Modeling 3D Projectile Motion with Drag and Spin: A Geometric and Vector-Based Derivation

Vyom N. Patel

Introduction

This article presents a vector-based formulation of projectile motion for a rigid body in 3D, factoring in both linear drag and rotational spin. Rather than simply describing angular velocity heuristically, this model connects the rotational force induced by air resistance with the object's surface geometry and spin. The key insight is that drag interacting with surface tangent vectors creates torque, which updates angular velocity.

Assumptions and Setup

We model:

- A rigid body (e.g., ball or disc) in 3D
- Initial velocity $\vec{v}_0$ at $t = 0$
- Gravity: $\vec{g} = -g\hat{y}$
- Linear drag: $\vec{F}_d = -k\vec{v}$
- Rotational torque from air resistance
- n labeled surface points with unit tangent vectors

Let the position of the center of mass be $\vec{r}(t)$ and angular velocity $\vec{\omega}(t)$. The position of each surface point i at time t is:

$$\vec{r}_i(t) = \vec{r}(t) + \vec{R}_{\text{rot}}(\vec{u}_i, \vec{\omega}, t)$$

where $\vec{u}_i$ is the relative position vector of point i from the center, and $\vec{R}_{\text{rot}}$ is the rotated vector due to spin.

Translational Motion

$$\frac{d^2\vec{r}}{dt^2} = -g\hat{y} - \frac{k}{m}\frac{d\vec{r}}{dt}$$

This is a second-order differential equation with linear damping.

Solution:

Let $\vec{v}(t) = \frac{d\vec{r}}{dt}$

$$\frac{d\vec{v}}{dt} = -g\hat{y} - \frac{k}{m}\vec{v}$$

Solving this yields:

$$\vec{v}(t) = \left(\vec{v}_0 + \frac{gm}{k}\hat{y}\right) e^{-\frac{k}{m}t} - \frac{gm}{k}\hat{y}$$

Integrating:

$$\vec{r}(t) = \vec{r}_0 + \left(\vec{v}_0 + \frac{gm}{k}\hat{y}\right)\left(\frac{m}{k}\left(1 - e^{-\frac{k}{m}t}\right)\right) - \frac{gtm}{k}\hat{y}$$

Rotational Motion

Let's assume n discrete points on the object, each with a tangent unit vector $\vec{t_i}$. The drag force at point i forms an angle β with the global x-axis. The vector $\vec{t_i}$ forms an angle α_i with the same x-axis.

The total torque on the object can be approximated by:

$$\vec{\tau} = \sum_{i=1}^{n} \left((\vec{F}_{d_i} \cdot \vec{t_i}) \cdot \vec{u}_i\right)$$

The rotational equation of motion is:

$$\vec{\tau} = I\frac{d\vec{\omega}}{dt}$$

where I is the moment of inertia tensor.

To compute $\vec{F}_{d_i}$ at point i, we take the drag direction $\hat{d}$ and evaluate:

$$\theta_i = \angle(\vec{t_i}, \hat{d}) = \alpha_i + \beta$$
$$F_{\text{rot},i} = F_d \cos(\theta_i)$$

Summing all such projections:

$$\vec{F}_{\text{rot}} = \sum_{i=1}^{n} F_d \cos(\alpha_i + \beta)\, \vec{u}_i$$

and the torque becomes:

$$\vec{\tau} = \sum_{i=1}^{n} \left(F_d \cos(\alpha_i + \beta)\, \vec{u}_i \times \vec{t_i}\right)$$

Then the angular velocity is:

$$\vec{\omega}(t) = \int \frac{\vec{\tau}(t)}{I}\, dt$$

Rotation of Points (Rodrigues' Formula)

To find the new position of each point due to spin, we use:

$$\vec{R}_{\text{rot}}(\vec{u}_i, \vec{\omega}, t) = \vec{u}_i \cos(\theta) + (\vec{\omega} \times \vec{u}_i)\sin(\theta) + \vec{\omega}(\vec{\omega} \cdot \vec{u}_i)(1 - \cos(\theta))$$

where $\theta = \|\vec{\omega}\| t$

Final Formula: Position of Each Surface Point

Putting it all together, the position of point i at time t is:

$$\vec{r}_i(t) = \vec{r}_0 + \left(\vec{v}_0 + \frac{gm}{k}\hat{y}\right)\left(\frac{m}{k}\left(1 - e^{-\frac{k}{m}t}\right)\right) - \frac{gtm}{k}\hat{y} + \vec{R}_{\text{rot}}(\vec{u}_i, \vec{\omega}(t), t)$$

This equation models:

- Drag-based damping of translational velocity
- Gravity
- Rotational change due to distributed surface drag
- 3D reorientation of points from angular velocity

Conclusion

This model attempts a semi-analytic, semi-simulation-friendly expression of projectile motion with spin in 3D. The approach combines:

- Analytical damped motion
- Torque as projection of drag onto local surface tangents
- Dynamic angular velocity updates and Rodrigues' rotation

While simplifications are used (like constant drag or discrete points), this method offers a bridge between classical mechanics and geometric simulation. It could be extended with continuous surface integrals, aerodynamic lift effects, or machine-learned corrections.