GPU-Optimized N-Body Simulation: Project Guide & Video Script

1. Project Overview & File Breakdown

The Core Files

- `optimized_barnes_hut.cu` **(The Engine)**
  - **What it does:** This is the source code written in CUDA C++. It implements the **Barnes-Hut algorithm**, which uses an Octree data structure to group distant particles. This reduces the computational complexity from O(N^2) to O(N log N).
  - **Key Feature:** It uses GPU parallel processing (kernels) to calculate gravitational forces for thousands of bodies simultaneously.
- `optimized_simulation` **(The Executable)**
  - **What it does:** This is the compiled binary file. When executed, it runs the actual simulation on the NVIDIA GPU.
  - **Process:** It initializes the bodies, builds the octree every frame, computes forces, and updates positions.
- `optimized_output.txt` **(The Data)**
  - **What it does:** This is the storage file for the simulation results.
  - **Content:** It stores the exact state of the bodies at every recorded time step.
  - **Format:** Rows of numbers representing `Position X`, `Position Y`, `Position Z`, and `Mass` for every body.
- `visualize_complete.py` **(The Visualization)**
  - **What it does:** A Python script using `matplotlib` and `numpy`.
  - **Function:** It reads the raw numbers from `optimized_output.txt` and converts them into visual data, specifically static images (Start/Middle/End) and a coherent animation (`.gif`) of the galaxy rotating.

2. The Workflow (Steps)

1. **Compilation:** The `.cu` file is compiled using `nvcc` to create the executable.
   - *Command:* `nvcc optimized_barnes_hut.cu -o optimized_simulation -O3 -arch=sm_70`
2. **Simulation:** Run the executable. It performs the math and writes data to the text file.
   - *Command:* `./optimized_simulation`
3. **Visualization:** Run the Python script to interpret the text data and generate the video.
   - *Command:* `python3 visualize_complete.py`