

An Inverted Pendulum on a Wheeled Base

May 11, 2026

Contents

Introduction	ii
1 Lagrangian Derivation of the Equations of Motion	1
1.1 System and Conventions	1
1.2 Kinematics	1
1.3 Kinetic Energy	3
1.4 Potential Energy	3
1.5 The Lagrangian	3
1.6 Generalized Forces	3
1.7 Euler–Lagrange Equations	4
1.8 Manipulator Form	4
1.9 Sanity Checks	4
2 Linearization About the Upright Equilibrium	6
2.1 Starting Point: the Nonlinear Dynamics	6
2.2 First-Order State-Space Form	6
2.3 The Upright Equilibrium	7
2.4 Linearization via Taylor Expansion	7
2.5 Computing the Acceleration Jacobians	7
2.6 Closed-Form A and B	9
3 Parametric LQR Stabilization of the Upright Equilibrium	10
3.1 Setup	10
3.2 The Linear–Quadratic Regulator	11
3.3 Choosing Q and R : Bryson’s Rule	12
3.4 Worked Example	12
3.5 Robustness Margins	14
3.6 What This Builds Toward	15
4 Drake Cross-Check of the Symbolic LQR	16
4.1 Why This Note Exists	16
4.2 One Source of Truth	16
4.3 The URDF Encoding	17
4.4 What Drake Returns at the Upright Equilibrium	17
4.5 Building A and B in State-Space Form	17
4.6 LQR Solution and Comparison	18
4.7 Closed-Loop Spectrum	18
4.8 What This Validates	19

5	Closed-Loop Simulation on the Linearized Plant	20
5.1	What This Chapter Shows	20
5.2	Stabilization	20
5.3	Step Tracking in Cart Position	20
5.4	What This Chapter Does Not Validate	21
6	LQR for the 3D Differential-Drive Segway via Drake	23
6.1	What This Chapter Does	23
6.2	Parameters	23
6.3	Generalized Coordinates and the Rolling Constraints	26
6.4	What Drake Returns at the Upright Equilibrium	26
6.5	Projection to the Constrained Dynamics	27
6.6	State-Space Linearization	28
6.7	LQR Design	29
6.8	Decoupling: Modal Form	29
6.9	Closed-Loop Spectrum	30
6.10	Bridge to MuJoCo	30
A	A Note About Bryson’s Rule: Choosing Q and R Without Choosing Numbers	31
A.1	The setup	31
A.2	The rule	31
A.3	Why this choice is natural	32
A.4	Why squared, not linear	32
A.5	What the rule does <i>not</i> do	32
A.6	Application: the 2D segway	33
A.7	Connecting back	33
B	From the Lagrangian to $C(q, \dot{q})$: Why Christoffel Symbols Appear	34
B.1	The setup	34
B.2	The derivation, line by line	34
B.3	The symmetrization trick	35
B.4	Reading off the Coriolis matrix	35
B.5	Why this C in particular: skew-symmetry of $\dot{M} - 2C$	36
B.6	A geometric remark	36
B.7	Application: the 2D segway	37
B.8	Connecting back	37
C	A Note About Pythagorean Cancellation: Why the Body’s Kinetic Energy Collapses So Cleanly	38
C.1	The setup	38
C.2	The cancellation, line by line	38
C.3	Why this is not algebraic luck	39
C.4	A picture	39
C.5	Why this matters in the dynamics	40

Introduction

This book is a self-contained theoretical tour of one toy system: an inverted pendulum mounted on a wheeled base. The 2D variant is the familiar cart-pole with rolling, no-slip wheel kinematics; the 3D variant adds yaw and a second drive wheel. The goal is to derive the equations of motion, linearize them about the upright equilibrium, design a stabilizing controller, and validate every step against an independent implementation in Drake.

What's in each chapter

Chapter 1 (Dynamics) derives the manipulator-form equations of motion from a Lagrangian. Out of this fall the mass matrix $M(q)$, the Coriolis matrix $C(q, \dot{q})$, the gravity term $\tau_g(q)$, and the input map B .

Chapter 2 (Linearization) expands the nonlinear dynamics about the upright equilibrium and produces the state-space matrices A and B in closed form, parametric in the physical constants.

Chapter 3 (LQR) sets up the linear-quadratic regulator, chooses cost weights Q and R via Bryson's rule (see Appendix ??), solves the algebraic Riccati equation, and reports the optimal feedback gain K .

Chapter 4 (Drake cross-check) regenerates the same plant in Drake from a URDF, recomputes A , B , and K , and asserts agreement with the symbolic pipeline. The point is to discover transcription errors in the hand derivation, not to claim Drake is more correct.

Chapter 5 (Closed-loop simulation) integrates the linearized plant under the Chapter 3 gain and exhibits the canonical responses: recovery from an initial tilt and tracking of a step in cart position.

Chapter 6 (3D LQR) repeats the controller design for the differential-drive 3D segway, where two wheels share the load and yaw becomes an independent degree of freedom.

Appendices

The appendices collect short notes on points that would otherwise clutter the main derivations: why Christoffel symbols appear in $C(q, \dot{q})$, why the body's kinetic energy collapses so cleanly via a Pythagorean cancellation, and what Bryson's rule is doing to Q and R .

How to read this

Each chapter is also a standalone PDF (the per-chapter `.tex` files in `output/` build on their own). This book is just those chapters glued together, with a table of contents and these appendices. The chapters were written to be read in order, but Chapters 4 and 5 can be skipped on a first pass without losing the plot.

Chapter 1

Lagrangian Derivation of the Equations of Motion

1.1 System and Conventions

We consider a planar inverted pendulum mounted on a single rolling wheel (a ‘2D segway’). The system consists of two rigid bodies:

- A **wheel** of mass m_w , radius r , and moment of inertia I_w about its axle.
- A **body** (the inverted pendulum) of mass m_b , with its centre of mass at a distance ℓ above the wheel axle along the body’s long axis, and moment of inertia I_b about that centre of mass.

The wheel rolls without slipping on a flat horizontal surface. A scalar motor torque u acts *between* the body and the wheel at the axle. By Newton’s third law it applies $+u$ to the wheel and $-u$ to the body. See Figure 1.1.

Generalized coordinates

The configuration of the system has three rigid-body coordinates: the wheel-centre horizontal position x , the wheel rotation φ , and the body tilt θ (measured from vertical, positive when the COM leans in the $+x$ direction). The rolling constraint $\dot{x} = r\dot{\varphi}$ is holonomic, so we may eliminate φ directly and use the two-DOF coordinate vector

$$q = \begin{bmatrix} x \\ \theta \end{bmatrix}.$$

1.2 Kinematics

Coordinate frame

All positions are measured in a fixed inertial frame with origin on the ground directly under the wheel’s initial location. The x -axis points horizontally in the direction of forward motion; the y -axis points vertically upward (opposite gravity). In this frame, the gravitational acceleration vector is $\mathbf{g} = (0, -g)^\top$ with $g > 0$.

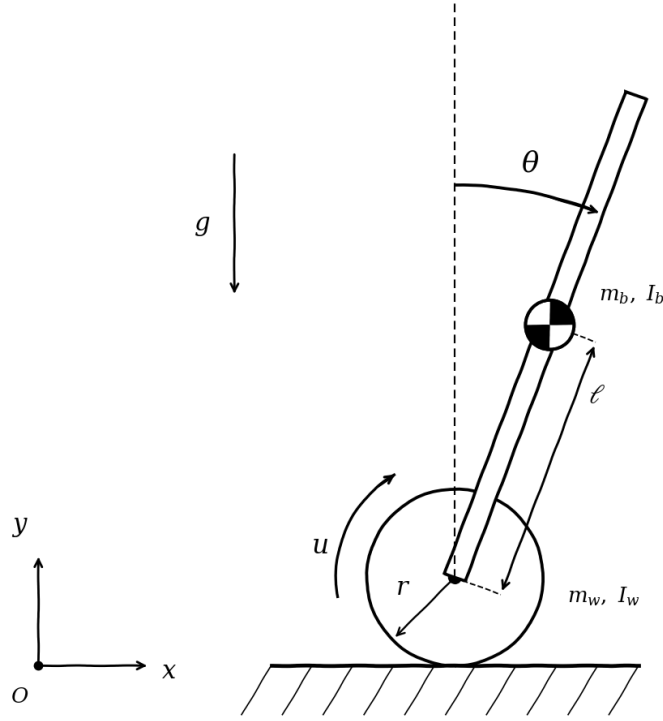


Figure 1.1: Diagram of a two dimensional inverted pendulum on a wheeled base.

Position vectors

The wheel axle sits at constant height r above the ground; only its horizontal position varies with x . The body's centre of mass lies a distance ℓ from the axle along the body axis. Writing each position as $\mathbf{p} = (\text{horizontal component}, \text{vertical component})^\top$,

$$\mathbf{p}_w = \begin{bmatrix} x_w \\ y_w \end{bmatrix} = \begin{bmatrix} x \\ r \end{bmatrix}, \quad \mathbf{p}_b = \begin{bmatrix} x_b \\ y_b \end{bmatrix} = \begin{bmatrix} x + \ell \sin \theta \\ r + \ell \cos \theta \end{bmatrix}.$$

Here x_b and y_b are simply names for the two components of $\mathbf{p}_b$; they are not new generalized coordinates.

Velocities

Differentiating $\mathbf{p}_b$ with respect to time, with x and θ both functions of time,

$$\dot{\mathbf{p}}_b = \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \end{bmatrix} = \begin{bmatrix} \dot{x} + \ell \cos \theta \dot{\theta} \\ -\ell \sin \theta \dot{\theta} \end{bmatrix}.$$

The no-slip rolling constraint relates the wheel spin rate to the translation rate:

$$\dot{\varphi} = \frac{\dot{x}}{r}.$$

1.3 Kinetic Energy

The wheel's kinetic energy is the sum of translation of its centre and rotation about that centre. Substituting $\dot{\varphi} = \dot{x}/r$,

$$T_w = \frac{1}{2}m_w\dot{x}^2 + \frac{1}{2}I_w\left(\frac{\dot{x}}{r}\right)^2 = \frac{1}{2}\left(m_w + \frac{I_w}{r^2}\right)\dot{x}^2.$$

The body's kinetic energy is translation of its COM plus rotation about the COM at angular rate $\dot{\theta}$. Expanding $\dot{x}_b^2 + \dot{y}_b^2 = \dot{x}^2 + 2\ell \cos \theta \dot{x}\dot{\theta} + \ell^2\dot{\theta}^2$ gives

$$T_b = \frac{1}{2}m_b\dot{x}^2 + m_b\ell \cos \theta \dot{x}\dot{\theta} + \frac{1}{2}(m_b\ell^2 + I_b)\dot{\theta}^2.$$

Collecting like terms in $\dot{x}^2$, $\dot{x}\dot{\theta}$, and $\dot{\theta}^2$, the total kinetic energy takes the canonical quadratic form

$$T = \frac{1}{2}\left(m_w + m_b + \frac{I_w}{r^2}\right)\dot{x}^2 + m_b\ell \cos \theta \dot{x}\dot{\theta} + \frac{1}{2}(m_b\ell^2 + I_b)\dot{\theta}^2.$$

The three coefficients above will reappear directly as the entries M_{11} , $M_{12} = M_{21}$, and M_{22} of the mass matrix.

1.4 Potential Energy

The wheel centre stays at fixed height r , so its gravitational potential is constant and may be dropped. Only the body contributes:

$$V = m_b g y_b = g m_b (\ell \cos \theta + r).$$

1.5 The Lagrangian

Combining the previous two sections,

$$\begin{aligned} \mathcal{L} = T - V = & \frac{1}{2}\left(m_w + m_b + \frac{I_w}{r^2}\right)\dot{x}^2 + m_b\ell \cos \theta \dot{x}\dot{\theta} + \frac{1}{2}(m_b\ell^2 + I_b)\dot{\theta}^2 \\ & - m_b g (r + \ell \cos \theta). \end{aligned}$$

The constant term $-m_b g r$ has no dynamical effect (its derivatives vanish) and may be dropped without consequence:

$$\mathcal{L} = \frac{1}{2}\left(m_w + m_b + \frac{I_w}{r^2}\right)\dot{x}^2 + m_b\ell \cos \theta \dot{x}\dot{\theta} + \frac{1}{2}(m_b\ell^2 + I_b)\dot{\theta}^2 - m_b g \ell \cos \theta.$$

1.6 Generalized Forces

The only non-conservative force is the motor torque u , which acts *internally* between wheel and body. Using virtual work and the rolling constraint $\delta\varphi = \delta x/r$,

$$\delta W = u \delta\varphi + (-u) \delta\theta = \frac{u}{r} \delta x - u \delta\theta.$$

Reading off the coefficients of δx and $\delta\theta$,

$$Q_x = \frac{u}{r}, \quad Q_\theta = -u.$$

1.7 Euler–Lagrange Equations

Applying $\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i} \right) - \frac{\partial \mathcal{L}}{\partial q_i} = Q_i$ to each generalized coordinate yields the equations of motion. Grouping the result by $\ddot{x}$, $\ddot{\theta}$, and $\dot{\theta}^2$ gives

$$\begin{aligned} \ddot{x} \left(\frac{I_w}{r^2} + m_b + m_w \right) + \ddot{\theta} \ell m_b \cos \theta - \dot{\theta}^2 \ell m_b \sin \theta &= \frac{u}{r}, \\ \ddot{x} \ell m_b \cos \theta + \ddot{\theta} (I_b + \ell^2 m_b) - \ell g m_b \sin \theta &= -u. \end{aligned}$$

1.8 Manipulator Form

Following the convention used by Tedrake’s *Underactuated Robotics* and Drake’s `MultibodyPlant`, every Lagrangian system can be written as

$$M(q) \ddot{q} + C(q, \dot{q}) \dot{q} = \tau_g(q) + B u,$$

where M is the mass matrix (coefficient of $\ddot{q}$), $\tau_g(q)$ is the generalized gravity force ($\tau_g = -\partial V / \partial q$), C is the Coriolis matrix computed from M via the Christoffel symbols

$$C_{ij} = \frac{1}{2} \sum_k \left(\frac{\partial M_{ij}}{\partial q_k} + \frac{\partial M_{ik}}{\partial q_j} - \frac{\partial M_{jk}}{\partial q_i} \right) \dot{q}_k,$$

and B describes how the actuation enters each coordinate. For our system,

$$\begin{aligned} M(q) &= \begin{bmatrix} \frac{I_w}{r^2} + m_b + m_w & \ell m_b \cos \theta \\ \ell m_b \cos \theta & I_b + \ell^2 m_b \end{bmatrix}, & C(q, \dot{q}) &= \begin{bmatrix} 0 & -\dot{\theta} \ell m_b \sin \theta \\ 0 & 0 \end{bmatrix}, \\ \tau_g(q) &= \begin{bmatrix} 0 \\ \ell g m_b \sin \theta \end{bmatrix}, & B &= \begin{bmatrix} \frac{1}{r} \\ -1 \end{bmatrix}. \end{aligned}$$

1.9 Sanity Checks

Several internal consistency conditions hold (verified symbolically by the accompanying script):

1. **Symmetry of M .** $M(q) = M(q)^\top$, as required for any kinetic-energy quadratic form.
2. **Positive-definiteness of M .** $M_{11} = \frac{I_w}{r^2} + m_b + m_w > 0$ and $\det M = \frac{I_b I_w}{r^2} + I_b m_b + I_b m_w + \frac{I_w \ell^2 m_b}{r^2} + \ell^2 m_b^2 \sin^2 \theta + \ell^2 m_b m_w > 0$ for all θ , because every term is non-negative and the terms not involving $\sin^2 \theta$ are strictly positive (so the bound holds even at $\theta = 0$ where $\sin \theta = 0$).

3. **Limit $\ell \rightarrow 0$.** The body collapses onto the axle and the equations decouple to $(m_w + m_b + I_w/r^2)\ddot{x} = u/r$ (a rolling cylinder) and $I_b\ddot{\theta} = -u$ (a pure rotor), as expected.
4. **Limit $m_w, I_w \rightarrow 0$.** A massless wheel removes $I_w/r^2 + m_w$ from M_{11} ; the remaining body equations match the textbook cart–pole on a frictionless slider with cart mass m_b .
5. **Energy conservation.** With $u = 0$, substituting the Euler–Lagrange equations into $\frac{d}{dt}(T + V)$ yields identically zero, as expected for an autonomous Lagrangian system.

Chapter 2

Linearization About the Upright Equilibrium

2.1 Starting Point: the Nonlinear Dynamics

From the Lagrangian derivation we have the manipulator-form equation,

$$M(q) \ddot{q} + C(q, \dot{q}) \dot{q} = \tau_g(q) + B_m u,$$

where $q = [x, \theta]^\top$ and

$$\begin{aligned} M(q) &= \begin{bmatrix} \frac{I_w}{r^2} + m_b + m_w & \ell m_b \cos \theta \\ \ell m_b \cos \theta & I_b + \ell^2 m_b \end{bmatrix}, & C(q, \dot{q}) &= \begin{bmatrix} 0 & -\dot{\theta} \ell m_b \sin \theta \\ 0 & 0 \end{bmatrix}, \\ \tau_g(q) &= \begin{bmatrix} 0 \\ \ell g m_b \sin \theta \end{bmatrix}, & B_m &= \begin{bmatrix} \frac{1}{r} \\ -1 \end{bmatrix}. \end{aligned}$$

We write B_m (the manipulator-form input map) instead of B to avoid clashing with the state-space input matrix B that appears below.

2.2 First-Order State-Space Form

Linearization is naturally expressed for first-order systems $\dot{\mathbf{x}} = f(\mathbf{x}, u)$. Following conventional notation, we stack positions and velocities into a single state vector

$$\mathbf{x} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} = \begin{bmatrix} x \\ \theta \\ \dot{x} \\ \dot{\theta} \end{bmatrix} \in \mathbb{R}^4.$$

The bold $\mathbf{x}$ is the full state; the plain italic x that appears as its first component is the wheel-center coordinate. Solving the manipulator equation for $\ddot{q}$ gives the explicit first-order form

$$\dot{\mathbf{x}} = f(\mathbf{x}, u) = \begin{bmatrix} \dot{q} \\ M(q)^{-1}[\tau_g(q) + B_m u - C(q, \dot{q})\dot{q}] \end{bmatrix}.$$

2.3 The Upright Equilibrium

An equilibrium is any pair $(\mathbf{x}^*, u^*)$ at which $f(\mathbf{x}^*, u^*) = 0$. Setting $u^* = 0$, the velocity rows of f vanish whenever $\dot{q}^* = 0$, and the acceleration rows reduce to $M(q^*)^{-1}\tau_g(q^*) = 0$, i.e. $\tau_g(q^*) = 0$. For our gravity force $\tau_g(q) = [0, m_b g \ell \sin \theta]^\top$ this requires $\sin \theta^* = 0$, giving two equilibria: $\theta^* = 0$ (*upright*) and $\theta^* = \pi$ (*hanging*). The upright equilibrium is the interesting one — it is unstable, and stabilizing it is the purpose of the robot's control system. Translational symmetry leaves x^* free; we pick $x^* = 0$ without loss of generality. In summary,

$$\mathbf{x}^* = \mathbf{0} \in \mathbb{R}^4, \quad u^* = 0.$$

2.4 Linearization via Taylor Expansion

Let $\delta \mathbf{x} = \mathbf{x} - \mathbf{x}^*$ and $\delta u = u - u^*$. Taylor-expanding f to first order about $(\mathbf{x}^*, u^*)$ gives

$$\delta \dot{\mathbf{x}} = f(\mathbf{x}^* + \delta \mathbf{x}, u^* + \delta u) \approx \underbrace{f(\mathbf{x}^*, u^*)}_{=0} + \underbrace{\frac{\partial f}{\partial \mathbf{x}} \Big|_{\text{eq}}}_A \delta \mathbf{x} + \underbrace{\frac{\partial f}{\partial u} \Big|_{\text{eq}}}_B \delta u.$$

Dropping the δ 's for compactness, the linear system is

$$\dot{\mathbf{x}} = A \mathbf{x} + B u, \quad A \in \mathbb{R}^{4 \times 4}, \quad B \in \mathbb{R}^{4 \times 1}.$$

Partitioning $\mathbf{x} = [q; \dot{q}]$ and $f = [\dot{q}; \ddot{q}]$, the Jacobian A takes the block form

$$A = \begin{bmatrix} \partial \dot{q} / \partial q & \partial \dot{q} / \partial \dot{q} \\ \partial \ddot{q} / \partial q & \partial \ddot{q} / \partial \dot{q} \end{bmatrix}_{\text{eq}} = \begin{bmatrix} 0_{2 \times 2} & I_{2 \times 2} \\ \frac{\partial \ddot{q}}{\partial q} \Big|_{\text{eq}} & \frac{\partial \ddot{q}}{\partial \dot{q}} \Big|_{\text{eq}} \end{bmatrix}.$$

The top rows are trivial; the work is in the bottom rows.

2.5 Computing the Acceleration Jacobians

From $\ddot{q} = M(q)^{-1}[\tau_g(q) + B_m u - C(q, \dot{q})\dot{q}]$, the chain rule gives

$$\frac{\partial \ddot{q}}{\partial q_k} = \frac{\partial M^{-1}}{\partial q_k} [\tau_g + B_m u - C\dot{q}] + M^{-1} \left[\frac{\partial \tau_g}{\partial q_k} - \frac{\partial (C\dot{q})}{\partial q_k} \right].$$

At the equilibrium the bracket $[\tau_g + B_m u - C\dot{q}]$ is exactly zero (that is what makes $(\mathbf{x}^*, u^*)$ an equilibrium), so the $\partial M^{-1} / \partial q$ term *vanishes*. This is the key simplification: we never need to differentiate a matrix inverse. What remains is

$$\begin{aligned}\left.\frac{\partial \ddot{q}}{\partial q}\right|_{\text{eq}} &= M(q^*)^{-1} \left[\frac{\partial \tau_g}{\partial q} - \frac{\partial (C\dot{q})}{\partial q} \right]_{\text{eq}}, \\ \left.\frac{\partial \ddot{q}}{\partial \dot{q}}\right|_{\text{eq}} &= -M(q^*)^{-1} \left[\frac{\partial (C\dot{q})}{\partial \dot{q}} \right]_{\text{eq}}, \\ \left.\frac{\partial \ddot{q}}{\partial u}\right|_{\text{eq}} &= M(q^*)^{-1} B_m.\end{aligned}$$

Note the sign asymmetry: τ_g sits on the RHS of the manipulator form, so its Jacobian enters with a +; $C\dot{q}$ sits on the LHS, so its Jacobian enters negated.

Mass matrix at the equilibrium

Substituting $\theta^* = 0$ (so $\cos \theta^* = 1$):

$$\begin{aligned}M(q^*) &= \begin{bmatrix} \frac{I_w}{r^2} + m_b + m_w & \ell m_b \\ \ell m_b & I_b + \ell^2 m_b \end{bmatrix}, \\ M(q^*)^{-1} &= \begin{bmatrix} \frac{r^2(I_b + \ell^2 m_b)}{I_b I_w + I_b m_b r^2 + I_b m_w r^2 + I_w \ell^2 m_b + \ell^2 m_b m_w r^2} & -\frac{\ell m_b r^2}{I_b I_w + I_b m_b r^2 + I_b m_w r^2 + I_w \ell^2 m_b + \ell^2 m_b m_w r^2} \\ -\frac{\ell m_b r^2}{I_b I_w + I_b m_b r^2 + I_b m_w r^2 + I_w \ell^2 m_b + \ell^2 m_b m_w r^2} & \frac{I_w + m_b r^2 + m_w r^2}{I_b I_w + I_b m_b r^2 + I_b m_w r^2 + I_w \ell^2 m_b + \ell^2 m_b m_w r^2} \end{bmatrix}.\end{aligned}$$

Gravity Jacobian

With $\tau_g(q) = [0, m_b g \ell \sin \theta]^\top$,

$$\left.\frac{\partial \tau_g}{\partial q}\right|_{\text{eq}} = \begin{bmatrix} 0 & 0 \\ 0 & \ell g m_b \end{bmatrix}.$$

Only the lower-right entry survives: $+m_b g \ell$ is the gravitational ‘anti-spring’ that an inverted pendulum supplies in place of a restoring force. That positive sign in $\partial \tau_{g,\theta} / \partial \theta$ is what destabilises the upright equilibrium.

Coriolis/centrifugal Jacobians (both vanish)

$C(q, \dot{q})\dot{q}$ is *quadratic* in $\dot{q}$ (it contains only the $\dot{\theta}^2$ term $-m_b \ell \sin \theta \dot{\theta}^2$). Differentiating a quadratic and evaluating at $\dot{q}^* = 0$ gives zero, so

$$\left.\frac{\partial (C\dot{q})}{\partial q}\right|_{\text{eq}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \left.\frac{\partial (C\dot{q})}{\partial \dot{q}}\right|_{\text{eq}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

This is a *generic* feature of mechanical systems linearized at rest: Coriolis terms always drop out at the linear level, and the linearization is determined entirely by $M(q^*)$, $\partial \tau_g / \partial q$, and B_m .

2.6 Closed-Form A and B

Assembling the blocks,

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{\ell^2 g m_b^2 r^2}{I_b I_w + I_b m_b r^2 + I_b m_w r^2 + I_w \ell^2 m_b + \ell^2 m_b m_w r^2} & 0 & 0 \\ 0 & \frac{\ell g m_b (I_w + m_b r^2 + m_w r^2)}{I_b I_w + I_b m_b r^2 + I_b m_w r^2 + I_w \ell^2 m_b + \ell^2 m_b m_w r^2} & 0 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 \\ 0 \\ \frac{r(I_b + \ell^2 m_b + \ell m_b r)}{I_b I_w + I_b m_b r^2 + I_b m_w r^2 + I_w \ell^2 m_b + \ell^2 m_b m_w r^2} \\ -\frac{I_w + \ell m_b r + m_b r^2 + m_w r^2}{I_b I_w + I_b m_b r^2 + I_b m_w r^2 + I_w \ell^2 m_b + \ell^2 m_b m_w r^2} \end{bmatrix}.$$

Rows 1 and 2 of A are kinematic identities: position rates are their corresponding velocities, by definition. Row 3 reflects the tilt-translation coupling: a nonzero θ produces a lateral gravitational force that drives wheel acceleration. Row 4 is the instability — the positive $\partial\tau_{g,\theta}/\partial\theta$ entry causes tilt to reinforce itself, unlike a hanging pendulum where the same entry is negative and restoring. The first column is zero throughout, a direct consequence of translational symmetry: the dynamics are independent of horizontal position.

Rows 1 and 2 of B are zero because torque acts through accelerations, not directly on positions. Row 3 is positive: torque accelerates the wheel in the direction it is applied. Row 4 is negative: by Newton's third law, the same torque acts oppositely on the body, tending to push it back toward vertical. This sign difference between rows 3 and 4 is the balancing mechanism — forward torque simultaneously drives the wheel under the falling body and applies a corrective moment to the body itself. That both entries are nonzero means a single motor has authority over both degrees of freedom, which is what makes the system controllable.

Chapter 3

Parametric LQR Stabilization of the Upright Equilibrium

3.1 Setup

From the linearization step we have the time-invariant linear system

$$\dot{\mathbf{x}} = A(p) \mathbf{x} + B(p) u, \quad \mathbf{x} = \begin{bmatrix} x \\ \theta \\ \dot{x} \\ \dot{\theta} \end{bmatrix} \in \mathbb{R}^4, \quad u \in \mathbb{R},$$

where the state and input are deviations from the upright equilibrium $(\mathbf{x}^*, u^*) = (\mathbf{0}, 0)$ and $p = \{m_w, m_b, \ell, I_w, I_b, r, g\}$ collects the physical parameters. In closed form (computed in `linearize.py`),

$$A(p) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{\ell^2 g m_b^2 r^2}{I_b I_w + I_b m_b r^2 + I_b m_w r^2 + I_w \ell^2 m_b + \ell^2 m_b m_w r^2} & 0 & 0 \\ 0 & \frac{\ell g m_b (I_w + m_b r^2 + m_w r^2)}{I_b I_w + I_b m_b r^2 + I_b m_w r^2 + I_w \ell^2 m_b + \ell^2 m_b m_w r^2} & 0 & 0 \end{bmatrix},$$

$$B(p) = \begin{bmatrix} 0 \\ 0 \\ \frac{r(I_b + \ell^2 m_b + \ell m_b r)}{I_b I_w + I_b m_b r^2 + I_b m_w r^2 + I_w \ell^2 m_b + \ell^2 m_b m_w r^2} \\ -\frac{I_w + \ell m_b r + m_b r^2 + m_w r^2}{I_b I_w + I_b m_b r^2 + I_b m_w r^2 + I_w \ell^2 m_b + \ell^2 m_b m_w r^2} \end{bmatrix}.$$

The upright equilibrium is unstable: $A(p)$ has one eigenvalue in the right half-plane (the gravitational ‘anti-spring’ $+m_b g \ell$ on the RHS of the manipulator form). The control problem is to design a state feedback $u(\mathbf{x})$ that drives any small perturbation back to the origin while remaining cheap with motor torque. The pair (A, B) is controllable, so stabilizing $u(\mathbf{x})$ exist; the question is which one to pick.

3.2 The Linear–Quadratic Regulator

LQR makes the choice by minimizing the infinite-horizon quadratic cost-to-go from the current state $\mathbf{x}$:

$$J^*(\mathbf{x}) = \min_{u(\cdot)} \int_0^\infty [\mathbf{x}^\top Q \mathbf{x} + u^\top R u] dt, \quad Q \succeq 0, \quad R \succ 0.$$

The matrix Q penalizes state error; R penalizes control effort. Two structural choices deserve emphasis. First, the cost is *quadratic*, which (as shown below) makes the optimal feedback *linear*. Second, the horizon is *infinite*, which makes the optimal gain *time-invariant*. Together these give the simplest implementable controller: a constant matrix K such that $u = -K\mathbf{x}$.

Hamilton–Jacobi–Bellman

By Bellman’s principle, applied infinitesimally to $J^*(\mathbf{x})$,

$$0 = \min_u \left\{ \mathbf{x}^\top Q \mathbf{x} + u^\top R u + \nabla J^*(\mathbf{x})^\top (A\mathbf{x} + B u) \right\}.$$

This is the HJB equation specialized to LQR. It is a partial differential equation in J^* , but the quadratic structure of the running cost suggests a quadratic ansatz

$$J^*(\mathbf{x}) = \mathbf{x}^\top S \mathbf{x}, \quad S = S^\top \succ 0,$$

which gives $\nabla J^*(\mathbf{x}) = 2S\mathbf{x}$. Substituting and performing the inner minimization over u (the integrand is convex quadratic in u , so set its gradient to zero) gives

$$2Ru + 2B^\top S \mathbf{x} = 0 \implies u^* = -R^{-1}B^\top S \mathbf{x}.$$

Substituting u^* back into the HJB equation, using $2\mathbf{x}^\top S A \mathbf{x} = \mathbf{x}^\top (A^\top S + S A) \mathbf{x}$ to symmetrize, and demanding that the result hold for all $\mathbf{x}$, yields the *continuous-time algebraic Riccati equation* (CARE):

$$A^\top S + S A - S B R^{-1} B^\top S + Q = 0.$$

The optimal feedback law is

$$u^*(\mathbf{x}) = -K \mathbf{x}, \quad K = R^{-1} B^\top S.$$

Closed-loop stability

Substituting $u = -K\mathbf{x}$ into the plant gives the closed loop $\dot{\mathbf{x}} = (A - BK) \mathbf{x}$. The cost-to-go itself, $J^*(\mathbf{x}) = \mathbf{x}^\top S \mathbf{x}$, serves as a Lyapunov function:

$$\begin{aligned} \frac{d}{dt}(\mathbf{x}^\top S \mathbf{x}) &= \mathbf{x}^\top [(A - BK)^\top S + S(A - BK)] \mathbf{x} \\ &= \mathbf{x}^\top [A^\top S + S A - 2S B R^{-1} B^\top S] \mathbf{x} \quad (\text{use } K = R^{-1} B^\top S) \\ &= \mathbf{x}^\top [-Q + S B R^{-1} B^\top S - 2S B R^{-1} B^\top S] \mathbf{x} \quad (\text{use CARE}) \\ &= -\mathbf{x}^\top [Q + K^\top R K] \mathbf{x} \leq 0. \end{aligned}$$

Provided the cost weights in §4 are chosen so that $Q \succ 0$, $\mathbf{x}^\top Q \mathbf{x} > 0$ whenever $\mathbf{x} \neq 0$, so the inequality is strict and the origin is globally asymptotically stable.

Existence and uniqueness

The CARE has a unique stabilizing solution $S \succ 0$ whenever (A, B) is controllable and $Q \succ 0$. Both hold here: controllability of (A, B) was verified in the linearization step, and the diagonal weights in §4 are strictly positive on every state (Bryson’s rule applied with every scale chosen finite), so $Q \succ 0$ by construction.

3.3 Choosing Q and R : Bryson’s Rule

The CARE machinery is mechanical, but the *design* sits in the choice of Q and R . Picking these matrices entry-by-entry is unprincipled; a much cleaner handle is Bryson’s rule, which parameterizes the cost in physically meaningful quantities:

$$Q_{ii} = \frac{1}{(x_{i,\max})^2}, \quad R_{jj} = \frac{1}{(u_{j,\max})^2}.$$

Each $x_{i,\max}$ is the *largest excursion of state i that you are willing to tolerate as “normal”*; analogously for $u_{j,\max}$. The reading is direct: when state i saturates its limit and all other variables are zero, its contribution to the integrand is exactly unity — so weights are dimensionally consistent across heterogeneous quantities (metres, radians, m/s, rad/s, N m). Tightening a limit multiplies the corresponding weight by a quadratic factor; doubling $u_{\max}$ quarters R , doubling the controller’s authority before any nonlinear actuator constraints come into play.

This gives two parametric handles for the design as a whole:

- the physical parameters $p = \{m_w, m_b, \ell, I_w, I_b, r, g\}$ (the *plant*), and
- the design scales $\{x_{\max}, \theta_{\max}, \dot{x}_{\max}, \dot{\theta}_{\max}, u_{\max}\}$ (the *cost*).

Both are exposed at the top of `lib/lqr.py`. Changing either re-derives K on re-run; the structure of the document below is unchanged.

3.4 Worked Example

Nominal physical parameters

Symbol	Value	Units	Meaning
g	9.81	m/s ²	gravitational acceleration
m_w	0.3	kg	wheel mass (planar lumped)
m_b	2	kg	body mass
r	0.04	m	wheel radius
ℓ	0.15	m	body COM height above axle
I_w	0.00024	kg m ²	wheel inertia (solid disc)
I_b	0.015	kg m ²	body inertia (slender rod)

I_w uses the solid-disc approximation $\frac{1}{2}m_w r^2$; I_b uses the slender-rod approximation $\frac{1}{12}m_b(2\ell)^2$ for a body whose COM sits at height ℓ above the axle.

Numerical A and B

Substituting the nominal parameters above:

$$A = \begin{bmatrix} 0.0000 & 0.0000 & 1.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 1.0000 \\ 0.0000 & -15.4895 & 0.0000 & 0.0000 \\ 0.0000 & 126.4974 & 0.0000 & 0.0000 \end{bmatrix},$$

$$B = \begin{bmatrix} 0.0000 \\ 0.0000 \\ 31.5789 \\ -174.5614 \end{bmatrix}.$$

The open-loop spectrum is $\{0.0000, 0.0000, 11.2471, -11.2471\}$: two eigenvalues at the origin (translational symmetry: x and $\dot{x}$ form an integrator chain that is uncoupled at this order), and a real saddle pair $\pm\sqrt{\alpha m_b g \ell / \det M_0}$ that encodes the inverted-pendulum instability. The single positive eigenvalue is what the controller must move into the left half-plane.

Design scales

Symbol	Value	Units
$x_{\max}$	0.5	m
$\theta_{\max}$	0.087	rad
$\dot{x}_{\max}$	1	m/s
$\dot{\theta}_{\max}$	1	rad/s
$u_{\max}$	1	N m

Cost weights

Bryson's rule gives

$$Q = \begin{bmatrix} 4.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 132.1178 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 1.0000 \end{bmatrix},$$

$$R = [1.0000].$$

Riccati solution and optimal gain

Solving the CARE numerically (`scipy.linalg.solve_continuous_are`) returns a unique symmetric positive-definite S :

$$S = \begin{bmatrix} 5.8317 & 3.2420 & 3.7511 & 0.6900 \\ 3.2420 & 15.6286 & 4.0366 & 0.8138 \\ 3.7511 & 4.0366 & 5.0304 & 0.9267 \\ 0.6900 & 0.8138 & 0.9267 & 0.1769 \end{bmatrix}.$$

The optimal feedback gain is then

$$K = R^{-1}B^{\top}S = [-2.0000 \quad -14.5933 \quad -2.9158 \quad -1.6210].$$

Reading the entries of K left-to-right gives the controller's priorities at a glance: it weights θ much more heavily than x , because tilt destabilizes whereas position drift is benign; $\dot{\theta}$ outweighs $\dot{x}$ for the same reason.

Closed-loop spectrum

With the optimal gain in place,

$$\lambda(A - BK) = \{-177.7393, -11.4094, -0.8686 + 0.7201i, -0.8686 - 0.7201i\}.$$

All four eigenvalues lie in the open left half-plane, confirming asymptotic stability. The slowest mode sets the dominant time constant of the closed loop; tightening $x_{\max}$ or $\theta_{\max}$ in the design scales accelerates the corresponding mode at the cost of larger control effort.

3.5 Robustness Margins

A property of LQR worth stating before reaching for fancier techniques: at every loop-breaking point the return-difference inequality

$$(I + L(j\omega))^*(I + L(j\omega)) \succeq I, \quad L(s) = K(sI - A)^{-1}B,$$

holds for every frequency ω . This single inequality implies classical guaranteed robustness margins:

$$\text{phase margin} \geq 60^\circ, \quad \text{gain margin} \in [\tfrac{1}{2}, \infty).$$

That is, gain in any single channel can be doubled to infinity, or halved, without destabilizing the closed loop, and up to 60° of phase lag is tolerable. These bounds are independent of the choice of Q and R : any LQR controller satisfies them. This is why LQR is a defensible default for stabilizing any controllable linear system; departures from it (LQG, H_∞ , MPC) buy specific things — noise rejection, worst-case performance, hard constraints — at the cost of more design machinery.

3.6 What This Builds Toward

The next step in this notebook is a Drake simulation. At that point the parametric handles in `NOMINAL` will be replaced by a 2D URDF whose physical parameters Drake reads directly, and the LQR machinery here will run against Drake's own `Linearize` of the plant rather than against the symbolic linearization in `linearize.py`. Cross-checking those two linearizations (symbolic vs. Drake-numerical) is the cleanest validation that the URDF encodes what the equations of motion in this notebook say it should.

Chapter 4

Drake Cross-Check of the Symbolic LQR

4.1 Why This Note Exists

Up to this point the LQR controller in `lqr.tex` has rested on two long chains of human work: a Lagrangian derivation of $M(q)$, $C(q, \dot{q})$, $\tau_g(q)$, and B in `derive_dynamics.py`, and a hand-computed linearization at the upright equilibrium in `linearize.py`. Both are mechanical, but both are also long enough that a typo in either is plausible.

The cleanest test of those two stages together is to compute K *independently*, by a route that shares none of the symbolic algebra, and check that we land on the same number. Drake's `MultibodyPlant` is exactly such a route: hand it a URDF and it returns $M(q)$ and $\tau_g(q)$ from its own internal multi-body machinery. If K from that pipeline matches K from the symbolic pipeline, the URDF and the equations of motion agree, and the linearization in between is correct.

4.2 One Source of Truth

The physical parameters live in `lib/config.py`:

Symbol	Value	Units	Meaning
g	9.81	m/s ²	gravitational acceleration
m_w	0.3	kg	wheel mass
m_b	2	kg	body mass
r	0.04	m	wheel radius
ℓ	0.15	m	body COM height above axle
I_w	0.00024	kg m ²	wheel inertia (solid disc)
I_b	0.015	kg m ²	body inertia (slender rod)

Three artefacts read from this single dictionary:

- `lqr.py` substitutes the values into the symbolic $A(p)$, $B(p)$ and produces K_{sym} via SciPy's CARE solver.
- `urdf_gen.py` writes `output/segway.urdf`, a URDF that encodes the same plant for Drake.

- `drake_lqr.py` (this script) loads that URDF, queries Drake for M and τ_g , builds A, B in state-space form, and produces K_{drake} via Drake’s CARE.

Editing `config.py` and re-running each of the three scripts regenerates all of `lqr.tex`, `segway.urdf`, and this note in lockstep.

4.3 The URDF Encoding

Drake’s URDF cannot represent the no-slip rolling constraint $\dot{\phi} = \dot{x}/r$ directly. However, the manipulator-form derivation already absorbs that constraint into a *lumped cart-pole* that has the same $M(q)$, $\tau_g(q)$:

$$\begin{aligned} m_{\text{cart}} &= m_w + \frac{I_w}{r^2} = 0.4500 \text{ kg}, \\ m_{\text{pole}} &= m_b, \\ \text{COM offset} &= \ell \quad (\text{pole COM at } z = \ell \text{ above the joint origin}), \\ I_{yy}^{\text{pole}} &= I_b \quad (\text{rotation axis is } +y). \end{aligned}$$

The extra I_w/r^2 in the cart’s mass absorbs the wheel’s rotational kinetic energy $\frac{1}{2}I_w\dot{\phi}^2 = \frac{1}{2}(I_w/r^2)\dot{x}^2$ under the rolling constraint, so cart translation alone reproduces the $M_{11} = m_w + m_b + I_w/r^2$ entry. The motor enters the dynamics through the actuation map $B_m = [1/r, -1]^\top$, derived from Newton’s 3rd law plus rolling; that map is applied in the script (outside the URDF), exactly as in the symbolic derivation.

The frame convention in the URDF: x is horizontal (cart axis), y is out-of-plane (pole’s revolute axis), z is vertical with gravity $-g\hat{z}$.

4.4 What Drake Returns at the Upright Equilibrium

At $q^* = 0$, $\dot{q}^* = 0$, Drake gives

$$\begin{aligned} M_{\text{drake}}(q^*) &= \begin{bmatrix} 2.4500 & 0.3000 \\ 0.3000 & 0.0600 \end{bmatrix}, \\ \left. \frac{\partial \tau_g}{\partial q} \right|_{q^*} &= \begin{bmatrix} 0.0000 & 0.0000 \\ 0.0000 & 2.9430 \end{bmatrix}. \end{aligned}$$

The (2,2) entry of $\partial \tau_g / \partial q$ is precisely the “anti-spring” $+m_b g \ell$: the gravitational torque that pulls the body away from upright once it tilts. This is the term that makes the open loop unstable.

4.5 Building A and B in State-Space Form

At the upright equilibrium $\partial(C\dot{q})/\partial\dot{q} = 0$ (because $C\dot{q}$ is quadratic in $\dot{q}$), so the partitioned state-space matrices reduce to the same closed form used in `linearize.py`:

$$A = \begin{bmatrix} 0 & I \\ M_0^{-1} \partial \tau_g / \partial q & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ M_0^{-1} B_m \end{bmatrix}, \quad B_m = \begin{bmatrix} 1/r \\ -1 \end{bmatrix}.$$

Substituting the Drake numbers above:

$$A_{\text{drake}} = \begin{bmatrix} 0.0000 & 0.0000 & 1.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 1.0000 \\ 0.0000 & -15.4895 & 0.0000 & 0.0000 \\ 0.0000 & 126.4974 & 0.0000 & 0.0000 \end{bmatrix},$$

$$B_{\text{drake}} = \begin{bmatrix} 0.0000 \\ 0.0000 \\ 31.5789 \\ -174.5614 \end{bmatrix}.$$

4.6 LQR Solution and Comparison

The Riccati equation is solved by `pydrake.systems.controllers.LinearQuadraticRegulator(A,B,Q,R)` with the same Q, R chosen by Bryson's rule in `lqr.py`. This yields

$$K_{\text{drake}} = \begin{bmatrix} -2.0000 & -14.5933 & -2.9158 & -1.6210 \end{bmatrix}.$$

The symbolic pipeline gave (`lqr.tex`)

$$K_{\text{sym}} = \begin{bmatrix} -2.0000 & -14.5933 & -2.9158 & -1.6210 \end{bmatrix}.$$

The componentwise residuals are

Quantity	$\ \cdot_{\text{drake}} - \cdot_{\text{sym}} \ _2$
M_0	$4.441e - 16$
A	$2.125e - 11$
B	$0.000e + 00$
S	$2.010e - 12$
K	$4.172e - 13$

M_0 and B agree to machine precision: both are extracted from exact arithmetic on each side (M_0 from Drake's symbolic multi-body composition vs. SymPy substitution; B from $M_0^{-1} B_m$ in both cases). A inherits the small finite-difference error in $\partial \tau_g / \partial q$ on the Drake side — of order ε^2 for central differences with step $\varepsilon = 10^{-6}$, so on the order of 10^{-12} . K inherits it through the CARE.

4.7 Closed-Loop Spectrum

The closed-loop poles agree across both pipelines:

Drake	Symbolic
-177.7393	-177.7393
-11.4094	-11.4094
$-0.8686 - 0.7201 i$	$-0.8686 - 0.7201 i$
$-0.8686 + 0.7201 i$	$-0.8686 + 0.7201 i$

Both lie in the open left half-plane. The slowest mode sets the dominant time constant of the closed loop.

4.8 What This Validates

Two failure modes are now ruled out. First, a typo in the symbolic M , C , or τ_g in `derive_dynamics.py` would have shown up as a discrepancy in M_0 or $\partial\tau_g/\partial q$, which we just compared directly against Drake’s independent derivation from the URDF. Second, an algebra slip in the linearization (`linearize.py`) would have shown up in A or B , which are also reproduced here from a re-derivation that uses only the manipulator-form pieces. The fact that both controllers K agree to $4e-13$ across two independent computational pipelines — symbolic SymPy and numerical Drake, sharing only `config.py` — is strong evidence that the URDF, the equations of motion, and the linearization all describe the same plant.

What this does *not* validate, and is the natural next step: behavior away from the upright equilibrium. Drake’s nonlinear simulator can integrate the URDF forward in time under $u = -K\mathbf{x}$ to check basin of attraction, transient behavior, and robustness to model error — all things the linearization is silent on.

Chapter 5

Closed-Loop Simulation on the Linearized Plant

5.1 What This Chapter Shows

The previous chapter handed us $K_{\text{drake}} \approx K_{\text{sym}}$, two pipelines in agreement on a 1×4 row vector. This one asks the next question: with the controller in the loop, what does the closed-loop response actually look like? We integrate the linearized model

$$\dot{\mathbf{x}} = (A - BK)\mathbf{x} + BK\mathbf{x}_{\text{ref}}, \quad u = -K(\mathbf{x} - \mathbf{x}_{\text{ref}}),$$

under two scenarios: pure stabilization from a perturbed initial condition, and a step command in cart position. Both runs use the `scipy.integrate.solve_ivp` adaptive RK4(5) integrator with tight tolerances.

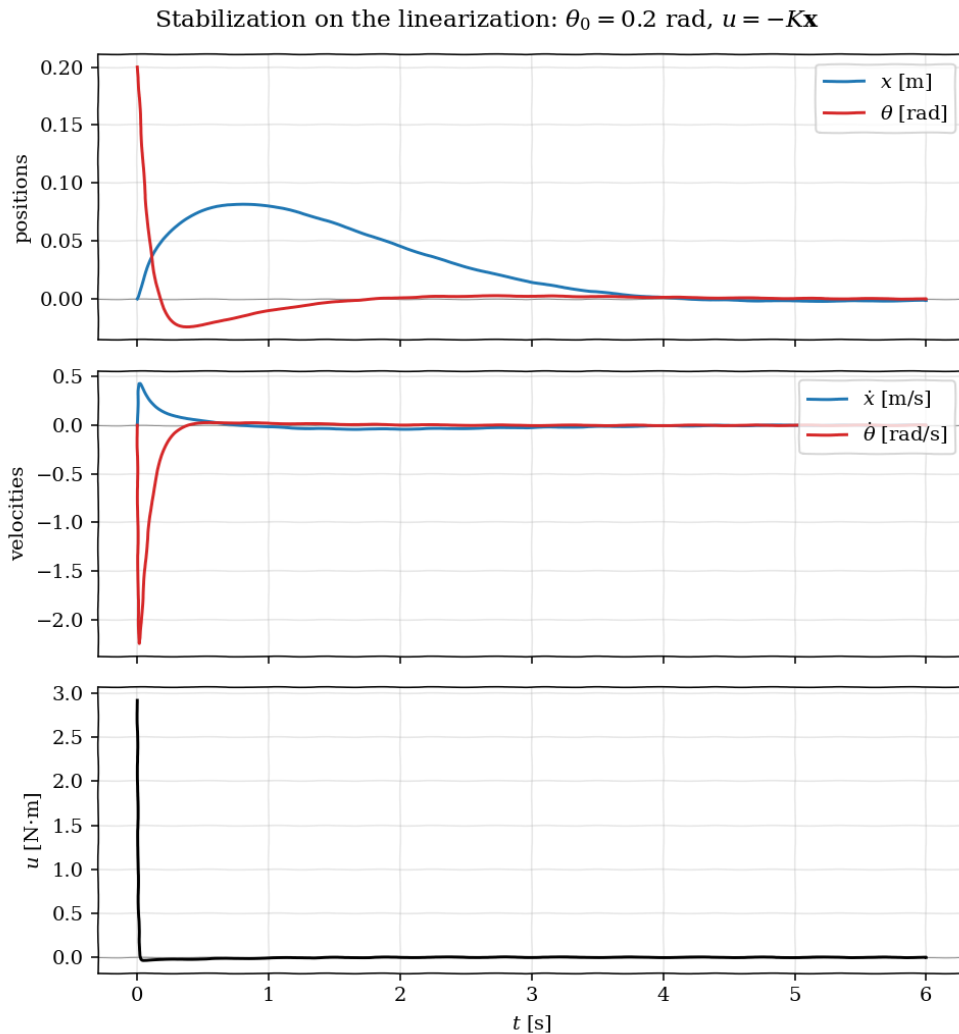
5.2 Stabilization

With $\mathbf{x}_{\text{ref}} = \mathbf{0}$ and initial tilt $\theta_0 = 0.2$ rad ($\approx 11.5^\circ$), the controller pulls the full state back to the upright equilibrium. All four states decay; the motor torque $u(t)$ is the LQR’s online verdict on how much actuation that costs.

5.3 Step Tracking in Cart Position

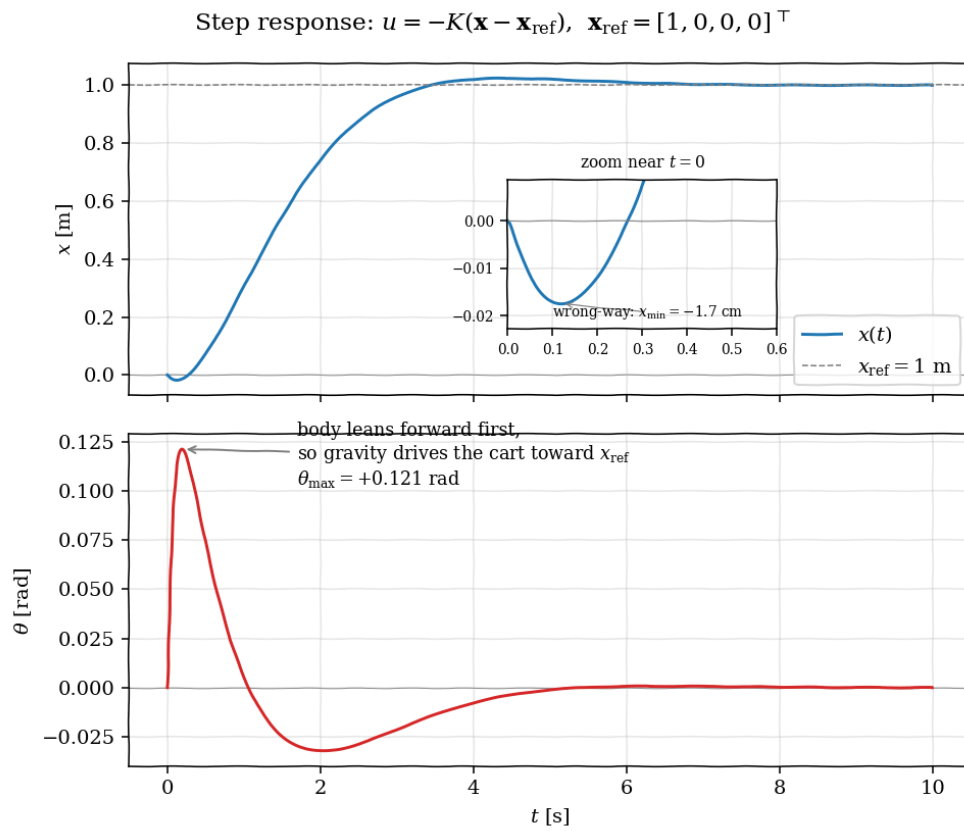
With $\mathbf{x}_{\text{ref}} = [1 \text{ m}, 0, 0, 0]^\top$ commanded from $t = 0$ and zero initial state, the same controller pushes the cart to its new setpoint — without redesign. This works because the upright equilibrium is translation-invariant in x (column zero of A is zero), so $\mathbf{x}_{\text{ref}} = [c, 0, 0, 0]^\top$ is itself a fixed point of the closed loop. A step controller on top of a regulator costs nothing.

The plot shows the non-minimum-phase signature of the underactuated wheeled inverted pendulum: the cart *dips backward* briefly before tracking forward, reaching $x_{\text{min}} = -0.0175$ m at $t = 0.120$ s. The body has to lean *forward* first ($\theta_{\text{max}} = +0.121$ rad at $t = 0.195$ s) so that gravity pulls the cart toward x_{ref} — the body must fall in the right direction before the cart can chase it. The wrong-way dip is small (about 2 cm) but visible in the inset zoom.

Figure 5.1: Linearized closed loop, stabilization from $\theta_0 = 0.2$ rad.

5.4 What This Chapter Does Not Validate

Everything here lives on the linearization. The controller is being asked to handle states that are still small enough for the linearization to be a faithful surrogate of the nonlinear plant; far from upright the agreement breaks down (see the phase portrait in the analysis figures). The natural next step — closing the loop on Drake’s nonlinear simulator and overlaying the two responses — will quantify when the linearization stops being a good predictor of the real plant’s behavior.

Figure 5.2: Linearized closed loop, step in x_{ref} .

Chapter 6

LQR for the 3D Differential-Drive Segway via Drake

6.1 What This Chapter Does

The previous chapters lived on a planar segway: one wheel, one body, one motor. Here we take the same machinery to a real differential-drive balancing robot: two coaxial wheels driven by two independent motors, plus a body that pitches above the axle. The system can translate along the ground, yaw about the vertical, and tilt about the wheel axle, all simultaneously.

A philosophical change comes with it. In the 2D chapter the URDF was a *lumped equivalent*: the wheel’s rotational inertia was absorbed into the cart’s translational mass to dodge URDF’s inability to encode the no-slip rolling constraint. That works for Drake-side LQR design but produces a URDF that cannot be reused for contact-based simulation in MuJoCo — there are no real wheels in it to make contact with.

This chapter takes the more honest route. The URDF is the *physical* model: real wheel links rotating on real axles. The no-slip rolling constraint is then applied *outside* the URDF as a velocity-level projection on top of Drake’s unconstrained mass matrix. The result is the same linearized plant, the same LQR gain, and a single URDF that ports unchanged to MuJoCo (where ground contact plus high friction reproduce no-slip rolling automatically).

6.2 Parameters

The 3D plant treats the body and each wheel as full rigid bodies with their own mass, COM offset, and 3×3 inertia tensor (in CAD frame). This carries enough flexibility to absorb any inertias and offsets that come out of the CAD model, including products of inertia, wheel imbalance (ℓ_x^w or ℓ_z^w nonzero), and left-vs-right asymmetry.

The body’s CAD frame need not be aligned with the equilibrium upright pose: a nonzero ℓ_x^b just means gravity, applied to the body’s CAD-zero pose, would torque it forward. We compensate inside the URDF by setting the pitch joint’s `rpy` to $(0, \alpha_{\text{eq}}, 0)$ with $\alpha_{\text{eq}} = \text{atan2}(-\ell_x^b, \ell_z^b)$, so that $\theta = 0$ in URDF coordinates is the upright equilibrium. The body’s inertial block carries CAD-frame numbers directly; no rotation of the inertia tensor is required.

Shared geometry

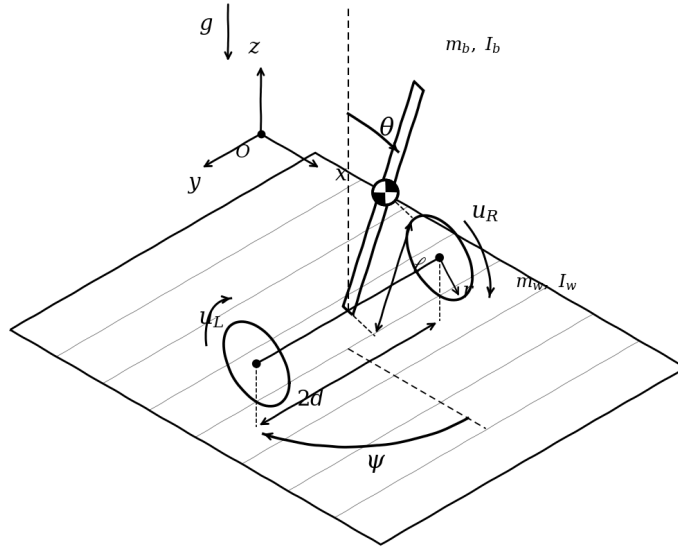


Figure 6.1: 3D differential-drive segway. Two coaxial wheels of radius r , half-track d , motors applying u_L and u_R ; the body pitches by θ about the axle and the chassis yaws by ψ about the vertical.

Symbol	Value	Units	Meaning
g	9.81	m/s ²	gravitational acceleration
r	0.04	m	wheel radius
d	0.1	m	half track-width
α_{eq}	-0	rad	pitch-joint rpy offset for upright equilibrium

Body (CAD frame)

Symbol	Value	Units	Meaning
m_b	2	kg	mass
ℓ_x^b	0	m	COM offset (x, CAD frame)
ℓ_y^b	0	m	COM offset (y, CAD frame)
ℓ_z^b	0.15	m	COM offset (z, CAD frame)
I_{xx}^b	0.015	kg m ²	inertia (xx)
I_{yy}^b	0.015	kg m ²	inertia (yy)
I_{zz}^b	0.0001	kg m ²	inertia (zz)
I_{xy}^b	0	kg m ²	product of inertia (xy)
I_{xz}^b	0	kg m ²	product of inertia (xz)
I_{yz}^b	0	kg m ²	product of inertia (yz)

Left wheel (CAD frame)

Symbol	Value	Units	Meaning
m_{wL}	0.3	kg	mass
ℓ_x^{wL}	0	m	COM offset (x, CAD frame)
ℓ_y^{wL}	0	m	COM offset (y, CAD frame)
ℓ_z^{wL}	0	m	COM offset (z, CAD frame)
I_{xx}^{wL}	0.00012	kg m ²	inertia (xx)
I_{yy}^{wL}	0.00024	kg m ²	inertia (yy)
I_{zz}^{wL}	0.00012	kg m ²	inertia (zz)
I_{xy}^{wL}	0	kg m ²	product of inertia (xy)
I_{xz}^{wL}	0	kg m ²	product of inertia (xz)
I_{yz}^{wL}	0	kg m ²	product of inertia (yz)

Right wheel (CAD frame)

Symbol	Value	Units	Meaning
m_{wR}	0.3	kg	mass
ℓ_x^{wR}	0	m	COM offset (x, CAD frame)
ℓ_y^{wR}	0	m	COM offset (y, CAD frame)
ℓ_z^{wR}	0	m	COM offset (z, CAD frame)
I_{xx}^{wR}	0.00012	kg m ²	inertia (xx)
I_{yy}^{wR}	0.00024	kg m ²	inertia (yy)
I_{zz}^{wR}	0.00012	kg m ²	inertia (zz)
I_{xy}^{wR}	0	kg m ²	product of inertia (xy)
I_{xz}^{wR}	0	kg m ²	product of inertia (xz)
I_{yz}^{wR}	0	kg m ²	product of inertia (yz)

6.3 Generalized Coordinates and the Rolling Constraints

The URDF carries six generalized coordinates,

$$q_{\text{full}} = (x, y, \psi, \varphi_L, \varphi_R, \theta)^\top,$$

where (x, y) is the wheel-axle midpoint on the ground, ψ is the chassis yaw, $\varphi_{L,R}$ are the wheel rotation angles, and θ is the body pitch. The wheels are independent rotational DOF in the URDF — nothing constrains them.

The no-slip rolling constraint says each wheel’s contact-point velocity vanishes. In standard differential-drive form this gives

$$\dot{x} = \frac{r}{2}(\dot{\varphi}_L + \dot{\varphi}_R) \cos \psi, \quad \dot{y} = \frac{r}{2}(\dot{\varphi}_L + \dot{\varphi}_R) \sin \psi, \quad \dot{\psi} = \frac{r}{2d}(\dot{\varphi}_R - \dot{\varphi}_L).$$

These are non-holonomic: they constrain velocities, not configurations. There is no reduction of the configuration space (the robot can still reach every (x, y, ψ)), but the velocity space drops from six dimensions to three.

For LQR design at the upright equilibrium $\psi = 0$, $\theta = 0$, the constraints linearize to constant coefficients:

$$\dot{x} = \frac{r}{2}(\dot{\varphi}_L + \dot{\varphi}_R), \quad \dot{y} = 0, \quad \dot{\psi} = \frac{r}{2d}(\dot{\varphi}_R - \dot{\varphi}_L).$$

The $\dot{y} = 0$ entry is the lateral drift mode — the robot cannot slide sideways at first order. Choosing the three independent velocities $\dot{q}_{\text{ind}} = (\dot{x}, \dot{\psi}, \dot{\theta})^\top$ and solving for the dependent ones, $\dot{\varphi}_{L,R} = (\dot{x} \mp d\dot{\psi})/r$, we get $\dot{q}_{\text{full}} = J \dot{q}_{\text{ind}}$ with the constant 6×3 Jacobian

$$J = \begin{bmatrix} 1.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 1.0000 & 0.0000 \\ 25.0000 & -2.5000 & 0.0000 \\ 25.0000 & 2.5000 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 \end{bmatrix}.$$

6.4 What Drake Returns at the Upright Equilibrium

Drake’s `MultibodyPlant`, parsing `output/segway_3d.urdf` and evaluated at $q^* = 0$, $\dot{q}^* = 0$, gives the unconstrained 6×6 mass matrix and the gravity Jacobian. The mass matrix is block-structured by the URDF’s joint order $(x, y, \psi, \varphi_L, \varphi_R, \theta)$:

$$M_{\text{free}}(q^*) = \begin{bmatrix} 2.6000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.3000 \\ 0.0000 & 2.6000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0063 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0002 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0002 & 0.0000 \\ 0.3000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0600 \end{bmatrix},$$

and the only nonzero entry of $\partial\tau_g/\partial q$ is on the (θ, θ) position — the destabilizing “anti-spring” $+m_b g \ell$ that drives the body away from upright once it tilts:

$$\left. \frac{\partial \tau_g}{\partial q} \right|_{q^*} = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 2.9430 \end{bmatrix}.$$

Note that the wheels appear in M_{free} purely through their own inertia tensors on the wheel-link diagonal blocks — there is no rolling-coupling into the cart’s translational mass. That coupling is what the projection adds back in.

6.5 Projection to the Constrained Dynamics

With $\dot{q}_{\text{full}} = J \dot{q}_{\text{ind}}$, the kinetic energy $T = \frac{1}{2} \dot{q}_{\text{full}}^\top M_{\text{free}} \dot{q}_{\text{full}}$ becomes $T = \frac{1}{2} \dot{q}_{\text{ind}}^\top (J^\top M_{\text{free}} J) \dot{q}_{\text{ind}}$, so the projected mass matrix is just

$$M = J^\top M_{\text{free}} J = \begin{bmatrix} 2.9000 & 0.0000 & 0.3000 \\ 0.0000 & 0.0093 & 0.0000 \\ 0.3000 & 0.0000 & 0.0600 \end{bmatrix}.$$

This is the *same* answer one would get by lumping the wheels’ rolling kinetic energies into the chassis-translation and yaw inertias by hand — the two routes are mathematically identical, but here the URDF stays physical. In the symmetric special case (axisymmetric body about its z -axis, axisymmetric wheels, identical L/R), the projected entries reduce to closed forms:

1. $M_{11} = 2m_w + m_b + 2I_{yy}^w/r^2$ is the forward effective mass. The $2I_{yy}^w/r^2$ term *does* appear, even though no such term lives in M_{free} — the projection delivers it.
2. M_{22} is the effective yaw inertia, $I_{zz}^b + 2(m_w + I_{yy}^w/r^2)d^2 + 2I_{zz}^w$: each wheel contributes its translation at radius d , its rolling KE scaled by d^2/r^2 , and its in-plane spin.
3. The forward and yaw subsystems decouple at the upright equilibrium: $M_{12} = M_{23} = 0$. Only the cart-pole coupling $M_{13} = m_b \ell_z^b$ remains, as in 2D.

Inertial asymmetries — products of inertia, lateral COM offsets, unequal L/R wheels — generally make M_{12} and M_{23} nonzero and couple the forward and yaw subsystems. The LQR pipeline below handles either case identically; the modal decoupling described in §6.8 is what gets sacrificed when the symmetry breaks.

The same projection applies to the gravity Jacobian and to the actuation map. Each wheel motor applies $+u_i$ to its wheel and an equal-and-opposite reaction torque $-u_i$ to the body about the pitch axis (Newton 3rd law), so the URDF generalized-force input map is

$$B_{\text{free}} = \begin{bmatrix} 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 1.0000 & 0.0000 \\ 0.0000 & 1.0000 \\ -1.0000 & -1.0000 \end{bmatrix},$$

with the bottom-right $-1, -1$ row carrying the motor reactions onto the body. Projecting,

$$J^\top \left(\frac{\partial \tau_g}{\partial q} \right) J = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 2.9430 \end{bmatrix},$$

$$B_{\text{in}} \equiv J^\top B_{\text{free}} = \begin{bmatrix} 25.0000 & 25.0000 \\ -2.5000 & 2.5000 \\ -1.0000 & -1.0000 \end{bmatrix}.$$

B_{in} has the structure one expects of a differential drive: both motors push the cart forward ($1/r$), differ in their effect on yaw ($\mp d/r$), and equally react against the body in pitch (-1).

6.6 State-Space Linearization

With $x_{\text{state}} = (x, \psi, \theta, \dot{x}, \dot{\psi}, \dot{\theta})^\top$ and at the upright equilibrium $\partial(C\dot{q})/\partial\dot{q} = 0$ (quadratic in $\dot{q}$), the partitioned linear plant is

$$A = \begin{bmatrix} 0 & I_3 \\ M^{-1}J^\top(\partial\tau_g/\partial q)J & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0_{3 \times 2} \\ M^{-1}B_{\text{in}} \end{bmatrix}.$$

Substituting the projected blocks above:

$$A = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 & 1.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 1.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 1.0000 \\ 0.0000 & 0.0000 & -10.5107 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 101.6036 & 0.0000 & 0.0000 & 0.0000 \end{bmatrix},$$

$$B = \begin{bmatrix} 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 21.4286 & 21.4286 \\ -267.6659 & 267.6659 \\ -123.8095 & -123.8095 \end{bmatrix}.$$

The open-loop spectrum of A shows the expected structure — four zero modes (rigid translation

in x , rigid yaw in ψ , and the two integrators that turn $\dot{x}, \dot{\psi}$ into positions) and a real saddle pair from the inverted pendulum:

$\lambda(A)$
0.0000
0.0000
0.0000
0.0000
10.0799
-10.0799

6.7 LQR Design

The cost weights follow Bryson’s rule: $Q_{ii} = 1/(\text{state}_i^{\max})^2$ and $R_{jj} = 1/u_{\max}^2$ *per motor*, so the bound applies to each wheel torque independently.

Symbol	Value	Units
$x_{\max}$	0.5	m
$\psi_{\max}$	0.785	rad
$\theta_{\max}$	0.087	rad
$\dot{x}_{\max}$	1	m/s
$\dot{\psi}_{\max}$	1	rad/s
$\dot{\theta}_{\max}$	1	rad/s
$u_{\max}$	1	N m

Solving the continuous-time algebraic Riccati equation $A^\top S + SA - SBR^{-1}B^\top S + Q = 0$ via `pydrake.systems.controllers.LinearQuadraticRegulator`:

$$K = R^{-1}B^\top S = \begin{bmatrix} -1.4142 & -0.9008 & -10.1587 & -2.0970 & -0.7095 & -1.1343 \\ -1.4142 & 0.9008 & -10.1587 & -2.0970 & 0.7095 & -1.1343 \end{bmatrix}.$$

The two rows are the gains for u_L and u_R . When the body and wheels are left/right symmetric, K ’s two rows agree on the symmetric states $(x, \theta, \dot{x}, \dot{\theta})$ and differ in sign on the antisymmetric states $(\psi, \dot{\psi})$. Inertial asymmetries break this and the two rows diverge; the gain is still optimal for whatever plant Drake assembled, just no longer cleanly modal.

6.8 Decoupling: Modal Form

The symmetry above suggests changing input basis from (u_L, u_R) to the modal pair

$$u_+ = u_L + u_R \quad (\text{forward push}), \quad u_- = u_R - u_L \quad (\text{yaw torque}),$$

which is the linear transform $\begin{bmatrix} u_+ \\ u_- \end{bmatrix} = T \begin{bmatrix} u_L \\ u_R \end{bmatrix}$ with $T = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$. In this basis the LQR gain becomes

$$K_{\text{modal}} = TK = \begin{bmatrix} -2.8284 & 0.0000 & -20.3174 & -4.1940 & 0.0000 & -2.2687 \\ 0.0000 & 1.8015 & 0.0000 & 0.0000 & 1.4190 & 0.0000 \end{bmatrix},$$

and the cleanest fact about the 3D plant falls out, *provided the body and wheels are left/right symmetric*:

- Row u_+ has zero gain on $(\psi, \dot{\psi})$ — the forward push ignores yaw entirely.
- Row u_- has zero gain on $(x, \theta, \dot{x}, \dot{\theta})$ — the yaw torque ignores the cart–pole.

In that case the 3D LQR is, to numerical precision, two independent controllers: the 2D cart–pole driven by u_+ and a single rotational degree of freedom (yaw) driven by u_- . The differential drive’s *symmetry*, not its complexity, is what makes the design clean. When the body or wheels carry asymmetric inertias or COM offsets, the off-block entries pick up small but nonzero values and the modal decomposition becomes approximate rather than exact.

6.9 Closed-Loop Spectrum

The closed-loop poles all sit in the open left half-plane, with the forward cart–pole and yaw modes well-separated by the cost choices above:

$\lambda(A - BK)$
–177.8999
–11.4176
–0.8474 + 0.7082 i
–0.8474 – 0.7082 i
–378.5346
–1.2739

The slowest pair sets the dominant time constant of the closed loop; the fastest poles sit deep in the cart–pole’s high-frequency stabilization, where Bryson’s rule has chosen aggressive gains to keep θ bounded.

6.10 Bridge to MuJoCo

What was bought by the projection step. The URDF in `output/segway_3d.urdf` is a faithful kinematic and inertial description of the robot — two real wheels rotating on real axles, no lumping — so it can be loaded by MuJoCo (or any contact-based simulator) and given a ground plane plus high-friction wheel contacts. The no-slip rolling constraint that we applied analytically here as a $J^\top M_{\text{free}} J$ projection emerges in MuJoCo as a consequence of contact dynamics. The LQR gain K designed above is exactly the controller to deploy on that simulated plant.

What this chapter does not validate, and is the natural next step: the closed loop on a fully nonlinear MuJoCo simulation, with contact forces, finite friction, and large yaw excursions. The projection’s linearization keeps faith with the upright equilibrium; how far that faith carries is an empirical question the simulator can answer.

Appendix A

A Note About Bryson's Rule: Choosing Q and R Without Choosing Numbers

A.1 The setup

The infinite-horizon LQR cost on a linear plant $\dot{x} = Ax + Bu$ is

$$J(x_0) = \int_0^\infty (x^\top Q x + u^\top R u) dt,$$

with $Q \succeq 0$ and $R \succ 0$. The minimizing control is $u = -Kx$ with $K = R^{-1}B^\top S$ and S the stabilizing solution of the continuous-time algebraic Riccati equation. The catch: *somebody* has to pick Q and R before any of that machinery runs, and a naive guess like $Q = I$, $R = I$ is almost always wrong — not because it yields an unstable closed loop (it usually doesn't), but because it silently mixes incompatible units.

Concretely, for our 2D segway the state is

$$x = [x_{\text{pos}}, \theta, \dot{x}_{\text{pos}}, \dot{\theta}]^\top \in (\text{m}, \text{rad}, \text{m/s}, \text{rad/s}),$$

so a literal $Q = I$ would treat one metre of cart drift as costing the same as one radian of body tilt as one metre per second of velocity. Five degrees of tilt (≈ 0.087 rad) is catastrophic; half a metre of cart wander is fine. Equating their costs is not a tuning choice — it is a unit error.

A.2 The rule

Bryson's rule is a recipe for Q and R that turns “what units are my states?” into “what excursion of each variable do I consider just barely acceptable?” Pick, for every state x_i , a positive scale $x_{i,\text{max}}$ representing the largest excursion you are willing to tolerate as “normal.” Do the same for each input u_j : a $u_{j,\text{max}}$ that is the largest control authority you would spend without flinching. Then set

$$Q_{ii} = \frac{1}{x_{i,\text{max}}^2}, \quad R_{jj} = \frac{1}{u_{j,\text{max}}^2}, \quad Q_{ik} = R_{jk} = 0 \text{ for } i \neq k.$$

Q and R are diagonal; off-diagonal terms are not part of the rule.

In our ‘lqr.py’, this is exactly the function ‘bryson_weights’, and the $x_{i,\max}$, $u_{j,\max}$ are the entries of the ‘DESIGN_LIMITS’ dict in ‘config.py’.

A.3 Why this choice is natural

Three properties make Bryson’s rule the canonical first guess.

Dimensional consistency. The integrand $x^\top Qx + u^\top Ru$ must be a single number (cost), so every term in it must be dimensionless. With Bryson weights,

$$x_i^2 Q_{ii} = \left(\frac{x_i}{x_{i,\max}} \right)^2, \quad u_j^2 R_{jj} = \left(\frac{u_j}{u_{j,\max}} \right)^2,$$

which are squared *ratios* — pure numbers, regardless of whether x_i was in metres or radians. All four entries of our segway Q end up on the same numerical footing despite mixing translational and angular units.

Unit-cost saturation. Set every variable to zero except one, and drive that one to its limit. The integrand becomes exactly 1. In other words, “ x_i saturates its acceptable excursion” contributes unit cost per unit time, and so does “ u_j saturates its acceptable authority.” Q and R are scaled so that their entries answer the same physical question, in the same units.

Tuning by physics, not numerology. Once weights are chosen this way, retuning means revisiting the design *limits*, not the matrix entries. Cutting $\theta_{\max}$ in half quadruples $Q_{\theta\theta}$ and the controller’s aversion to lean. This is far easier to reason about than nudging matrix entries until simulations “look right.”

A.4 Why squared, not linear

Why $1/x_{\max}^2$ rather than $1/x_{\max}$? Because the cost is *quadratic* in x . We want a single scalar knob $x_{\max}$ to mean “the value of x at which the integrand has unit magnitude”:

$$Q_{ii} x_i^2 = 1 \quad \text{when} \quad x_i = x_{i,\max}.$$

Solving forces $Q_{ii} = 1/x_{i,\max}^2$. A linear $1/x_{\max}$ would be dimensionally incorrect (units of $1/x$, not $1/x^2$) and would make the saturation point depend on the size of $x_{\max}$.

A.5 What the rule does *not* do

A few honest limitations are worth being explicit about, since Bryson’s rule is a starting point, not a final answer.

It is a heuristic, not an optimum. Nothing about the Riccati equation singles this Q and R out. Two engineers with the same plant and the same design limits will produce the same Bryson weights, but a different pair of engineers with different opinions about “acceptable excursion” will produce different controllers — both correct.

It ignores cross-coupling. Real systems frequently care about *products* of state errors — e.g., “lean is fine if the cart is already moving in the right direction.” Such preferences live in off-diagonal entries of Q , which Bryson’s rule sets to zero. Adding them is a tuning step beyond the rule.

It says nothing about closed-loop bandwidth or robustness. The resulting eigenvalues of $A - BK$ are whatever they are; if they sit too close to the imaginary axis or too far into the left-half plane, the remedy is to revisit the limits, not to abandon the rule.

It assumes the limits are achievable. If $u_{\max}$ is much smaller than what the plant physically needs to keep x inside $x_{\max}$, the controller will saturate in practice. Bryson’s rule does not check this — it just produces a self-consistent cost.

A.6 Application: the 2D segway

For the 2D segway, the design limits in ‘config.py’ are

$$x_{\max} = 0.5 \text{ m}, \quad \theta_{\max} = 0.087 \text{ rad}, \quad \dot{x}_{\max} = 1 \text{ m/s}, \quad \dot{\theta}_{\max} = 1 \text{ rad/s}, \quad u_{\max} = 1 \text{ N m}.$$

Plugging into the rule gives

$$Q = \text{diag}\left(\frac{1}{0.5^2}, \frac{1}{0.087^2}, \frac{1}{1^2}, \frac{1}{1^2}\right) \approx \text{diag}(4, 132, 1, 1), \quad R = [1].$$

The lopsided ratio of $Q_{\theta\theta}$ to the others is not a bug: it reflects the physics. Tilt has a tight tolerance ($\theta_{\max}$ is small), so its weight is large, and the controller is correspondingly intolerant of even small lean angles. Cart position has a generous tolerance ($x_{\max}$ is large), so its weight is small, and the controller allows the wheel to wander while it nurses the body upright. This is the right priority for an inverted pendulum, and we did not have to choose it directly — it followed from telling Bryson’s rule what “acceptable” meant.

A.7 Connecting back

The numerical Q and R above are exactly what ‘lqr.py’ feeds into ‘scipy.linalg.solve_continuous_are’ to produce the gain K that appears in ‘output/ch03_lqr.tex’. The matrices that look opaque on the page — a diagonal of 4, 132, 1, 1 — are not magic numbers: they are five physically meaningful limits run through one line of arithmetic. That is the whole point of the rule.

Appendix B

From the Lagrangian to $C(q, \dot{q})$: Why Christoffel Symbols Appear

B.1 The setup

Suppose we have a mechanical system whose kinetic energy is the canonical quadratic form

$$T = \frac{1}{2} \dot{q}^\top M(q) \dot{q} = \frac{1}{2} \sum_{j,k} M_{jk}(q) \dot{q}_j \dot{q}_k,$$

where $M(q)$ is symmetric and positive definite. The Euler–Lagrange equations applied to $L = T - V$ read

$$\frac{d}{dt} \left(\frac{\partial T}{\partial \dot{q}_i} \right) - \frac{\partial T}{\partial q_i} = -\frac{\partial V}{\partial q_i} + Q_i.$$

Our goal is to evaluate the left-hand side and watch it organise itself into the manipulator-form pieces $M(q)\ddot{q} + C(q, \dot{q})\dot{q}$ — and to discover, along the way, that the matrix C is built out of Christoffel symbols not by convention but by algebraic necessity.

B.2 The derivation, line by line

Step 1. Compute $\partial T / \partial \dot{q}_i$.

Differentiating the quadratic form with respect to $\dot{q}_i$ and using the symmetry $M_{ij} = M_{ji}$:

$$\frac{\partial T}{\partial \dot{q}_i} = \frac{1}{2} \sum_{j,k} M_{jk} (\delta_{ji} \dot{q}_k + \dot{q}_j \delta_{ki}) = \frac{1}{2} \left(\sum_k M_{ik} \dot{q}_k + \sum_j M_{ji} \dot{q}_j \right) = \sum_j M_{ij} \dot{q}_j.$$

Step 2. Differentiate in time.

M_{ij} depends on t only through $q(t)$, so the chain rule gives $dM_{ij}/dt = \sum_k (\partial M_{ij} / \partial q_k) \dot{q}_k$, and

$$\frac{d}{dt} \left(\frac{\partial T}{\partial \dot{q}_i} \right) = \sum_j M_{ij} \ddot{q}_j + \sum_{j,k} \frac{\partial M_{ij}}{\partial q_k} \dot{q}_k \dot{q}_j.$$

The first sum is the inertia term we expected ($M\ddot{q}$); the second is quadratic in velocity — the trace of the configuration-dependence of M .

Step 3. Compute $\partial T / \partial q_i$.

Only M_{jk} depends on q , so

$$\frac{\partial T}{\partial q_i} = \frac{1}{2} \sum_{j,k} \frac{\partial M_{jk}}{\partial q_i} \dot{q}_j \dot{q}_k.$$

Step 4. Subtract.

$$\frac{d}{dt} \left(\frac{\partial T}{\partial \dot{q}_i} \right) - \frac{\partial T}{\partial q_i} = \sum_j M_{ij} \ddot{q}_j + \sum_{j,k} \left[\frac{\partial M_{ij}}{\partial q_k} - \frac{1}{2} \frac{\partial M_{jk}}{\partial q_i} \right] \dot{q}_j \dot{q}_k.$$

The bracket is asymmetric in (j, k) . The next step turns that into a feature, not a bug.

B.3 The symmetrization trick

The bracketed coefficient in step 4 is multiplied by $\dot{q}_j \dot{q}_k$, which is *symmetric* in (j, k) . Any anti-symmetric part of the coefficient is annihilated when summed against a symmetric tensor, so we may freely replace the coefficient by its symmetric part without changing the sum. In particular,

$$\sum_{j,k} \frac{\partial M_{ij}}{\partial q_k} \dot{q}_j \dot{q}_k = \sum_{j,k} \frac{1}{2} \left(\frac{\partial M_{ij}}{\partial q_k} + \frac{\partial M_{ik}}{\partial q_j} \right) \dot{q}_j \dot{q}_k,$$

because relabelling $j \leftrightarrow k$ in the right-hand sum reproduces the left-hand sum. Substituting back into step 4 collapses the three partial-derivative terms into a single bracket:

$$\frac{d}{dt} \left(\frac{\partial T}{\partial \dot{q}_i} \right) - \frac{\partial T}{\partial q_i} = \sum_j M_{ij} \ddot{q}_j + \sum_{j,k} \underbrace{\frac{1}{2} \left[\frac{\partial M_{ij}}{\partial q_k} + \frac{\partial M_{ik}}{\partial q_j} - \frac{\partial M_{jk}}{\partial q_i} \right]}_{\Gamma_{ijk}} \dot{q}_j \dot{q}_k.$$

The bracketed quantity is the **Christoffel symbol of the first kind** Γ_{ijk} for the metric $M(q)$. We did not impose Christoffel symbols on the problem — they fell out of the algebra of its own accord.

B.4 Reading off the Coriolis matrix

To express the velocity-quadratic piece as $C(q, \dot{q})\dot{q}$, we factor out one velocity:

$$\sum_{j,k} \Gamma_{ijk} \dot{q}_j \dot{q}_k = \sum_j \underbrace{\left(\sum_k \Gamma_{ijk} \dot{q}_k \right)}_{= C_{ij}(q, \dot{q})} \dot{q}_j.$$

This gives the standard formula

$$C_{ij}(q, \dot{q}) = \frac{1}{2} \sum_k \left(\frac{\partial M_{ij}}{\partial \dot{q}_k} + \frac{\partial M_{ik}}{\partial \dot{q}_j} - \frac{\partial M_{jk}}{\partial \dot{q}_i} \right) \dot{q}_k$$

along with the manipulator-form identity

$$\frac{d}{dt} \left(\frac{\partial T}{\partial \dot{q}_i} \right) - \frac{\partial T}{\partial q_i} = \sum_j [M_{ij} \ddot{q}_j + C_{ij} \dot{q}_j].$$

A subtle point: the factorisation $C\dot{q}$ of a quadratic form in $\dot{q}$ is *not unique*. Replacing $C \rightarrow C + S$ with any matrix S satisfying $S\dot{q} = 0$ leaves the product unchanged. But the Christoffel choice above is canonical, for the structural reason explained next.

B.5 Why this C in particular: skew-symmetry of $\dot{M} - 2C$

The Christoffel choice produces a remarkable identity:

$$\dot{M}(q, \dot{q}) - 2C(q, \dot{q}) \quad \text{is skew-symmetric,}$$

that is, $(\dot{M} - 2C)^\top = -(\dot{M} - 2C)$, equivalently $\dot{q}^\top (\dot{M} - 2C) \dot{q} = 0$ for every $\dot{q}$.

This is not a curiosity — it is *equivalent* to energy conservation in the kinetic term. Differentiating $T = \frac{1}{2} \dot{q}^\top M \dot{q}$ and substituting the Tedrake-form manipulator equation $M\ddot{q} + C\dot{q} = \tau_g + Bu$,

$$\frac{dT}{dt} = \dot{q}^\top M \ddot{q} + \frac{1}{2} \dot{q}^\top \dot{M} \dot{q} = \dot{q}^\top (\tau_g + Bu - C\dot{q}) + \frac{1}{2} \dot{q}^\top \dot{M} \dot{q} = \dot{q}^\top (\tau_g + Bu) + \frac{1}{2} \dot{q}^\top (\dot{M} - 2C) \dot{q}.$$

Because $\tau_g = -\partial V / \partial q$, the first piece equals $-dV/dt + \dot{q}^\top Bu$, so

$$\frac{d(T + V)}{dt} = \dot{q}^\top Bu + \frac{1}{2} \dot{q}^\top (\dot{M} - 2C) \dot{q}.$$

For energy to be conserved up to actuation work — i.e. for $d(T + V)/dt$ to equal $\dot{q}^\top Bu$ on *every* trajectory — the bracketed form must vanish identically. That happens iff $\dot{M} - 2C$ is skew-symmetric. The Christoffel formula is precisely the factorisation that delivers it.

B.6 A geometric remark

In Riemannian geometry, a symmetric positive-definite inner product on the tangent spaces of a manifold (a *metric*) determines a unique torsion-free, metric-compatible connection (the *Levi-Civita connection*); the connection's coefficients are the Christoffel symbols of the metric.

For a mechanical system, the configuration space $\mathcal{Q}$ becomes a Riemannian manifold when equipped with the kinetic-energy metric $\langle \dot{q}, \dot{q} \rangle = \dot{q}^\top M(q) \dot{q}$. In that geometric picture, the unforced equation

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} = 0$$

is exactly the *geodesic equation* on $\mathcal{Q}$. The Coriolis matrix is not arbitrary algebra; it is the curvature of the configuration manifold making itself felt. Gravity (τ_g) and actuation (Bu) appear as external forces that bend the system off its natural geodesic path.

B.7 Application: the 2D segway

Our segway has the mass matrix

$$M(q) = \begin{bmatrix} m_w + m_b + I_w/r^2 & m_b \ell \cos \theta \\ m_b \ell \cos \theta & m_b \ell^2 + I_b \end{bmatrix},$$

with $q = [x, \theta]^\top$. Of the four entries, three are constants; only the off-diagonal $M_{12} = M_{21} = m_b \ell \cos \theta$ depends on the configuration, and only on θ . Every Christoffel symbol is therefore zero except for those involving $\partial M_{12}/\partial \theta = -m_b \ell \sin \theta$.

Working through the formula entry by entry, the only surviving piece is

$$C_{12} = \frac{1}{2} \left[\frac{\partial M_{12}}{\partial \theta} + \frac{\partial M_{12}}{\partial \theta} - \frac{\partial M_{22}}{\partial x} \right] \dot{\theta} = \frac{1}{2} [-m_b \ell \sin \theta - m_b \ell \sin \theta - 0] \dot{\theta} = -m_b \ell \sin \theta \dot{\theta}.$$

The other three C_{ij} vanish: C_{11} and C_{22} from the diagonal entries being constant; C_{21} from the two $\partial M_{12}/\partial \theta$ contributions exactly cancelling. The result is

$$C(q, \dot{q}) = \begin{bmatrix} 0 & -m_b \ell \sin \theta \dot{\theta} \\ 0 & 0 \end{bmatrix}.$$

Two observations are worth dwelling on.

Only C_{12} survives. Multiplied into the manipulator equation, $C_{12} \dot{\theta} = -m_b \ell \sin \theta \dot{\theta}^2$ appears in the x -row: a swinging body at nonzero lean exerts a horizontal reaction force on the wheel. This is the centrifugal/Coriolis effect that you feel at your feet when a weight on a string swings near the ground; here the wheel is your feet and the body is the weight.

C_{21} vanishes identically. There is no Coriolis cross term in the θ -equation. Cart motion still couples to pendulum tilt — through the inertial term $M_{21} \ddot{x}$ — but no $\dot{x}$ -driven Coriolis force tilts the body. All of the configuration-dependent velocity coupling is concentrated in a single entry.

B.8 Connecting back

This C is what enters the Tedrake-form manipulator equation

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} = \tau_g(q) + Bu$$

derived in `equations.tex`. And it is what the linearisation in `linearization.tex` appeals to when noting that $\partial(C\dot{q})/\partial \dot{q}$ vanishes at $\dot{q}^* = 0$: that quadratic-in- $\dot{q}$ structure — the property that lets every Coriolis Jacobian die at rest, and lets linearisation about an equilibrium reduce so cleanly to $M(q^*)$, $\partial \tau_g/\partial q$, and B — traces back to the symmetrization step in section 3.

Appendix C

A Note About Pythagorean Cancellation: Why the Body's Kinetic Energy Collapses So Cleanly

C.1 The setup

The body's centre of mass sits at

$$\mathbf{p}_b = \begin{bmatrix} x_b \\ y_b \end{bmatrix} = \begin{bmatrix} x + \ell \sin \theta \\ r + \ell \cos \theta \end{bmatrix}.$$

Differentiating gives the COM velocity:

$$\dot{\mathbf{p}}_b = \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \end{bmatrix} = \begin{bmatrix} \dot{x} + \ell \cos \theta \dot{\theta} \\ -\ell \sin \theta \dot{\theta} \end{bmatrix}.$$

The kinetic energy of the body needs $\|\dot{\mathbf{p}}_b\|^2 = \dot{x}_b^2 + \dot{y}_b^2$.

C.2 The cancellation, line by line

We compute $\dot{x}_b^2 + \dot{y}_b^2$ step by step.

Step 1. Square each component.

$$\begin{aligned} \dot{x}_b^2 &= (\dot{x} + \ell \cos \theta \dot{\theta})^2 = \dot{x}^2 + 2\ell \cos \theta \dot{x} \dot{\theta} + \ell^2 \cos^2 \theta \dot{\theta}^2, \\ \dot{y}_b^2 &= (-\ell \sin \theta \dot{\theta})^2 = \ell^2 \sin^2 \theta \dot{\theta}^2. \end{aligned}$$

Step 2. Add the two squares.

$$\dot{x}_b^2 + \dot{y}_b^2 = \dot{x}^2 + 2\ell \cos \theta \dot{x} \dot{\theta} + \ell^2 (\cos^2 \theta + \sin^2 \theta) \dot{\theta}^2.$$

Step 3. Apply the Pythagorean identity $\cos^2 \theta + \sin^2 \theta = 1$. The combined $\ell^2 \dot{\theta}^2$ coefficient becomes a clean 1, with no remaining θ -dependence:

$$\dot{x}_b^2 + \dot{y}_b^2 = \dot{x}^2 + 2\ell \cos\theta \dot{x}\dot{\theta} + \ell^2 \dot{\theta}^2$$

What *looked* like a messy expression with $\cos^2\theta$ and $\sin^2\theta$ scattered through it collapses into three terms. The angular dependence survives only in the cross term $2\ell \cos\theta \dot{x}\dot{\theta}$, which couples translation and rotation.

C.3 Why this is not algebraic luck

The cancellation has a clean geometric reason.

The body's COM velocity is the sum of two vectors:

$$\dot{\mathbf{p}}_b = \underbrace{\begin{bmatrix} \dot{x} \\ 0 \end{bmatrix}}_{\mathbf{v}_{\text{wheel}}} + \underbrace{\dot{\theta} \ell \begin{bmatrix} \cos\theta \\ -\sin\theta \end{bmatrix}}_{\mathbf{v}_{\text{rot}}}.$$

- $\mathbf{v}_{\text{wheel}}$ is the translation of the axle, dragging the body along.
- $\mathbf{v}_{\text{rot}}$ is the velocity contribution from the body rotating about the axle. It points *tangent* to the circle the COM sweeps as θ changes.

The key observation: the magnitude of $\mathbf{v}_{\text{rot}}$ is *independent* of θ . Indeed,

$$\|\mathbf{v}_{\text{rot}}\| = |\dot{\theta}| \ell \sqrt{\cos^2\theta + \sin^2\theta} = |\dot{\theta}| \ell,$$

which is just *angular rate* $\times$ *radius* — the speed of any point on a rigid body undergoing pure rotation at distance ℓ from the centre. Different values of θ change the *direction* of $\mathbf{v}_{\text{rot}}$ but not its length.

So when we expand $\|\dot{\mathbf{p}}_b\|^2$ via the law of cosines,

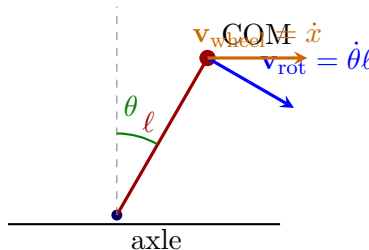
$$\|\mathbf{v}_{\text{wheel}} + \mathbf{v}_{\text{rot}}\|^2 = \|\mathbf{v}_{\text{wheel}}\|^2 + \|\mathbf{v}_{\text{rot}}\|^2 + 2 \mathbf{v}_{\text{wheel}} \cdot \mathbf{v}_{\text{rot}},$$

the first two summands are purely θ -free (they are $\dot{x}^2$ and $\ell^2 \dot{\theta}^2$ respectively). All of the θ -dependence of the problem has been pushed into the dot product, which equals

$$\mathbf{v}_{\text{wheel}} \cdot \mathbf{v}_{\text{rot}} = \dot{x} (\ell \dot{\theta} \cos\theta) = \ell \cos\theta \dot{x} \dot{\theta}.$$

This is exactly the cross term we got before. The Pythagorean identity is the algebraic shadow of a geometric statement: **the speed of a point on a rotating rigid body does not care where the body is pointing.**

C.4 A picture



The blue arrow $\mathbf{v}_{\text{rot}}$ has length $\ell\dot{\theta}$ regardless of how the rod is oriented; only its *direction* rotates with θ . The orange arrow $\mathbf{v}_{\text{wheel}}$ is always horizontal. The total COM velocity is their vector sum, and Pythagoras' theorem (in its inner-product form) gives the square of its length.

C.5 Why this matters in the dynamics

Because the angular dependence has collapsed to a single cross term, the body's kinetic energy is also a tidy three-term expression:

$$T_b = \frac{1}{2}m_b \|\dot{\mathbf{p}}_b\|^2 + \frac{1}{2}I_b\dot{\theta}^2 = \frac{1}{2}m_b \dot{x}^2 + m_b\ell \cos\theta \dot{x}\dot{\theta} + \frac{1}{2}(m_b\ell^2 + I_b)\dot{\theta}^2.$$

The three coefficients above will reappear unchanged as the entries of the mass matrix:

$$M(q) = \begin{bmatrix} m_w + m_b + I_w/r^2 & m_b\ell \cos\theta \\ m_b\ell \cos\theta & m_b\ell^2 + I_b \end{bmatrix}.$$

Note that the configuration dependence of M lives entirely in the off-diagonal $\cos\theta$ — which is exactly the dot-product term $\mathbf{v}_{\text{wheel}} \cdot \mathbf{v}_{\text{rot}}$ from the geometric picture. The diagonal entries are constant precisely because of the Pythagorean cancellation.