Elastic Collision Brownian Motion: Emergence from Pure Collision Dynamics

Andrea Simone Costa

October 11, 2025

1 Introduction

Unlike the previous simulations where Brownian motion was implemented through random walks with artificial stochastic steps, this project demonstrates how Brownian motion emerges naturally from the fundamental physics of elastic collisions. A single large particle (mass $M = 30$, radius $R = 8$) starts completely at rest and is bombarded by approximately 1000 small thermal particles (mass $m = 1$, radius $r = 1.5$) whose velocities follow a Maxwell-Boltzmann distribution. The only dynamics are deterministic elastic collisions - no random walks, no stochastic forces, no artificial noise.

This approach more faithfully represents the historical discovery of Brownian motion by Robert Brown in 1827, who observed pollen grains moving erratically in water due to bombardment by invisible water molecules. The simulation validates key predictions of statistical mechanics, particularly the velocity autocorrelation function and the emergence of diffusive behavior from microscopic deterministic dynamics.

2 Statistical Mechanics Background

2.1 Maxwell-Boltzmann Distribution

In thermal equilibrium, the velocity distribution of particles follows the Maxwell-Boltzmann distribution. For a 2D system, each velocity component (v_x, v_y) follows a Gaussian distribution:

$$P(v_x, v_y) = \frac{m}{2\pi k_B T} \exp\left(-\frac{m(v_x^2 + v_y^2)}{2k_B T}\right) \tag{1}$$

where m is the particle mass, k_B is Boltzmann's constant, and T is the temperature. In the implementation, we use reduced units where $k_B = 1$, so each velocity component is sampled from a Gaussian with standard deviation:

$$\sigma = \sqrt{\frac{T}{m}} \tag{2}$$

The Box-Muller transform is used to efficiently generate Gaussian random numbers from uniform random numbers:

$$\begin{cases} Z_1 = \sqrt{-2\ln U_1}\cos(2\pi U_2) \\ Z_2 = \sqrt{-2\ln U_1}\sin(2\pi U_2) \end{cases} \tag{3}$$

where $U_1, U_2 \sim \mathcal{U}[0, 1]$ are uniform random numbers and $Z_1, Z_2 \sim \mathcal{N}(0, 1)$ are standard normal variates.

2.2 Elastic Collision Physics

When two particles collide elastically, both momentum and kinetic energy are conserved. For particles with masses m_1, m_2 and velocities $\mathbf{v}_1$, $\mathbf{v}_2$, the post-collision velocities along the collision normal $\hat{\mathbf{n}}$ are:

$$v'_{1,\text{new}} = \frac{(m_1 - m_2)v'_1 + 2m_2 v'_2}{m_1 + m_2} \tag{4}$$

$$v'_{2,\text{new}} = \frac{(m_2 - m_1)v'_2 + 2m_1 v'_1}{m_1 + m_2} \tag{5}$$

where v' denotes the velocity component projected onto the collision normal:

$$v' = \mathbf{v} \cdot \hat{\mathbf{n}}, \quad \hat{\mathbf{n}} = \frac{\mathbf{r}_2 - \mathbf{r}_1}{|\mathbf{r}_2 - \mathbf{r}_1|} \tag{6}$$

Importantly, the tangential velocity components (perpendicular to $\hat{\mathbf{n}}$) remain unchanged, as the collision force acts only along the line connecting the particle centers.

2.3 Velocity Autocorrelation Function

The velocity autocorrelation function (VACF) is a key observable for detecting Brownian motion:

$$C(\tau) = \frac{\langle \mathbf{v}(t) \cdot \mathbf{v}(t+\tau) \rangle}{\langle |\mathbf{v}(t)||\mathbf{v}(t+\tau)| \rangle} \tag{7}$$

This normalized form measures the directional correlation: $C(\tau) = 1$ indicates perfect correlation (particle maintains its direction), while $C(\tau) = 0$ indicates complete decorrelation (random direction).

For a particle undergoing Brownian motion in the overdamped regime, the VACF decays exponentially:

$$C(\tau) = \exp\left(-\frac{\tau}{\tau_c}\right) \tag{8}$$

where $\tau_c = M/\gamma$ is the velocity relaxation time, with M the particle mass and γ the friction coefficient. The characteristic decay time occurs when $C(\tau_c) = 1/e \approx 0.368$.

3 Implementation

3.1 Particle Initialization

The simulation initializes particles with the following properties:

- **Large particle**: mass $M = 30$, radius $R = 8$, initial velocity $(v_x, v_y) = (0, 0)$, positioned at the center
- **Small particles**: mass $m = 1$, radius $r = 1.5$, count $N \approx 1000$, velocities sampled from Maxwell-Boltzmann distribution at temperature $T = 0.5$

Small particles are initialized at random positions at least $3R$ away from the large particle to prevent immediate collisions. This ensures the system starts in a physically reasonable configuration.

3.2 Collision Detection and Resolution

Naive collision detection has $O(N^2)$ complexity, checking all particle pairs. For $N \approx 1000$ particles, this requires ∼500,000 checks per time step. The simulation implements spatial hashing to reduce this cost:

1. Divide the domain into cells of size $2r$ (twice the small particle radius)
2. Assign each particle to its containing cell
3. For collision detection, check only particles in the same cell and the 8 neighboring cells (a 3×3 grid)

For the large particle with radius $R = 8$ and small particles with radius $r = 1.5$, the maximum collision distance is $R + r = 9.5$. With cell size $2r = 3$, this spans approximately 4 cells, so the large particle checks a 9×9 neighborhood (cells within distance 4).

Collision throttling: To prevent rapid repeated collisions between the same pair (which can occur due to numerical overlap), a minimum collision interval is enforced. Particles cannot collide with the same partner more than once per $\Delta t_{\text{min}} = 1$ tick.

3.3 Velocity Sampling and Analysis

The velocity of the large particle is sampled every 10 ticks to construct the velocity history used for autocorrelation analysis. This sampling interval reduces noise while maintaining sufficient temporal resolution to capture the decorrelation dynamics.

The VACF is calculated for time lags $\tau = 0, 1, 2, \ldots, \tau_{\text{max}}$ where $\tau_{\text{max}} = \min(25, N_{\text{samples}}/4)$. The division by 4 ensures sufficient statistical samples for reliable correlation estimates at large lags.

Brownian motion detection criterion: The simulation detects Brownian behavior when $C(\tau = 3) < 0.7$. This threshold indicates significant velocity decorrelation, signaling the transition from ballistic motion (where the large particle maintains its direction between collisions) to diffusive motion (where direction becomes randomized by collisions).

3.4 Boundary Conditions

Particles undergo elastic reflection at domain boundaries. When a particle crosses a boundary at position x_{boundary}, its position and velocity are updated:

$$x_{\text{new}} = x_{\text{boundary}} - (x - x_{\text{boundary}}), \quad v_x \to -|v_x| \tag{9}$$

This preserves kinetic energy while reversing the velocity component normal to the boundary.

4 Physics Validation

4.1 Emergence of Brownian Motion

The simulation successfully demonstrates several key features:

1. **Large particle activation**: Initially at rest, the large particle begins moving after sufficient collisions with small thermal particles. The buildup of momentum occurs gradually, as each collision transfers a small amount of momentum (proportional to the mass ratio $m/M \approx 1/30$).
2. **Velocity autocorrelation decay**: After an initial ballistic phase, the VACF begins to decay. The decay rate depends on the collision frequency, which in turn depends on the small particle density and thermal velocity.

3. **Brownian detection**: Typically within 300-500 ticks (≈30-50 velocity samples), the criterion $C(3) < 0.7$ is satisfied, indicating the onset of Brownian motion.
4. **No artificial forces**: All motion emerges from elastic collisions. The large particle's trajectory is completely determined by the collision history - there are no random number generators influencing its motion directly.

4.2 Energy Conservation

A critical validation is energy conservation. In an isolated system with purely elastic collisions, the total kinetic energy should remain constant:

$$E_{\text{total}}(t) = \frac{1}{2}M|\mathbf{v}_{\text{large}}|^2 + \sum_{i=1}^{N}\frac{1}{2}m|\mathbf{v}_i|^2 = \text{constant} \tag{10}$$

Any deviation indicates numerical errors or incorrect collision physics. The simulation maintains energy conservation to within numerical precision by ensuring:

- Collision velocities computed using exact analytical formulas
- Particle separation after collision to prevent persistent overlap
- Collision throttling to avoid multiple impulses from a single physical collision

4.3 Comparison with Mean Squared Displacement

The previous simulations used Mean Squared Displacement (MSD) as the primary observable. However, MSD has limitations in bounded domains: it saturates when particles explore the entire available space. The velocity autocorrelation function is superior in confined geometries because it measures the rate of directional randomization, which remains meaningful even when spatial exploration is limited.

Furthermore, VACF provides information about the microscopic timescale of the collision process (τ_c), whereas MSD characterizes the macroscopic diffusion coefficient (D). Both are related through the fluctuation-dissipation theorem:

$$D = \int_0^\infty \langle \mathbf{v}(0) \cdot \mathbf{v}(\tau) \rangle \, d\tau \tag{11}$$

5 Software Architecture

5.1 System Overview

The collision-driven simulation uses a modular architecture separating physics, analysis, and rendering:

```
index.ts (UI Controller)
    |
    v
ElasticModel (AgentScript Model)
    |-- Large particle (M=30, r=8, initially at rest)
    |-- Small particles (~1000, m=1, r=1.5, Maxwell-Boltzmann)
    |-- step() -> Move → Collide → Analyze → Render
    |
    +-- elasticCollisions.ts -> Spatial hash + elastic physics
    +-- brownianAnalysis.ts -> Velocity autocorrelation (Chart.js)
    +-- simulation.ts -> Canvas rendering
```

5.2 Core Modules

ElasticModel (`elasticModel.ts`): Manages simulation state. Creates large particle at rest and small particles with `maxwellBoltzmannVelocity2D(T, m)` for thermal velocities. Samples large particle velocity every 10 ticks for analysis. Implements reset modes: "all", "count", "temperature", "nothing".

ElasticCollisions (`elasticCollisions.ts`): Implements spatial hashing with cell size $2r$. `performElasticCollision()` calculates post-collision velocities using $v_1' = [(m_1 - m_2)v_1 + 2m_2v_2]/(m_1+m_2)$ along collision normal. Includes collision throttling (`minCollisionInterval`) and particle separation to prevent overlap.

BrownianAnalysis (`brownianAnalysis.ts`): Calculates directional velocity autocorrelation $C(\tau) = \langle \hat{v}(t) \cdot \hat{v}(t+\tau) \rangle$. Maintains 500-point velocity history. Brownian detection when $C(3) < 0.7$. Updates Chart.js every 5 ticks.

Simulation (`simulation.ts`): Canvas rendering with coordinate transform (physics origin at center, Y-up). Handles device pixel ratio for high-DPI displays. Draws particles as circles with `ctx.arc()`.

5.3 Execution Flow

Each simulation step (`ElasticModel.step()`):

1. **Ballistic Motion**: `moveParticle()` updates position $\mathbf{r} \to \mathbf{r} + \mathbf{v}\Delta t$ with boundary reflection
2. **Collision Detection**: `handleAllCollisions()` builds spatial grid, checks cell neighborhoods
3. **Elastic Response**: For each collision pair:
   - Check overlap: $d < r_1 + r_2$
   - Check relative velocity: skip if separating
   - Apply impulse along normal: $\Delta v = [(m_1 - m_2)v_1 + 2m_2v_2]/(m_1 + m_2) - v_1$
   - Separate particles by `overlap + collisionBuffer`
4. **Velocity Sampling**: Every 10 ticks, record large particle velocity for autocorrelation
5. **Analysis Update**: Calculate $C(\tau)$ for lags $0 \ldots 25$, update chart
6. **Rendering**: Clear canvas, transform coordinates, draw all particles

5.4 Maxwell-Boltzmann Implementation

Small particles initialized with thermal velocities using Box-Muller transform:

```
function gaussianRandom() {
  u = Math.random(), v = Math.random()
  mag = sqrt(-2 * ln(u))
  return mag * cos(2*PI*v)  // Standard normal N(0,1)
}

function maxwellBoltzmannVelocity2D(T, m) {
  sigma = sqrt(T/m)
  return { vx: sigma*gaussianRandom(),
           vy: sigma*gaussianRandom() }
}
```

Large particle always initialized with $v_x = v_y = 0$ (completely at rest).

5.5 Spatial Hashing

Hash map with string keys `"cellX,cellY"`. For small particles, check 3×3 neighborhood. For large particle (radius 8), check 9×9 neighborhood to ensure all potential collisions detected.

5.6 Canvas Coordinate Transform

Physics space (origin at center) mapped to canvas space (origin top-left):

```
ctx.save()
ctx.scale(pixelsPerUnit, pixelsPerUnit)
ctx.translate(canvasWidth/(2*pixelsPerUnit),
              canvasHeight/(2*pixelsPerUnit))
// Draw in physics coordinates
ctx.restore()
```

6 Discussion

This simulation bridges the microscopic and macroscopic descriptions of Brownian motion. At the microscopic level, particles undergo deterministic elastic collisions governed by Newton's laws. At the macroscopic level, the large particle exhibits random walk behavior characterized by the diffusion equation.

The key insight is that randomness emerges from complexity. Although each collision is deterministic, the large number of collisions with unpredictable timing and angles produces effectively random motion. This is the essence of statistical mechanics: macroscopic randomness from microscopic determinism.

6.1 Limitations and Extensions

The current implementation assumes:

- **Point masses**: Particles have no rotational degrees of freedom
- **Hard-sphere collisions**: Instantaneous elastic collisions rather than soft potentials
- **2D dynamics**: Real colloidal systems exist in 3D (though 2D systems can be realized experimentally with particles confined to interfaces)
- **No hydrodynamics**: Real fluids have complex hydrodynamic interactions beyond simple collisions

Despite these simplifications, the simulation captures the essential physics of Brownian motion and provides quantitative agreement with theoretical predictions.

7 Conclusion

This elastic collision simulation demonstrates a fundamental principle of statistical mechanics: how macroscopic phenomena (Brownian motion, diffusion, random walks) emerge from microscopic deterministic dynamics (elastic collisions, Newton's laws). By eschewing artificial random walks in favor of pure collision physics, the simulation provides a more faithful representation of the physical processes underlying Brownian motion. The velocity autocorrelation function serves as a robust diagnostic for detecting the transition from ballistic to diffusive behavior, working effectively even in confined geometries where MSD-based methods fail. This project illustrates the power of computational physics to reveal emergent behavior and validate theoretical predictions from first principles.