# Simulating Solid and Hollow Rigid Bodies on Inclined Planes

Nishanth Shubhakar Mandala, Kevin Bojia Zhang, Justin Liu
New York University
nishanth.mandala@nyu.edu, kbz2005@nyu.edu, jjl9909@nyu.edu

May 3, 2025

## Contents

## 1. Introduction

Rolling motion is a classic example of how different physical forces work together. Even though it looks simple, a body rolling down a slope involves gravity, friction, rotation, and energy loss all at the same time [3]. Understanding how these forces interact helps build a stronger intuition for how real objects move.

In this project, we simulate the motion of different objects—solid spheres, hollow spheres, and cylinders—rolling down an inclined plane. Each object's motion depends not only on its mass and size, but also on how that mass is distributed, which affects how easily it can spin [3]. As the objects move downhill, we track how their energy changes: how potential energy turns into kinetic energy, how friction slows them down, and how some energy is lost over time.

We also look closely at how rotational and translational motion are connected. For an object to roll without slipping, its spin and forward movement have to match a specific relationship [3]. By simulating these behaviors and comparing the results to what physics predicts, this project explores how even everyday motions reveal important ideas about forces, energy, and motion.

## 2. Theoretical Background

Before setting up the simulation, it's important to understand the basic physics principles that guide how the objects behave.

The starting point is realizing that each object is treated as a rigid body. A rigid body is an idealized object that doesn't deform as it moves, meaning that the distances between all its points stay constant over time [2]. Assuming rigidity simplifies the simulation because we can focus on motion without worrying about bending or warping.

Gravity plays a central role by providing a constant downward force that pulls each object down the slope. In this model, gravity points in the negative Z-direction at a uniform rate, consistent with conditions near Earth's surface. Because the slope itself is inclined at a fixed angle, gravity's pull has two effects: it presses the object against the slope and also causes it to accelerate downhill [6].

To handle collisions between the objects and the slope, the simulation uses an impulse-based plane collision model. When an object makes contact with the slope, it checks whether the object has penetrated into the surface. If so, a correction is applied to push it back out, following a collision response based on conservation of momentum and energy loss through the coefficient of restitution. This coefficient controls how "bouncy" the collision is — a

value of 1 means a perfectly elastic bounce, while a value less than 1 means some energy is lost during impact [4]. We use a value of 1 to represent perfectly elastic collisions; we do not want to remove any energy during the collisions and ensure the non-slipping condition is met [5].

Another key element is friction. Friction at the point of contact prevents the object from sliding freely and instead causes it to roll. This static friction force also applies a torque, spinning up the object as it moves downhill. If the object's center-of-mass speed and rotational speed satisfy the condition $v = R\omega$ (where $R$ is radius and $\omega$ is angular velocity), the object is said to be rolling without slipping. Otherwise, some slipping occurs, and energy is dissipated as heat [5].

The simulation accounts for both translational and rotational kinetic energy. Translational kinetic energy comes from the forward motion of the center of mass, while rotational kinetic energy comes from the object's spinning motion around its center. Potential energy, based on height above the ground, steadily converts into these two forms as the object rolls downhill [5].

Energy conservation provides an important check on the simulation's accuracy. Ideally, if there were no friction or energy loss, the sum of potential, translational, and rotational energy would remain constant [5]. In this model, friction introduced expected energy losses, which was carefully tracked and compared against total mechanical energy to validate the realism of the results.

Different types of objects—solid spheres, hollow spheres, solid cylinders, and hollow cylinders—are treated by assigning different moments of inertia based on how their mass is distributed. For example, a hollow sphere has a higher moment of inertia than a solid sphere of the same size and mass, making it harder to spin up and influencing how fast it rolls [4].

Together, these physical models—rigid body motion, gravitational acceleration, frictional torque, collision response, and energy accounting—allow the simulation to realistically capture the behavior of rolling objects on an inclined surface. By observing how these forces and motions interact, we gain deeper intuition into the mechanics underlying everyday experiences like rolling balls, tires, and wheels.[1].

[1]Please note that the explanations and derivations provided in the Appendix are pulled from a collection of academic literature and do not represent novelty or rigor, but a representation optimized and simplified for the sake of simulation.

## 3. Governing Equations

The simulation of rolling bodies on an inclined slope with collisions and friction is governed by the following set of mechanical equations.

### 3.1 Time Evolution: Symplectic Euler Integration

To update the velocity $V$ and position $S$ of a body under constant gravitational acceleration $g$, we use the semi-implicit (symplectic) Euler method (See Appendix 9):

$$V := U + g \cdot \Delta t \tag{1}$$

$$S := S + V \cdot \Delta t \tag{2}$$

Where:

- $U$ is the velocity before gravity is applied.
- $g$ is the acceleration due to gravity, given by 9.8 $\mathrm{m/s^2}$.
- $\Delta t$ is the time step.
- $S$ is the position.

### 3.2 Slope Surface Detection via Plane Equation

The slope surface is defined by a plane (See Appendix 9.2):

$$\vec{n} \cdot \vec{r} + d = 0 \tag{3}$$

Where:

- $\vec{n}$ is the unit normal vector of the slope plane.
- $\vec{r}$ is the position vector of the center of mass.
- $d$ is the scalar offset from the origin based on plane height.

Contact is detected when:

$$\vec{n} \cdot \vec{r} + d - R < 0$$

## 3.3 Contact Point Velocity

The velocity of the contact point is given by (See Appendix 9.7):

$$\vec{v}_{\text{contact}} = \vec{v}_{\text{cm}} + \vec{\omega} \times \vec{r} \tag{4}$$

Where:

- $\vec{v}_{\text{cm}}$ is the center of mass linear velocity.
- $\vec{\omega}$ is the angular velocity vector.
- $\vec{r}$ is the vector from the center of mass to the contact point.

## 3.4 Normal Collision Impulse

When a rigid body contacts the slope surface, the normal impulse due to collision is given by (See Appendix 9.3):

$$\vec{J_n} := -(1 + e) \cdot v_n \cdot m \tag{5}$$

Where:

- $e$ is the coefficient of restitution.
- $v_n$ is the normal component of the contact point velocity.
- $m$ is the mass of the body.

## 3.5 Tangential (Friction) Impulse

The tangential impulse due to Coulomb friction is bounded by the product of the friction coefficient $\mu$ and the magnitude of the normal impulse (See Appendix 9.4):

$$J_t = \min\left(\mu \cdot |J_n|, \frac{\|\vec{v}_t\|}{k_t}\right) \tag{6}$$

Where:

- $\mu$ is the coefficient of kinetic friction.
- $|J_n|$ is the magnitude of the normal impulse.
- $\vec{v}_t$ is the tangential velocity at the contact point.
- $k_t$ is the effective tangential mass.

## 3.6 Effective Mass for Tangential Impulse

The effective mass $k_t$ accounts for both linear and rotational effects (See Appendix 9.4):

$$k_t = \frac{1}{m} + \frac{\|\vec{r}\|^2 - (\vec{r} \cdot \hat{n})^2}{I} \tag{7}$$

Where:

- $\vec{r}$ is the vector from the center of mass to the contact point.
- $\hat{n}$ is the unit normal of the slope.
- $I$ is the moment of inertia about the center of mass.

## 3.7 Impulse Vector Update

Resolving both impulse vectors, The total impulse vector $\vec{J}$ is:

$$\vec{J} = \vec{J_n} - J_t \cdot \frac{\vec{v_t}}{\|\vec{v_t}\|} \tag{8}$$

Where:

- $\vec{J_n}$ is the normal impulse vector $J_n \hat{n}$.
- $\vec{v_t}$ is the tangential velocity vector at contact.

## 3.8 Momentum and Angular Momentum Updates

After computing the impulse, the body's linear and angular momentum are updated by (See Appendix 9.5):

$$\vec{v} := \vec{v} + \frac{\vec{J}}{m} \tag{9}$$

$$\vec{\omega} := \vec{\omega} + \frac{\vec{r} \times \vec{J}}{I} \tag{10}$$

## 3.9 Energy Accounting

To track conservation and dissipation, the total energy of the system (See Appendix 9.6) is computed as:

$$E_{\text{total}} = \underbrace{\frac{1}{2}mv^2}_{\text{Translational Kinetic}} + \underbrace{\frac{1}{2}I\omega^2}_{\text{Rotational Kinetic}} + \underbrace{mgh}_{\text{Potential Energy}} + \underbrace{E_{\text{friction}}}_{\text{Friction Loss}} \tag{11}$$

Where:

- $E_{\text{total}}$ is the total mechanical energy including losses.
- $m$ is the mass.
- $v$ is the linear speed of the center of mass.
- $I$ is the moment of inertia.
- $\omega$ is the angular speed.
- $h$ is the vertical position of the center of mass.
- $g$ is gravitational acceleration.
- $E_{\text{friction}}$ is the cumulative work done by friction (See Appendix 9.6):

$$E_{\text{friction}} \approx \mu \cdot |J_n| \cdot \|\vec{v}_t\| \cdot \Delta t$$

## 3.10 Angular Momentum of Rigid Bodies

The angular momentum is given by:

$$\vec{L} = I\vec{\omega} \tag{12}$$

## 3.11 Rolling Without Slipping Condition

For rolling without slipping at the contact point (See Appendix 9.7):

$$\vec{v}_{\text{cm}} - \vec{\omega} \times \vec{R} = 0 \tag{13}$$

Where:

- $\vec{v}_{\text{cm}}$ is the velocity of the center of mass.
- $\vec{\omega}$ is angular velocity.
- $\vec{R}$ is the vector from the center to the contact point.

## 4. Numerical Method

We simulate the motion of multiple rigid bodies (spheres and cylinders) rolling down a triangular prism slope under the influence of gravity, surface collisions, and Coulomb friction. Each body evolves according to Newtonian mechanics with both translational and rotational dynamics.

The motion is governed by the following quantities for each body:

- **Position** $\vec{r}(t)$ — the location of the body's center of mass in 3D space.
- **Linear velocity** $\vec{v}(t)$ — the velocity of the center of mass.
- **Angular velocity** $\boldsymbol{\omega}(t)$ — the body's rotational speed about its center of mass.

### 4.1 Symplectic Euler Integration

To numerically solve the time evolution of each body, we discretize time into steps of size $\Delta t$, and apply the **symplectic Euler method**, which updates velocity before position.

Unlike standard Euler integration, symplectic Euler preserves geometric structures such as phase-space volume and is more stable over long simulations, especially in systems with collisions and conservative forces [1].

1. **Velocity Update (Gravity Only)**:

$$\vec{v}^{n+1} = \vec{v}^{n} + \vec{g} \cdot \Delta t$$

**Where:**

- $\vec{v}^{n}$ — velocity of the center of mass at time step $n$
- $\vec{g}$ — gravitational acceleration vector (typically $[0, 0, -9.81]$ m/s$^2$)
- $\Delta t$ — timestep duration
- $\vec{v}^{n+1}$ — updated velocity at the next timestep

2. **Position Update (Using New Velocity)**:

$$\vec{r}^{n+1} = \vec{r}^{n} + \vec{v}^{n+1} \cdot \Delta t$$

**Where:**

- $\vec{r}^n$ — position of the center of mass at time $n$
- $\vec{r}^{n+1}$ — updated position at time $n+1$

3. **Collision Response (If Contact)**:

   When a body contacts the slope surface, we compute the net impulse $\vec{J}$ that corrects for interpenetration, elastic bounce, and friction. The impulse modifies both the translational and rotational velocities:

$$\vec{v}^{n+1} = \vec{v}^{n+1} + \frac{\vec{J}}{m}, \qquad \boldsymbol{\omega}^{n+1} = \boldsymbol{\omega}^{n+1} + \frac{\vec{r_c} \times \vec{J}}{I}$$

   **Where:**

   - $\vec{J}$ — total impulse applied during collision
   - $m$ — mass of the rigid body
   - $\vec{r_c}$ — vector from the center of mass to the contact point
   - $\boldsymbol{\omega}^{n+1}$ — angular velocity after the impulse
   - The first term updates linear motion (Newton's 2nd Law); the second updates rotational motion using torque $\vec{r_c} \times \vec{J}$

**Why Symplectic Euler?** Compared to explicit Euler integration, the symplectic method is better suited to mechanical systems with conservative forces and constraints. By applying the force-based update to velocity before updating position, energy drift over time is significantly reduced. This makes it ideal for long-term simulations of rolling, colliding, and rotating bodies on a slope, where energy and momentum accounting is essential.

## 5. Validation

To ensure that our rigid-body simulator behaves physically and converges numerically, we performed three validation checks: Energy conservation, time-step sensitivity, and slip metric (See Appendix 9.6).

### 5.1 Energy-Conservation

We define each body's total energy as

$$E_{\text{total}}(t) = E_{\text{mech}}(t) + W_{\text{friction}}(t), \tag{14}$$

Figure 1 shows the total system energy for all four bodies, all with $\Delta t = 0.001$. In each case, the curves remain essentially flat, validating the conservation of energy in our simulation.

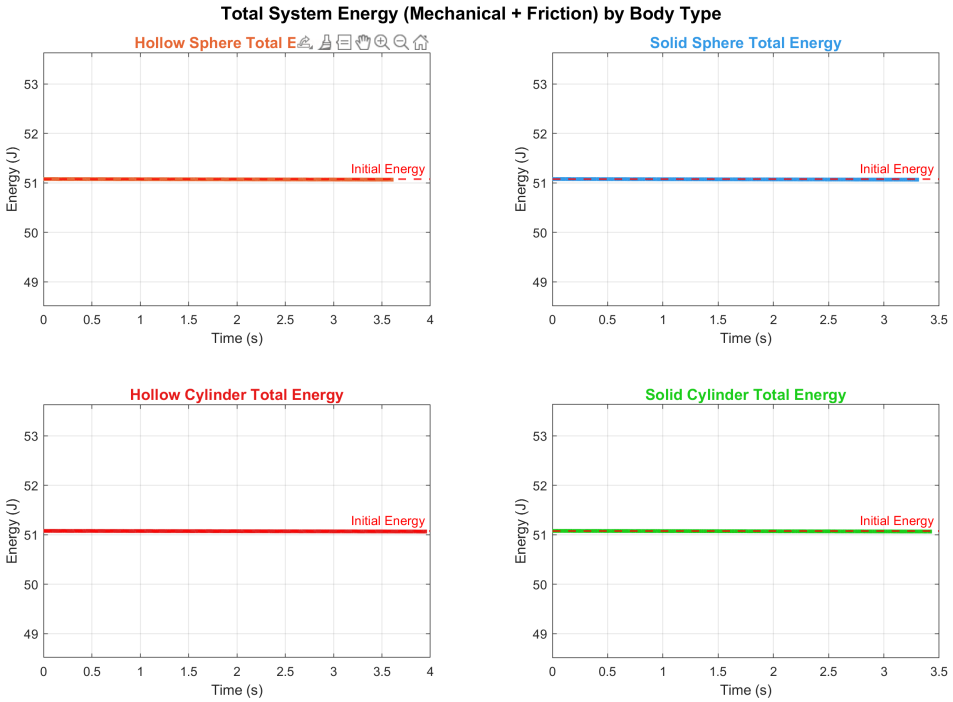


Figure 1: Total system energy vs. time for all four bodies.

## 5.2 Time-Step Sensitivity Study

To verify time-step sensitivity, we reran the same scenario with

$$\Delta t \in \{0.01,\ 0.001,\ 0.00025\}\,,$$

keeping all other parameters identical and checking that the total system energy converges to a single curve as $\Delta t \to 0$.

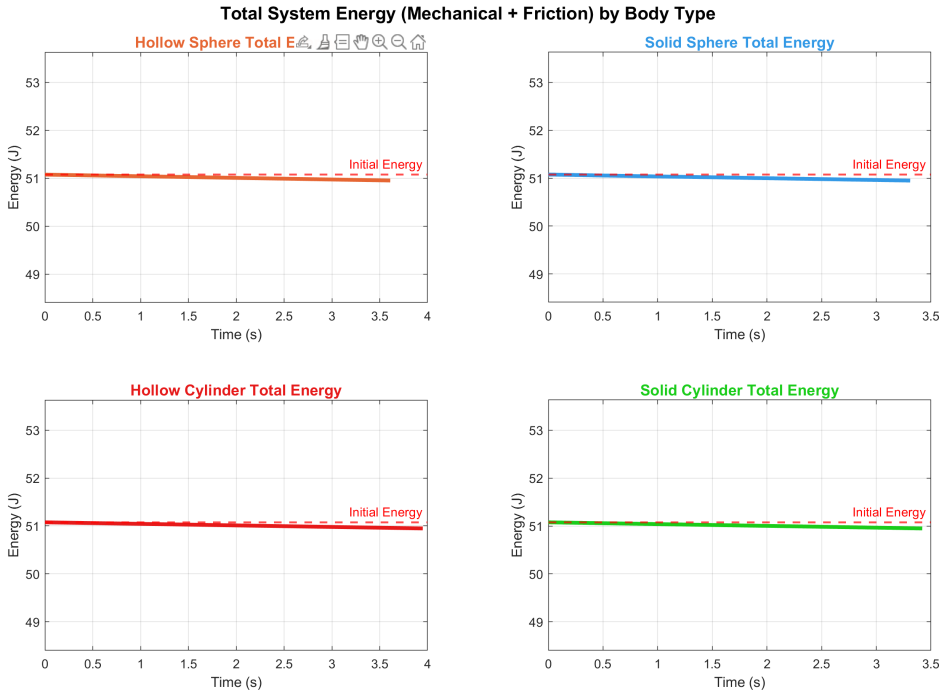


Figure 2: Total system energy vs. time for $\Delta t = 0.01$.

**Figure 2:** With $\Delta t = 0.01$s, the energy curve shows noticeable downward drift, indicating energy loss that can be attributed to integration error.

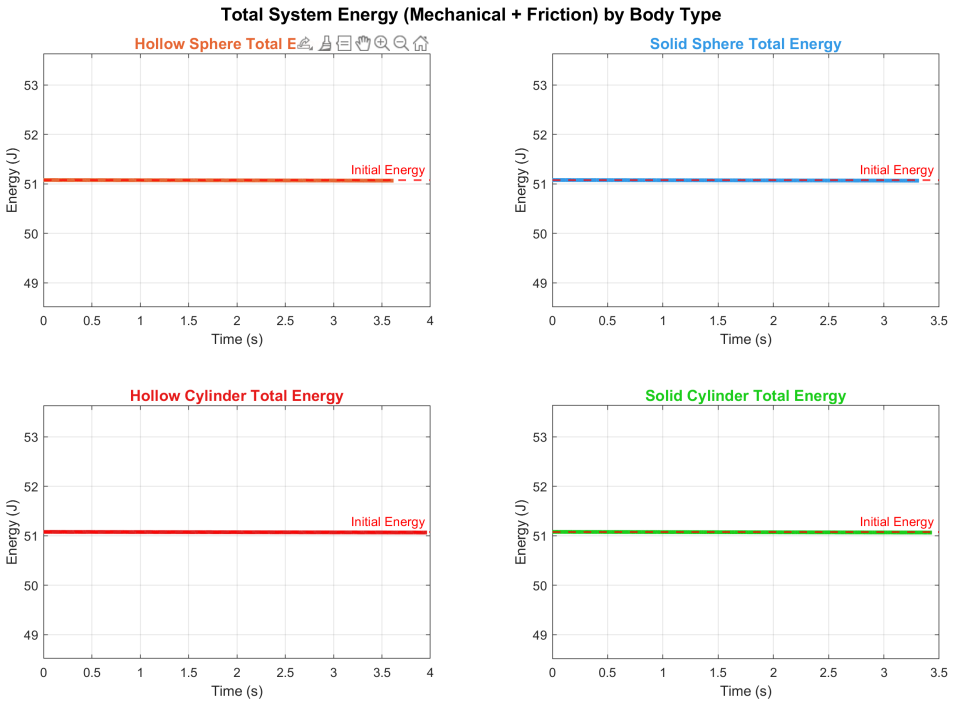


Figure 3: Total system energy vs. time for $\Delta t = 0.001$.

**Figure 3:** At $\Delta t = 0.001$, which is the reference timestep, the energy is conserved.

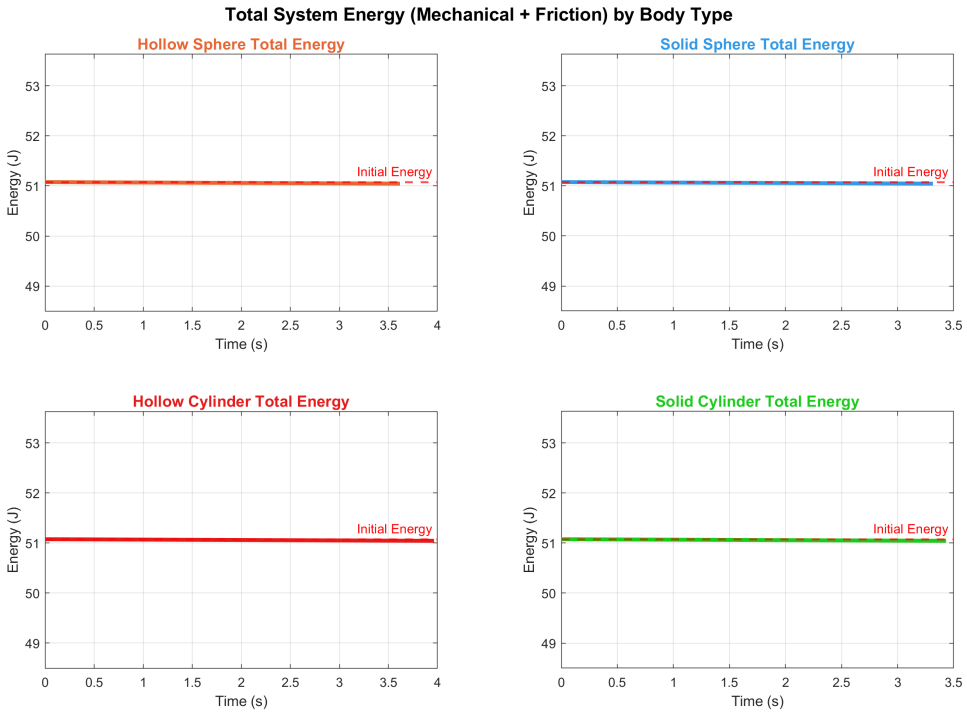


Figure 4: Total system energy vs. time for $\Delta t = 0.00025$.

**Figure 4:** At $\Delta t = 0.00025$s, the total system energy looks nearly identical to the graphs for $\Delta t = 0.001$s, confirming that the timestep is small enough and validating conservation of energy. With a timestep this small, the simulation takes substantially longer to run.

These findings show that it makes the most sense to use $\Delta t = 0.001$s for our simulation, as it maintains a balance between conserving energy but still running the simulation in a reasonable amount of time.

## 5.3 Slip Metric

To assess whether each body rolls without slipping, we define the *slip metric*

$$\Delta v(t) \;=\; \omega(t)\, R \;-\; \|\mathbf{v}(t)\|,$$

where $\omega$ is the body's instantaneous angular speed, $R$ its radius, and $\|\mathbf{v}\|$ its translational speed. In pure rolling, $\omega R = v$ and thus $\Delta v = 0$. Deviations from zero indicate slip or spin-only motion.

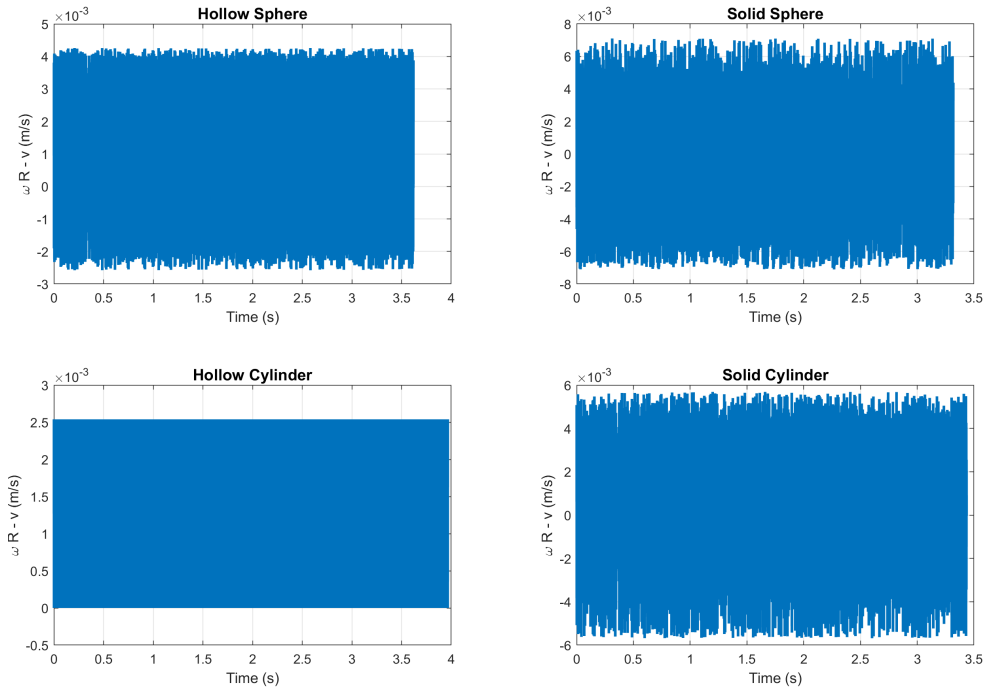


Figure 5: Slip metric $\omega R - v$ vs. time for all four bodies at $\Delta t = 0.001$.

Figure 5 shows that $\Delta v$ remains very close to 0 throughout the full simulation for all bodies, confirming that our friction and collision model enforces nearly pure rolling with minimal slip.

## 6. Implementation and Code Considerations

This section presents a detailed implementation and explanation of our rigid-body slope simulation. The MATLAB code models spheres and cylinders rolling down a triangular-prism wedge under gravity, with collision impulses, Coulomb friction, and energy tracking.

### 6.1 Simulation Constants and Mode Configuration

```
dt          = 0.001;        % timestep (s)
simTime     = 5;              % total duration (s)
g           = [0 0 -9.81];    % gravity vector (m/s)
mu          = 0.30;           % kinetic friction coefficient
coefRest    = 1;              % coefficient of restitution
CYL_LEN_FAC= 0.6;             % cylinder length factor ( = 0.6R)
```

Listing 1: Simulation Constants

**Explanation:** $\Delta t$ determines the temporal resolution of the integrator—smaller values improve accuracy (especially during collisions) at the cost of longer run time. $\mu$ controls how much kinetic energy is dissipated by sliding friction. coefRest sets how elastic collisions are (1 = perfectly elastic, 0 = perfectly inelastic).

### 6.2 Slope Geometry Setup

```
planeH   = 5;                     % height at x=0 (m)
slopeAng = 15*pi/180;             % tilt angle (rad)
prismLen = 10;                    % length in X (m)
halfW    = 1.5;                   % halfwidth in Y (m)
nSlope   = [sin(slopeAng) 0 cos(slopeAng)];
nSlope   = nSlope./norm(nSlope);  % unit normal vector
dSlope   = -planeH * nSlope(3);   % plane offset: nx + d = 0
```

Listing 2: Slope Geometry Initialization

**Explanation:** We define the inclined plane via its normal $n$ and offset $d$. A point $\mathbf{r}$ is below the surface if $n \cdot \mathbf{r} + d < 0$.

## 6.3 Body Initialization

```
R      = 0.20;
lanesY = [-1.125 -0.375 0.375 1.125];
startX = 0;  startZ = planeH + R;
bodies(1) = makeSphere(R,1,[startX lanesY(1) startZ],'hollow');
bodies(2) = makeSphere(R,1,[startX lanesY(2) startZ],'solid');
bodies(3) = makeCylinder(R,1,[startX lanesY(3) startZ],'hollow');
bodies(4) = makeCylinder(R,1,[startX lanesY(4) startZ],'solid');
```

Listing 3: Body Creation at Start Line

**Explanation:** We place four bodies just touching the slope. Each factory sets mass $m$, radius $R$, moment of inertia $I$, initial position, zero linear and angular velocity, and distinguishes solid vs. hollow.

## 6.4 Visualization Setup

```
fig = figure('Color','w','Renderer','opengl');
hold on; grid on; axis equal vis3d;
rotate3d(fig,'on'); view(35,20);
plotPrism(slopeAng,planeH,prismLen,halfW);
lighting gouraud; camlight headlight; material dull;
```

Listing 4: Graphics Initialization

**Explanation:** We draw the wedge, enable 3D rotation, and set lighting to visually distinguish solid vs. hollow bodies.

## 6.5 Diagnostics Preallocation

```
steps         = round(simTime/dt);
numBodies     = numel(bodies);
for k = 1:numBodies
  energy_logs{k}   = zeros(steps,7);  energy_logs{k}(:,7)=1;
  angVel_logs{k}   = zeros(steps,3);
  velocity_logs{k} = zeros(steps,1);
end
reached_end   = zeros(1,numBodies);
friction_work = zeros(1,numBodies);
```

Listing 5: Diagnostics Preallocation

**Explanation:** We reserve arrays for each body to log time, translational KE, rotational KE, potential energy, mechanical energy, friction loss, and a validity flag. Angular and linear speeds are stored separately.

## 6.6 Main Time-Stepping Loop

We split the core loop into four logical stages.

### 6.6.1 Store Pre-collision State

```
for k = 1:numBodies
  pre_vel{k}    = bodies(k).vel;
  pre_angVel{k} = bodies(k).angVel;
end
```

Listing 6: Pre-collision State Storage

**Explanation:** We snapshot velocities before any collisions so that frictional work can be computed from the change in momentum.

### 6.6.2 Symplectic Euler Free Motion

```
for k = 1:numBodies
  bodies(k).vel = bodies(k).vel + dt*g;
  bodies(k).pos = bodies(k).pos + dt*bodies(k).vel;
end
```

Listing 7: Free Motion Integration

**Explanation:** This semi-implicit update (velocity first, then position) is symplectic: it preserves the Hamiltonian structure of gravity and improves long-term energy behavior.

### 6.6.3 Collision Detection & Friction

```
for k = 1:numBodies
  is_on = (bodies(k).pos(1)>=0 && bodies(k).pos(1)<=prismLen && ...
           abs(bodies(k).pos(2))<=halfW && ...
           dot(nSlope,bodies(k).pos)+dSlope-bodies(k).R < 0);
  bodies(k) = collideTop(bodies(k),nSlope,dSlope,prismLen,halfW,coefRest,mu);
  if is_on
    r         = -bodies(k).R * nSlope;
    vcp       = bodies(k).vel + cross(bodies(k).angVel, r);
    v_slide   = vcp - dot(vcp,nSlope)*nSlope;
```

```
    v_slide_m = norm(v_slide);
    delta_v   = bodies(k).vel - pre_vel{k};
    normal_impulse = dot(delta_v,nSlope)*bodies(k).mass;
    normal_force   = abs(normal_impulse)/dt;
    friction_force = mu * normal_force;
    friction_work_step = friction_force * v_slide_m * dt;
    if v_slide_m>1e-6
      friction_work(k) = friction_work(k)+friction_work_step;
    end
  end
end
```

Listing 8: Collision and Friction Handling

**Explanation:** We first test if the body is over the slope. After the plane collision impulse ('collideTop'), we compute sliding velocity at the contact point, approximate normal force via impulse, and accumulate frictional work when sliding occurs.

### 6.6.4 Energy and Velocity Logging

```
currentTime = (step-1)*dt;
for k = 1:numBodies
  angVel_logs{k}(step,:) = bodies(k).angVel;
  velocity_logs{k}(step) = norm(bodies(k).vel);
  % Kinetic and potential energies
  ke_t = 0.5*bodies(k).mass*sum(bodies(k).vel.^2);
  ke_r = 0.5*bodies(k).I*sum(bodies(k).angVel.^2);
  pe   = -bodies(k).mass*g(3)*bodies(k).pos(3);
  energy_logs{k}(step,1:6) = [currentTime ke_t ke_r pe ke_t+ke_r+pe friction_work(k)];
end
```

Listing 9: Energy & Diagnostics Logging

**Explanation:** We record mechanical energies and cumulative friction loss for later analysis, along with angular and translational speeds.

## 6.7 Plotting Results

```
figure('Name','Translational KE','Position',[50,50,800,600],'Color','w');
hold on;
for k=1:numBodies
  idx = energy_logs{k}(:,7)==1;
  plot(energy_logs{k}(idx,1), energy_logs{k}(idx,2),'LineWidth',2);
```

```
end
legend(bodyNames,'Location','best'); grid on; hold off;
```

Listing 10: Translational KE Comparison Plot

**Explanation:** Similar blocks produce plots of rotational KE, potential energy, total mechanical energy, friction loss, angular momentum, velocities, and slip metrics to validate dynamics and energy conservation.

**Code Considerations:**

- *Symplectic Euler:* ensures better long-term energy behavior in conservative systems.
- *Impulse-based collision:* handles restitution and Coulomb friction without stiff ODEs.
- *Preallocation:* critical for MATLAB performance inside large loops.
- *Modular design:* breaking the loop into clear stages aids readability and debugging.
- *Numeric parameters:* $\Delta t$, $\mu$, and e can be tuned for stability vs. realism.

## 7. Results and Discussion

Here we outline the parametric studies for simulating the four rolling objects. The control case for all objects had mass $m = 1.0\,\text{kg}$. We then systematically varied the mass of each object while keeping the others at their control mass. We found that variations in mass did not affect the translational motion: in every case, the bodies reached the bottom of the ramp in the same order: solid sphere, solid cylinder, hollow sphere, hollow cylinder. However, increasing the mass did lead to proportional increases in total system energy, translational kinetic energy, angular momentum, and heat energy.

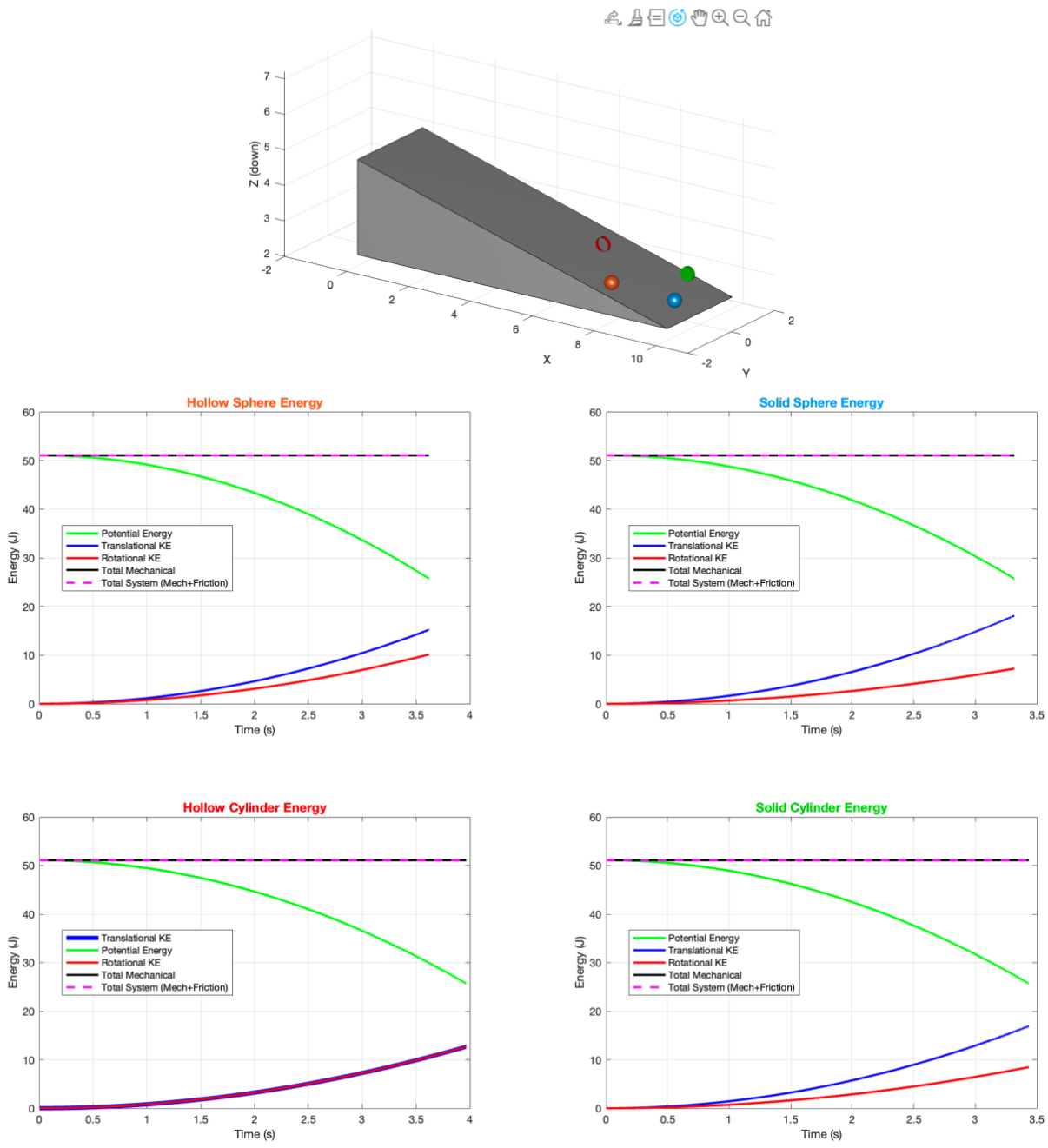


Figure 6: Frame from the simulation near the end of the ramp, with energy traces overlaid for each object (control case, $m = 1.0\,\text{kg}$). The 3D animation is identical for all mass-variation cases, so only the control-case frame is shown.

This frame shows that all four bodies follow identical trajectories down the slope in the control case. The overlaid energy graphs highlight their relative mechanical states just before exiting the ramp. Because total energy is conserved for all cases, proportionally increasing by the mass scaling, the energy graphs are omitted for all varied masses as they don't convey additional information.

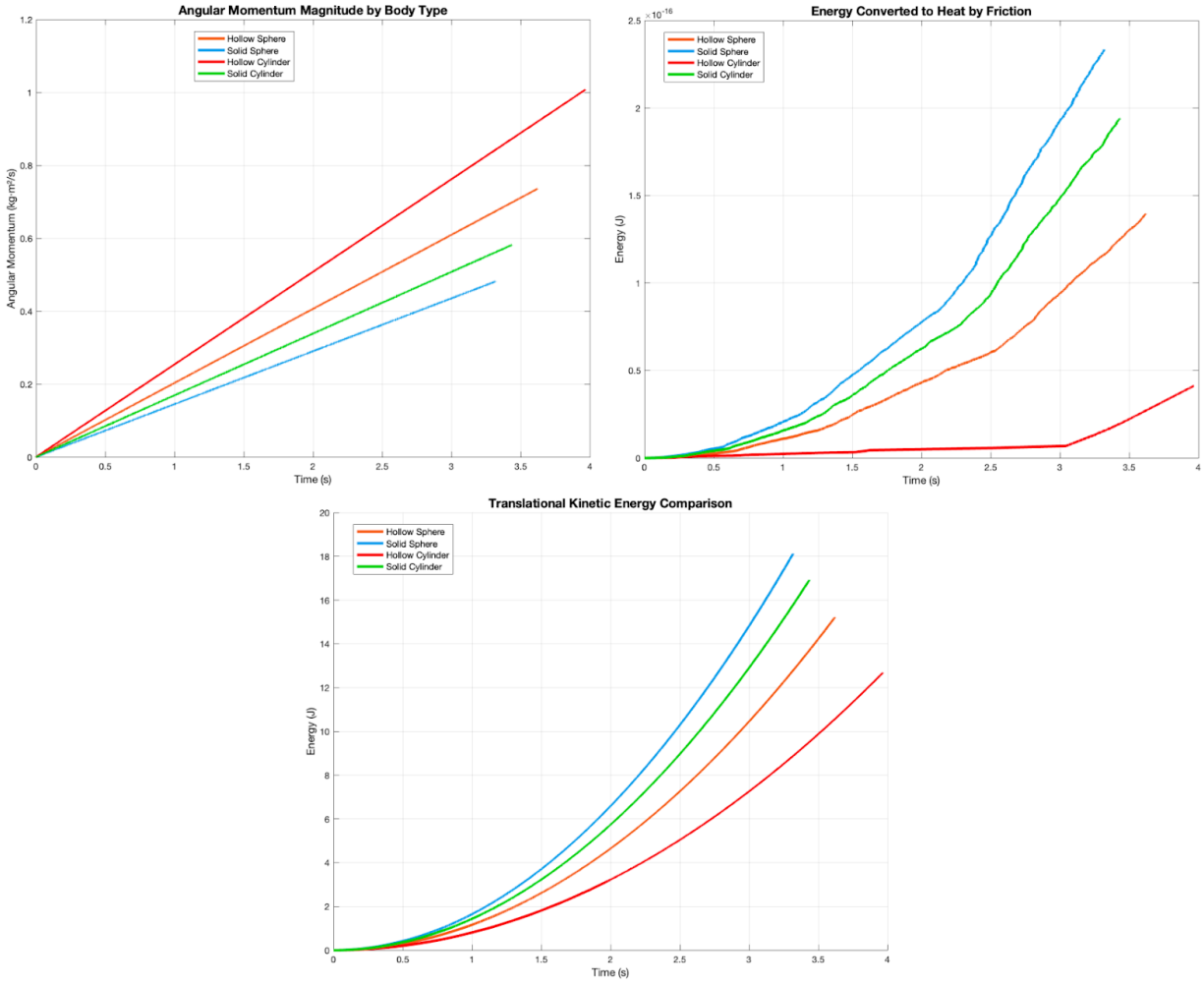


Figure 7: Angular momentum, energy converted to heat, and translational kinetic energy vs. time for all four objects (control case, $m = 1.0\,\text{kg}$).

In this plot, the solid sphere rapidly builds translational kinetic energy and angular momentum, while the hollow cylinder accumulates less heat energy slower than the rest of the bodies due to its larger moment of inertia.

## 7.1 Parametric Studies

For each of the following cases, one object's mass was increased either twofold or tenfold, while all others remained at $m = 1.0\,\text{kg}$. In every scenario, the trajectories and order of arrival were unchanged. Only the energy and momentum-based quantities scaled with mass.

**Solid Sphere,** $m = 2.0\,\textbf{kg}$

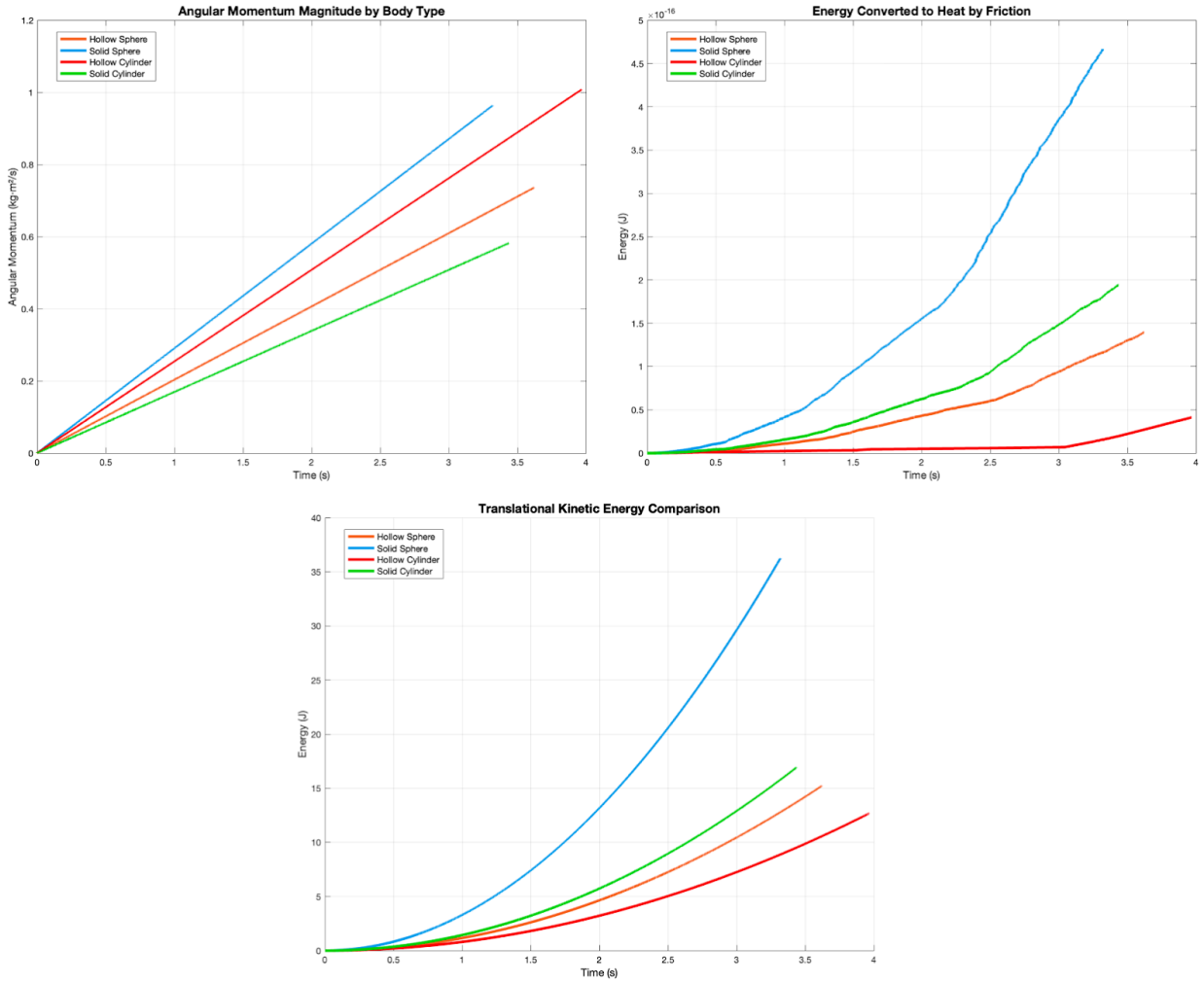


Figure 8: Angular momentum and heat conversion vs. time for solid sphere at $m = 2.0\,\text{kg}$.

Doubling the mass of the solid sphere increases its angular momentum, frictional heat, and kinetic energy roughly by a factor of two.

**Solid Sphere,** $m = 10.0\,\textbf{kg}$

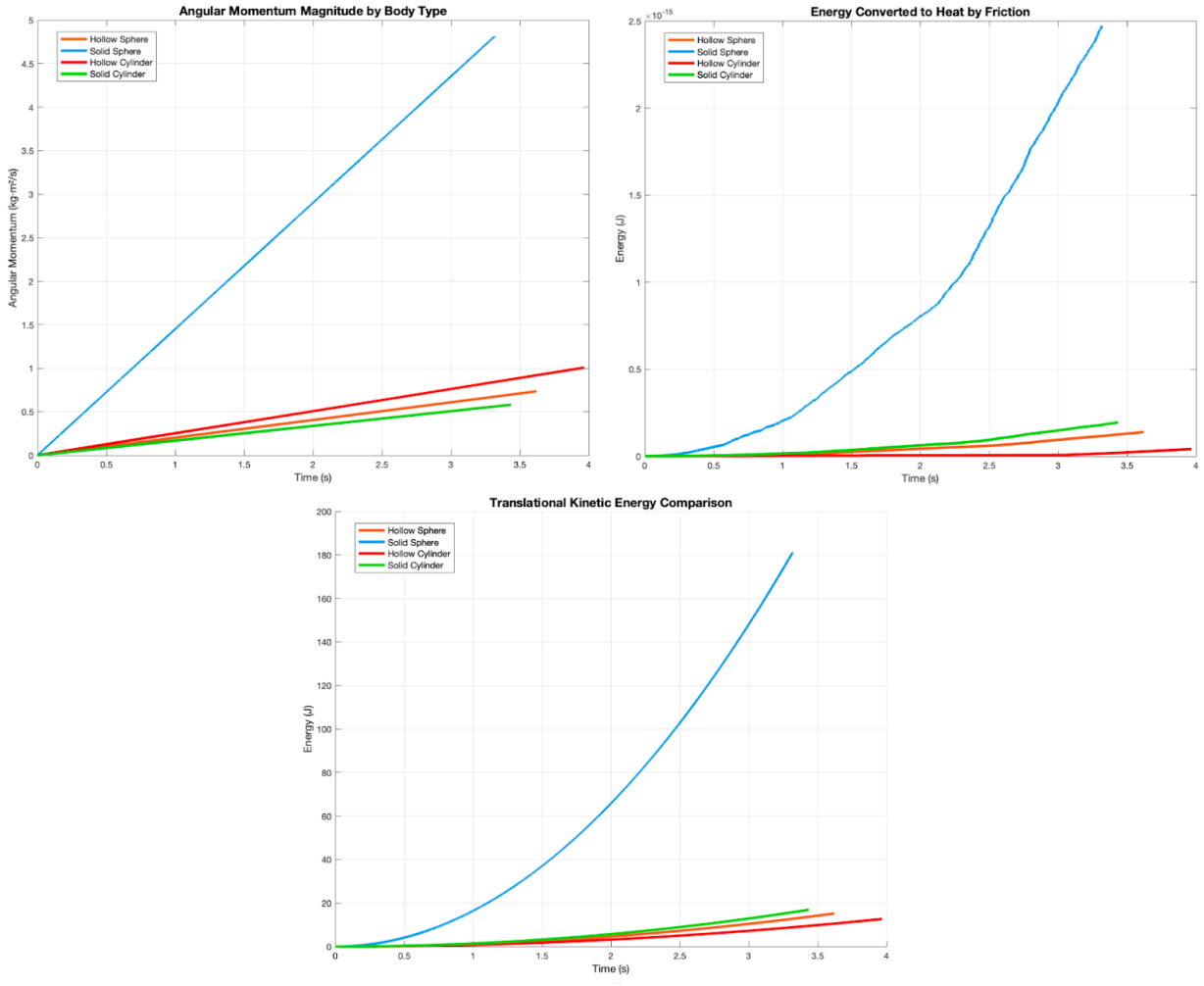


Figure 9: Angular momentum and heat conversion vs. time for solid sphere at $m = 10.0\,\text{kg}$.

At ten times the mass, the solid sphere exhibits a ten-fold increase in the quantities.

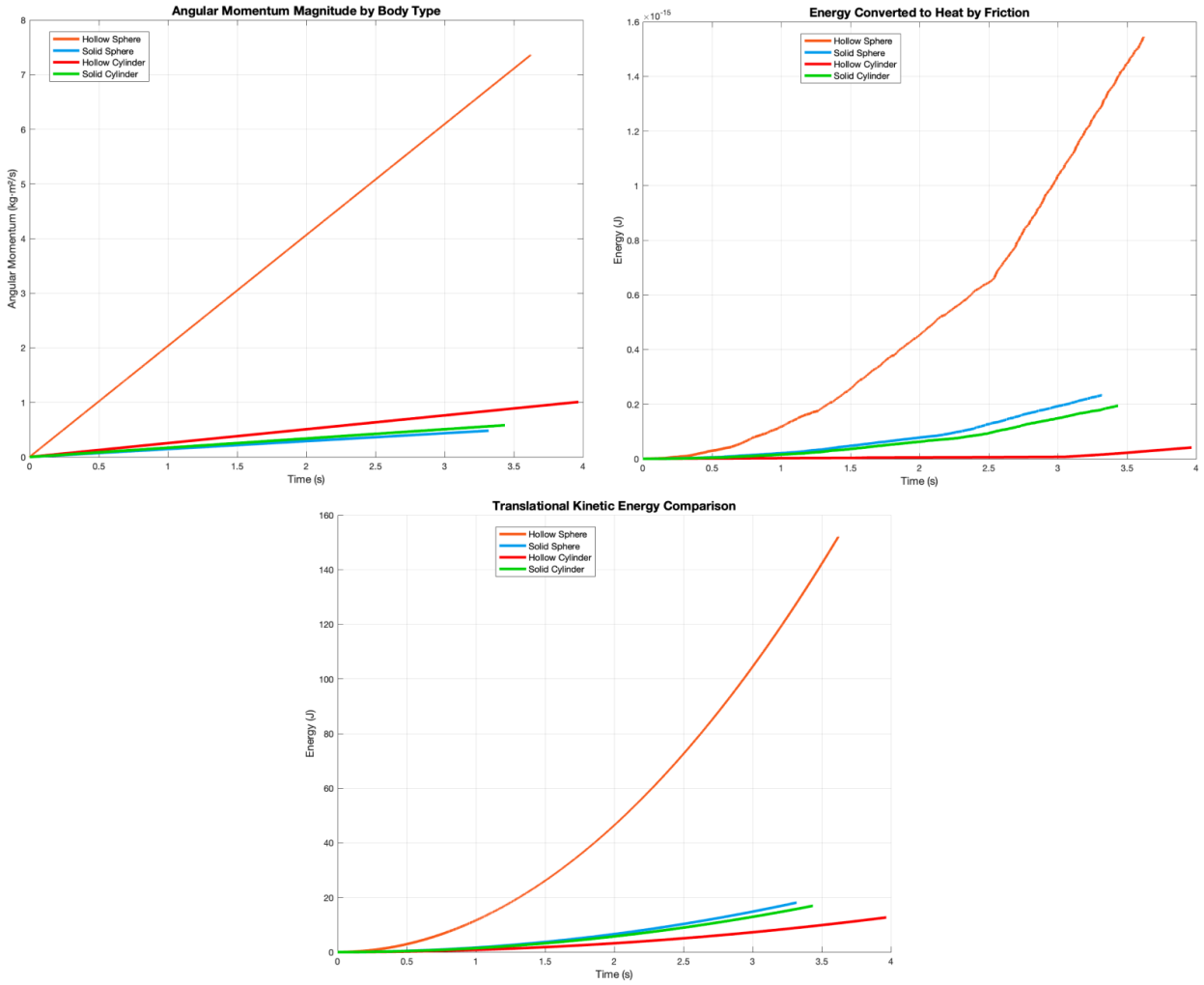


Figure 10: Angular momentum and heat conversion vs. time for hollow sphere at $m = 2.0\,\text{kg}$.

The hollow sphere at 2.0 kg shows similar scaling in momentum and heat as the solid sphere, but with overall lower values due to its larger moment of inertia.

**Hollow Sphere,** $m = 10.0\,\text{kg}$

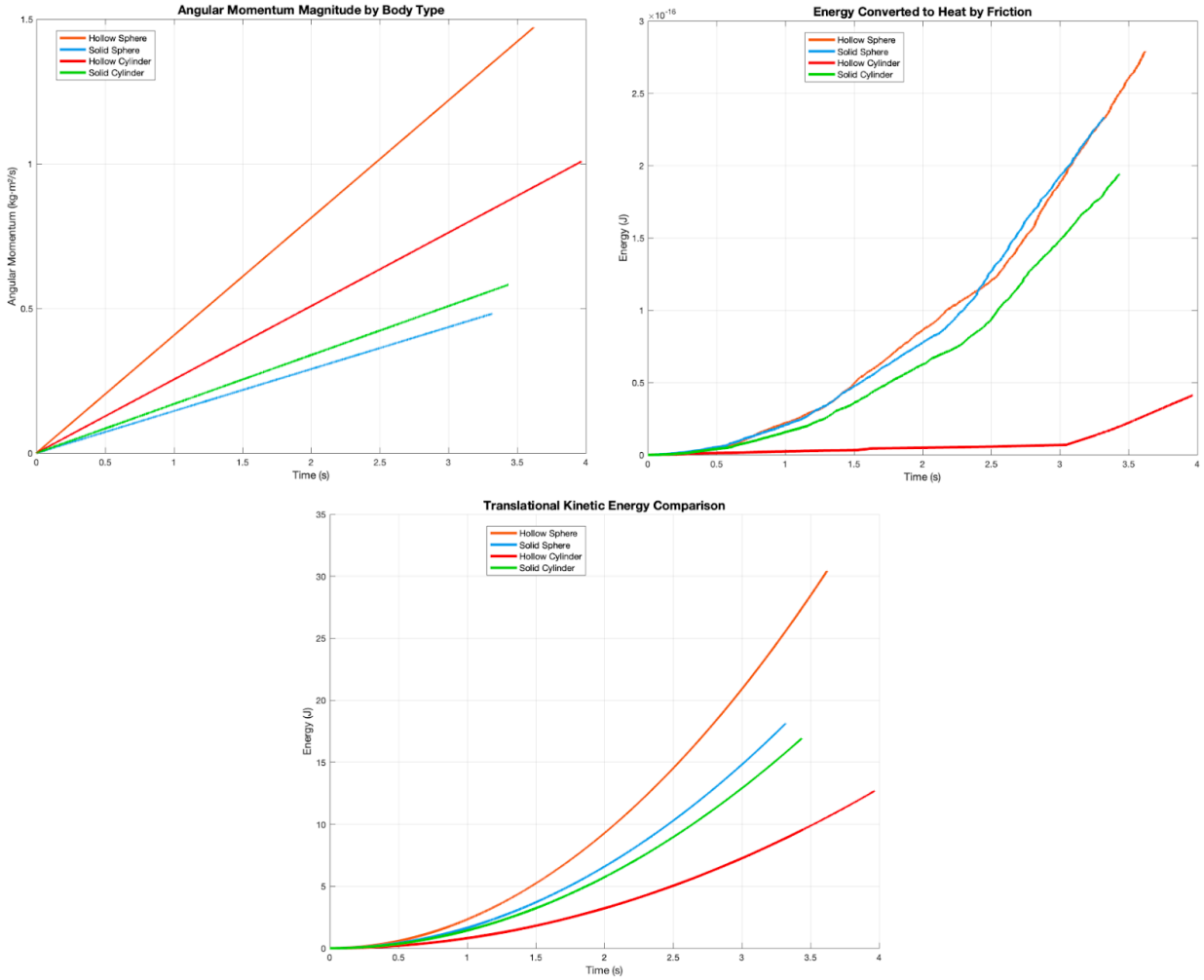


Figure 11: Angular momentum and heat conversion vs. time for hollow sphere at $m = 10.0\,\text{kg}$.

Increasing the mass of the hollow sphere to 10.0 kg leads to a proportional increase in the quantities.

**Solid Cylinder,** $m = 2.0\,\mathbf{kg}$

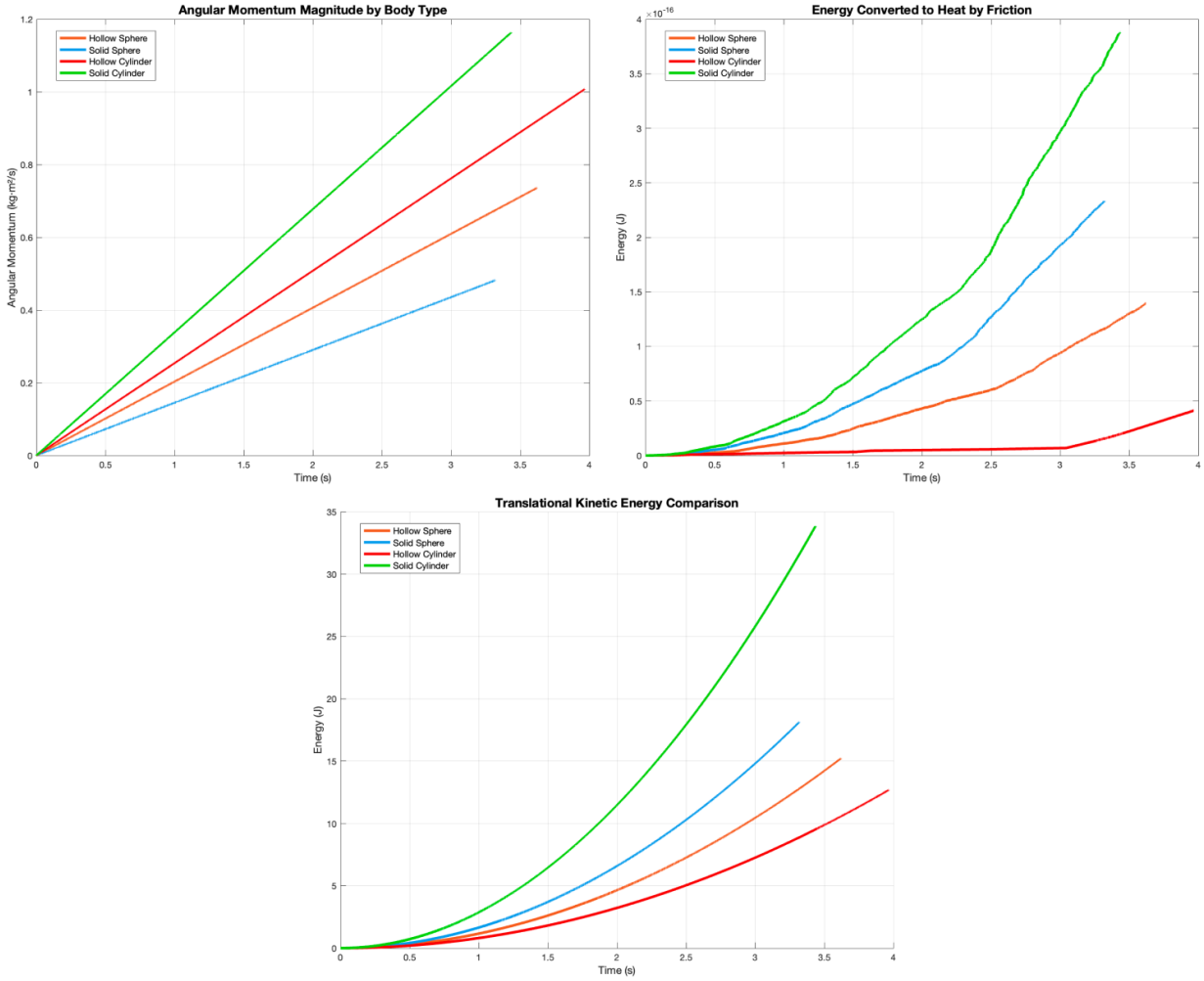


Figure 12: Angular momentum and heat conversion vs. time for solid cylinder at $m = 2.0\,\mathrm{kg}$.

The solid cylinder's angular momentum, heat energy, and kinetic energy all double relative to the control case, mirroring the behavior seen in spherical bodies.

**Solid Cylinder,** $m = 10.0\,\mathbf{kg}$

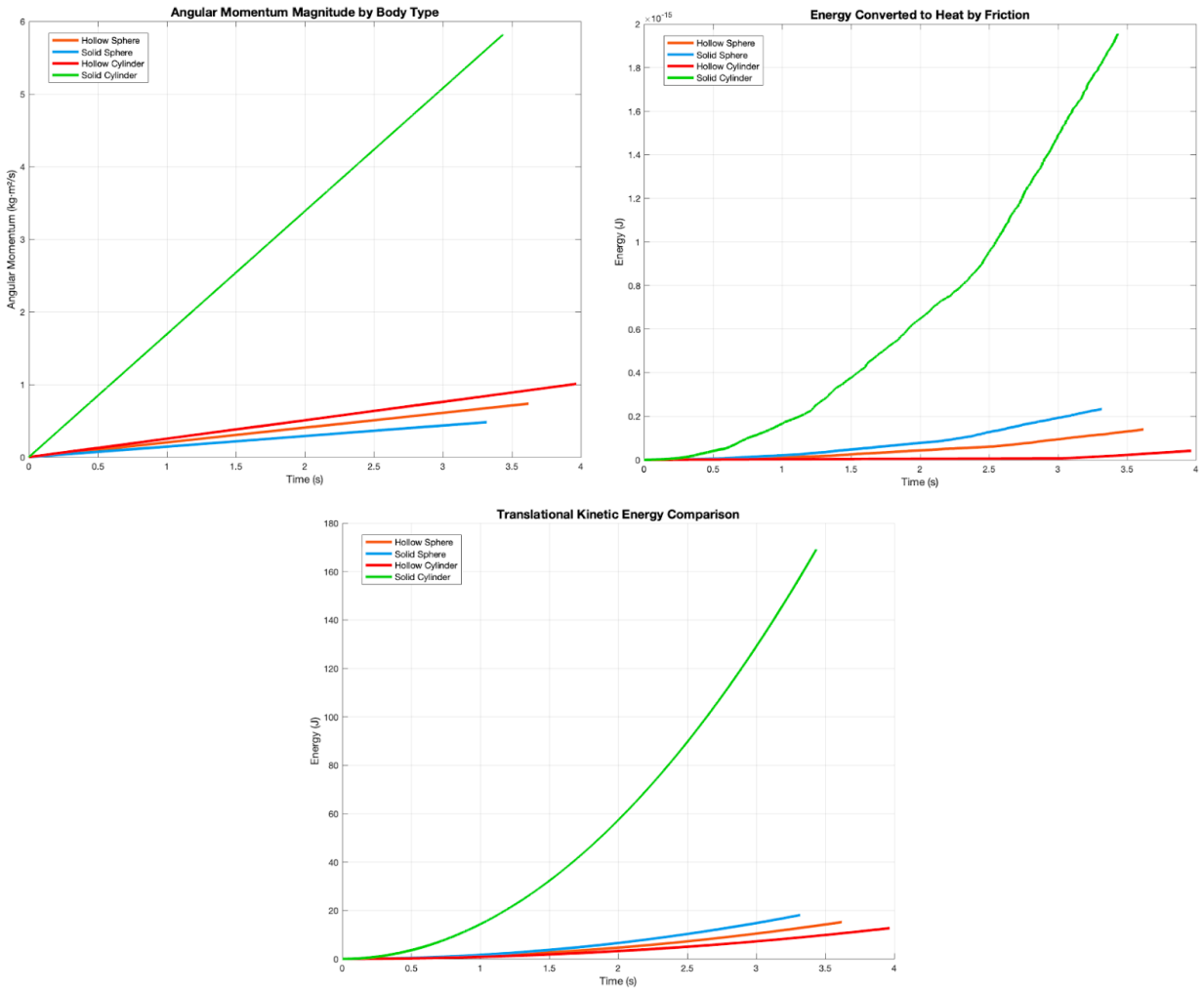


Figure 13: Angular momentum and heat conversion vs. time for solid cylinder at $m = 10.0\,\text{kg}$.

At ten times the mass, the quantities scale linearly with mass, confirming the relationship.

**Hollow Cylinder,** $m = 2.0\,\mathbf{kg}$

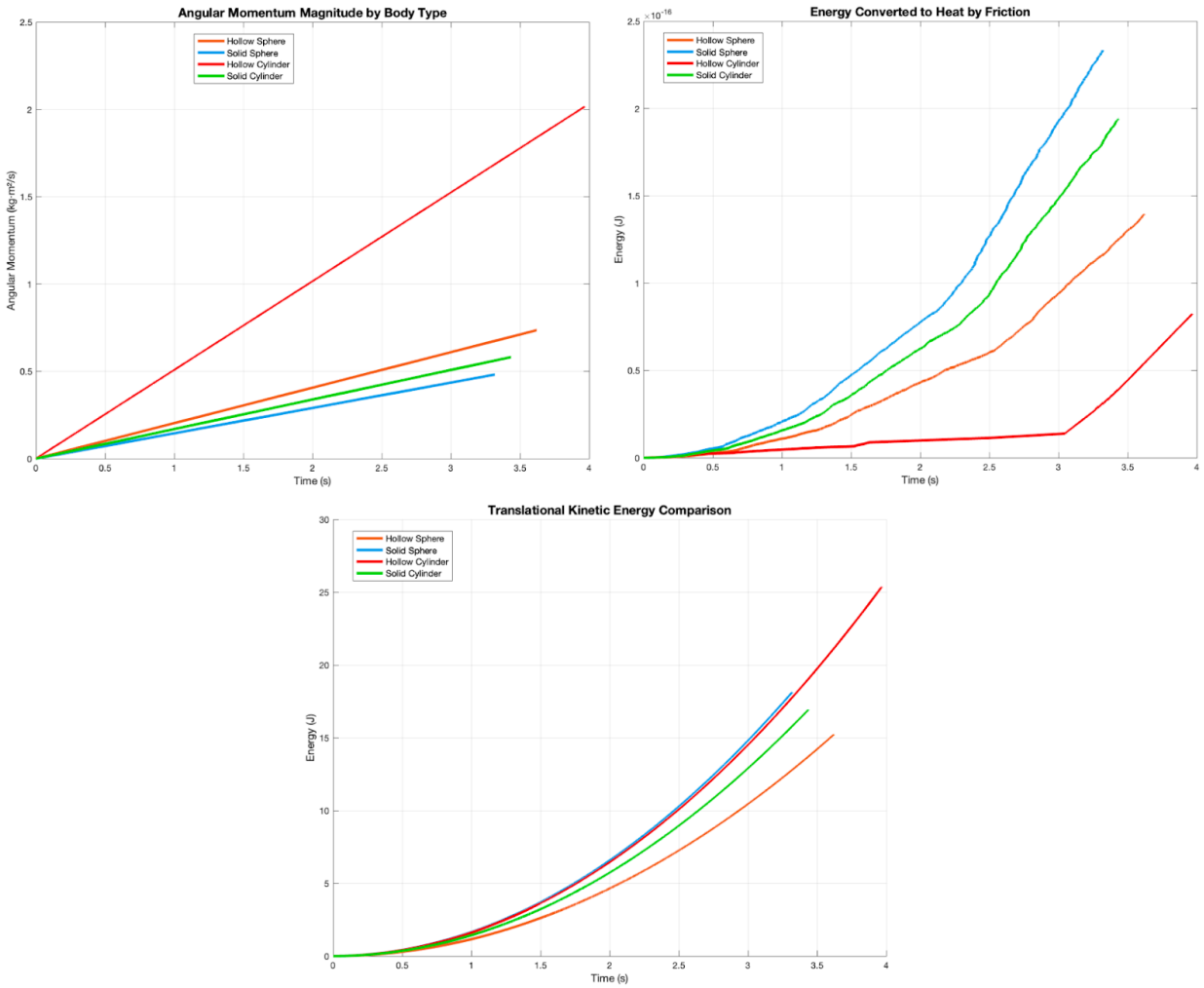


Figure 14: Angular momentum and heat conversion vs. time for hollow cylinder at $m = 2.0\,\mathrm{kg}$.

Similarly, the hollow cylinder shows an increase in the quantities when doubled in mass.

**Hollow Cylinder,** $m = 10.0\,\mathbf{kg}$

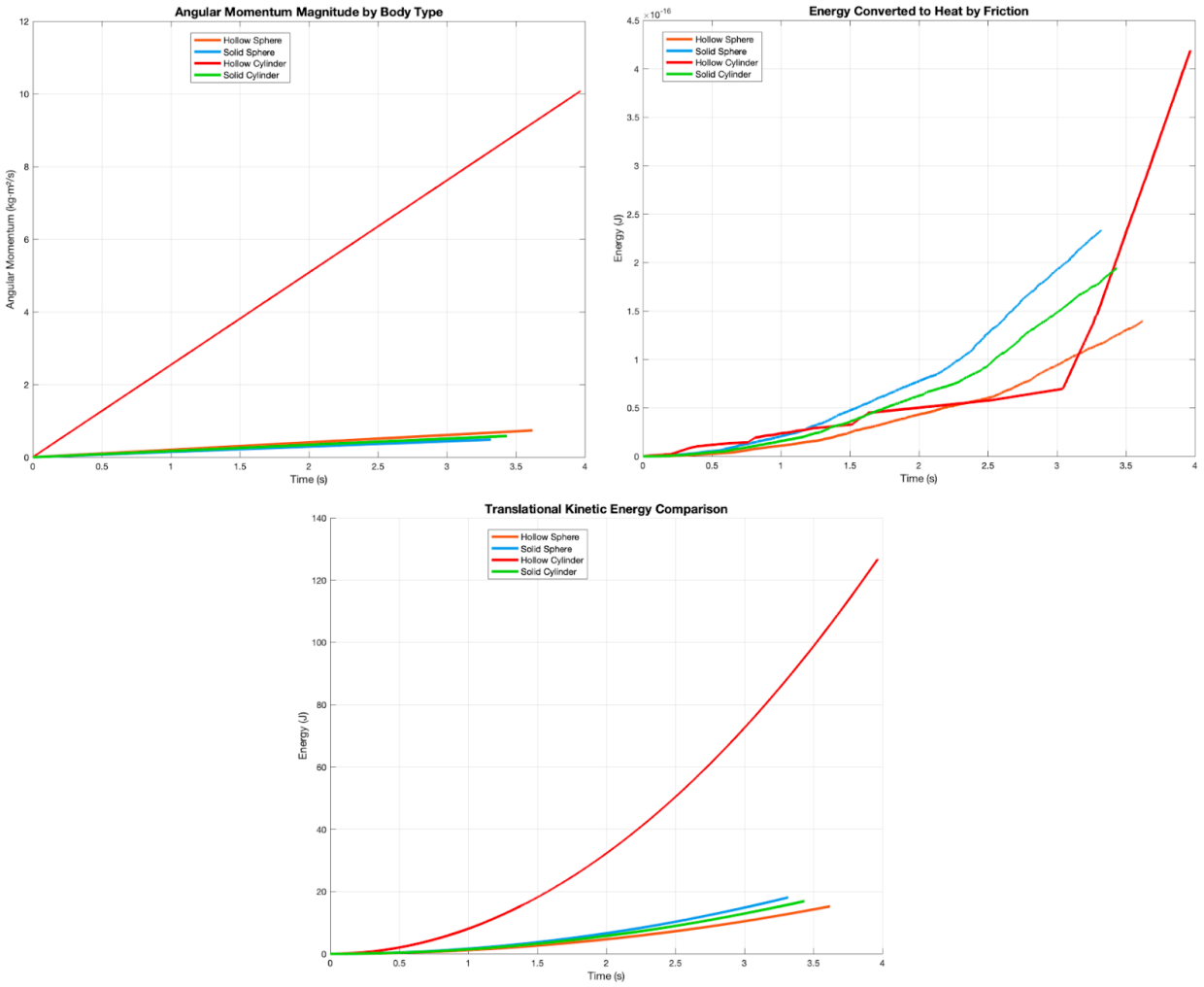


Figure 15: Angular momentum and heat conversion vs. time for hollow cylinder at $m = 10.0\,\text{kg}$.

Increasing the hollow cylinder's mass by a factor of ten results in the largest absolute increase in angular momentum and heat among all bodies, reflecting its combined mass and inertia effects.

## 8. Conclusions

This project involved simulating the motion of rigid bodies, in particular how bodies with different mass distributions rolled down an inclined plane. In addition to the theoretical elements such as gravity, linear and angular momentum, kinetic and potential energy, and energy converted to heat due to friction, we also accounted for simulation-specific factors such as elastic surface collision between the bodies and inclined plane. To get as close as possible to simulating pure rolling, we also accounted for and minimized slipping. The mechanism for simulating motion for a given body involved computing the net impulse upon surface collision and updating the linear and angular momentum. The numerical implementation we used was symplectic Euler method because it was optimal for conservative forces. Indeed, our simulation conserved total energy throughout time for small timesteps. After building a functional simulation, we studied how changes in the mass of each body affected results. As expected, changes in mass did not affect changes in motion, and the order in which the bodies reached the bottom was always (from first to last) solid sphere, solid cylinder, hollow sphere, hollow cylinder. However changes in mass did effect changes in mechanical energy, energy lost to friction, and angular momentum.

Through this project, we gained fundamental insights into rotational dynamics, energy partitioning, and the effects of mass distribution on motion. We found that, for pure rolling, acceleration was independent from mass, which is consistent with theoretical predictions. Also, the moment of inertia was critical in determining energy partitioning, as hollow objects consistently demonstrated different rotational-to-translational energy ratios compared to their solid counterparts. These results reinforce core physics principles while highlighting the importance of careful parameter selection in computational modeling of mechanical systems.

## 9. Appendix

Everything that happens in the code can be reduced to six recurring questions:

1. *Where is the body?* $\mathbf{r}(t)$
2. *How is it moving?* $\mathbf{v}(t)$ and $\boldsymbol{\omega}(t)$

3. *How far is it from the slope?* $\Phi(\mathbf{r})$
4. *What is the velocity of the contact point?* $\mathbf{v}_{cp}$
5. *What impulse removes penetration and slip?* $\mathbf{J} = \mathbf{J}_n + \mathbf{J}_t$
6. *How do we update momenta and check energy?*

## 9.1 Global Coordinate System

The right-handed orthonormal basis of the inertial frame $\mathcal{F}$ is

$$\hat{\mathbf{i}} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad \hat{\mathbf{j}} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \qquad \hat{\mathbf{k}} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

- **Basis directions**

$$\begin{aligned} \hat{\mathbf{i}} &: \text{ downslope } (+x), \\ \hat{\mathbf{j}} &: \text{ across lanes } (+y), \\ \hat{\mathbf{k}} &: \text{ upward}^2 \ (+z). \end{aligned}$$

  Any vector assumes the expansion

$$\mathbf{v} = v_x\hat{\mathbf{i}} + v_y\hat{\mathbf{j}} + v_z\hat{\mathbf{k}} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}.$$

- **Gravity vector**

$$\mathbf{g} = -\,9.81\,\hat{\mathbf{k}} = \begin{bmatrix} 0 \\ 0 \\ -9.81 \end{bmatrix} \ \mathrm{m\,s^{-2}}.$$

  Because $\hat{\mathbf{k}}$ is upward, $\mathbf{g}$ has a negative $z$-component. In the collision code we frequently evaluate the scalar projection $\mathbf{g}\cdot\hat{\mathbf{n}}$; the sign choice ensures that a body compressed against the slope has a *positive* normal load, simplifying the Coulomb-friction logic.

[2]The simulation's $z$-axis points upward while the MATLAB scene labels $Z$ as "down". Internally we keep the classical physics sign convention; the axis label in the figure is inverted so that increasing $Z$ visually corresponds to moving downward on screen.

## 9.2 The Slope as a Plane

**Constructing the Unit Normal.** Rotating a horizontal surface by an angle $\alpha$ about the $y$-axis produces a triangular wedge. A convenient (yet *unnormalised*) outward normal is

$$\tilde{\mathbf{n}} \;=\; \begin{bmatrix} \sin\alpha \\ 0 \\ \cos\alpha \end{bmatrix}.$$

To obtain a direction vector of unit length we divide by its norm:

$$\mathbf{n} \;=\; \frac{\tilde{\mathbf{n}}}{\|\tilde{\mathbf{n}}\|} \;=\; \frac{1}{\sqrt{\sin^2\alpha + \cos^2\alpha}} \begin{bmatrix} \sin\alpha \\ 0 \\ \cos\alpha \end{bmatrix} \;=\; \begin{bmatrix} \sin\alpha \\ 0 \\ \cos\alpha \end{bmatrix}, \qquad \|\mathbf{n}\| = 1.$$

*Physical meaning.* $\mathbf{n}$ is a **pure direction**: its magnitude is exactly one, so any dot-product of the form $\mathbf{n}\cdot\mathbf{r}$ returns the signed perpendicular component (*distance*) of $\mathbf{r}$ relative to the plane. In the collision routine this scalar tells us whether a body's center lies above (+) or penetrates (−) the slope surface.

**Completing the Plane Equation.** The plane passes through point $P_0 = (0, 0, h)$ at the top of the wedge. For any spatial point $\mathbf{r}$ the level-set function is

$$f(\mathbf{r}) = \mathbf{n}\cdot(\mathbf{r} - P_0) = 0.$$

Expanding and gathering constants:

$$\mathbf{n}\cdot\mathbf{r} \;+\; d = 0, \quad d = -\,\mathbf{n}\cdot P_0 = -h\,n_z.$$

*Why bother with $d$?* It lets the penetration test be a *single* dot-product no matter where the plane is in world space.

**Signed Distance for a Sphere (vector form).** Let the center of mass be

$$\vec{r}_{\text{cm}} \;=\; \begin{bmatrix} x_{\text{cm}} \\ y_{\text{cm}} \\ z_{\text{cm}} \end{bmatrix}, \qquad R = \text{sphere radius}.$$

With the unit normal, $\vec{n} = \begin{bmatrix} \sin\alpha \\ 0 \\ \cos\alpha \end{bmatrix}$, and plane offset $d = -\,h\,\cos\alpha$ ($h$ is the slope height at $x = 0$), the signed gap function becomes

$$\boxed{\Phi(\vec{r}_{\mathrm{cm}}) \;=\; \vec{n}^{\mathsf{T}}\,\vec{r}_{\mathrm{cm}} \;+\; d \;-\; R}$$

1. $\vec{n}^{\mathsf{T}}\vec{r}_{\mathrm{cm}} + d$ is the perpendicular distance from the sphere's *center* to the plane.
2. Subtract $R$ to convert the center–plane distance into the distance from the *surface* of the sphere to the plane.

Hence $\Phi > 0 \Rightarrow$ separated, $\Phi = 0 \Rightarrow$ touching, $\Phi < 0 \Rightarrow$ penetration depth $|\Phi|$.
Choosing $Q$ as the contact point, $\mathbf{r}_Q - \mathbf{r}_{cm} = -R\,\mathbf{n}$.

## 9.3 Normal Impulse $J_n$

A collision generates an *impulse*—a finite kick $\vec{J} = J_n\,\hat{\imath}_n + \vec{J}_t$—applied over an infinitesimal time. Here we derive the scalar normal component $J_n$ that enforces the desired coefficient–of–restitution behaviour.

**Pre-contact kinematics.** At the instant just before impact [3]

$$\vec{v}_{cp}^{\,-} \;=\; \vec{v}_{cm}^{\,-} \;+\; \vec{\omega}^{-}\times\vec{r}, \qquad v_n^- \;=\; \hat{\imath}_n^{\mathsf{T}}\vec{v}_{cp}^{\,-},$$

where

- $\vec{r} = -R\,\hat{\imath}_n$ is the radius vector from the CM to the contact point (sphere against plane $\vec{r} \parallel \hat{\imath}_n$);
- the superscript "$-$" denotes *pre-impact* values.

**Restitution target.** Newton's impact law stipulates [4]

$$v_n^+ \;=\; -\,e\,v_n^-,$$

with $0 \le e \le 1$; "$e = 1$" is perfectly elastic, "$e = 0$" is perfectly plastic. Thus the required *change* in normal velocity is

$$\Delta v_n \;=\; v_n^+ - v_n^- \;=\; -(1+e)\, v_n^-.$$

**Linear momentum update.** A normal impulse changes the CM velocity by linear momentum conservation:

$$\Delta \vec{v}_{cm} = \frac{\vec{J}}{m}.$$

Taking the dot product with the unit normal $\hat{\imath}_n$:

$$\Delta v_n = \hat{\imath}_n^{\mathsf{T}} \Delta \vec{v}_{cm} = \frac{\hat{\imath}_n^{\mathsf{T}} \vec{J}}{m} = \frac{J_n}{m},$$

because $\hat{\imath}_n \cdot \hat{\imath}_n = 1$ and $\hat{\imath}_n \cdot \vec{J}_t = 0$ (tangential impulse is orthogonal).

**Rotational contribution (why it *vanishes*).** The impulse also produces an angular kick

$$\Delta \vec{\omega} = \frac{\vec{r} \times \vec{J}}{I}.$$

For a sphere–plane contact $\vec{r} \parallel \hat{\imath}_n$, so $\vec{r} \times \vec{J}_n \hat{\imath}_n = \vec{0}$; the normal impulse does *not* alter $\vec{\omega}$. Hence the only effective inertia in the normal direction is the translational mass $m$.

**Solve for the impulse.** Equate the two expressions for $\Delta v_n$:

$$\frac{J_n}{m} = -(1+e)\, v_n^- \quad \Longrightarrow \quad \boxed{J_n = -\,(1+e)\, v_n^-\, m}.$$

- A *positive* $v_n^-$ means the contact point is moving *into* the plane, giving a *negative* impulse (pushes the body upward).
- If $e = 0$ the post-impact normal velocity is zero (perfect sticking); if $e = 1$ the magnitude is preserved but directed outward.

**Angular Contribution.** Because the line from CM to contact is *parallel to* $\mathbf{n}$, the angular impulse $\mathbf{J}$ produces *no* additional normal velocity at the same point (cross-product is perpendicular), so $\Delta v_n$ above is complete.

## 9.4 Tangential Impulse $J_t$

The collision impulse has two orthogonal pieces $\vec{J} = J_n\,\hat{\jmath}_n \;+\; \vec{J}_t$. Since we already fixed $J_n$, we can now construct the *tangential* component $\vec{J}_t$.

**Identify the slip velocity.** Immediately before impact the contact-point velocity is

$$\vec{v}_{cp}^{\,-} = \vec{v}_{cm}^{\,-} \;+\; \vec{\omega}^{-}\times\vec{r}, \quad \text{with } \vec{r} = -R\,\hat{\jmath}_n.$$

Split it into normal and tangential parts:

$$v_n^- = \hat{\jmath}_n^{\mathsf{T}}\vec{v}_{cp}^{\,-}, \qquad \boxed{\vec{v}_t^{\,-} = \vec{v}_{cp}^{\,-} - v_n^-\,\hat{\jmath}_n}.$$

Only $\vec{v}_t^{\,-}$ needs to be cancelled for perfect rolling.

**How $\vec{J}_t$ alters $\vec{v}_t$.** A purely tangential impulse $\vec{J}_t = -J_t\,\hat{\jmath}_t$ ($\hat{\jmath}_t \equiv \vec{v}_t^{\,-}/\|\vec{v}_t^{\,-}\|$) produces

$$\Delta\vec{v}_{cm} = \frac{\vec{J}_t}{m}, \quad \Delta\vec{\omega} = \frac{\vec{r}\times\vec{J}_t}{I}.$$

For a sphere $\vec{r} = -R\,\hat{\jmath}_n$, hence

$$\vec{r}\times\vec{J}_t = (-R\hat{\jmath}_n)\times(-J_t\hat{\jmath}_t) = R\,J_t\,(\hat{\jmath}_n\times\hat{\jmath}_t) \quad \parallel\ \hat{\jmath}_t.$$

The contact point's tangential change is therefore

$$\Delta\vec{v}_t = \frac{\vec{J}_t}{m} + (\Delta\vec{\omega})\times(-R\hat{\jmath}_n) = \left(\frac{1}{m} + \frac{R^2}{I}\right)\vec{J}_t.$$

Define the *effective tangential mass*

$$\boxed{k_t = \frac{1}{m} + \frac{R^2}{I}}, \quad \Longrightarrow \quad \Delta\vec{v}_t = k_t\,\vec{J}_t.$$

**Static-friction (ideal) impulse.** To make the post-impact slip zero ($\vec{v}_t^{\,+} = 0$) one would need

$$\Delta\vec{v}_t = -\vec{v}_t^{\,-} \quad\Longrightarrow\quad \boxed{J_t^{\text{ideal}} = \frac{\|\vec{v}_t^{\,-}\|}{k_t}}$$

directed opposite $\vec{v}_t^{\,-}$.

**Coulomb limit.** Real contacts obey $\|\vec{J}_t\| \le \mu\,|J_n|$. Hence

$$\boxed{J_t \;=\; \min\!\big(\mu\,|J_n|,\; J_t^{\text{ideal}}\big), \qquad \vec{J}_t = -\,J_t\,\hat{\jmath}_t}$$

- If $J_t^{\text{ideal}} \le \mu|J_n|$ the surface can supply the full static-friction impulse, the slip is erased, and the body *sticks* (rolls without sliding) for this time–step.
- Otherwise the contact is in kinetic-friction regime; we apply the maximum admissible impulse $\mu|J_n|$, reducing but not eliminating slip.

## 9.5 Post-Impact Updates

$$\begin{cases} \mathbf{v}_{cm}^{+} &= \mathbf{v}_{cm}^{-} + \dfrac{\mathbf{J}}{m}, \\ \boldsymbol{\omega}^{+} &= \boldsymbol{\omega}^{-} + \dfrac{(-R\mathbf{n}) \times \mathbf{J}}{I}. \end{cases}$$

*Why the cross-product?* Angular momentum about the CM is $\mathbf{L} = I\,\boldsymbol{\omega}$. An impulse has moment $\boldsymbol{\tau}\Delta t = (\mathbf{r}_{cp} - \mathbf{r}_{cm}) \times \mathbf{J}$, and $\Delta\mathbf{L} = \boldsymbol{\tau}\Delta t$.

## 9.6 Energy Book-Keeping

For every body we store *four* contributions at each time–step and then form running totals to monitor conservation and dissipation [2]. The mathematics below shows *exactly* how each term arises.

**Translational kinetic energy.** A rigid body of mass $m$ whose center–of–mass (CM) moves with linear velocity $\vec{v}_{cm}$ possesses

$$\boxed{E_{\text{trans}} = \frac{1}{2}\,m\,\|\vec{v}_{cm}\|^2} \quad \text{J}.$$

**Rotational kinetic energy.** For rotation about any fixed principal axis the kinetic energy is

$$\boxed{E_{\text{rot}} = \frac{1}{2}\,I\,\|\vec{\omega}\|^2} \quad \text{J},$$

where $\vec{\omega}$ is the angular–velocity vector and $I$ the moment of inertia about that axis.

**Gravitational potential energy.** With the zero of potential at the slope's lower plane $z = 0$,

$$\boxed{E_{\text{pot}} = m\,g\,z_{cm}} \quad \text{J},$$

$g = \|\vec{g}\| = 9.81\ \mathrm{m/s^2}$.

**Frictional work (dissipation).** During a single time increment $\Delta t$ the kinetic–friction force at the contact is

$$\vec{F}_f = -\mu\, N\, \hat{\jmath}_t, \qquad N = \frac{|J_n|}{\Delta t},$$

where $N$ is the average normal force inferred from the *normal impulse* $J_n$.

The differential work is then

$$\boxed{\Delta W_f = \vec{F}_f \cdot \vec{v}_t\ \Delta t = \mu\, |J_n| \|\vec{v}_t\|\, \Delta t} \quad \text{J}.$$

1. The sign is always *negative* because $\vec{F}_f$ opposes the slip velocity $\vec{v}_t$.
2. The simulation accumulates $W_f^{\mathrm{cum}}(t+\Delta t) = W_f^{\mathrm{cum}}(t) + \Delta W_f$ to obtain the total energy irreversibly lost to heat.

**Mechanical-energy check.** At every step we store

$$E_{\mathrm{mech}} = E_{\mathrm{trans}} + E_{\mathrm{rot}} + E_{\mathrm{pot}}, \qquad E_{\mathrm{total}} = E_{\mathrm{mech}} + W_f^{\mathrm{cum}}.$$

**Physical meaning (why bother?).** Tracking $\left\{E_{\mathrm{trans}}, E_{\mathrm{rot}}, E_{\mathrm{pot}}, W_f\right\}$ exposes whether the impulses are applied correctly, whether the time-step is sufficiently small, and whether unexpected energy is leaking from the numerical scheme.

## 9.7 Rolling–Without–Slipping Condition

**Contact–point velocity.** For *any* rigid body the linear velocity of a material point at position $\vec{r} = \vec{r}_{cm} + \vec{\rho}$ relative to the center of mass is

$$\boxed{\vec{v}(\vec{r}) = \vec{v}_{cm}\ +\ \vec{\omega} \times \vec{\rho}} \quad \text{(rigid–body kinematics)}.$$

- $\vec{v}_{cm}$ — translational drift of the whole body.
- $\vec{\omega} \times \vec{\rho}$ — *pure spin* contribution; a right-hand rule rotation about $\vec{\omega}$.

**The rolling constraint.** For a sphere or cylinder of radius $R$ on a plane with outward unit normal $\vec{n}$, the contact point is $\vec{r}_{cp} = \vec{r}_{cm} - R\,\vec{n}$ so that $\vec{\rho} = -R\,\vec{n}$.

$$\vec{v}_{cp} = \vec{v}_{cm} + \vec{\omega} \times (-R\vec{n}).$$

**Pure rolling** (no slipping) demands

$$\boxed{\vec{v}_{cp} = \mathbf{0} \implies \vec{v}_{cm} = \vec{\omega} \times (R\vec{n}) = -\vec{\omega} \times (-R\vec{n})}$$

1. The cross-product ensures the CM velocity is *tangent* to the surface and perpendicular to $\vec{\omega}$.

2. Magnitude condition for a sphere/cylinder: $\|\vec{v}_{cm}\| = \|\vec{\omega}\| R$. A faster spin would *advance* the contact patch (+ slip); a slower spin would *drag behind* (– slip).

**Slip–metric used in the code.** To quantify how close the numerical motion is to perfect rolling we define

$$\boxed{\sigma \;=\; R\,\|\vec{\omega}\| \;-\; \|\vec{v}_{cm}\|}$$

and plot $\sigma(t)$ for each body:

- $\sigma = 0$ ideal rolling.
- $\sigma > 0$ *underspin* (the surface points slip *backwards* relative to the slope).
- $\sigma < 0$ *overspin* (surface points are trying to run *ahead* of the CM).

**Why this specific definition?** Because both terms are *scalars with identical units*, the difference gives an immediate measure ($\mathrm{ms}^{-1}$) of tangential slip at the contact line:

$$\|\vec{v}_{cp}\| = \big|\, R\,\|\vec{\omega}\| - \|\vec{v}_{cm}\| \,\big| \;=\; |\sigma|.$$

Hence the sign of $\sigma$ tells the *direction* of slip, while $|\sigma|$ is the slip speed that Coulomb friction must oppose.

## References

[1] S. Brorson. Symplectic integrators. `https://math.libretexts.org/Bookshelves/Differential_Equations/Numerically_Solving_Ordinary_Differential_Equations_(Brorson)/01:_Chapters/1.07:_Symplectic_integrators`, 2020. Accessed: 2025-05-02.

[2] Daniel Kleppner and Robert Kolenkow. *An Introduction to Mechanics*. Cambridge University Press, 2 edition, 2013. Chapter 6: Rigid Body Mechanics.

[3] OpenStax. *College Physics 2e*. OpenStax, Rice University, 2016. Chapter 10: Introduction to Rotational Motion and Angular Momentum.

[4] OpenStax. *College Physics 2e*. OpenStax CNX, Rice University, 2016. Chapter 8.7: Perfectly Inelastic Collisions; Chapter 8.8: Collisions in Two Dimensions.

[5] OpenStax. *College Physics 2e*. OpenStax CNX, Rice University, 2016. Chapter 10.5: Rolling Motion.

[6] Raymond A. Serway and John W. Jewett. *Physics for Scientists and Engineers*. Cengage Learning, 10 edition, 2018. Chapter 5: Applications of Newton's Laws (Inclined Planes).