base:} & n = n_j \\ \text{Planar (2D):} & n = 3 + n_j \\ \text{Floating (3D):} & n = 6 + n_j \end{cases}$$

Contact Constraints $n_c = \sum n_{c,i}$

- **Point Contact** no-slip: $n_c = 3$ (3D) or $n_c = 2$ (2D) per foot
- **Surface Contact** $n_c = 6$ (3D) per foot

Mobility and Actuation Analysis

- **Uncontrollable DoF** dim($\mathbf{q}_b$) − rank($\mathbf{J}_{\text{contact}}$) (e.g., Planar base + 1 point contact: $3 - 2 = 1$ uncontrollable)
- **Degree of Underactuation** dim($\mathbf{q}_b$) $- n_c$ (If > 0, system is underactuated)

System Mobility (Net DoF) $\delta_{sys} = n_{total} - n_c$

Example: Quadruped, $n = 18$

- 3 Stance Legs (Point): $n_c = 3 \times 3 = 9$
- Net DoF: $18 - 9 = 9$

9.2 Dynamics Equations

Core equation for floating-base system:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \mathbf{J}_c^T\mathbf{F}_c = \mathbf{S}^T\boldsymbol{\tau}$$

Trick: For support-consistent dynamics, project into null-space of $\mathbf{J}_c$ (removes $\mathbf{F}_c$ dependency): $\mathbf{N}_c = \mathbf{I} - \mathbf{J}_c^+\mathbf{J}_c$.

Centroidal Momentum (Balance Control)

Centroidal momentum matrix $A_G(q)$:

$\dot h_G = A_G\dot q = \sum m_i(J_{P,i}^T v_i + J_{R,i}^T(I_i\omega_i + \omega_i \times I_i\omega_i))$

CoM tasks: Control $\dot h_G$ via contacts (underactuated base)

9.3 Contact Constraints

Underactuation: Torque can't directly control base, use contacts.

- Velocity level: $\mathbf{J}_c\dot{\mathbf{q}} = 0$ (no motion at contact points)

- Acceleration level: $\mathbf{J}_c\ddot{\mathbf{q}} + \dot{\mathbf{J}}_c\dot{\mathbf{q}} = 0$

- Friction: cone constraints on $\mathbf{F}_c$ (e.g., $\|\mathbf{F}_{c,\perp}\| \le \mu\mathbf{F}_{c,\|}$)

Exam hint: Always enforce as highest priority in hierarchical control to avoid slippage.

9.4 Inverse Differential Kinematics (Swing Leg Task)

Given constraints $\mathbf{J}_c\dot{\mathbf{q}} = 0$ and task $\mathbf{J}_{\text{swing}}\dot{\mathbf{q}} = \dot{\mathbf{r}}_{\text{swing}}^{\text{des}}$

- Stacked (may be singular): $\dot{\mathbf{q}} = \begin{bmatrix} \mathbf{J}_c \\ \mathbf{J}_{\text{swing}} \end{bmatrix}^+ \begin{bmatrix} 0 \\ \dot{\mathbf{r}}_{\text{swing}}^{\text{des}} \end{bmatrix}$

- Hierarchical/null-space (preferred, robust):

$$\dot{\mathbf{q}} = \mathbf{J}_c^+ \cdot 0 + \mathbf{N}_c\dot{\mathbf{q}}_0 \quad \dot{\mathbf{q}}_0 = (\mathbf{J}_{\text{swing}}\mathbf{N}_c)^+\dot{\mathbf{r}}_{\text{swing}}^{\text{des}}$$

Trick: Use for tasks like swing foot tracking; extend to acceleration level for dynamics.

9.5 Control Strategies

Common approaches (by robustness/exam frequency):

1. High-gain joint PD/PID: $\boldsymbol{\tau} = \mathbf{K}_p(\mathbf{q}^* - \mathbf{q}) + \mathbf{K}_d(\dot{\mathbf{q}}^* - \dot{\mathbf{q}})$ (poor for impacts due to instability on uneven terrain, prefer torque-based for compliance)
2. Inverse dynamics + low-gain: $\boldsymbol{\tau} = \boldsymbol{\tau}_{FB} + \boldsymbol{\tau}_{FF}$ (model-based feedforward)
3. Support-consistent ID: Project desired $\ddot{\mathbf{q}}^*$ into $\mathbf{N}_c$ null-space.
4. Task-space control: Regulate tasks (e.g., CoM, feet) via QP.
5. Hierarchical QP: Strict priorities (dynamics/contact highest).

Hint: High-gain fails due to impacts/unmodeled terrain; prefer torque control for compliance.

9.6 Actuator Types

Type	Torque Method	Pros/Cons
High-gear + SEA/sensor	Elasticity/sensor	Robust but slower response
Low-gear + current	Current $\approx$ torque	Fast, backdrivable impact-resistant
Hydraulic	Pressure valve	High power, hard to scale

- Low-gear best for dynamic locomotion (e.g.Mini Cheetah)
- Why torque control? Compliance for rough terrain (vs. stiff position control).

9.7 Whole-Body Control (WBC) Hierarchy

Typical priority order:
1. Dynamics: $\mathbf{M}\ddot{\mathbf{q}} + \mathbf{h} = \mathbf{J}_c^T \mathbf{f}_c + \mathbf{S}^T \boldsymbol{\tau}$.
2. Contact stability: $\mathbf{J}_c \ddot{\mathbf{q}} + \dot{\mathbf{J}}_c \dot{\mathbf{q}} = 0$.
3. Tasks (CoM/base/swing): e.g., $\mathbf{J}_{com}\ddot{\mathbf{q}} = \ddot{\mathbf{x}}_{com}^{ref} - \dot{\mathbf{J}}_{com}\dot{\mathbf{q}}$.
4. Regularization: min $\|\boldsymbol{\tau}\|$ or $\|\mathbf{f}_c\|$.

Torque command:

$$\boldsymbol{\tau}^d = \mathbf{M}_j(\mathbf{q})\ddot{\mathbf{q}}^* + \mathbf{h}_j(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{J}_{s,j}^T(\mathbf{q})\boldsymbol{\lambda}^*$$

Final: $\boldsymbol{\tau}^{ref} = \boldsymbol{\tau}^d + k_p \tilde{\mathbf{q}} + k_d \dot{\tilde{\mathbf{q}}}$.

9.8 Optimization-Based Control (QP/SLQ)

QP for WBC: min $\|\mathbf{A}x - \mathbf{b}\|^2$ s.t. constraints (tasks, limits)
Finite-time OCP (SLQ/DDP):

$$\min_{u(\cdot)} \Phi(x(T)) + \int_0^T L(x(t), u(t), t)\, dt \quad \text{s.t. constraints}$$

Trick: SLQ linear in horizon length (faster than DDP); use for MPC in legged systems.

10 Rotorcrafts

10.1 Key Assumptions and Modeling

- Core assumptions: CoG at body frame origin; rigid, symmetric structure; rigid propellers; neglect fuselage drag; near-hover (hub forces & rolling moments ≈ 0).
- Generalized coordinates for aerial robots:
$$q = [C_{BW}, v, \omega, \alpha_i, \omega_i]^T \in \mathrm{SO}(3) \times \mathbb{R}^{6+10}$$
(incl. base orientation, velocities, arm angles, rotor speeds)
- Newton-Euler equations (full dynamics):
$$\sum F_{\text{ext}} = m(\alpha_i, \omega_i)\,\dot{v} + b(v, \omega, \alpha_i, \omega_i) + g(C_{BW}, \alpha_i, \omega_i)$$
$$\sum T_{\text{ext}} = I_B(\alpha_i, \omega_i)\,\dot{\omega} + b(v, \omega, \alpha_i, \omega_i) + g(C_{BW}, \alpha_i, \omega_i)$$
- Simplified for quadrotor: $q = [\Theta, v, \omega]^T \in \mathbb{R}^3 \times \mathbb{R}^6$.
- Derivation methods: Newton-Euler momentum theory, Lagrange, or virtual work principle.

10.2 Propulsion and Thrust Models

- Thrust and drag torque (hover/low-speed): $T_i = b\omega_{p,i}^2, Q_i = d\omega_{p,i}^2$
- Classical momentum theory (ideal hover):
$$T = 2\rho A v_i^2, \quad P_{\text{ideal}} = T v_i = \frac{T^{3/2}}{\sqrt{2\rho A}}.$$
- Figure of Merit: $FM = P_{\text{ideal}}/P_{\text{actual}} < 1$.
- Dependencies: Thrust $T \propto \omega^2$ (independent of forward speed near hover); drag torque $Q \propto \omega^2$.
- Hover: Total $T = mg$, per rotor $\omega_i = \sqrt{T/(4b)}$ (quad).
- Allocation matrix: Solve $A\omega^2 = [T, M_x, M_y, M_z]^T$ (pseudo-inverse for over-actuated; min $\sum \omega^2$).
- Tilt for accel: $\theta_d = \arcsin(a_x/g)$, check drag $D \propto \omega^2$.
- Yaw signs: Positive d for CCW rotors (CW torque on body).

10.3 Control Allocation (Quadrotor, X-Configuration)

- Virtual inputs:
$$\begin{aligned} u_1 &= b(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) && \text{(collective thrust)},\\ u_2 &= lb(\omega_4^2 - \omega_2^2) && \text{(roll moment)},\\ u_3 &= lb(\omega_1^2 - \omega_3^2) && \text{(pitch moment)},\\ u_4 &= d(-\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2) && \text{(yaw moment)}. \end{aligned}$$
- Allocation matrix (solve for ω_i^2):
$$\begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} \frac{1}{b} & 0 & \frac{1}{lb} & -\frac{1}{d} \\ \frac{1}{b} & -\frac{1}{lb} & 0 & \frac{1}{d} \\ \frac{1}{b} & 0 & -\frac{1}{lb} & -\frac{1}{d} \\ \frac{1}{b} & \frac{1}{lb} & 0 & \frac{1}{d} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}.$$
Note: For + configuration, adjust signs (e.g., $u_2 = lb(\omega_1^2 + \omega_3^2 - \omega_2^2 - \omega_4^2)$) exams often specify X-config.
- For over-actuated systems (e.g., hexacopter): Multiple solutions; choose minimum-energy (min rotor-speed norm) or min max(ω_i).

10.4 Attitude Dynamics (Linearized near Hover)

- Linearized equations:
$$\ddot{\phi} \approx \frac{u_2}{I_{xx}} = \frac{lb}{I_{xx}}(\omega_4^2 - \omega_2^2), \quad \ddot{\theta} \approx \frac{u_3}{I_{yy}}, \quad \ddot{\psi} \approx \frac{u_4}{I_{zz}}.$$
- PD attitude controller (decoupled double integrators):
$$\begin{aligned} u_2 &= k_{p\phi}(\phi_d - \phi) - k_{d\phi}\dot{\phi},\\ u_3 &= k_{p\theta}(\theta_d - \theta) - k_{d\theta}\dot{\theta},\\ u_4 &= k_{p\psi}(\psi_d - \psi) - k_{d\psi}\dot{\psi}. \end{aligned}$$
- Tuning hint: k_p sets rise time/natural frequency; k_d sets damping (avoid large k_d to prevent saturation).

10.5 Position and Altitude Control

- Altitude (small-angle approx.):
$$\ddot{z} \approx g - u_1 m^{-1} \cos\phi \cos\theta$$
$$u_1 = (m(g - k_{p_z}(z_d - z) - k_{d_z}\dot{z}))(\cos\phi\cos\theta)^{-1}$$
- Thrust vector control (desired ${}^E\mathbf{T} = [T_x, T_y, T_z]^T$):
$$u_1 = \|{}^E\mathbf{T}\|, \theta_d = \arcsin\left(\frac{T_x}{u_1}\right), \phi_d = -\arcsin\left(\frac{T_y}{u_1}\right)$$
- Forward acceleration: $\propto \sin\theta$ or $\sin\phi$ (indep. of yaw ψ)

Tricks and Hints

- Singularities: Euler angles singular at $\theta = \pm 90°$; use quaternions for aggressive maneuvers.
- Derivatives: Near hover, $\dot{\phi} \approx p, \dot{\theta} \approx q, \dot{\psi} \approx r$.
- Linearization: Small-angle approx. $\sin\phi \approx \phi$, $\cos\phi \approx 1$.
- Power: $\propto \omega^3$; minimize max(ω_i) over sum ω_i^2.
- Underactuation: Position/attitude coupled; check config. (X vs. +) for allocation.
- Common pitfalls: Division by $\cos\phi\cos\theta$ (numerical issues); neglect drag in hover.
- Thrust vector: arcsin can cause singularities at high tilts; use quaternions for large angles (as in slides).

11 Fixed-Wing Aircraft

Key Assumptions and Simplifications

- **Rigid symmetric structure** Constant, diagonal inertia matrix
- **Constant mass** Neglect fuel burn (electric/short duration)
- **No stall** Operating strictly in the linear lift domain
- **Lumped Aerodyn.** Lift/drag of wing and fuselage combined
- **No Side-Force** Sideslip β is regulated to 0
- **No Interference effects** e.g. prop wash on control surfaces
- **CoM Origin** Body frame origin placed at Center of Mass

11.1 Reference Frames and Kinematics

- **Inertial Frame E (NED)** North (x_E), East (y_E), Down (z_E)
- **Body Frame B** x_B (nose), y_B (right wing), z_B (down)
- **Wind Frame W** x_W aligned with air-mass velocity vector $\mathbf{V}_a$

Body Velocity in B(ody frame): $\mathbf{V}_a^B = (u, v, w)^T$
Airspeed: $V = \|\mathbf{V}_a^B\| = \sqrt{u^2 + v^2 + w^2}$ (always positive)
Body Rates (in B): $\boldsymbol{\omega}^B = (p, q, r)^T$

Airflow Angles:

$$\begin{aligned} \alpha &= \arctan(w/u) && \text{(Angle of Attack)}\\ \beta &= \arcsin(v/V) && \text{(Sideslip Angle)} \end{aligned}$$

Euler Angles: $\mathbf{X} = (\phi, \theta, \psi)^T$ (roll, pitch, yaw)
Rotation Matrix C_{EB} (ZYX sequence):

$$\mathbf{V}_a^E = C_{EB}\,\mathbf{V}_a^B, \quad C_{EB} = C_z(\psi)\,C_y(\theta)\,C_x(\phi)$$

Angular Rates Relation:

$$\begin{pmatrix} p \\ q \\ r \end{pmatrix} = \begin{pmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{pmatrix} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix}.$$

Singularity at $\theta = \pm\pi/2$ (gimbal lock, use quaternions)

Polar Coordinates (No Wind Assumption)

Longitudinal θ (pitch), γ (flight path), α, Relation: $\theta = \alpha + \gamma$
Lateral ψ (yaw/heading), ξ (course)

Why no wind? If wind = 0, ground velocity equals airspeed, simplifying position derivatives.

With Steady Wind $\mathbf{v}_g = \mathbf{v}_a + \mathbf{v}_w$ (Wind Triangle) Difference between Heading ψ and Course angle χ (direction of $\mathbf{v}_g$) χ is the *Crab Angle*.

11.2 Aerodynamics

Bernoulli Equation (Incompressible) $\frac{1}{2}v^2 + gh + \frac{P}{\rho} = \text{const}$
Dynamic Pressure: $q = \frac{1}{2}\rho V^2$ (ρ: air density, V: airspeed)
Aerodynamic Forces & Moments

Type	2D Airfoil (per unit span)	3D Whole Aircraft
Lift ($\perp$)	$dL = q\,c\,C_l\,dy$	$L = q\,S\,C_L$
Drag ($\|$)	$dD = q\,c\,C_d\,dy$	$D = q\,S\,C_D$
Moment	$dM = q\,c^2\,C_m\,dy$	$M = q\,S\,\bar{c}\,C_M$

q: dynamic pressure, S: wing area, $c/\bar{c}$: chord/mean
$C_{l,d,m}$ and $C_{L,D,M}$ functions of: α, Re, Ma (Mach)
Reynolds Number: Re $= \frac{VL}{\nu}$ (L: char. length, ν: kin. viscosity)
Stall: Occurs at high α; C_L drops post-max. Avoid in operation!

- Coefficients from CFD (Xfoil, Javafoil), wind tunnel, flight data sysid or linear approximations ($C_L \approx C_{L0} + C_{L\alpha}\alpha$)
- Control surfaces modify C_L/C_M: Ailerons (Roll), Elevator (Pitch), Rudder (Yaw)
- $T = \frac{1}{2}\rho v^2 S C_D, C_L = \frac{2mg}{\rho v^2 S}$
- Max range at max C_L/C_D;
- Max endurance at max $C_L^{1.5}/C_D$
- gliding angle $\tan\gamma = (L/D)^{-1}$.

11.3 Equation of Motion and Dynamics

Model as single rigid body (Newton-Euler):

$$\begin{aligned} m\dot{\mathbf{v}}^B &= \mathbf{F}^B - \boldsymbol{\omega}^B \times m\mathbf{v}^B\\ \mathbf{I}\dot{\boldsymbol{\omega}}^B &= \mathbf{T}^B - \boldsymbol{\omega}^B \times \mathbf{I}\boldsymbol{\omega}^B \end{aligned}$$

Forces in B: $\mathbf{F}^B = \mathbf{F}^B_{\text{aero}} + \mathbf{F}^B_{\text{prop}} + \mathbf{F}^B_{\text{grav}}$
Simplified Aero Forces (in wind frame, then rotate):

- Assume no side force ($Y = 0$).
- Lift/Drag lumped.

11.4 Key Flight Conditions

Steady Level Flight

$$C_L = \frac{2mg}{\rho V^2 S}, \quad T = D.$$

Note: Drag $D \propto m$ (since $V \propto \sqrt{2mg/(\rho S C_L)}$ at fixed α)
Coordinated Turn (no sideslip, constant altitude/speed)

$$\tan\phi = \frac{V^2}{g R_{\text{turn}}}, \quad L = \frac{mg}{\cos\phi}$$

Stall speed increases by $\sqrt{n}$ where $n = \frac{1}{\cos\phi} = \frac{L}{W}$ load Factor (check $C_L < C_{L,\text{max}}$ for margin)
Gliding (Thrust $T = 0$)

$$\gamma = -\arctan(D/L) \approx -C_D/C_L \text{(for small angles)}$$

Note: Max range occurs at max L/D

- Best: **Range** max C_L/C_D **Endurance** max $C_L^{3/2}/C_D$
- **Stall Speed** $V_{\text{stall}} = \sqrt{\frac{2mg}{\rho S C_{L,\text{max}}}}$
- **Stall Margin** Required C_L increases in turn, check $C_{L,\text{max}}$
- **Guidance** Use Ground speed for navigation, Airspeed for control loop (aerodynamic stability)

11.5 Control Strategies

- **Cascaded Control** Position (Outer) $\rightarrow$ Velocity/Attitude (Mid) $\rightarrow$ Rates (Inner).
- **Attitude Control** PID often sufficient.
- **Guidance** Feedforward is useful for wind rejection.
- **Underactuated** Cannot control all 6 DOFs independently (e.g., rolling creates yaw/sideslip).

12 Recipes - Steps to Solve Common Tasks

12.1 Kinematics

Consider a 2D mobile manipulator robot with a wheeled base (horizontal position x_b), a prismatic joint for height (z_l), and a 3-joint arm (ϕ_1, ϕ_2, ϕ_3). Links have lengths l_1, l_2, l_3. Generalized coordinates: $q = (x_b, z_l, \phi_1, \phi_2, \phi_3)^T$.

1. Write the forward kinematics for the end-effector position $r_e = (x_e, z_e)^T$ in the world frame.
2. Derive the geometric Jacobian $J_e(q)$ for the end-effector linear velocity.
3. Provide a singular configuration and explain the lost degree of freedom.

4. For a desired end-effector velocity $v_e^* = (1, 0)^T$ m/s, compute the minimal-norm joint velocity $\dot{q}^*$ using pseudoinverse (assume non-singular q).
5. The system is redundant. Formulate a multi-task control where primary task is v_e^* and secondary task is to keep $\phi_1 = 0$ and $\dot{z}_l = 0$. Use null-space projection for equal priority.
6. If secondary task has higher priority, how would you modify the control law?
7. Discuss how to handle singularities using damped least-squares.

Solution

1. $x_e = x_b + l_1 \cos\phi_1 + l_2 \cos(\phi_1 + \phi_2) + l_3 \cos(\phi_1 + \phi_2 + \phi_3)$

   $z_e = z_l + l_1 \sin\phi_1 + l_2 \sin(\phi_1 + \phi_2) + l_3 \sin(\phi_1 + \phi_2 + \phi_3)$
2. $J_e = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

   $\begin{matrix} -l_1 s_1 - l_2 s_{12} - l_3 s_{123} & -l_2 s_{12} - l_3 s_{123} & -l_3 s_{123} \\ l_1 c_1 + l_2 c_{12} + l_3 c_{123} & l_2 c_{12} + l_3 c_{123} & l_3 c_{123} \end{matrix}\Bigg]$

   (where $s_1 = \sin\phi_1$, etc.)
3. E.g., $\phi_2 = \phi_3 = 0$, arm fully extended; loses control in radial direction.
4. $\dot{q}^* = J_e^+ v_e^*$, where $J_e^+ = J_e^T (J_e J_e^T)^{-1}$.
5. Stacked Jacobian $J = \begin{bmatrix} J_e \\ J_{sec} \end{bmatrix}$, where

   $J_{sec} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$, desired $\dot{q}^* = J^+ \begin{bmatrix} v_e^* \\ 0 \end{bmatrix}$
6. Prioritize secondary:
   $\dot{q}^* = J_{sec}^+ 0 + N_{sec} J_e^+ (v_e^* - J_e J_{sec}^+ 0)$,
   where $N_{sec} = I - J_{sec}^+ J_{sec}$.
7. Use damped pseudoinverse $J^+ = J^T (J J^T + \lambda I)^{-1}$ to avoid high velocities near singularities.

12.2 Dynamics

The equations of motion are $M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) = \tau$.

1. Derive a joint-space PD controller with gravity compensation for tracking q^*.

2. Formulate an inverse dynamics control law for desired acceleration $\ddot{q}^*$.

3. For a floating-base version (add base orientation θ_b), explain why M is not full rank.

4. Propose a task-space impedance controller for end-effector force tracking.

5. Discuss redundancy resolution in torque space for over-actuated systems.

Solution

1. $\tau = g(q) + K_p(q^* - q) - K_d \dot{q}$.

2. $\tau = M\ddot{q}^* + C\dot{q} + g$.

3. Floating base has unactuated DOFs (linear momentum conservation); M has rank deficiency.

4. $\tau = J_e^T (K_p \Delta r_e - K_d v_e + f^*) + g$, where $\Delta r_e = r_e^* - r_e$.

5. Use null space: $\tau = \tau_{task} + N\tau_0$, where $N = I - J^+ J$, τ_0 optimizes secondary objectives like joint limits.

12.3 Legged Robots

Consider a quadruped with 3 DOF per leg, point feet, two feet in contact.

1. (3 pts) What is the dimension of the null space for joint torques?

2. (4 pts) Formulate the contact-consistent dynamics and how to project accelerations.

3. (3 pts) Describe a simple balance controller using CoM projection.

Solution

1. Total DOF: 6 (floating base) + 12 (joints) = 18
constraints: 2 feet × 3 = 6; null space dim = 18 - (18 - 6) = 6
(redundancy in torque)

2. $M\ddot{q} + h = S\tau + J_c^T \lambda$
project to consistent $\ddot{q} = (I - J_c^+ J_c)\ddot{q}_0 + J_c^+ \dot{J}_c \dot{q}$

3. Control torques to keep projected CoM within support polygon, e.g., via virtual model control.

12.4 Rotorcraft

For a quadcopter.

1. (3 pts) Write the thrust allocation for position control.

2. (3 pts) Explain coupling between position and attitude.

3. (2 pts) Why are feedforward terms useful in tracking?

Solution

1. Total thrust $f = m(\ddot{z}^* + g)$, moments from differential rotor speeds.

2. Underactuated: attitude must tilt to generate horizontal forces.

3. Compensate nonlinearities for better tracking without relying solely on feedback.

12.5 Fixed-Wing Aircraft

For a fixed-wing UAV in level flight.

1. (3 pts) Derive the required bank angle for a coordinated turn of radius R at speed V.

2. (2 pts) How does wind affect guidance?

3. (2 pts) Explain airspeed vs. groundspeed in control.

Solution

1. $\phi = \tan^{-1}(V^2/(gR))$.

2. Wind disturbs position; guidance uses groundspeed feedback for correction.

3. Airspeed for aerodynamic stability, groundspeed for navigation.