Chapter Outline

Propulsion, Forces, and Observation Modeling

Chapter 4
Vehicle Motion Modeling and Simulation
4.1 Modeling Fundamentals and Kinematics
4.2 Rigid-Body Dynamics and Solvers
4.3 Propulsion, Forces, and Observations
4.4 Trim, Linearization, and Dynamics
4.5 Multicopter Modeling and Simulation
4.6 Measurement and Static Identification
4.7 Dynamic Identification and Refinement
4.8 Typical Vehicles and Extensions
4.9 Modeling Tools and Deployment
4.10 API and SDK Modeling Interfaces
4.11 Examples and Integrated Practice
4.12 Exercises and Course Projects

5 Units in This Section

4.3.1 Actuator Subsystem Modeling

Textbook: Sec. 4.1.4.3; Eqs. (4.29–30) in Sec. 4.1.4.4

4.3.2 Force and Torque Synthesis

Textbook: Sec. 4.1.4.4, Force and Torque Synthesis Subsystem

4.3.3 Environment Subsystem Modeling

Textbook: Secs. 4.1.4.2 and 4.1.4.4

4.3.4 Terrain and Contact

Textbook: Secs. 4.1.4.2 and 4.1.4.4

4.3.5 Sensor Subsystem Modeling

Textbook: Secs. 4.1.4.6 and 4.1.4.7, Sensors and 3D Environment

Section Introduction

Add Propulsion and Observations to the Four-Layer Model

- **Input:** Commands drive motors; speed becomes body force through thrust and reaction-torque relations. Section 4.2 prescribed four propeller thrusts; this section adds their source.
- **Environment:** Wind and density change relative airflow and aerodynamic forces; contact changes ground support. Both enter the force sum.
- **Output:** Rigid-body states are ground truth. Accelerometer and gyroscope measurements also require mounting orientation, bias, and sampling before the autopilot can use them.

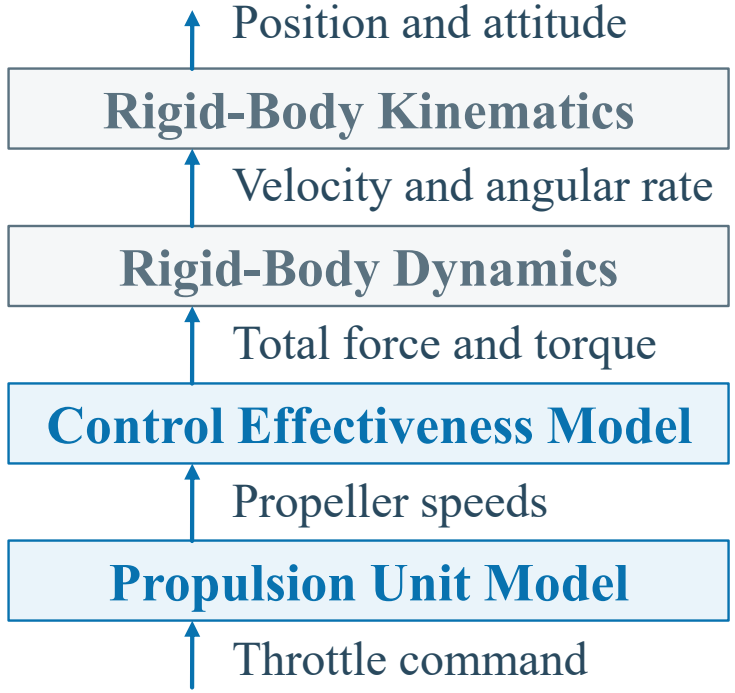


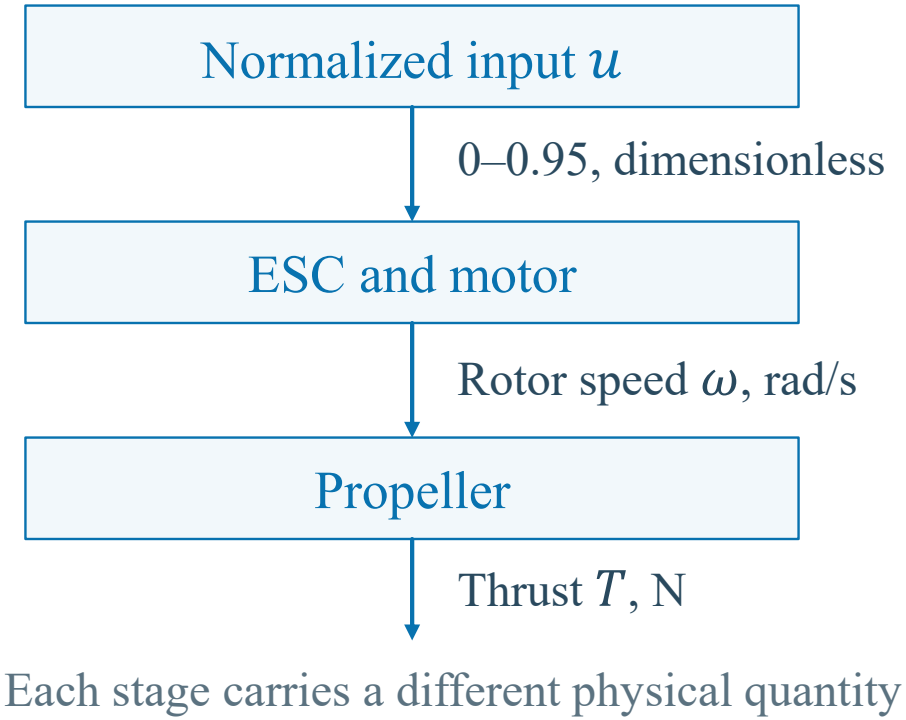
Figure: Four-layer motion model

Real inputs pass through propulsion; motion ground truth passes through an observation model.

4.3.1 Actuator Subsystem Modeling: Signal Meaning

Commands Become Thrust Through the ESC, Motor, and Propeller

- Normalized command u: dimensionless. PWM (pulse-width modulation) encodes microsecond pulse widths. Electronic speed controller (ESC) → motor → propeller → thrust. $u = 0.5$ is not half rated thrust.
- **M0:** the e0_MinModelTemp minimal template. [installation directory]/RflySimAPIs/ 4.RflySimModel/1.BasicExps/ e0_MinModelTemp/Exp1_MinModelTemp.slx
- inPWMs: 16 normalized channels. Motor inputs are limited to 0–0.95. Check units at every connection.



A normalized command sets the input scale; the propulsion model determines the physical output.

4.3.1 Actuator Subsystem Modeling: Time-Domain Dynamics

Why Motor Speed Cannot Jump with the Command

- The first-order approximation combines the main electrical and rotational-inertia lags in time constant τ_m; ω_∞ is the steady-state target for the current input.
- When the target exceeds current speed, the right-hand difference is positive and the motor accelerates. As it approaches the target, the gap and speed change both decrease.
- A larger time constant gives a smaller speed derivative for the same gap. This is an approximation over a specified operating range; no prior Laplace transform is needed.

$$\tau_m \dot{\omega} + \omega = \omega_\infty, \qquad \dot{\omega} = \frac{\omega_\infty - \omega}{\tau_m}$$

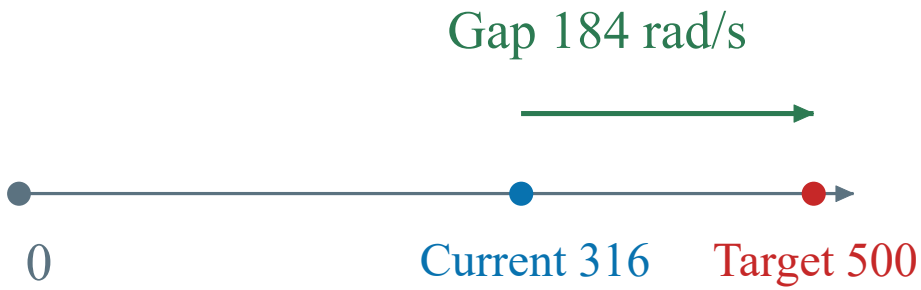
A first-order model sets acceleration from the gap between current and target speed.

4.3.1 Actuator Subsystem Modeling: Target and Response

Find the Target Speed, Then Evolve the Current State

- Steady-state map $\omega_\infty = C_R u + \omega_{\text{bias}}$: endpoint after transients decay at fixed input. Measure each point; remeasure if voltage, propeller, load, or airflow changes. Static-thrust bench tests also have transients.
- Acceleration uses target minus current speed; smaller gap or larger τ_m slows response. With a 0.10 s time constant: 184/0.10 = 1840 rad/s² instantaneously.

Two quantities at the same instant



Smaller gap, slower acceleration

Target sets endpoint; state sets current output

$$\dot{\omega} = (\omega_\infty - \omega)/\tau_m$$

The steady-state curve gives the endpoint; it does not give the time needed to reach it.

4.3.1 Actuator Subsystem Modeling: Step Example

Read the Time Constant from the Response Curve

- From zero speed to a 500 rad/s target, blue $\tau_m = 0.05$ s reaches 316.1 at 0.05 s and 475.1 rad/s at 0.15 s.
- Orange $\tau_m = 0.10$ s reaches only about 196.7 rad/s at the same 0.05 s. At one respective time constant, both reach 316.1 rad/s: 63.2% of the change.

$$\omega(t) = \omega_\infty + (\omega_0 - \omega_\infty)e^{-t/\tau_m}$$

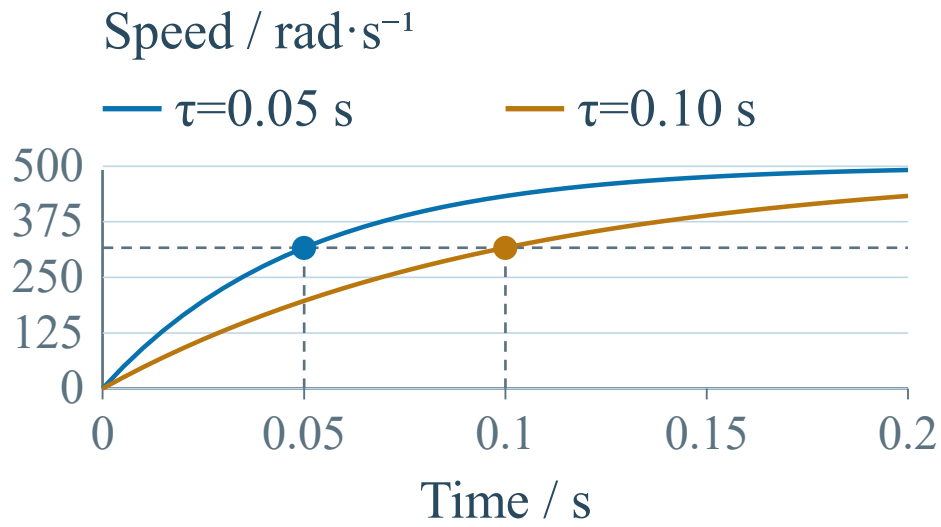


Figure: Dashed line marks 63.2% (316 rad/s)

63.2%: 316 rad/s
Blue 0.05 s; orange 0.10 s

A first-order speed response completes about 63.2% of its total change in one time constant.

4.3.1 Actuator Subsystem Modeling: The Square Law

Convert Speed Units Before Squaring for Thrust

- Low-speed approximation: $T = c_T\omega^2$, $Q = c_M\omega^2$. For $c_T = 2 \times 10^{-5}$ N/(rad/s)², 500 rad/s gives 5 N; half speed gives 1.25 N.

- c_T and c_M are thrust and reaction-torque coefficients (textbook Eq. 4.29; kT and kQ in code). Step-example thrust after one time constant is only about 40% of final. 6000 rpm = 628.3 rad/s gives 7.90 N.

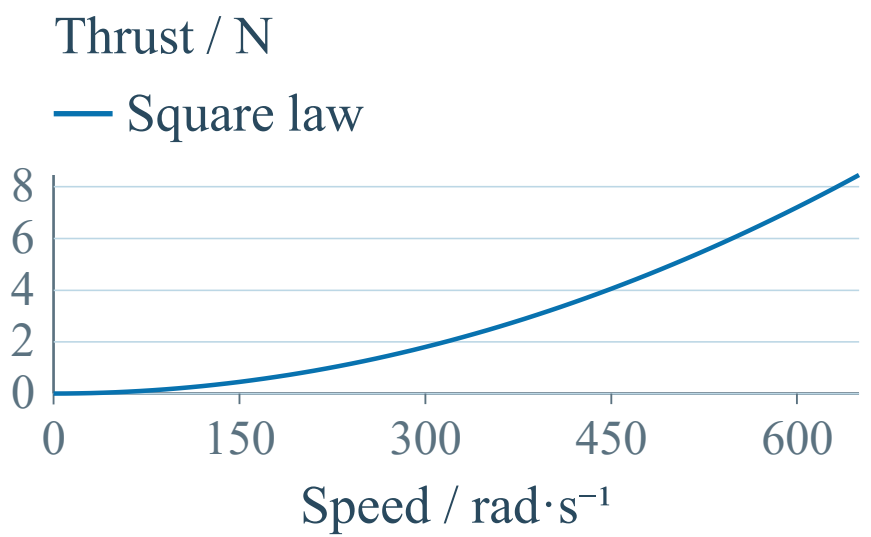


Figure: Halving speed reduces thrust to one quarter

$$\omega = (2\pi/60)n_{\text{rpm}}$$

Thrust depends on current speed squared, so it does not increase in direct proportion to speed.

4.3.1 Actuator Subsystem Modeling: Square-Law Conditions

Advance Ratio and Varying Thrust and Torque Coefficients

$$J = \frac{V}{nD}$$

Advance ratio compares inflow speed with propeller rotation. It is dimensionless; see textbook Eq. (4.30).

V: Inflow along the propeller axis, normal to its disk, in m/s.
n: Speed in revolutions per second; divide rpm by 60 first.
D: Propeller diameter in m.

Varying coefficients: Even for similar propellers and conditions, c_T and c_M vary with advance ratio J.

Validity: In hover or low-speed flight, use constant coefficients only if J is small and coefficient changes are negligible. For substantial inflow changes, use condition-specific tables or identify the coefficients.

4.3.1 Actuator Subsystem Modeling: Textbook Fig. 4.9

The Complete Chain from Throttle to Thrust

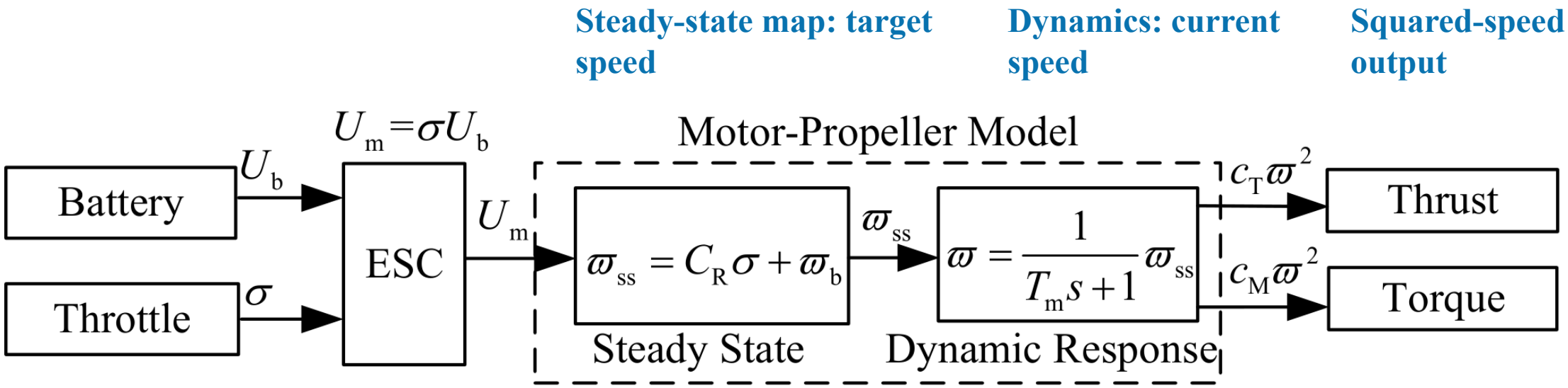


Figure 4.9 Motor-Propeller Actuator Output Response

- **Notation:** throttle $\sigma \rightarrow u$, $\varpi_{ss} \rightarrow \omega_\infty$, $\varpi \rightarrow \omega$, $T_m \rightarrow \tau_m$, $\varpi_b \rightarrow \omega_{bias}$; retain c_T, c_M (kT, kQ in code). Section 4.4 introduces s.

4.3.1 Actuator Subsystem Modeling: Actual Template

M0: Limit the Target, Then Compute Steady-State Thrust

- M0 clips input to [0,0.95], giving u_c. For $u_c < 0.05$, the target is zero; otherwise use $842.1u_c + 22.83$. Time constant: 0.0214 s.
- $u = 0.5$ gives target $842.1 \times 0.5 + 22.83 = 443.88$ rad/s. With $c_T = 1.681 \times 10^{-5}$, one propeller gives 3.312 N; four give 13.25 N.
- Mass 1.515 kg; here $g = 9.8$. Weight 14.85 N > four-propeller thrust 13.25 N: downward acceleration only when airborne and level.

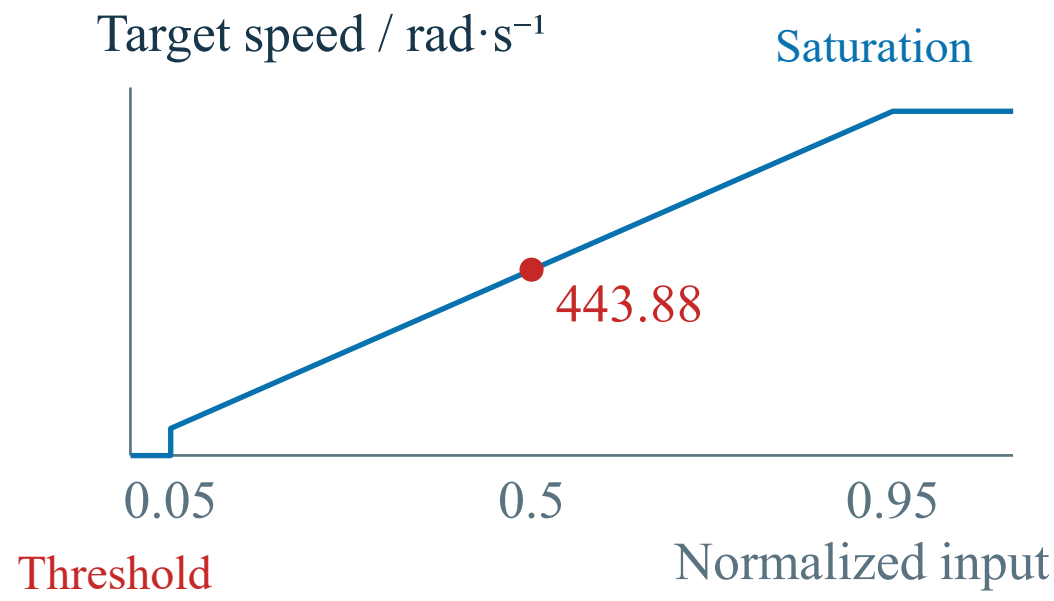


Figure: Threshold and saturation change the steady-state target

The threshold decides whether the target is zero; actual speed still evolves continuously.

4.3.1 Actuator Subsystem Modeling: Step and Shutdown

Track Command, Speed, and Thrust Across Shutdown

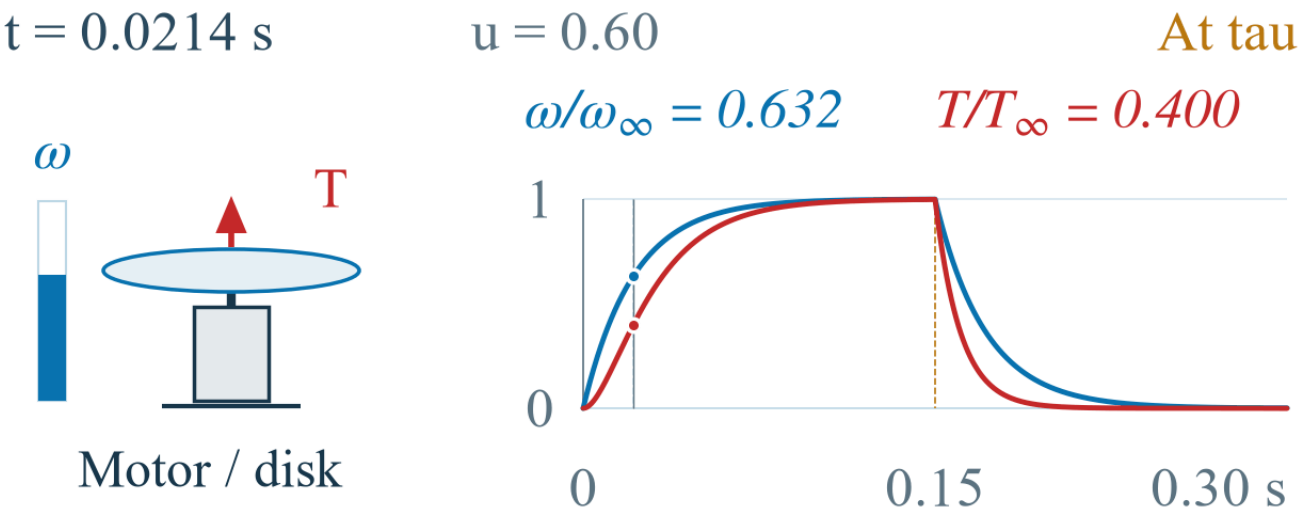


Figure: After shutdown, speed decays continuously and thrust follows its square

- Shut off input 0.6: target speed → zero; actual speed decays continuously. M0 threshold test: 0.04/0.06 → targets 0/73.36 rad/s; saturation test: 0.95/1.0 → 822.8 rad/s for both.

Shutdown changes the target first; speed and thrust then decay continuously.

Chapter Outline

Propulsion, Forces, and Observation Modeling

Chapter 4
Vehicle Motion Modeling and Simulation
4.1 Modeling Fundamentals and Kinematics
4.2 Rigid-Body Dynamics and Solvers
4.3 Propulsion, Forces, and Observations
4.4 Trim, Linearization, and Dynamics
4.5 Multicopter Modeling and Simulation
4.6 Measurement and Static Identification
4.7 Dynamic Identification and Refinement
4.8 Typical Vehicles and Extensions
4.9 Modeling Tools and Deployment
4.10 API and SDK Modeling Interfaces
4.11 Examples and Integrated Practice
4.12 Exercises and Course Projects

5 Units in This Section

4.3.1 Actuator Subsystem Modeling

Textbook: Sec. 4.1.4.3; Eqs. (4.29–30) in Sec. 4.1.4.4

4.3.2 Force and Torque Synthesis

Textbook: Sec. 4.1.4.4, Force and Torque Synthesis Subsystem

4.3.3 Environment Subsystem Modeling

Textbook: Secs. 4.1.4.2 and 4.1.4.4

4.3.4 Terrain and Contact

Textbook: Secs. 4.1.4.2 and 4.1.4.4

4.3.5 Sensor Subsystem Modeling

Textbook: Secs. 4.1.4.6 and 4.1.4.7, Sensors and 3D Environment

4.3 Propulsion, Forces, and Observations

4.3.2 Force and Torque Synthesis: Physical Connections

Textbook Fig. 4.3: Propulsion, Body, Environment, and Observations

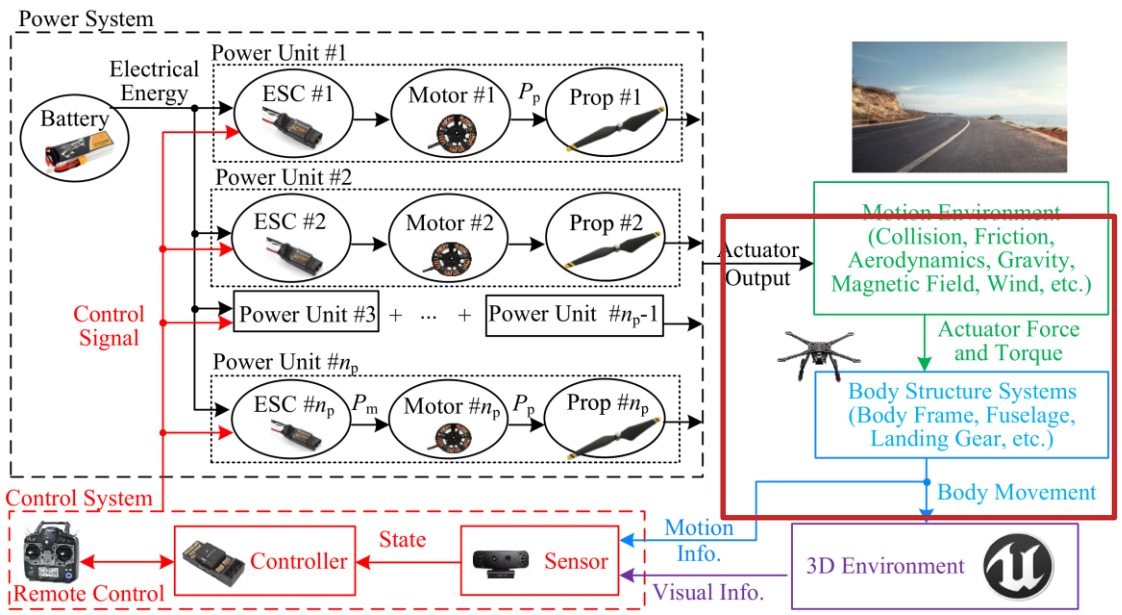


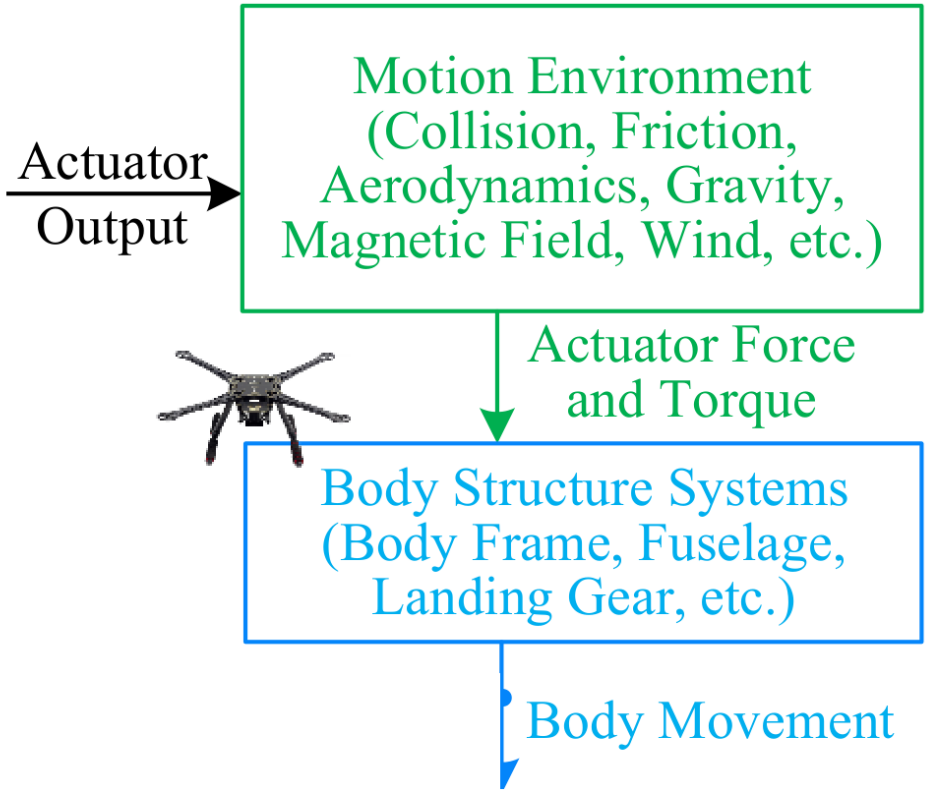
Fig. 4.3 Multicopter component geometry and topology

- Propulsion units produce thrust and reaction torque, sent through the frame to the body. Collision, friction, and aerodynamic effects from the motion environment act on the same body.
- Mounting position, axis direction, and spin direction set their contributions to rigid-body equations. Equal propeller outputs at different positions give different resultant torques, all about the center of mass.

The same thrust at different locations produces body torques in different directions.

4.3.2 Force and Torque Synthesis: Fig. 4.3 Detail

Actuators Act on the Body Through the Motion Environment



- **Actuator deflection:** Propulsion-unit thrust and reaction torque enter the motion environment together with collision, friction, aerodynamics, gravity, magnetic field, and wind.
- **Forces and torques:** Their resultant acts on the body system—arms, fuselage, and landing gear. This unit computes that resultant force and torque about the center of mass.
- **Body motion:** Motion reaches the 3D environment and sensors, then returns through the controller to the ESCs, closing the loop.

Figure 4.3 (Detail) Actuators → Motion Environment → Body System

4.3.2 Force and Torque Synthesis: Frames and Moment Arms

Use the Center of Mass as Origin for Each Propeller Force

- As in Section 4.1, body frame b is forward-right-down (FRD); Earth frame e is north-east-down (NED). R maps body column vectors to the Earth frame. Take every torque about the center of mass.
- First assume all four rotor axes are parallel to the body down axis. Thrust magnitude $T_i \geq 0$ acts upward, so the force vector is $[0,0,-T_i]^T$.
- Moment arm $\rho_i = [x_i, y_i, z_i]^T$ points from the center of mass to rotor i. Update it when a payload shifts the center of mass.

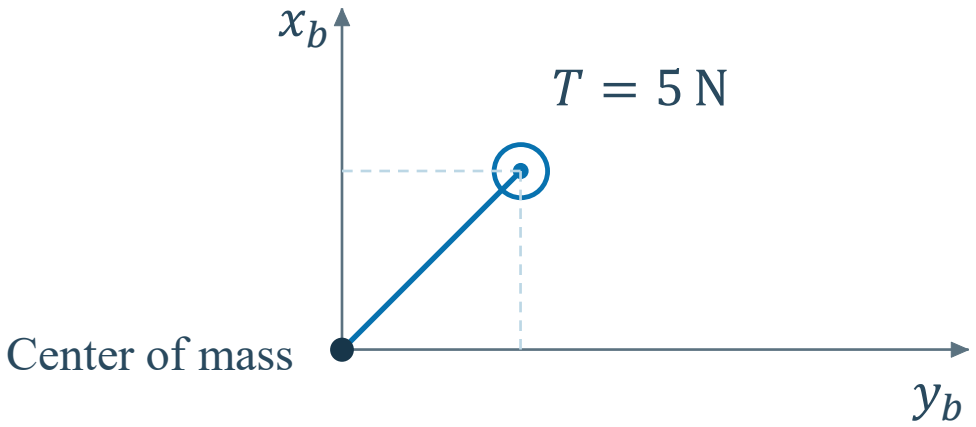
$$\mathbf{F}_{i,b} = \begin{bmatrix} 0 \\ 0 \\ -T_i \end{bmatrix}, \qquad \boldsymbol{\rho}_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$$

In FRD, upward thrust has a negative third component; the arm points from center of mass to rotor.

4.3.2 Force and Torque Synthesis: Cross-Product Example

One Propeller Gives Both Roll and Pitch Moments

- FRD axes are positive forward, right, and down. Upward thrust $T_i \geq 0$ is $[0,0,-T_i]^T$; arm $\boldsymbol{\rho}_i$ points from center of mass to rotor.
- Cross product: $\boldsymbol{\rho}_i \times \mathbf{F}_i = [-y_i T_i, x_i T_i, 0]^T$. Increasing front-right thrust gives negative roll and positive pitch, with no yaw moment from this arm.
- With $x_i = y_i = 0.2$ m and $T_i = 5$ N, the result is $[-1,1,0]^T$ N·m: both the right side and nose tend to rise.



⊙ Upward, out of the page

$M_x = -1,\ M_y = +1$

Figure: Two moment-arm contributions from one propeller

Lateral position sets the roll moment of upward thrust; fore-aft position sets its pitch moment.

4.3.2 Force and Torque Synthesis: Reaction-Torque Signs

Yaw Reaction Torque Comes from Rotation Direction

- As the motor drives the rotor, the body receives opposite torque, with steady-state magnitude $Q_i = c_{\mathrm{M}}\omega_i^2$. Add it separately to the moment-arm cross product.
- From above, positive rotation about down is clockwise. For body reaction torque, the textbook uses $s_i = \pm 1$: odd rotors positive, even rotors negative. Follow the actual model's numbering and spin directions.

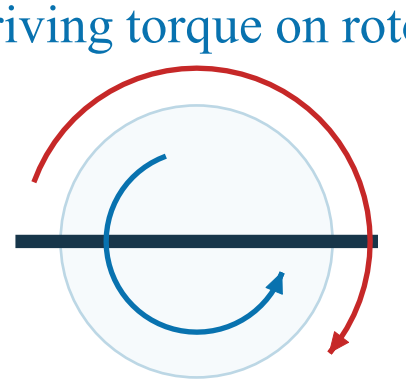


Figure: Driving and reaction torques are opposite

$$\mathbf{M}_{i,b} = \boldsymbol{\rho}_i \times \mathbf{F}_{i,b} + [0,0,s_i Q_i]^{\mathsf{T}}$$

Body reaction torque opposes the driving torque applied to the rotor.

4.3.2 Force and Torque Synthesis: Fig. 4.10 Geometry

General Rotor Positions and X-Layout Numbering

- Angle and radial distance describe a general rotor position. Geometry determines each rotor's contribution column.

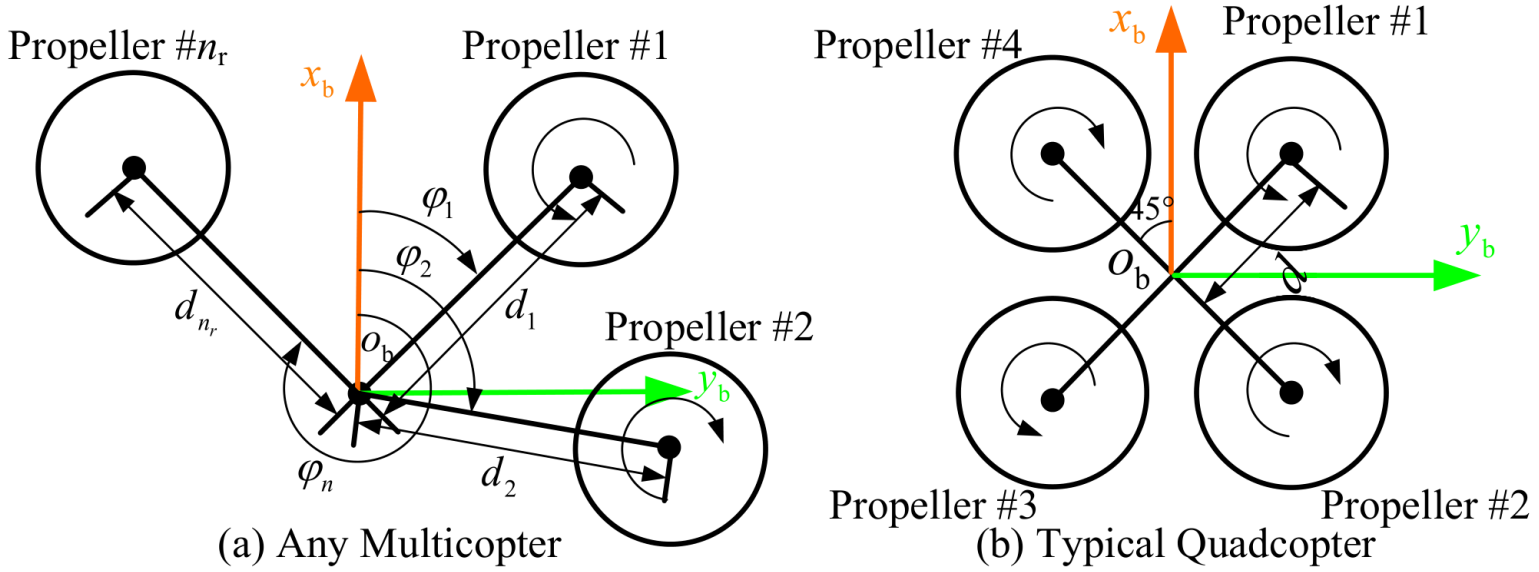


Figure 4.10 Force Synthesis for Multicopter Actuators

- X layout: start at front-right rotor 1; number clockwise at 45°, 135°, 225°, 315°.
- Radial arm $d = 0.25$ m. Both axis arms are projections: $a = d/\sqrt{2} = 0.1768$ m. Yaw signs follow the equation.

4.3.2 Force and Torque Synthesis: Fig. 4.10 Geometry

Express X-Layout Numbering as Signed Rotor Coordinates

Arm projection

- Use the textbook X numbering and 2 kg mass. Radial arm $d = 0.25\,\text{m}$ has projection $a = d/\sqrt{2} \approx 0.1768\,\text{m}$. Body axes are positive forward, right, and down.

Four rotor coordinates

- Relative to the center of mass, planar coordinates of rotors 1–4 are $(+a, +a)$, $(-a, +a)$, $(-a, -a)$, $(+a, -a)$, all with $z = 0$. Set position signs before deriving each column.

Single-rotor moment

- Rotor 1 has upward force $[0,0,-T_1]^T$. Position crossed with force gives $(\tau_x, \tau_y) = (-aT_1, +aT_1)$. Apply the same rule to the other rotors; add yaw reaction torque separately.

4.3.2 Force and Torque Synthesis: Meaning of Each Column

Derive the First Column Before the Full Synthesis Matrix

- Define resultant vector $w = [T_\Sigma, \tau_x, \tau_y, \tau_z]^T$. Its first entry is the positive magnitude of upward thrust; use a negative body-z component in the rigid-body force equation.
- Front-right rotor 1 is at $[a, a, 0]^{\mathrm{T}}$, with textbook sign $s_1 = +1$. For $\eta = c_{\mathrm{M}}/c_{\mathrm{T}}$, each 1 N of its thrust contributes $[1, -a, a, \eta]^{\mathrm{T}}$.
- For each remaining column, substitute that rotor's fore-aft position, lateral position, and reaction-torque sign. This transfers more easily than memorizing one frame's entire matrix.

$$\eta = \frac{c_{\mathrm{M}}}{c_{\mathrm{T}}}, \quad \mathbf{w}_1 = [1, -a, a, \eta]^{\mathrm{T}} T_1$$

Each matrix column is one actuator's contribution to total thrust and three torques.

4.3.2 Force and Torque Synthesis: Full X-Layout Equation

Read Each Column as a Motor and Each Row as an Effect

- Let $\mathbf{w} = [T_\Sigma, \tau_x, \tau_y, \tau_z]^{\mathrm{T}}$, with upward thrust magnitude first. $\mathbf{t}$ contains the four thrusts; $\eta = c_{\mathrm{M}}/c_{\mathrm{T}}$.
- Each extra 1 N from front-right rotor 1 contributes $[1, -a, a, \eta]^T$. Arrange all four position and rotation contributions as columns below; sum each row for the resultant.

$$\mathbf{w} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -a & -a & a & a \\ a & -a & -a & a \\ \eta & -\eta & \eta & -\eta \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix}, \quad a = \frac{d}{\sqrt{2}}$$

Geometry and rotation directions set the forward matrix; squared speeds weight its columns.

4.3.2 Force and Torque Synthesis: Forward Examples

Three Input Patterns Check Four-Rotor Synthesis

- Teaching values: $d = 0.25$ m, $\eta = 0.02$ m. 4.9 N per rotor gives 19.6 N total thrust and zero torque on all three axes.
- Rotor 1 only, +1 N: total thrust +1 N; torque increment $[-0.1768, 0.1768, 0.02]^T$ N·m.
- Diagonal rotors 1 and 3: +1 N each; 2 and 4: −1 N each. Only yaw torque changes, to 0.08 N·m. Cancellation must be in thrust or squared speed; equal speed changes need not cancel.

Symmetric

4.9 4.9 4.9 4.9

All three torques zero

Increase rotor 1 only

5.9 4.9 4.9 4.9

Roll − / pitch + / yaw +

Opposing diagonal changes

5.9 3.9 5.9 3.9

Yaw only: +0.08 N·m

Symmetry; single-rotor and yaw increments

Symmetric thrust checks cancellation; a single-rotor increment checks an entire column's signs.

4.3.2 Force and Torque Synthesis: Inverse Map and Feasibility

X-Layout Inverse Map: Resultant Force and Torques to Thrusts

- Use textbook numbering, $a = d/\sqrt{2}$ and $\eta = c_{\mathrm{M}}/c_{\mathrm{T}}$. Compute each thrust, then check nonnegative and upper bounds.

$$T_1 = \frac{1}{4}\left(T_\Sigma - \frac{\tau_x}{a} + \frac{\tau_y}{a} + \frac{\tau_z}{\eta}\right), \quad T_2 = \frac{1}{4}\left(T_\Sigma - \frac{\tau_x}{a} - \frac{\tau_y}{a} - \frac{\tau_z}{\eta}\right)$$

$$T_3 = \frac{1}{4}\left(T_\Sigma + \frac{\tau_x}{a} - \frac{\tau_y}{a} + \frac{\tau_z}{\eta}\right), \quad T_4 = \frac{1}{4}\left(T_\Sigma + \frac{\tau_x}{a} + \frac{\tau_y}{a} - \frac{\tau_z}{\eta}\right)$$

- **Check:** Substitute all four thrusts into the forward matrix on the synthesis-matrix page. They should recover the specified total thrust and three torques.

An algebraic solution gives mathematical balance; real propellers also have nonnegative and upper limits.

4.3.2 Force and Torque Synthesis: Inverse Map and Feasibility

Check Whether Each Requested Rotor Thrust Is Achievable

- After applying the X-layout inverse map, require $0 \le T_i \le T_{i,\max}$. An algebraic solution must also be achievable by the actuators.
- Example: Request yaw only, with $T_\Sigma = 19.6\,\mathrm{N}$, $\tau_z = 0.48\,\mathrm{N \cdot m}$, $\eta = 0.02\,\mathrm{m}$, and zero other torques. The yaw term adds or subtracts $\tau_z/(4\eta) = 0.48/0.08 = 6\,\mathrm{N}$ per rotor.

$$T_1 = T_3 = \frac{T_\Sigma}{4} + \frac{\tau_z}{4\eta} = 4.9 + 6 = 10.9\,\mathrm{N}$$

$$T_2 = T_4 = \frac{T_\Sigma}{4} - \frac{\tau_z}{4\eta} = 4.9 - 6 = -1.1\,\mathrm{N}$$

Check: The other pair requires $-1.1\,\mathrm{N} < 0$, impossible for unidirectional rotors.

4.3.2 Force and Torque Synthesis: Actual Model Numbering

Map M0 Motor Order to the Textbook Order

- M0 angles are 45°, 225°, 315°, and 135°. Template motors 1–4 are front-right, rear-left, front-left, and rear-right.
- By physical position, textbook rotors 1, 2, 3, 4 connect to M0 motors 1, 4, 2, 3. An X layout alone does not justify copying input indices.
- The template rotor-direction array is $[-1,-1,+1,+1]$. Body reaction torque has the opposite sign: M0 motors 1 and 2 give positive yaw; 3 and 4 give negative yaw.



Same position, different input index

$1 \rightarrow 1,\ 2 \rightarrow 4,\ 3 \rightarrow 2,\ 4 \rightarrow 3$

Figure: Two numbering schemes, same layout

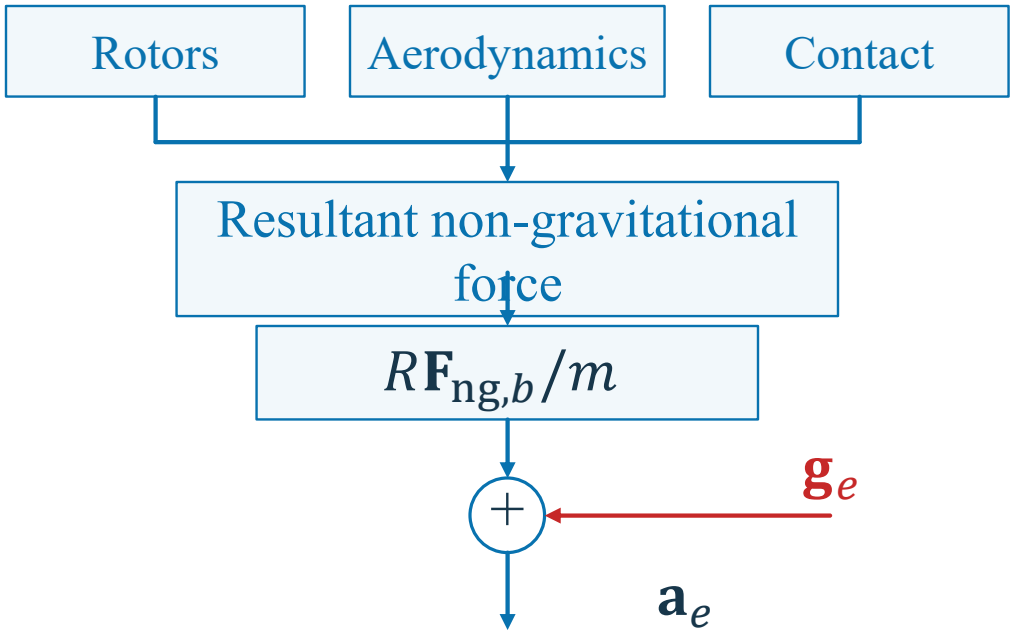
Align motors by physical position and rotation direction before comparing matrices.

4.3.2 Force and Torque Synthesis: Submodel Integration

Return Resultant Forces and Rotational Effects to Dynamics

- Express propeller thrusts, aerodynamic forces, and contact forces in body components. Sum them at the center of mass and pass them to the rigid-body equations in Section 4.2.
- If a block already receives total force with gravity, do not add gravity again outside. Rotor gyroscopic effects add torque from rotor angular momentum as the body turns; count this separately from body inertial coupling.

$$\mathbf{a}_e = \mathbf{g}_e + R\mathbf{F}_{\mathrm{ng},b}/m$$



Sum non-gravitational inputs; add gravity once

Count gravity once; treat rotor gyroscopic effects separately from rigid-body inertial coupling.

Chapter Outline

Propulsion, Forces, and Observation Modeling

Chapter 4
Vehicle Motion Modeling and Simulation

4.1 Modeling Fundamentals and Kinematics
4.2 Rigid-Body Dynamics and Solvers
4.3 Propulsion, Forces, and Observations
4.4 Trim, Linearization, and Dynamics
4.5 Multicopter Modeling and Simulation
4.6 Measurement and Static Identification
4.7 Dynamic Identification and Refinement
4.8 Typical Vehicles and Extensions
4.9 Modeling Tools and Deployment
4.10 API and SDK Modeling Interfaces
4.11 Examples and Integrated Practice
4.12 Exercises and Course Projects

5 Units in This Section

4.3.1 Actuator Subsystem Modeling

Textbook: Sec. 4.1.4.3; Eqs. (4.29–30) in Sec. 4.1.4.4

4.3.2 Force and Torque Synthesis

Textbook: Sec. 4.1.4.4, Force and Torque Synthesis Subsystem

4.3.3 Environment Subsystem Modeling

Textbook: Secs. 4.1.4.2 and 4.1.4.4

4.3.4 Terrain and Contact

Textbook: Secs. 4.1.4.2 and 4.1.4.4

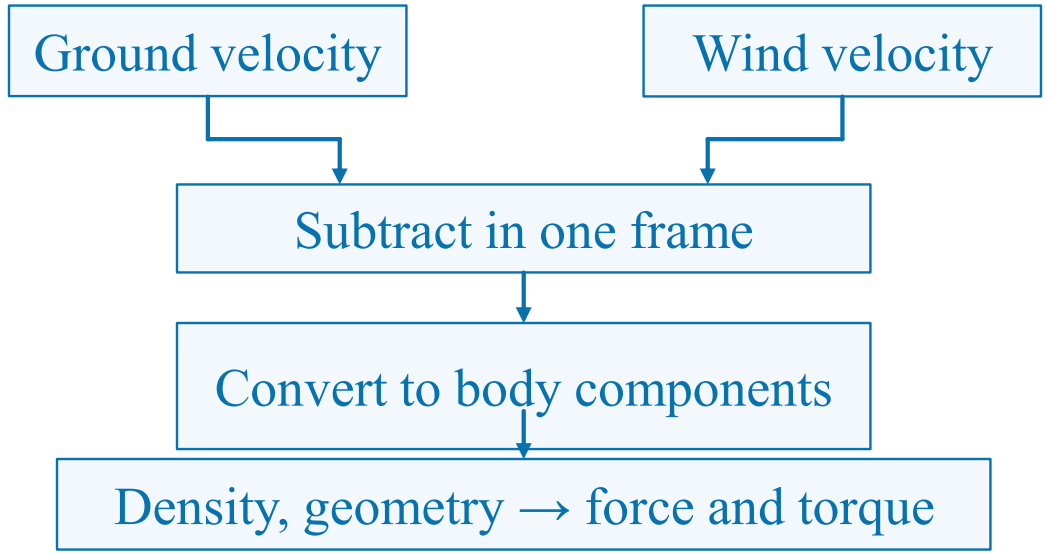
4.3.5 Sensor Subsystem Modeling

Textbook: Secs. 4.1.4.6 and 4.1.4.7, Sensors and 3D Environment

4.3.3 Environment Subsystem Modeling: Environmental Inputs

The Environment Changes Relative Airflow, Then Forces

- Wind: air motion over the ground. Subtract ground and wind velocities in NED; multiply by R^T for FRD air-relative velocity. A meteorological north wind moves south.
- Position queries density, gravity, terrain; velocity and attitude set force. Environment alters aerodynamics and contact; dynamics changes motion. The introductory ideal model uses zero aerodynamics for analytical checks.



Use consistent frames before computing force

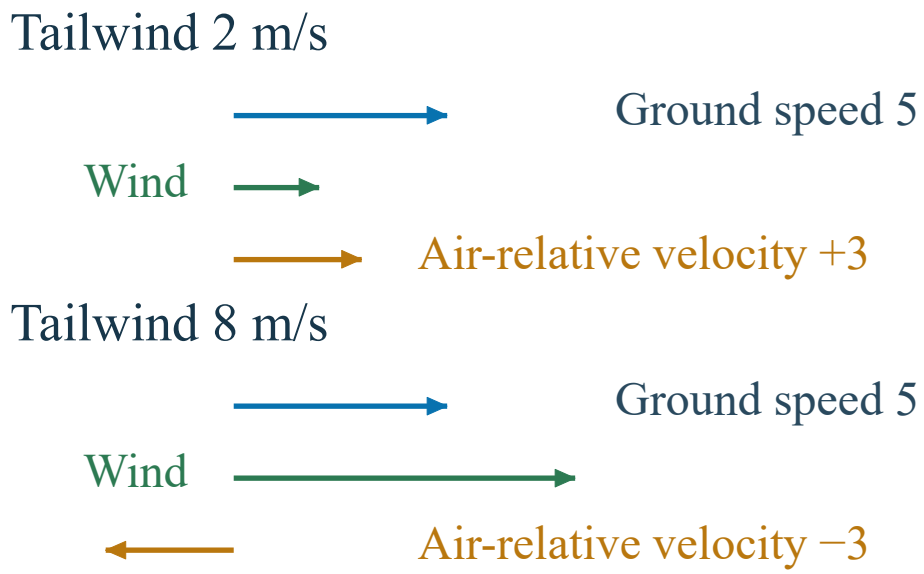
$$\mathbf{v}_{\mathrm{rel},b} = R^T(\mathbf{v}_e - \mathbf{v}_{\mathrm{wind},e})$$

Relative airflow, aerodynamic parameters, and external forces change the motion response.

4.3.3 Environment Subsystem Modeling: Relative-Flow Example

Tailwind Can Reverse Drag While the Aircraft Still Flies North

- Aircraft: 5 m/s north; air: 2 m/s north. Northward air-relative velocity is 3 m/s. With $F_N = -kv_N|v_N|$, drag points south.
- If air moves north at 8 m/s, air-relative velocity becomes −3 m/s and drag points north. Faster northbound air now pushes the aircraft.
- With $k = 0.055$ N/(m/s)^2, the forces are −0.495 N and +0.495 N. This local hand calculation only borrows M0's drag coefficient.



Tailwind magnitude changes relative motion

Drag opposes motion relative to the air; it need not oppose ground velocity.

4.3.3 Environment Subsystem Modeling: Airflow Angles

Airspeed, Angle of Attack, and Sideslip Have Different Meanings

- Air-relative velocity $[u_a, v_a, w_a]^T$ gives speed and direction. α: longitudinal-plane angle; β: angle out of that plane. These are not body attitude angles.
- $[10,1,2]^T$ m/s → 10.25 m/s, 11.31°, 5.60°. atan2 preserves the quadrant; bound the arcsine argument. Zero airspeed: direction undefined. Near zero: use velocity components directly.

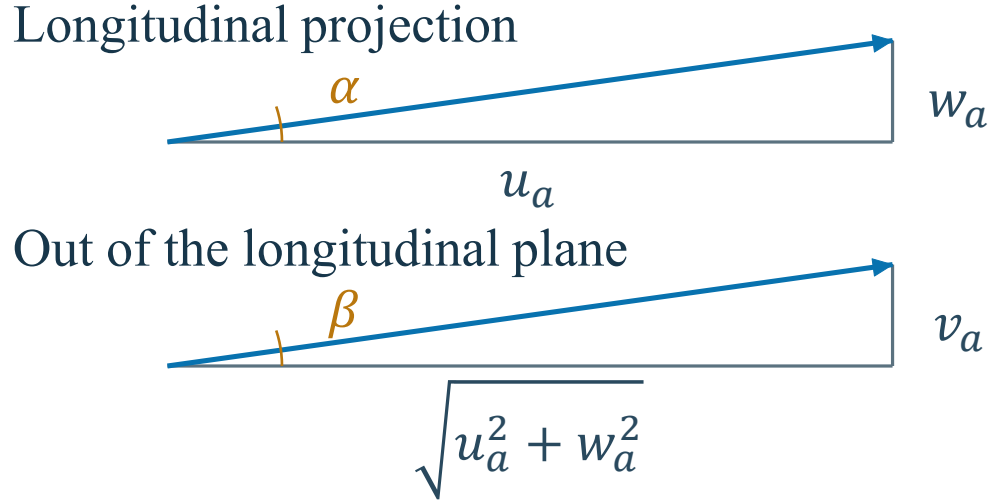


Figure: Longitudinal flow angle and sideslip

$$V_a = \sqrt{u_a^2 + v_a^2 + w_a^2}, \quad \alpha = \text{atan2}(w_a, u_a), \quad \beta = \arcsin(v_a / V_a)$$

Flow angles relate velocity direction to body axes; they are undefined at zero airspeed.

4.3.3 Environment Subsystem Modeling: Drag Direction and Model

Quadratic Drag Opposes Relative Velocity and Flattens at Zero

- 1D model $F_D = -kv|v|$ keeps squared magnitude and velocity sign: positive velocity → negative force; negative → positive; zero → zero.
- Zero slope at the origin means no first-order quadratic-drag term near rest in still air. This does not hold with wind or at moving operating points.
- 3D: $-\mathbf{k} \odot \mathbf{v} \odot |\mathbf{v}|$ uses componentwise multiplication ($\odot$). It differs from $-k \parallel \mathbf{v} \parallel \mathbf{v}$ and linear drag. Match coefficients to the model.

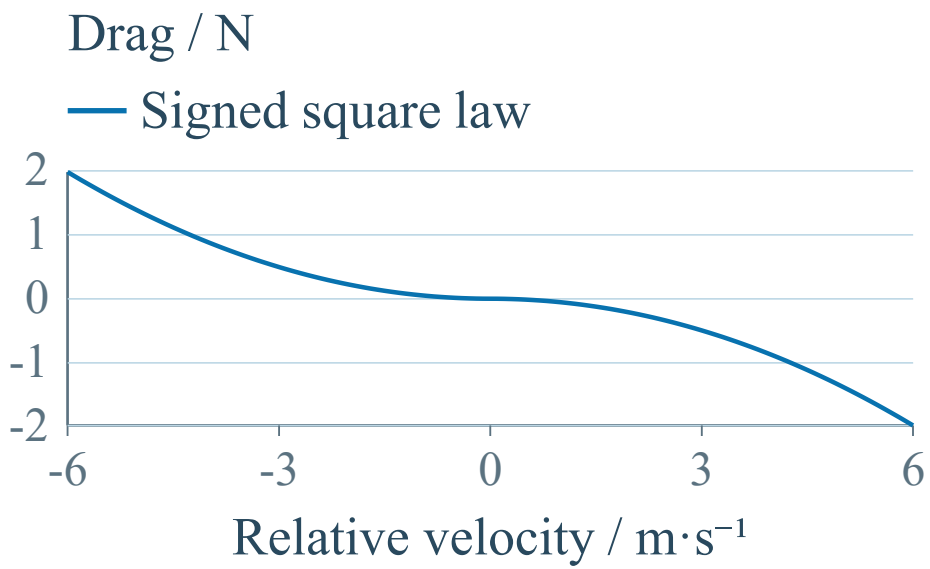


Figure: Drag reverses with relative velocity

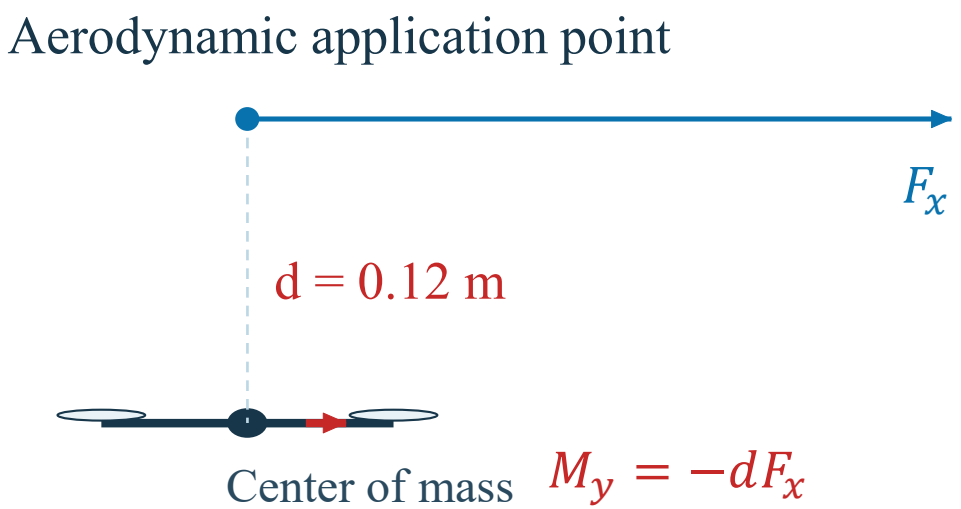
Drag reverses when velocity reverses, and this drag term vanishes at zero velocity.

4.3.3 Environment Subsystem Modeling: Actual Aerodynamic Function

How M0 Computes Body Drag and Rotational Damping

- M0 translational drag: $\mathbf{F}_D = -C_d \mathbf{v} \odot |\mathbf{v}|$, with dimensional coefficient $C_d = 0.055\ \mathrm{N} \cdot \mathrm{s}^2/\mathrm{m}^2$.

- Rotational damping opposes angular rate: 1 rad/s roll gives −0.0035 N·m. An effective application point 0.12 m above the center of mass also creates torque $[dF_y, -dF_x, 0]^T$.



Offset drag application point also gives torque

$$\mathbf{M}_{\text{damp}} = -\mathbf{C}_\omega \odot \boldsymbol{\omega}_b \odot |\boldsymbol{\omega}_b|, \quad \mathbf{C}_\omega = [0.0035, 0.0039, 0.0034]^T$$

Drag opposes air-relative velocity; rotational damping opposes angular rate.

4.3.3 Environment Subsystem Modeling: Atmosphere and Dynamic Pressure

Pressure × Area Gives Force; Add Length for Torque

- Dynamic pressure $q_a = \rho V_a^2/2$ depends on density and airspeed squared, in Pa. With $\rho = 1.2$ kg/m³ and $V_a = 10$ m/s, it is 60 Pa.
- Lift, drag, and side-force coefficients C_L, C_D, C_Y multiply $q_a S$. Moment coefficients C_l, C_m, C_n also require span b or chord c, using their defined reference quantities.
- With $S = 0.4$ m² and $C_L = 0.5, C_D = 0.1$, lift is 12 N and drag 2.4 N. Coefficients vary with flow and control-surface angles. The International Standard Atmosphere (ISA) is a reference; actual weather can differ.

$$L = q_a S C_L, \quad D = q_a S C_D, \quad Y = q_a S C_Y$$

$$M_x = q_a S b C_l, \quad M_y = q_a S c C_m, \quad M_z = q_a S b C_n$$

Density and airspeed squared set dynamic pressure; coefficients describe shape and state effects.

4.3.3 Environment Subsystem Modeling: Coefficients and Units

Equal Coefficients Can Give Different Roll and Pitch Moments

- **Teaching example:** $q_a = 60\,\text{Pa}$, $S = 0.4\,\text{m}^2$, span $b = 1.0\,\text{m}$, mean aerodynamic chord $c = 0.2\,\text{m}$, and $C_l = C_m = 0.1$.
- Roll moment uses span b; pitch moment uses chord c. With equal dynamic pressure, area, and coefficients, a fivefold reference-length difference gives a fivefold moment difference.
- Here C_l is the roll-moment coefficient, not lift coefficient C_L. Coefficients must match their reference area and length; coefficient values alone cannot compare moments.

$$M_x = q_a S b C_l = 60 \times 0.4 \times 1.0 \times 0.1 = 2.4\,\text{N}\cdot\text{m}$$

$$M_y = q_a S c C_m = 60 \times 0.4 \times 0.2 \times 0.1 = 0.48\,\text{N}\cdot\text{m}$$

Aerodynamic force uses dynamic pressure × area; aerodynamic torque also needs a reference length.

4.3.3 Environment Subsystem Modeling: Wind Axes to Body Axes

Transform Lift and Drag Back into the Body Frame

- Wind-frame forward follows air-relative velocity; upward lift and backward drag give $[-D, Y, -L]^T$. Transform to body axes.
- At zero sideslip, rotate longitudinal components by angle of attack. At zero attack angle, check $F_x = -D$ and $F_z = -L$.
- For $\alpha = 10°$, $D = 2.4$ N, and $L = 12$ N, obtain $F_x \approx -0.28$ N and $F_z \approx -12.23$ N.

$$F_x = -D\cos\alpha + L\sin\alpha$$

$$F_z = -D\sin\alpha - L\cos\alpha \quad (\beta = 0)$$

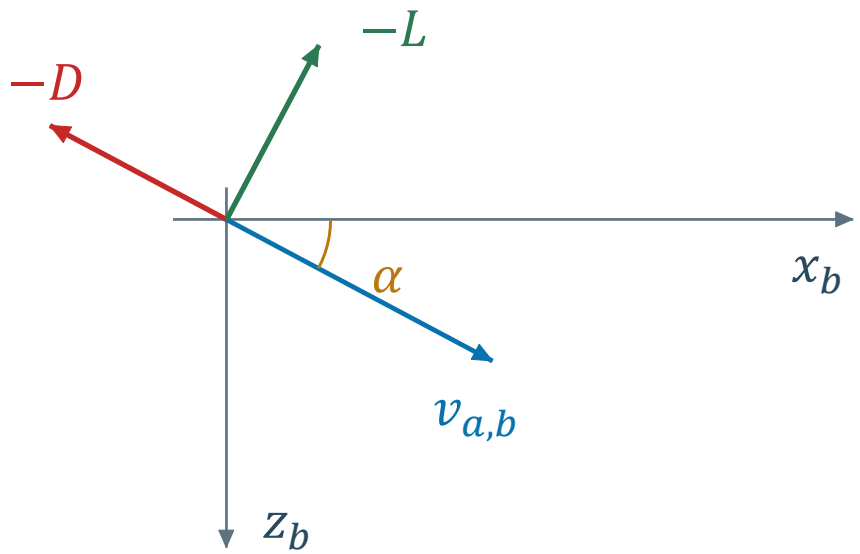


Figure: One aerodynamic force in two frames

Airflow defines lift and drag directions; express both in body components before rigid-body dynamics.

4.3.3 Environment Subsystem Modeling: 3D Equation Reference

The Full Airflow Transform Includes Attack and Sideslip Angles

- Let $C_{b\leftarrow w}$ map wind to body axes, distinct from body-to-Earth matrix R in Section 4.1. Use the same flow-angle definitions for α, β.
- First compute $C_{b\leftarrow w}[-D, Y, -L]^T$; multiply by R only if Earth components are needed. At $\alpha = \beta = 0$, this matrix must reduce to the identity.

$$C_{b\leftarrow w} = \begin{bmatrix} \cos\alpha\cos\beta & -\cos\alpha\sin\beta & -\sin\alpha \\ \sin\beta & \cos\beta & 0 \\ \sin\alpha\cos\beta & -\sin\alpha\sin\beta & \cos\alpha \end{bmatrix}$$

$$\mathbf{F}_{\text{aero,b}} = C_{b\leftarrow w}[-D, Y, -L]^{\mathrm{T}}$$

The full aerodynamic map takes wind-axis components to body-axis components.

4.3.3 Environment Subsystem Modeling: Wind-Field Models

Constant Wind, Gust, Shear, and Turbulence Describe Different Changes

- Constant wind is fixed; a gust changes over a finite time; shear varies with spatial position; turbulence has specified random statistics.
- Dryden shaping filters generate temporally correlated wind. Intensity, length scale, and power spectrum describe amplitude, variation scale, and frequency distribution.
- Before superposition, align frames and units; save parameters and seeds. Frozen turbulence (carried by the mean wind) requires sufficient airspeed; justify its use in hover separately.

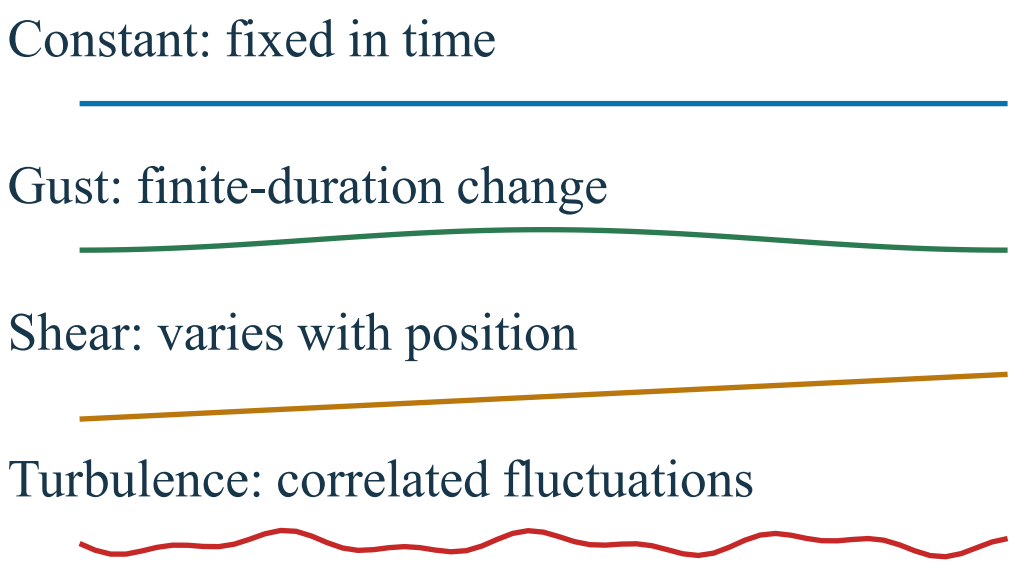


Figure: Time, space, and randomness are different dimensions

Time variation, spatial variation, and randomness are different dimensions of a wind model.

4.3.3 Environment Subsystem Modeling: Atmosphere and Wind Data

Altitude Changes Density; Measurements Can Reconstruct Wind

- ISA gives temperature and density by altitude: each 1000 m rise lowers temperature by about 6.5 °C and air density by about 10%–12%.
- At 6000 m, density roughly halves. For the same propeller and speed with nearly constant coefficients, thrust roughly halves too. Check vehicle-specific takeoff/landing thrust margin and suitable propellers.
- Data-driven wind fields: Use sensors across the target area for long-term wind speed/direction records. Reconstruct representative disturbances by statistics, curve fitting, or basis expansions.

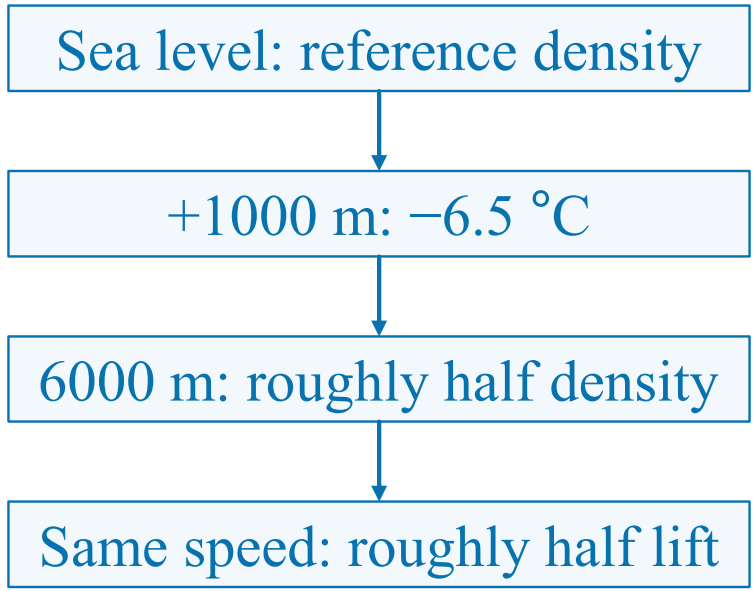


Figure: Altitude effects on density and lift in ISA

Altitude changes propulsion and aerodynamics through density; wind fields can use physical models or data.

4.3.3 Environment Subsystem Modeling: Practice and Acceptance

Check Wind from Air-Relative Velocity Through to Force

- Save an M0 practice copy. Hold attitude and ground velocity fixed; apply no wind, 2 m/s tailwind, then 2 m/s headwind. Record air-relative velocity, drag, and Earth components.
- M0 injects WindDist by subtracting transformed wind from body velocity. Check intermediate values before releasing motion integrators to inspect trajectories.
- **Criterion:** With R=I, northward ground speed 5 m/s, and Cd=0.055, northward air-relative speeds are 5, 3, 7 m/s; drag forces are −1.375, −0.495, −2.695 N.

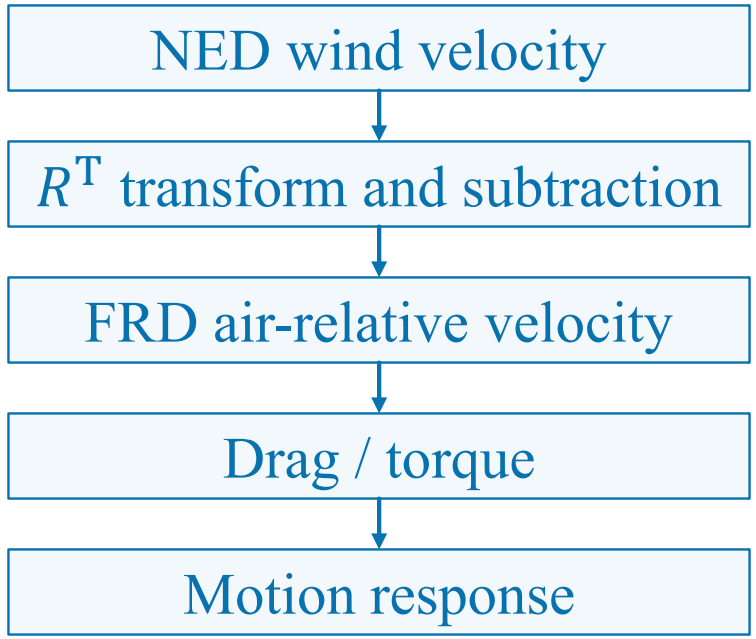


Figure: Traceable wind-disturbance injection path

Wind becomes a dynamics disturbance only when connected to relative airflow and force calculations.

Chapter Outline

Propulsion, Forces, and Observation Modeling

Chapter 4
Vehicle Motion Modeling and Simulation
4.1 Modeling Fundamentals and Kinematics
4.2 Rigid-Body Dynamics and Solvers
4.3 Propulsion, Forces, and Observations
4.4 Trim, Linearization, and Dynamics
4.5 Multicopter Modeling and Simulation
4.6 Measurement and Static Identification
4.7 Dynamic Identification and Refinement
4.8 Typical Vehicles and Extensions
4.9 Modeling Tools and Deployment
4.10 API and SDK Modeling Interfaces
4.11 Examples and Integrated Practice
4.12 Exercises and Course Projects

5 Units in This Section

4.3.1 Actuator Subsystem Modeling

Textbook: Sec. 4.1.4.3; Eqs. (4.29–30) in Sec. 4.1.4.4

4.3.2 Force and Torque Synthesis

Textbook: Sec. 4.1.4.4, Force and Torque Synthesis Subsystem

4.3.3 Environment Subsystem Modeling

Textbook: Secs. 4.1.4.2 and 4.1.4.4

4.3.4 Terrain and Contact

Textbook: Secs. 4.1.4.2 and 4.1.4.4

4.3.5 Sensor Subsystem Modeling

Textbook: Secs. 4.1.4.6 and 4.1.4.7, Sensors and 3D Environment

4.3.4 Terrain and Contact: Contact Geometry

Locate the Lowest Contact Point, Then Find Penetration

- Locate a contact point by $\mathbf{p}_{P,e} = \mathbf{p}_{C,e} + R\boldsymbol{\rho}_{P,b}$. At equal center-of-mass height, a tilted side may touch first. The scene supplies geometry; dynamics computes support.
- NED is positive downward. On flat ground $z_G = 0$, point heights −0.02, 0, +0.01 m mean 2 cm clearance, contact, and 1 cm penetration. A penalty model treats small penetration as elastic compression.

$$\delta = \max(0, z_P - z_G)$$

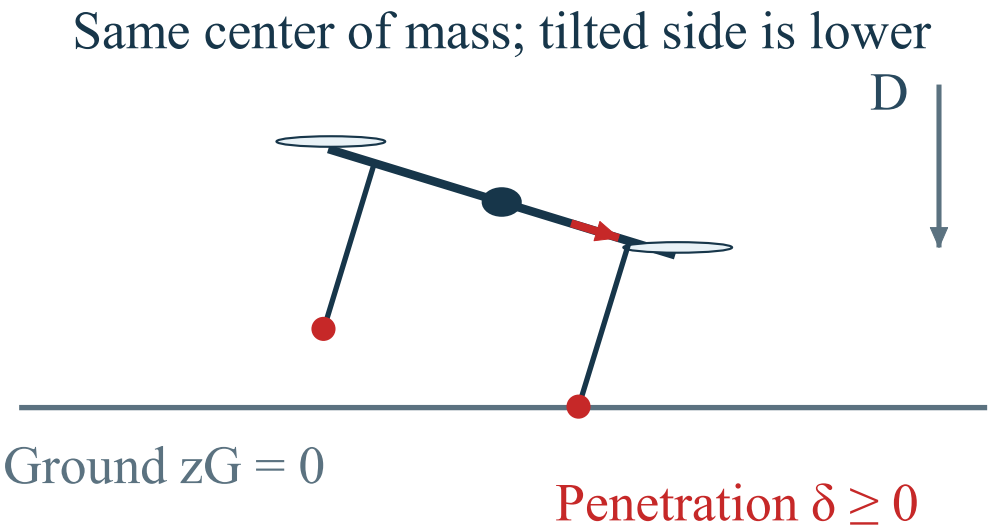


Figure: The lowest point sets contact; positive penetration represents compression

Center-of-mass height alone cannot determine contact; attitude and landing-gear geometry also matter.

4.3.4 Terrain and Contact: Unilateral Spring and Damper

The Ground Pushes, Never Pulls: Compression and Support

- During contact, $N = \max(0, k\delta + c\dot{\delta})$ gives flat-ground support $[0,0,-N]^T$. After separation it is zero; clipping prevents an adhesive downward pull during rebound.
- Take $m = 2$ kg, $k = 1000$ N/m, and $c = 20$ N·s/m. At 0.01 m penetration and 0.2 m/s downward velocity, support is only 14 N, below the 19.6 N weight.
- Downward acceleration is still 2.8 m/s². Static compression is 0.0196 m. At the original compression with 1 m/s upward rebound, clip negative support to zero.

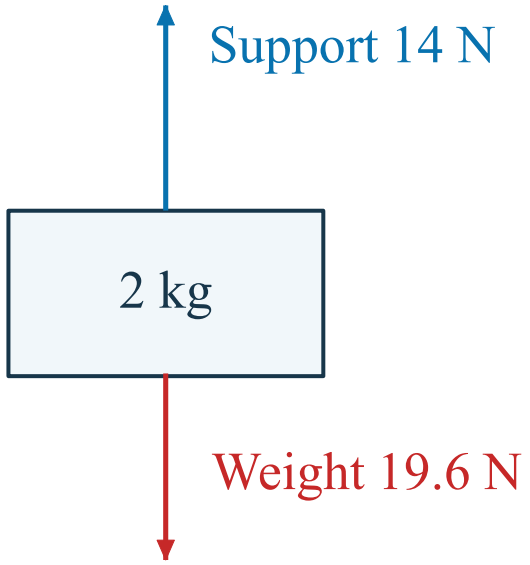


Figure: Contact does not imply force balance

Support resists compression and landing velocity; release it after separation.

4.3.4 Terrain and Contact: Contact-Force Example

How to Handle Negative Candidate Support During Rebound

- Use $m = 2\,\text{kg}$, $k = 1000\ \text{N/m}$, $c = 20\,\text{N}\cdot\text{s/m}$. At penetration $\delta = 0.01\,\text{m}$, upward rebound at 1 m/s gives $\dot{\delta} = -1\ \text{m/s}$.
- Spring force is 10 N and damping force −20 N, giving candidate support −10 N. Unilateral ground can push but cannot pull, so clip this negative value to zero.
- With no other non-gravitational vertical force, downward acceleration is $a_D = 9.8\ \text{m/s}^2$. Velocity can still be upward while its magnitude decreases; velocity and acceleration need not share a direction.

$$N = \max(0,\ 1000 \times 0.01 + 20 \times (-1)) = 0\,\text{N}$$

Contact and resultant-force balance are separate decisions.

4.3.4 Terrain and Contact: Friction and Contact Torque

Tangential Contact Friction Affects Translation and Rotation

- Sliding friction has magnitude μN opposite slip. Static friction balances external force up to its limit: for N=19.6 N and μ=0.4, the limit is 7.84 N.
- After transforming contact force to the body frame, compute $\boldsymbol{\rho}_P \times \mathbf{F}_{P,b}$. For forward slip, backward friction below the center of mass produces a nose-down moment.

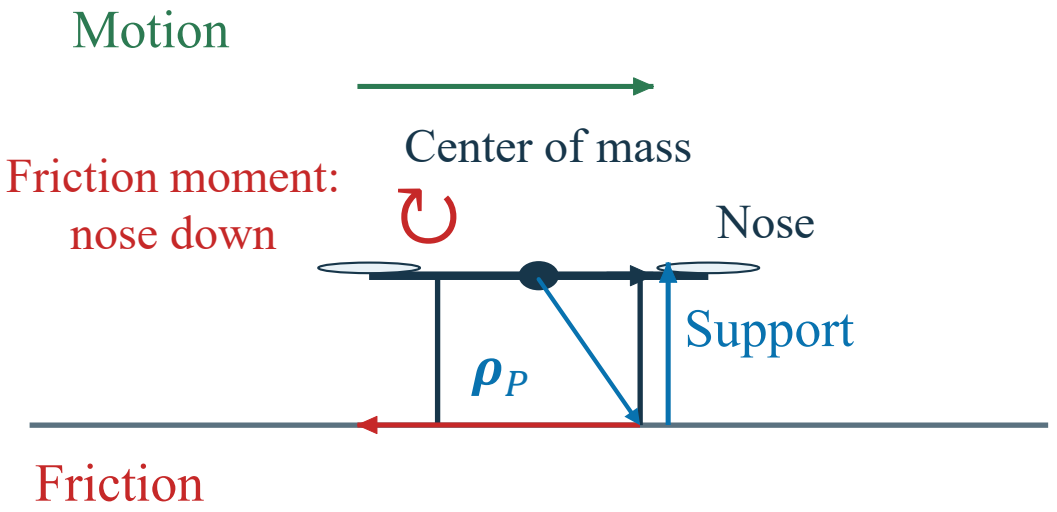


Figure: Contact moments about the center of mass

$$\mathbf{F}_t = -\mu N \mathbf{v}_t / \| \mathbf{v}_t \| \ (\| \mathbf{v}_t \| > 0), \quad \| \mathbf{F}_{t,\text{static}} \| \leq \mu N$$

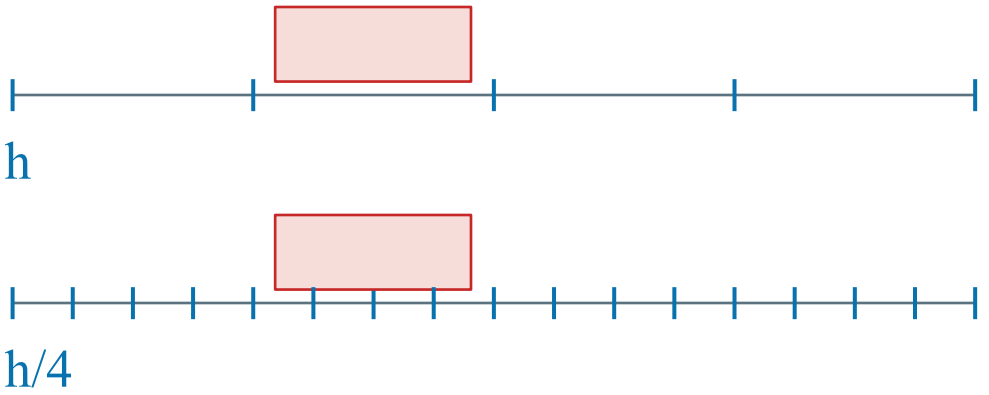
Friction opposes relative tangential slip and produces torque when applied away from the center of mass.

4.3.4 Terrain and Contact: Contact Timescale

Stiffer Contact Requires a Smaller Solver Timestep

- Spring-mass timescale: about $\sqrt{m/k}$. At fixed mass, quadrupling stiffness roughly halves response time; coarse steps are likelier to skip contact.
- With $m = 2$ kg and $k = 1000$ N/m, this scale is about 0.0447 s. An excessive step allows deep penetration before an excessive restoring force, causing false rebound or divergence.
- Hold physical parameters fixed. Compare contact time, maximum penetration, and peak support at h and $h/2$. Interpret material stiffness only after numerical results stabilize.

One contact process, two time grids



Narrow red region: rapid contact changes

Figure: Red region: none at h, three at h/4
h/4 shows sampling; h and h/2 test convergence

Greater contact stiffness shortens the response; a coarse step can skip the whole collision.

4.3.4 Terrain and Contact: Actual Ground-Model Limits

Test M0 Ground Constraints and Ideal Free Fall Separately

- M0 uses the first TerrainIn elevation and body $z - z_G$ for restoring force/torque and horizontal damping. It has no per-foot contact and includes gravity cancellation.
- M0 supports whole-body flat-ground behavior. Model unilateral material and sloped-surface contact separately. Test airborne, penetrated, and initial landing states.
- Ideal free fall: contact off, motors confirmed stopped, aerodynamics zero. Initialize each run; save identical terrain, initial state, and timestep.

Ground test	Ideal free fall
Contact enabled	Contact disabled
Record support and torque	Motors stopped
Initialize ground state	Zero aerodynamic force

Figure: The two tests retain different forces

The minimal template uses elevation and body state for ground constraints, introducing interfaces and effects.

4.3.4 Terrain and Contact: Terrain Data and Obstacles

Local Terrain Files Give Heights; Trajectories Describe Obstacles

- 3D-scene terrain streaming: high data volume, delay, packet loss. CopterSim defaults to a terrain server with local files stored in advance.
- Terrain uses a regular grid with Earth-frame nodes. RflySim stores Z elevation in .png and XY bounds, spacing, and origin latitude, longitude, and altitude in .txt.
- Obstacle positions vary with time in Earth axes; a constant trajectory is static. RflySim can create obstacles and update them in real time through Python interfaces.

Terrain server	Online queries
.png: Z elevation	From the 3D scene
.txt: bounds and origin	Large data volume
Offline, stable, low overhead	Delay and packet loss

Figure: Two ways to obtain terrain data

Offline grid files provide terrain height; obstacle position trajectories enter the same environment model.

Chapter Outline

Propulsion, Forces, and Observation Modeling

Chapter 4
Vehicle Motion Modeling and Simulation

4.1 Modeling Fundamentals and Kinematics
4.2 Rigid-Body Dynamics and Solvers
4.3 Propulsion, Forces, and Observations
4.4 Trim, Linearization, and Dynamics
4.5 Multicopter Modeling and Simulation
4.6 Measurement and Static Identification
4.7 Dynamic Identification and Refinement
4.8 Typical Vehicles and Extensions
4.9 Modeling Tools and Deployment
4.10 API and SDK Modeling Interfaces
4.11 Examples and Integrated Practice
4.12 Exercises and Course Projects

5 Units in This Section

4.3.1 Actuator Subsystem Modeling

Textbook: Sec. 4.1.4.3; Eqs. (4.29–30) in Sec. 4.1.4.4

4.3.2 Force and Torque Synthesis

Textbook: Sec. 4.1.4.4, Force and Torque Synthesis Subsystem

4.3.3 Environment Subsystem Modeling

Textbook: Secs. 4.1.4.2 and 4.1.4.4

4.3.4 Terrain and Contact

Textbook: Secs. 4.1.4.2 and 4.1.4.4

4.3.5 Sensor Subsystem Modeling

Textbook: Secs. 4.1.4.6 and 4.1.4.7, Sensors and 3D Environment

4.3.5 Sensor Subsystem Modeling: Overall Motion Framework

Motion Ground Truth, Environment Data, and Observations Have Sources

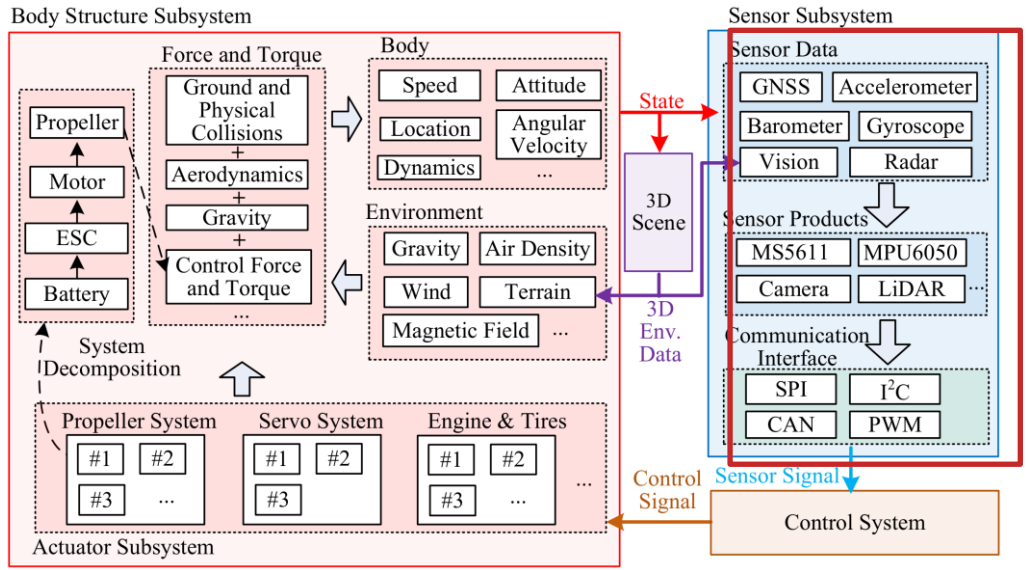


Figure 4.7 Overall Vehicle Motion Model Framework for Unmanned Systems

Ground truth ≠ noisy measurement ≠ estimate

- **Motion and environment:** Forces/torques update body motion ground truth. Pose sets the 3D scene for vision, terrain contact, and spatial interaction.
- **Observation sources:** Sensors use ground truth or the scene, then add mounting, sampling, and errors. True position, noisy measurement, and filtered estimate are different variables.
- **Read the loop:** Observations enter the control system; commands drive the body. Ground truth helps debugging; connect controllers through the agreed observation interface.

Compute physical motion first, then observations to close the feedback-control loop.

4.3.5 Sensor Subsystem Modeling: Fig. 4.7 Detail

Sensors: Ideal Data, Product Effects, and Communication

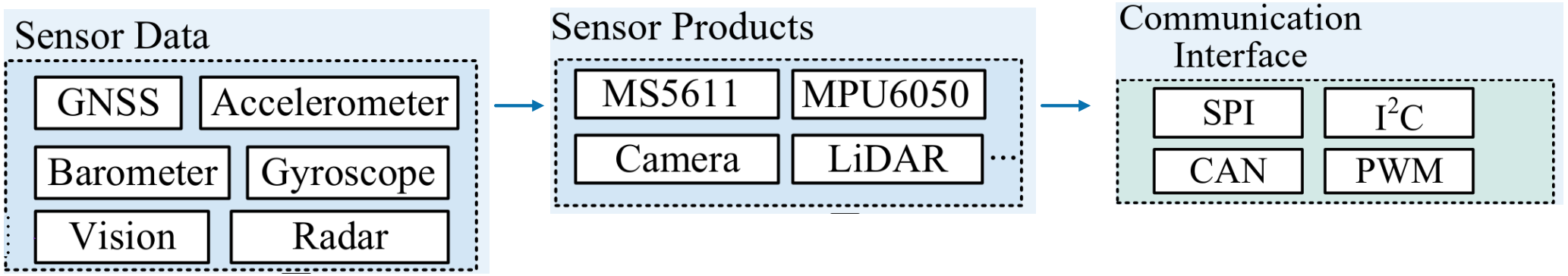


Figure 4.7 (Detail) Three Sensor-Subsystem Layers

- **Three layers:** Data: ideal physical quantities; product: noise, quantization, calibration error; interface: Serial Peripheral Interface (SPI), Inter-Integrated Circuit (I²C), Universal Asynchronous Receiver/Transmitter (UART), Controller Area Network (CAN), Universal Serial Bus (USB) formats.
- **Model depth:** In hardware-in-the-loop (HITL; HIL in the textbook), sensor data reaches the autopilot over serial MAVLink; SPI/I²C emulation is unnecessary. Protocol models are needed when a field-programmable gate array (FPGA) replaces a sensor chip.

4.3.5 Sensor Subsystem Modeling: Layers and Interface Conventions

Ground Truth Becomes Data Through Observation and Product Errors

- Rigid-body integrators store motion ground truth. Physical observation maps it to ideal measurements, then bias, noise, mounting error, and sampling are added.
- Interface conventions specify inputs, outputs, units, frames, rates, and timing between modules. M0 outputs observations through HILSensor30d, HILGPS30d, and other ports, while retaining separate ground truth.
- For each field, record quantity, frame, unit, timestamp, and period. Disable product errors first; check specific force under stationary support, ideal free fall, and known acceleration.

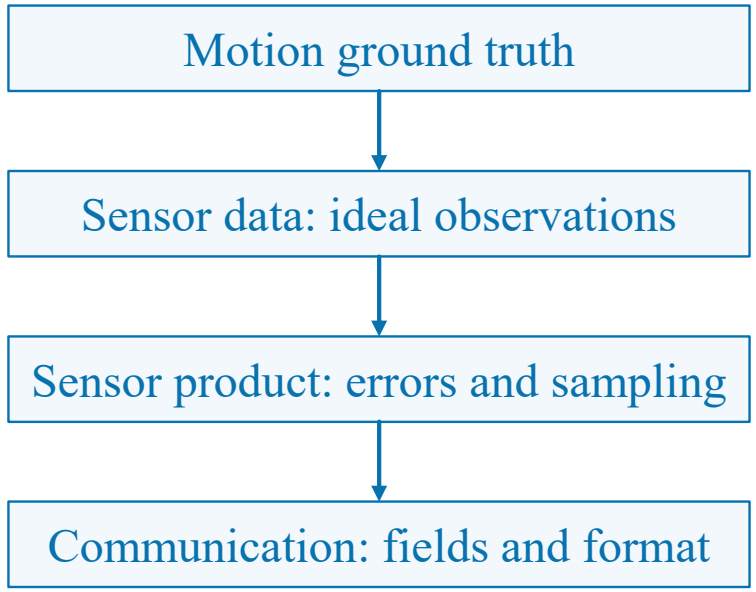


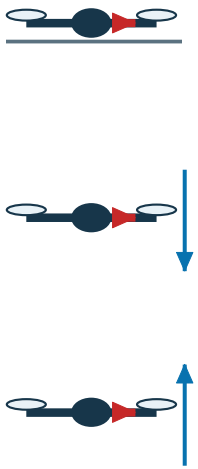
Figure: Data, product, and interface layers turn ground truth into observations

Motion ground truth, ideal measurements, and imperfect product outputs serve different roles.

4.3.5 Sensor Subsystem Modeling: Three Specific-Force Cases

Subtract Gravity from Motion Acceleration for Specific Force

- At the center of mass, theoretical specific force is $\mathbf{f}_b = R^T(\mathbf{a}_e - \mathbf{g}_e)$. Subtract gravity from motion acceleration, then transform to sensor axes.
- Here body axes align with NED and an error-free sensor is at the center of mass. Stationary motion acceleration is zero, but specific force is −9.8 m/s².
- Ideal free fall requires zero thrust, contact, and aerodynamic force: downward acceleration is 9.8, specific force zero. At upward acceleration 1, downward specific force is −10.8.



Stationary support

$$a_D = 0,\ f_D = -9.8$$

Free fall

$$a_D = 9.8,\ f_D = 0$$

Upward acceleration

$$a_D = -1,\ f_D = -10.8$$

Figure: Ideal specific force in three cases

Three known cases jointly check gravity handling, signs, and acceleration units.

4.3.5 Sensor Subsystem Modeling: Proof-Mass Mechanism

How Proof-Mass Displacement Produces a Specific-Force Reading

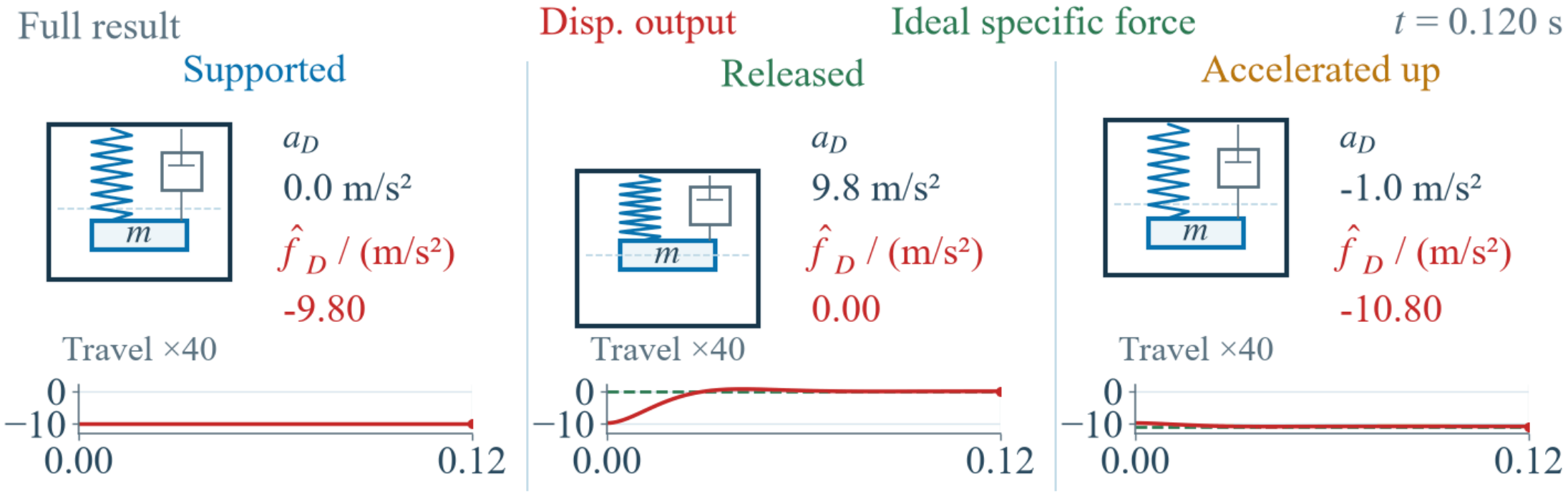


Figure: Spring-damper proof-mass transients (single-axis teaching model; internal displacement enlarged)

Three known cases jointly check gravity handling, signs, and acceleration units.

4.3.5 Sensor Subsystem Modeling: Angular Rate and Mounting

Handle Gyroscope Angular Rate in the Sensor Mounting Frame

- A gyroscope measures angular rate about sensor axes. Project with mounting matrix $C_{s\leftarrow b}$ before comparing with body angular rate; this is not an Euler-angle derivative.
- Here sensor forward points body-right, sensor right points body-rear, and down is unchanged. Rows of $C_{s\leftarrow b}$ are [0,1,0], [−1,0,0], [0,0,1]; body-forward [1,0,0] maps to [0,−1,0].

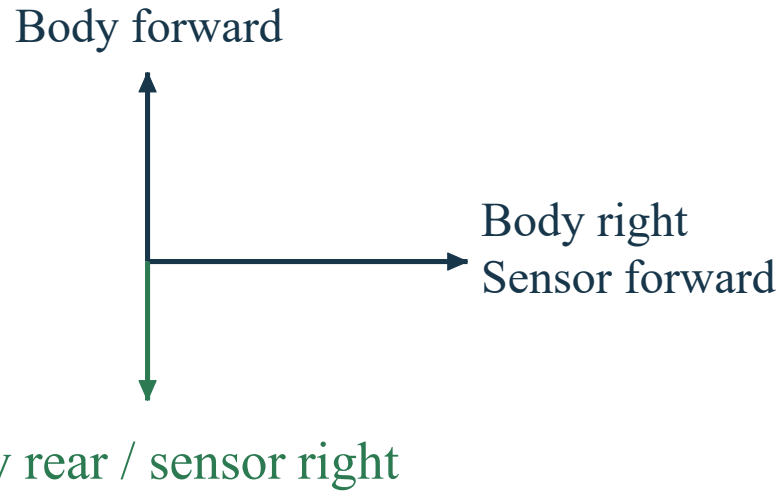


Figure: Mounting rotation differs from body attitude

$$\boldsymbol{\omega}_s = C_{s\leftarrow b}\boldsymbol{\omega}_b, \quad \mathbf{f}_s = C_{s\leftarrow b}\mathbf{f}_b$$

A gyroscope measures angular rate about sensor axes; transform it before comparing with body equations.

4.3.5 Sensor Subsystem Modeling: Sensor Lever Arm

An Offset Sensor Also Feels Rotational Acceleration

- All body points share angular velocity, but an offset sensor also has tangential and centripetal acceleration. $\boldsymbol{\rho}_s$ is its mounting lever arm; do not simply copy center-of-mass specific force.
- At constant 1 rad/s about the down axis, a sensor 0.1 m forward of the center of mass has extra forward acceleration −0.1 m/s². The three basic cases use a centered sensor to exclude this lever-arm effect.

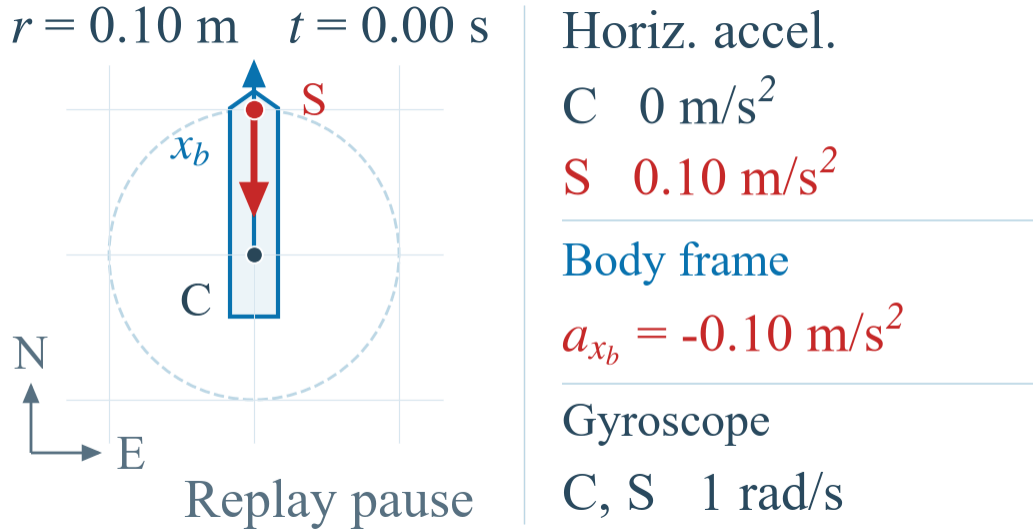


Figure: Same rotation, different point accelerations

$$\mathbf{f}_s = C_{s\leftarrow b}[\mathbf{f}_{C,b} + \dot{\boldsymbol{\omega}}_b \times \boldsymbol{\rho}_s + \boldsymbol{\omega}_b \times (\boldsymbol{\omega}_b \times \boldsymbol{\rho}_s)]$$

Rigid-body points share angular velocity but can have different linear accelerations at different locations.

4.3.5 Sensor Subsystem Modeling: GNSS and Reference Origin

One Ascent Has Different GNSS and Pressure Observations

- A global navigation satellite system (GNSS) needs a geographic origin; N and E in meters are not angles. Altitude $h = h_0 - z_D$: origin 50 m and $z_D = -2$ m give 52 m.
- **Pressure falls during ascent:** near ground, $\rho = 1.2$ kg/m³ gives about −117.6 Pa over a 10 m rise. Large height changes need a full atmosphere model. Distinguish ellipsoid, sea-level, and above-ground heights; 1 hPa=100 Pa.

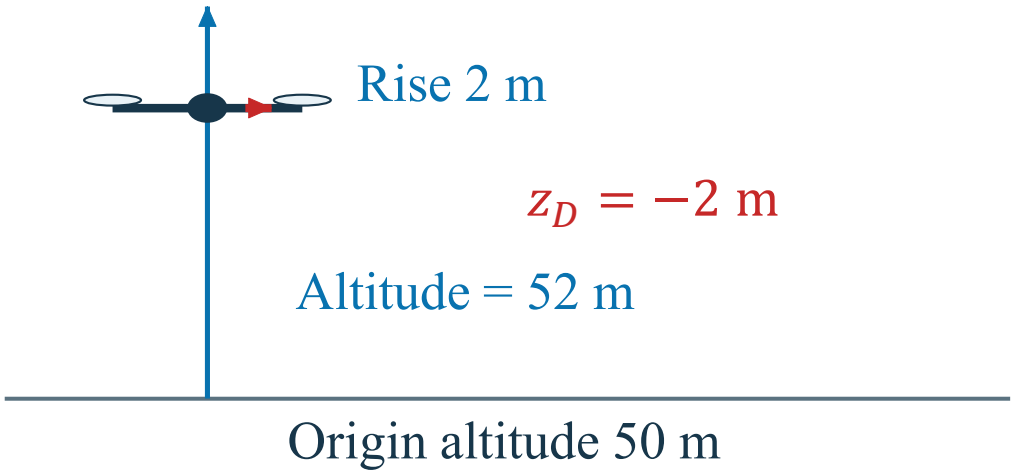


Figure: Ascent raises altitude and lowers pressure

$$\Delta p \approx -\rho g \Delta h$$

Converting NED position to latitude, longitude, and altitude requires an origin and a vertical convention.

4.3.5 Sensor Subsystem Modeling: Magnetic-Field Direction

A Magnetometer Measures Field Components in Its Own Axes

- Transform the local field by attitude and mounting. For NED field $[20,0,40]^T$ μT and aligned body/sensor axes, facing north leaves it unchanged; facing east gives $[0,-20,40]^T$ μT.
- Attitude changes axis components, not the local field. Add fields from ferromagnetic materials and currents, plus product errors, after checking ideal directions.

$$\mathbf{B}_s = C_{s\leftarrow b} R^{\mathsf{T}} \mathbf{B}_e$$

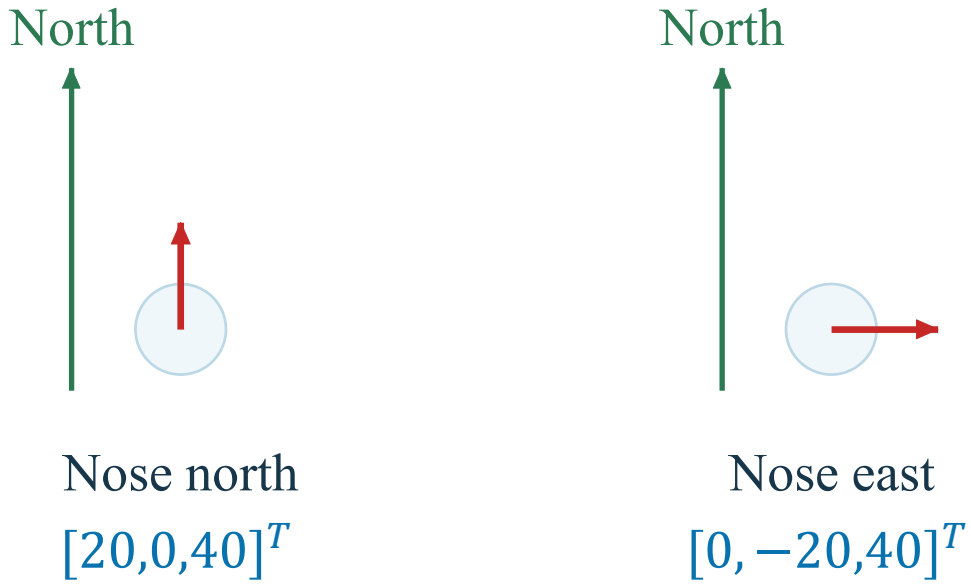


Figure: Fixed field, changing axis components

Changing attitude changes field components without changing the local geomagnetic field itself.

4.3.5 Sensor Subsystem Modeling: Geodetic and Magnetic Models

Use Global Position to Query Gravity, Density, and Magnetic Field

- Home is latitude, longitude, altitude (LLA). Use Home to convert center-of-mass NED position to LLA in World Geodetic System 1984 (WGS 84). Angles are degrees; ellipsoid height is positive upward.
- Use the WGS Earth model for local gravity and ISA for density and temperature. GNSS also combines satellite data to generate positioning output.
- The World Magnetic Model (WMM) maps LLA to field strength, inclination, and declination, supplying ideal magnetometer data and magnetic-heading correction.

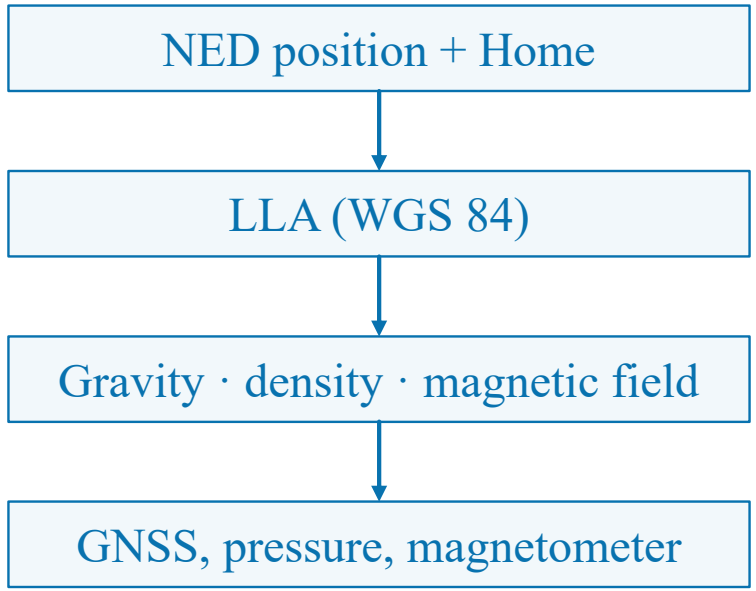


Figure: Query global environmental models from local position

Latitude, longitude, and altitude connect local motion to global gravity, atmosphere, and magnetic models.

4.3.5 Sensor Subsystem Modeling: Bias and Random Noise

Bias, Scale, and Mounting Errors Change Different Features

- Project to sensor axes first, then add scale K, bias $\mathbf{b}$, and noise $\mathbf{n}$. Bias 0.2 shifts the mean at truth 1 to 1.2; scale 1.02 turns 10 into 10.2.
- Mounting error also mixes axes; averaging cannot remove fixed bias. Errors act on observations first and may indirectly affect motion through feedback. State each term's frame and order.

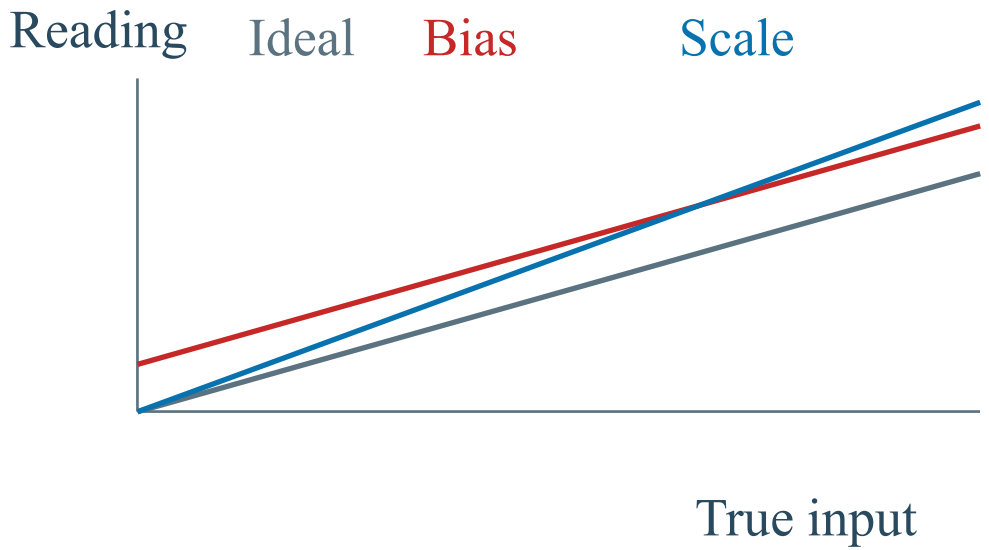


Figure: Bias shifts zero; scale changes slope

$$\mathbf{y}_s = C_{s\leftarrow b}\mathbf{y}_b, \quad \mathbf{y}_m = K\mathbf{y}_s + \mathbf{b} + \mathbf{n}$$

Bias shifts measurements as a whole; random noise scatters them around a level.

4.3.5 Sensor Subsystem Modeling: Resolution and Range

Finite Resolution and Saturation Also Change Digital Measurements

- Quantization rounds at interval Δ to finite steps. With nearest rounding and no saturation, error is at most $\Delta/2$.
- For $\Delta = 0.02$ m/s², 3.456 rounds to 3.46. Small changes within one step may leave digital output unchanged.
- With upper limit 4, true input 5 produces only 4, flattening the peak. Quantization, saturation, and random noise are three distinct mechanisms.

$$y_q = \Delta \text{round}(y/\Delta)$$

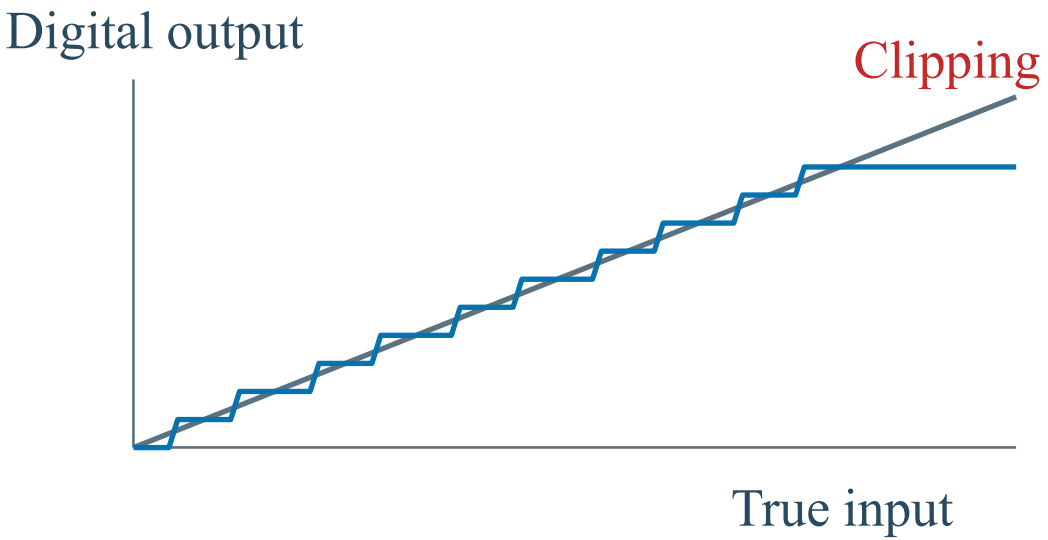


Figure: Quantization and clipping; step sizes are schematic

Quantization rounds continuous values to discrete steps; saturation clips values at the range boundary.

4.3.5 Sensor Subsystem Modeling: Drift and Noise Units

Slow Drift Differs from White Noise at Each Sample

- White noise gives independent sample-to-sample jitter; a random walk accumulates increments, slowly moving the bias baseline. The model below scales with sample time.
- ξ_k is independent standard normal noise. σ_b has units of measurement unit/√s; per-sample noise standard deviation has measurement units. Convert noise density using sampling and bandwidth; do not interchange them.

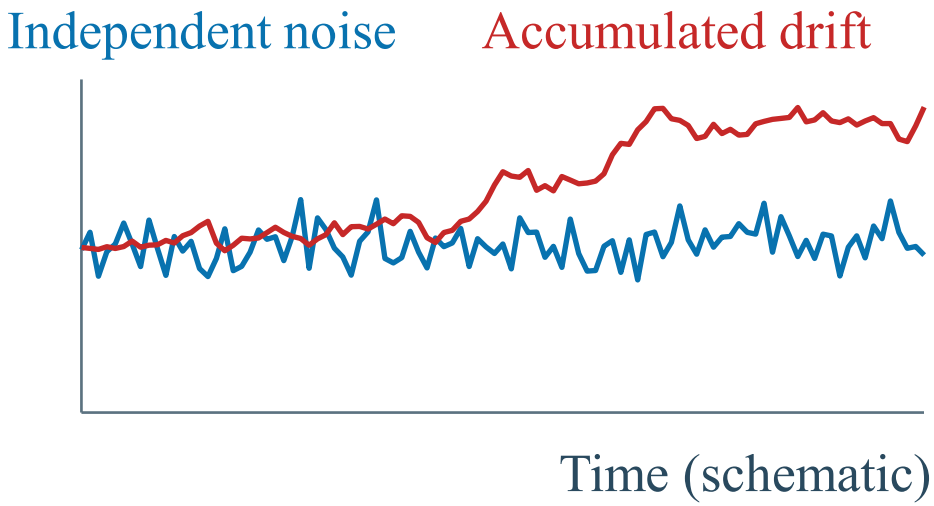


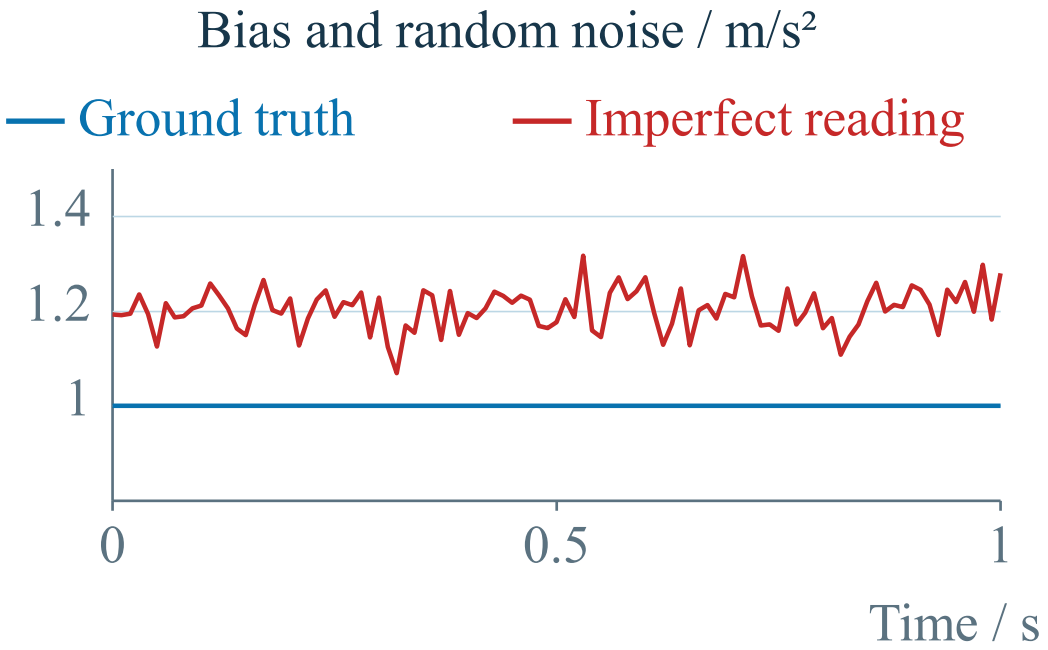
Figure: Independent jitter and accumulated drift

$$b_{k+1} = b_k + \sigma_b \sqrt{T_s}\, \xi_k, \qquad \xi_k \sim \mathcal{N}(0,1)$$

Bias drift accumulates over time; discrete noise parameters must match the sampling definition.

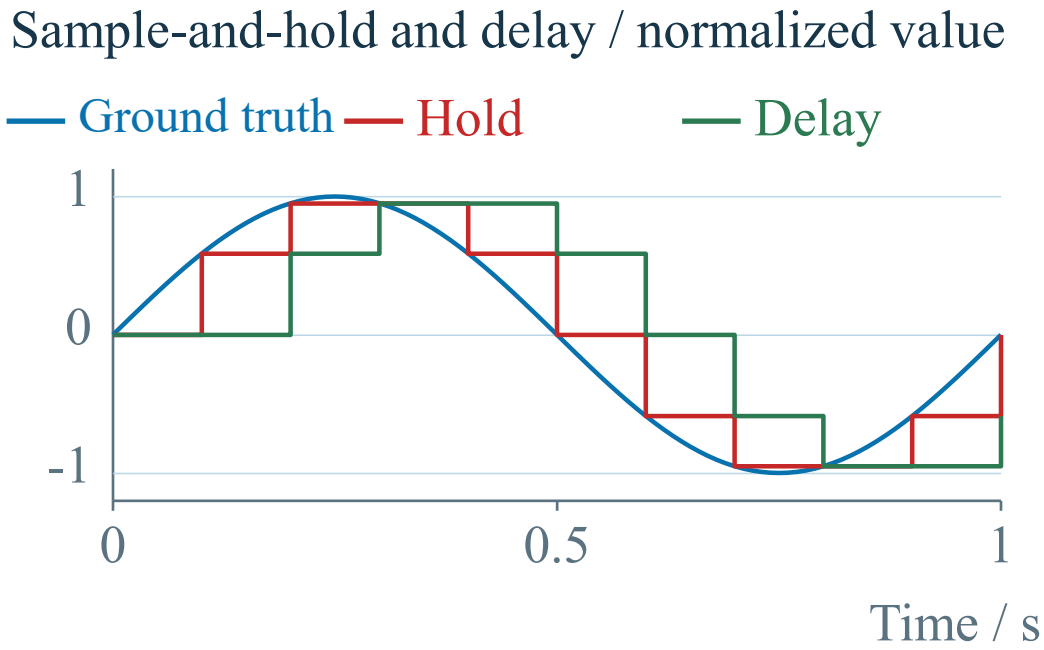
4.3.5 Sensor Subsystem Modeling: Recognizing Error Sources

Distinguish Bias, Sample-and-Hold, and Delay in Curves



Bias: mean shifts by +0.2

- Truth is 1; the imperfect reading has mean 1.2, with random noise scattered around that mean.



Sampling: 0.1 s steps **Delay: shifted 0.1 s later**

1 Hz ground truth; steps and lag occur even without random noise.

4.3.5 Sensor Subsystem Modeling: Timestamps and Delay

What Happens Between Sampling and Reception?

Sample period: 0.1 s. Transmission delay: 0.1 s. The timestamp records acquisition time.

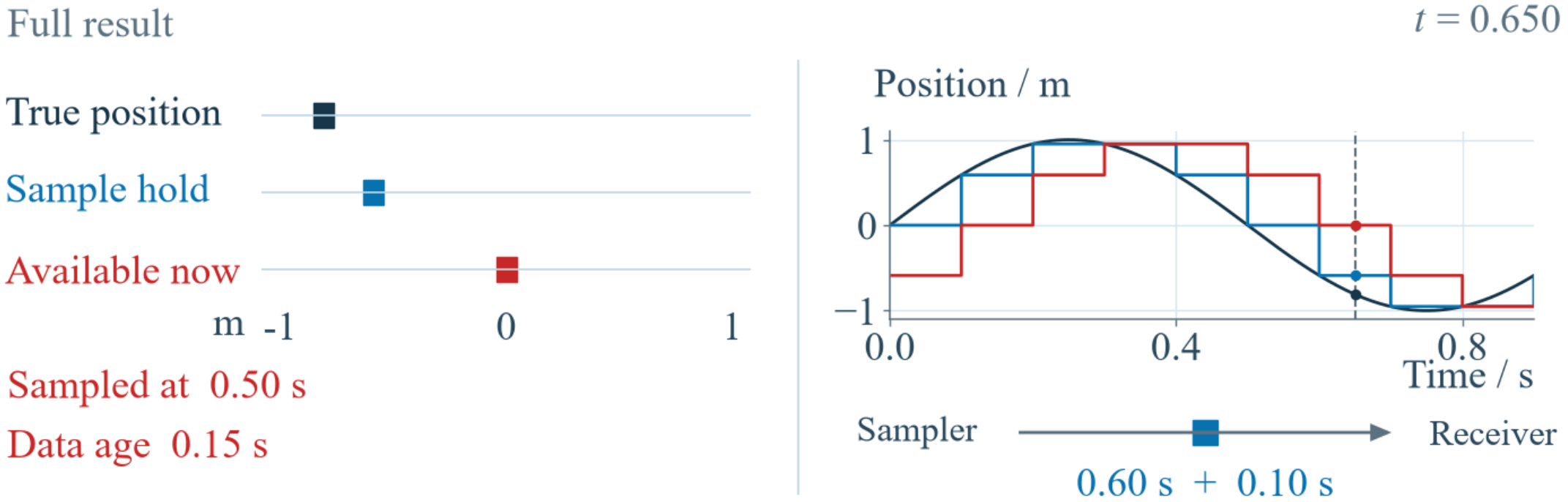


Figure: A 1 Hz position signal, sampled at 10 Hz with a 0.1 s observation delay

4.3.5 Sensor Subsystem Modeling: Vibration and Visual Data

Speed-Dependent Noise and Product Effects in Visual Sensors

- Without vibration, noise standard deviation can be constant. Motor/propeller vibration makes it vary with rotational speed; identify this from measurements.
- Optical flow: downsample camera images, then simulate flow processing to estimate velocity. Raw images come from the 3D environment subsystem.
- Camera and laser point-cloud models add blur, distortion, and noise to ideal data. Unreal Engine (UE), Unity, and other renderers simulate lighting, reflection, and fog.

Vector data	Image data
Inertial measurement unit (IMU), pressure, field	Camera, optical flow, point cloud
Speed-dependent noise	Blur, distortion, noise
Drift, bias, scale	Lighting, reflection, fog

Figure: Product effects in two sensor-data types
Images and point clouds come from RflySim3D (Chapter 3)

Vibration makes noise speed-dependent; visual sensors also add product errors to ideal data.

4.3.5 Sensor Subsystem Modeling: Component Self-Checks

Run Component Self-Checks and Compare Identical Conditions

- **Directory:** [installation directory]/RflySimAPIs/4.RflySimModel/1.BasicExps/e6_ModelingLearningLab/02_ComponentChecks/; read Readme.pdf first.
- Run with the Python 3 standard library or basic MATLAB; Simulink is unnecessary. Change to this directory. Run line 1 in a system terminal or line 2 in the MATLAB Command Window.
- **Success:** motor_step.csv is generated (Python also: checks.json; MATLAB also: checks.mat, motor_step.png). Target: 528.09 rad/s. At 0.0214 s: 333.82 rad/s and 1.873 N.
- **Failures:** If python or the file is missing, check the interpreter and working directory. Without write permission, choose a writable output directory.

```
python component_checks.py --out-dir results
component_checks('results_matlab')
```

Offline component calculations give explicit references for comparison with matching minimal-template blocks.

Class Exercise: Staged Experiment A1—Actuators and Synthesis

Experiment A1: Compute Resultant Force and Torque from Inputs

- **Task 1:** Set TAU in the self-check script (tau in MATLAB) to 0.0428 s. Predict steady-state speed, final thrust, and response speed.
- **Task 2:** $d = 0.25\,\mathrm{m}$, $\eta = 0.02\,\mathrm{m}$; each rotor starts at 4.9 N. Add 0.5 N each to rotors 1 and 2. Find total thrust, three torques, and roll acceleration ($J_x = 0.05\,\mathrm{kg \cdot m^2}$).
- **Staged experiments A1–A4:** A1 here, physical calculations; A2 (Sec. 4.7), parameter validation; A3 (Sec. 4.10), interface experiments; A4 (Sec. 4.12), one model change.

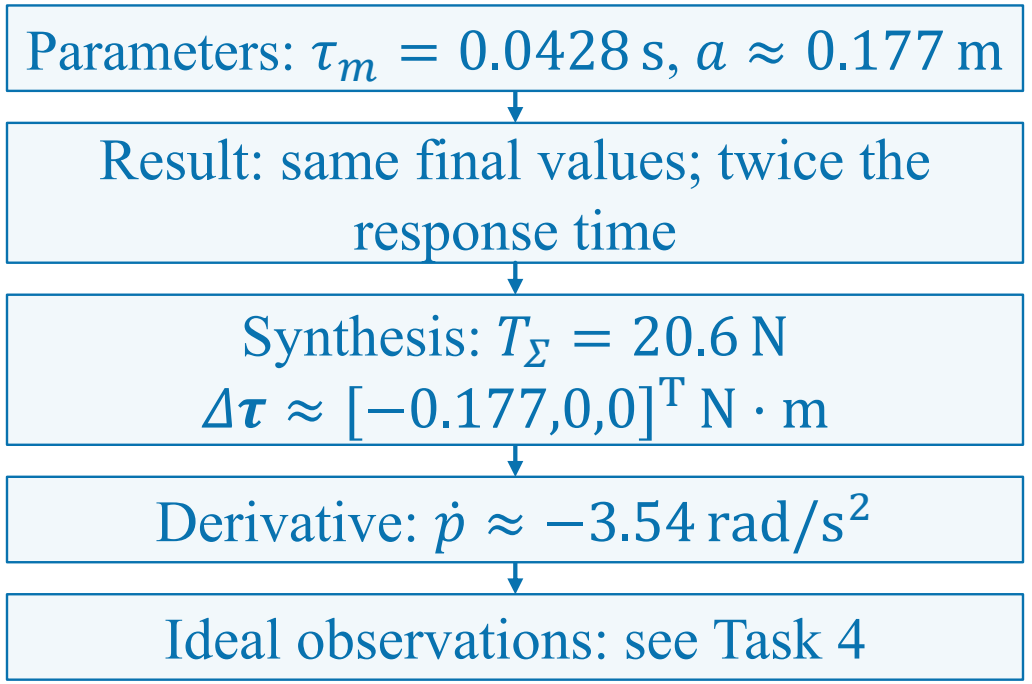


Figure: Check references along the computation chain

Use actual actuator outputs for force and torque synthesis, then inertia for angular acceleration.

Class Exercise: Staged Experiment A1—Environment and Sensors

Experiment A1: Compute Wind Effects and Check Ideal Observations

- **Task 3:** Nose east; NED ground velocity $[0,5,0]^{\mathrm{T}}$, wind $[3,4,0]^{\mathrm{T}}$ m/s, $k = 0.055\ \mathrm{N}\cdot\mathrm{s}^2/\mathrm{m}^2$. Find body-frame air-relative velocity and drag.
- **Task 4:** Let g=9.8 and align a centered sensor with body axes. Find specific force for $R = I$ with 2 m/s² upward acceleration, and at rest on a slope giving 30° body roll.
- Submit frame conventions, parameters, equations, input–speed–thrust curves, and explanations.

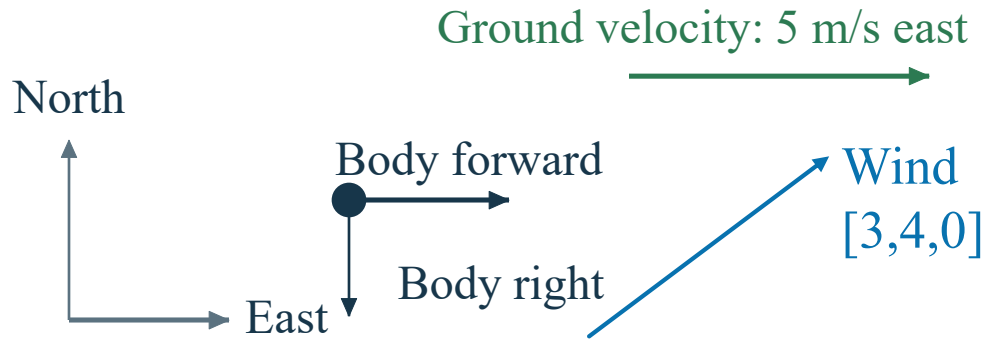


Figure: Top view; north up, east right, nose east

- **Reference 3:** $\mathbf{v}_{\mathrm{rel},b} = [1,3,0]^{\mathrm{T}}$ m/s; drag $\approx [-0.055, -0.495, 0]^{\mathrm{T}}$ N.
- **Reference 4:** $\mathbf{f}_b = [0,0,-11.8]^{\mathrm{T}}$; on the slope, $\mathbf{f}_b \approx [0, -4.90, -8.49]^{\mathrm{T}}$ m/s².

State the frame, units, and operating condition for every result to make the model reproducible.

Section Summary

From Commands to Observations: Methods, Judgments, and Pitfalls

- **Method:** Saturation and threshold set target speed; first-order dynamics gives current speed, then $T = c_T \omega^2$. Positions and rotation directions set force/torque synthesis. Environment changes relative airflow and contact; sensors turn ground truth into observations.
- **Actuators:** Distinguish target from current speed; use current speed squared for thrust. Convert rpm to rad/s before applying the thrust coefficient.
- **Synthesis:** Check motor numbering and rotation directions before comparing matrices. Inverse-map thrusts must lie between zero and each rotor's maximum.
- **Environment and contact:** Drag opposes air-relative velocity, not ground velocity. Ground pushes but cannot pull; clip negative support to zero. For stiffer contact, reduce the timestep and check convergence.
- **Observations:** Specific force = motion acceleration minus gravity. Ground truth, measurement, and estimate are distinct. R maps body vectors to Earth axes; R^T reverses this. Interchanging them gives incorrect component signs.