THE TETRAHEDRAL PROTOCOL

A Resolution-Independent Standard for Volumetric Vignettes

Author: Kevin Forseth

Date: January 2026

Status: Draft v1.0

1. INTRODUCTION

The history of 3D computer graphics has been defined by the pursuit of the "Infinite World." From the earliest flight simulators to modern open-world games, the fundamental assumption of the rendering pipeline has been **Egocentric**: a camera floating inside an unbounded Cartesian grid, looking out at a horizon.

This model has served the screen-based era well. However, the rise of Spatial Computing (Augmented Reality, Virtual Reality, and Holographic Displays) has revealed a critical gap in our digital infrastructure.

We do not have a native format for **"The Thing."**

When a user places a virtual chessboard on their coffee table, or an engineer inspects a holographic engine part, they are not interacting with a "World." They are interacting with a **Vignette**—a bounded, self-contained volume of high-fidelity data. Current polygon-based engines struggle to render these objects convincingly. They suffer from aliasing when viewed up close, "clipping" artifacts when touched, and massive file sizes that prevent easy sharing.

The Tetrahedral Engine proposes a new paradigm: Object-Space Volumetric Rendering.

By abandoning the infinite horizon in favor of a bounded **Root Tetrahedron**, and replacing the perspective camera with an **Exocentric Ray-Net**, we can create digital artifacts that possess the physical integrity of real matter. These objects are not "pictures" of 3D models; they are **mathematically continuous volumes** defined by a Rhombic Dodecahedral lattice.

This paper outlines the architecture of a system designed not to render a world, but to perfect the **Volumetric Vignette**: a resolution-independent, physically interactive, and mathematically pure digital object.

2. THE PROBLEM: THE "POLYGON TRAP"

Current 3D standards (GLTF, USD, OBJ) are surface-based. They define an object by wrapping a hollow skin of triangles around an empty void.

1. **The Interaction Failure:** If a user reaches their hand into a polygon object, they clip through the skin and see the empty back-face of the mesh. The illusion of solidity is broken immediately.
2. **The Resolution Failure:** As the user leans in to inspect a detail, the illusion of curvature breaks down. A round cup reveals its jagged polygonal edges. To fix this, developers must use "LODs" (Level of Detail), swapping models in and out, which causes visual popping.
3. **The "Skybox" Waste:** Traditional engines are optimized to render pixels on a screen, processing the background sky and the distant terrain with the same logic as the hero object. For a Vignette, this is wasted computation.

3. THE ATOMIC UNIT: BEYOND THE CUBE

In a traditional voxel engine (like Minecraft or medical MRI), the world is divided into a grid of **Cubes**. While mathematically simple, the Cube is a poor approximation of physical space.

- **The Diagonal Problem:** A Cube has 6 face neighbors, but 20 "corner" or "edge" neighbors. This creates "anisotropic" (direction-dependent) physics. A fluid flowing diagonally moves faster than a fluid flowing straight, breaking the simulation.
- **The Surface Problem:** A Cube has sharp corners. When used to represent organic shapes (a face, a cloud, a stone), it creates the jagged "staircase" effect known as aliasing.

The Solution: The Rhombic Dodecahedron (RD)

The Tetrahedral Protocol utilizes the Rhombic Dodecahedron as its fundamental spatial unit. This shape is the geometric dual of the Face-Centered Cubic (FCC) lattice, the tightest possible packing of spheres in 3D space.

3.1 The "Super-Pixel" of 3D Space

The RD offers three critical advantages for Vignette rendering:

1. **Isotropic Connectivity:** Each RD voxel has **12 identical neighbors**. This allows for "Round Physics." Simulation data (heat, liquid, stress) flows through the lattice with near-perfect uniformity, eliminating the "Manhattan Distance" artifacts of cubic grids.
2. **Gapless Tessellation:** Unlike spheres (which leave gaps) or complex polyhedra (which don't stack), RD voxels stack perfectly to fill 100% of the volume.
3. **The Integer Inversion:** The RD is geometrically derived as the inversion of a Cube. This allows the engine to maintain a **2:1 Integer Volume Ratio**, enabling high-speed integer arithmetic for coordinate lookups while retaining the physical benefits of a complex 12-sided shape.

3.2 "Solid Matter" Encoding

Because the Vignette is finite, we do not need to optimize for "empty air" as aggressively as an open-world engine. Instead, we optimize for **Density**.

The RD Lattice allows us to treat the object as a solid block of data. A "cut" through the object reveals the internal grain of the 12-sided cell structure, providing a visual texture that feels closer to crystalline matter than hollow polygons.

4. THE OPTICAL ENGINE: EXOCENTRIC RAY-CASTING

The defining challenge of volumetric rendering is projecting a 3D field onto a 2D plane without losing information. Standard methods (like "Splatting" or "Marching Cubes") approximate the volume by drawing thousands of tiny billboards or extracting a hollow surface mesh. Both methods introduce aliasing and destroy the internal data structure.

The Tetrahedral Engine utilizes a novel **Exocentric Ray-Casting** method known as the **Tetrahedral Skewer**.

4.1 The Skew-Axis Coordinate System (S, T, U)

Instead of a traditional Camera (which uses a 4\times4 perspective matrix), the engine defines the view using two non-parallel slits (The "Event Horizons").

- **Ray Definition:** Every pixel on the display corresponds to a unique ray defined by the coordinates (S, T), where S is the position along the first slit and T is the position along the second.
- **The "Skewer" Calculation:** This (S, T) ray pierces the Rhombic Dodecahedral lattice. Because the lattice is derived from integer geometry, the intersection of the ray with the voxel cells can be calculated using bitwise logic rather than expensive floating-point trilinear interpolation.

4.2 Resolution Independence

This optical path allows for **Infinite Zoom**.

- In a polygon engine, zooming in on an object reveals the flat facets of the geometry.
- In the Tetrahedral Engine, zooming in simply changes the sampling density of the ray. As the ray passes through the lattice, it can sample the "Solid Matter" at any frequency. The visual result is an object that appears to have no discrete "resolution," much like looking at a physical object through a magnifying glass.

4.3 The "Vignette" Advantage

Because the system is rendering a bounded Vignette, the Ray-Caster can employ **"Early Ray Termination"** aggressively. The moment a ray exits the back of the Root Tetrahedron, computation stops. This makes the engine highly efficient for AR "Tabletop" scenarios, as no computation is wasted on the empty environment surrounding the object.