Gravitational Painting:
N-Body Simulations in Dimensionless Units

1 Introduction

We present a systematic approach to generating training data for neural networks that learn gravitational N-body dynamics. Our method produces paired images: initial conditions (point configurations) and their temporal evolution (trajectory paintings), enabling machine learning models to predict dynamical outcomes from static initial states.

2 Dimensionless Units and Scaling Symmetry

2.1 Newtonian Gravity Scaling

The equations of motion for N-body gravitational systems are:

$$\ddot{\mathbf{r}}_i = -G \sum_{j \neq i} \frac{m_j(\mathbf{r}_i - \mathbf{r}_j)}{|\mathbf{r}_i - \mathbf{r}_j|^3} \tag{1}$$

These equations exhibit a scaling symmetry under the transformation:

$$m \to \lambda_m m, \quad \mathbf{r} \to \lambda_r \mathbf{r}, \quad t \to \lambda_t t \tag{2}$$

provided the constraint

$$\frac{\lambda_m}{\lambda_r^3} = \lambda_t^{-2} \tag{3}$$

is satisfied. This implies that the dimensionless quantity Gmt^2/r^3 is invariant under rescaling.

2.2 Choice of Units

This scaling symmetry allows us to fix any two of the three scales (mass, length, time), with the third determined by the constraint. We adopt the standard convention in N-body simulations:

- **Gravitational constant**: $G = 1$
- **Total mass**: $M = \sum_{i=1}^{N} m_i = 1$
- **Length scale**: Characteristic radius ~ 1

Time is then measured in dynamical units:

$$t_{\text{dyn}} \sim \sqrt{\frac{r^3}{GM}} \tag{4}$$

Benefits:

- All simulations are directly comparable in the same units
- No arbitrary physical scales (kg, meters, seconds)
- Physics is universal and scale-invariant
- Optimal for neural network training (normalized inputs/outputs)

3 Initial Conditions

3.1 Mass Distribution

Individual masses are drawn from a log-uniform distribution:

$$\log_{10} m_i \sim \mathcal{U}(0, \log_{10} \lambda) \tag{5}$$

where λ is the mass ratio (default: $\lambda = 10$). Masses are then normalized:

$$m_i \leftarrow \frac{m_i}{\sum_{j=1}^{N} m_j} \tag{6}$$

ensuring $\sum m_i = 1$.

Rationale: Log-uniform distribution samples the large dynamic range of astrophysical mass ratios (e.g., planetary systems, stellar clusters) while ensuring reasonable diversity.

3.2 Position Distribution

Positions are uniformly distributed in a disk of unit radius:

$$\theta_i \sim \mathcal{U}(0, 2\pi) \tag{7}$$

$$r_i \sim \sqrt{\mathcal{U}(0, 1)} \tag{8}$$

$$\mathbf{r}_i = r_i(\cos\theta_i, \sin\theta_i) \tag{9}$$

The $\sqrt{r}$ scaling ensures uniform density in the disk (area element $\propto r\,dr$).

3.3 Velocity Distribution

All bodies start with **zero initial velocity**:

$$\mathbf{v}_i(t = 0) = \mathbf{0}, \quad \forall i \tag{10}$$

Rationale: This represents a "cold start" or gravitational collapse scenario, where the system begins from rest and evolves purely under mutual gravitational attraction. This choice:

- Simplifies the initial state (positions and masses only)
- Produces interesting, non-trivial dynamics (collapse, close encounters, ejections)
- Reduces the input space for the neural network

4 Numerical Integration

We use the 4th-order Runge-Kutta (RK4) method to integrate the equations of motion:

$$\mathbf{r}_i(t+\Delta t),\ \mathbf{v}_i(t+\Delta t) = \text{RK4}(\mathbf{r}_i(t), \mathbf{v}_i(t), \Delta t) \tag{11}$$

Parameters:

- Simulation time: $T = 50$ (dynamical units, default)
- Number of steps: $N_{\text{steps}} = 10{,}000$ (default)
- Time step: $\Delta t = T/N_{\text{steps}} = 0.005$

The RK4 method provides a good balance between accuracy ($\mathcal{O}(\Delta t^4)$ per step) and computational efficiency.

5 Visualization

5.1 Initial Points Image

Bodies are rendered as Gaussian spots with radii proportional to $\log_{10} m_i$:

$$r_{\text{dot},i} = r_{\min} + (r_{\max} - r_{\min}) \cdot \frac{\log_{10} m_i - \log_{10} m_{\min}}{\log_{10} m_{\max} - \log_{10} m_{\min}} \tag{12}$$

where $r_{\min} = 2$ pixels, $r_{\max} = 10$ pixels.

Coordinate frame: Centered at the center of mass, with extent $\pm 1.25 \times r_{\max}$ where $r_{\max} = \max_i |\mathbf{r}_i - \mathbf{r}_{\text{COM}}|$.

5.2 Trajectory Painting

Trajectories are rendered by accumulating intensity along each body's path. At each time step k, we add a Gaussian spot with:

Line width: Proportional to mass (25% of initial dot size):

$$r_{\text{line},i} = 0.25 \times r_{\text{dot},i} \tag{13}$$

Intensity: Proportional to $\log_{10} |\mathbf{v}_i|$, normalized and scaled:

$$I_{k,i} = 0.3 \times \frac{\log_{10} |\mathbf{v}_{k,i}| - \log_{10} v_{\min}}{\log_{10} v_{\max} - \log_{10} v_{\min}} \tag{14}$$

where $v_{\min} = 1.0$ avoids $\log(0)$ and the factor 0.3 limits maximum brightness to 30%.

Final positions: Overlaid as white dots (same size as initial points) with black outlines for contrast.

Rationale:

- Logarithmic scaling captures wide velocity range (slow $\rightarrow$ fast motion)
- Lower intensity (30%) keeps paintings from saturating
- Thinner lines prevent visual clutter
- Additive accumulation shows trajectory density

6 Data Generation for Neural Networks

6.1 Dataset Structure

Each sample consists of a paired image set:

- **Input**: Initial points image (positions + masses)
- **Output**: Trajectory painting (evolved dynamics)

Files are named: `prefix_n{N}_seed{S}_{points|painting}.png`

6.2 Recommended Dataset Sizes

- **Proof of concept**: 500–1,000 examples per N
- **Production training**: 5,000–10,000 examples per N
- **Large-scale**: 20,000–50,000 examples for $N \geq 10$

Note: Train separate models for each N or include N as an input feature, since complexity scales as $\mathcal{O}(N^2)$.

6.3 Data Augmentation

The physics is rotationally invariant, enabling augmentation:

- Random rotations by $\theta \in [0, 2\pi)$
- Horizontal/vertical flips
- Effectively multiplies dataset size by ~ 8

7 Neural Network Architecture Considerations

Input: 2048×2048 grayscale image (initial points)
Output: 2048×2048 grayscale image (trajectory painting)
Recommended architectures:

- U-Net: Standard for image-to-image translation
- Conditional GAN (pix2pix): For realistic texture generation
- Physics-informed neural networks: Incorporate gravitational constraints

Loss functions:

- L1 or L2 pixel-wise loss
- Perceptual loss (VGG features)
- Adversarial loss (if using GAN)

8 Computational Details

Image resolution: 2048×2048 pixels (grayscale, 8-bit)

Rendering optimization: Local bounding boxes around each trajectory point (3σ cutoff) reduce computation from $\mathcal{O}(N \cdot N_{\text{steps}} \cdot W \cdot H)$ to $\mathcal{O}(N \cdot N_{\text{steps}} \cdot r^2)$ where $r \ll \min(W, H)$.

Typical generation time: $\sim$20–60 seconds per sample (depending on N and N_{steps}).

9 Summary

Our simulation framework leverages the scale-invariance of Newtonian gravity to produce a universal, dimensionless dataset of N-body dynamics. Key design choices:

1. Dimensionless units ($G = 1$, $M = 1$) ensure scale-invariance
2. Zero initial velocities simplify input and produce interesting collapse dynamics
3. Log-scaling (mass $\rightarrow$ size, velocity $\rightarrow$ intensity) captures wide dynamic ranges
4. High-resolution rendering (2048^2) preserves fine trajectory details
5. Systematic naming convention enables organized dataset management

This approach is optimized for training neural networks to learn gravitational dynamics from static initial conditions.