Simulation Requirements – MagNanoBridge (Updated with Expert Feedback)

1. Introduction

This document outlines the strict and detailed requirements for the development of a realistic magnetic nanoparticle simulation intended to model the temporary bridging of a bone fracture using magnetically guided particles. The simulation must be physically plausible, modular, clearly documented, and allow both interactive and pre-configured operation modes.

2. Project Folder Structure

The project must follow a modular, well-organized folder structure, suitable for collaborative scientific development and future expansion. The structure must separate core simulation logic, visualization, and documentation, with clear entry points and file responsibilities. A single monolithic file is explicitly not allowed.

The required structure (or a functionally equivalent layout) is as follows:

```
MagNanoBridge/
├── src/                # Core Python code for the simulation
│   ├── main.py             # Main simulation entry point (minimal logic)
│   ├── physics/            # Modules: magnetism, drag, dipole forces, collisions
│   ├── control/            # PID/AI-based control of field dynamics
│   ├── utils/            # Math helpers, constants, tools
│   └── config/             # YAML/JSON configs for full reproducibility
├── visualization/          # Web interface (Three.js/HTML) synced with simulation state
│   ├── index.html
│   ├── threejs/            # Modular JS components
│   └── static/           # Icons, stylesheets, shaders
├── notebooks/              # Jupyter/Colab exploration, analysis, optimization
├── data/               # Simulation results: raw, processed, export
├── tests/              # Unit + integration tests for physics/logic (required)
├── doc/                # User/dev documentation, architecture, manuals
└── requirements.txt        # Dependencies for Colab and offline use
```

Additional requirements:

- • Python source code must be modular. No single-file architecture is allowed.
- • The Python code must run in Google Colab with necessary installation cells.
- • Three.js visualization must reflect the actual simulation state (not animations).
- • Code must follow PEP-8 or equivalent standards and be documented.
- • Offline and online usage must be supported.
- • README.md must describe usage, setup, structure, and contribution notes.

3. Core Functional Requirements

The simulation must:

- • Allow the definition of a 3D FOCUS FIGURE (e.g., narrow cylinder or STL mesh) as target volume.
- • Support manual or file-based (CSV/JSON) configuration of this focus zone.
- • Simulate the movement of magnetic nanoparticles influenced by multiple external coil fields.
- • Each nanoparticle must have: realistic mass, volume, dipole, and plastic shell.
- • Dipole-dipole interaction between particles must be implemented (not approximated).
- • Dipole logic must scale to >1000 particles. If needed, Fast Multipole Method or approximation must be discussed.
- • Plastic shell must have adjustable thickness, separately visualized and physically present.
- • Collisions must follow a Lennard-Jones or equivalent soft-sphere model.
- • Brownian motion must be optionally included for nanoscale realism (via Langevin dynamics).
- • Motion integration should use a stable method (e.g., Velocity Verlet or RK4).
- • Field sources (coils) must be real (e.g., Biot-Savart) and not ideal dipoles unless justified.
- • PID or AI-based control system must guide particles into focus zone in 'focus mode'.
- • The objective function (e.g., density, volume coverage) for bridging success must be clearly defined.
- • Real-time 3D rendering (Three.js) of full particle system with trails, shell, size.
- • Allow slow-motion visual playback of movement, with toggle.
- • Enable ID-based tracking and highlighting of individual particles.
- • Allow switch between 3D and 2D (cross-section) debug views.
- • Export CSV or JSON of position, velocity, forces over time.
- • Batch mode for non-visual, AI-optimized or multi-run experiments must be included.

4. Grounds for Rejection

The project will be rejected and payment withheld if any of the following conditions are met:

- • Any single core requirement is missing or only partially implemented.
- • 3D focus zone is not functioning, controllable, or is not volumetric.
- • Particles behave unrealistically (e.g., teleport, ignore forces).
- • No mutual dipole-dipole force simulation or poorly implemented version.
- • Collision logic missing or unphysical.
- • Simulation logic and rendering not aligned (e.g., faked visuals).
- • Missing or incomplete modular structure.
- • No data export, test suite, or config-based reproducibility.
- • Time-varying or induced fields implemented against specifications.

- • No PID/AI logic or no defined objective for 'focus mode'.

Note: Partial delivery, 'work in progress', or placeholder elements will not be accepted unless explicitly agreed upon in advance and in writing.

6. Ownership and Legal Enforcement

1. **Exclusive Ownership**:
   Upon acceptance of this project, **all intellectual property rights, source code, simulations, documentation, visualizations, and derivative works become the sole and exclusive property of the commissioning party (Oscar van Maarseveen)**. The developer or contractor waives any and all rights to reuse, resell, republish, or relicense any portion of the work, unless explicitly authorized in writing by the commissioning party.
2. **Non-Disclosure and Exclusivity**:
   The contractor agrees not to share, discuss, or demonstrate any part of the developed codebase, logic, data, or visualization techniques with any third party, under any circumstances.
3. **Breach and Penalty Clause**:
   In the event of unauthorized reuse, reproduction, publication, or distribution of the code, concepts, or simulation system, **a contractual penalty of €100,000 per day will be due**, plus **€2,500 per additional calendar day**, without prejudice to any further claims for actual damages.
4. **Jurisdiction**:
   This agreement shall be governed by Dutch law. In the event of any legal dispute, the competent court shall be the **Rechtbank Midden-Nederland, location Utrecht**, unless otherwise mandated by law.