System Architecture: The "Dual-Engine" Approach

The Tetrahedral Engine utilizes a split-architecture design to maximize performance for both simulation and rendering. This document outlines the separation of concerns between the **Volumetric Lattice (Physics)** and the **Tetrahedral Skewer (Rendering)**.

1. The Storage Layer: Rhombic Dodecahedral Lattice

The fundamental unit of the world is the **Rhombic Dodecahedron (RD)**.

- **Why:** The RD is the space-filling dual of the Face-Centered Cubic (FCC) lattice. It provides 12 equidistant neighbors, superior packing density, and integer-based coordinates.
- **Data Structure:** The world is stored not as a list of coordinates, but as a **Graph of Connected Cells**.

Navigation & Physics ("The Hopper")

Movement within the engine is defined as **Graph Traversal**.

- **Movement:** An entity does not "slide" along float coordinates. It "hops" from Cell A to Neighbor B (Directions 1-12).
- **Collision:** Collision detection is instant O(1) lookup. The entity queries the state of its 12 neighbors to determine valid paths.
- **Fluid Dynamics:** Simulation of gases and liquids utilizes Cellular Automata rules over the 12-neighbor graph, preventing "Manhattan Distance" artifacts common in 6-neighbor cubic grids.

2. The Rendering Layer: Tetrahedral Skewer

While the data is stored in RDs, the viewing system utilizes the **Tetrahedral Coordinate System** (S, T, U).

- **Why:** Rendering requires converting 3D volume data into a 2D plane. The Tetrahedral Skewer offers a mathematically perfect way to "pierce" the RD lattice and sample density without aliasing.

Visualisation ("The Skewer")

- **Ray-Casting:** A viewing ray is defined by (S, T) slit coordinates.
- **Intersection:** The ray pierces the volume. Because the (S, T, U) coordinates map 1:1 to the RD Lattice, the ray can instantly calculate which sequence of cells it passes through.
- **Sampling:** The engine samples the color/density of the intersected RD cells and accumulates the pixel value.

3. The Bridge: Coordinate Mapping

The system relies on a seamless translation between the two spaces:

- **Input (Ingest):** XYZ \rightarrow STU \rightarrow \text{RD-ID} (Bitwise Sorting)
- **Simulation:** \text{RD-ID} \leftrightarrow \text{RD-ID} (Neighbor Hopping)
- **Output (Render):** \text{Ray}(S,T) \rightarrow \text{RD-ID} \rightarrow \text{Pixel}
-

Architecture defined Jan 2026.