Modeling dust clouds in GEANT4

Ara N. Knaian, Ph.D.
Nathaniel MacFadden
NK Labs, LLC
Cambridge, MA, USA
18-January-2019

Motivation

- Simulate a particle beam passing through a dusty environment.
- Need to efficiently model billions of dust particles in the 1 μm - 1 mm range
- Generic Example Applications:
  - Ionization type smoke detector
  - Industrial dusty plasma
  - Space dusty plasma
  - Cosmic rays on a rainy day
- We have developed a special geometry class to make this easy and efficient.

Potential Geometry Approaches

- Approximate as a low density solid
  - Does not effectively model the physics; some particles stop within a single dust grain, others travel a long way and interact with fields
- Brute-force geometry creation
  - Uses a large amount of RAM.
  - 1% density of 50μm dust particles in 1 m^3 = 80B sphere instances = out of memory
- Parametrized Volume
  - Saves RAM, but uses CPU time to close the geometry
- Replicated Volume
  - Create a small cube of randomly-placed particles and replicate it in X,Y,Z
  - Spheres do not cross the faces of the replica; results in lower density near the boundaries
  - Statistics are less random for particles travelling along the cartesian directions
- Model the whole cloud as a single solid

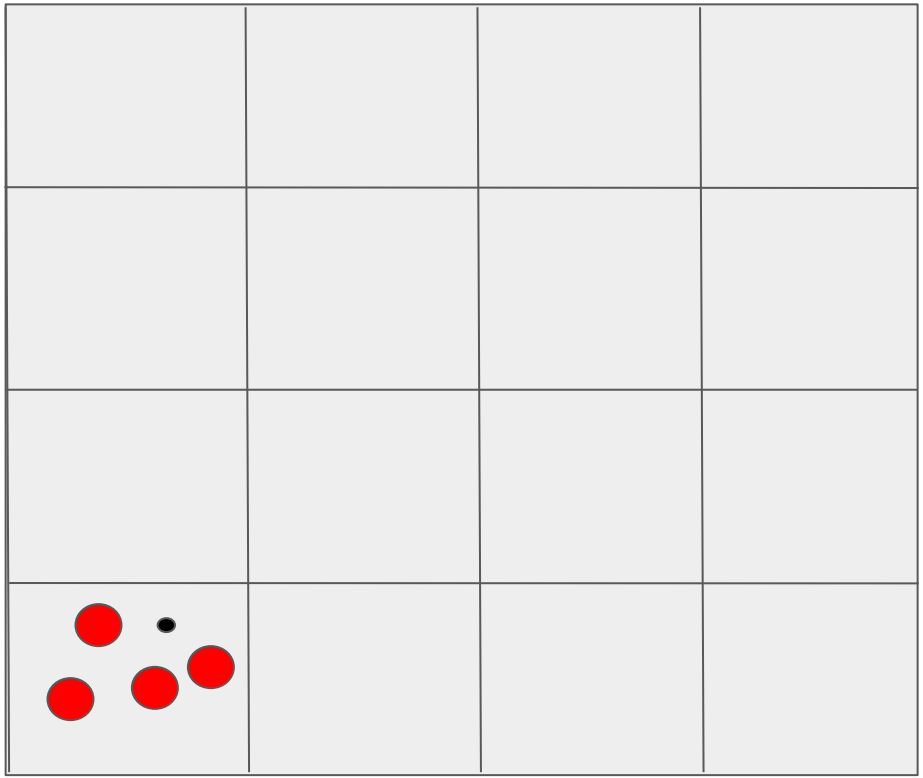
Our Approach

- Model the whole cloud as a single solid
- Compute the sphere locations dynamically, only as they are needed to respond to geometry callbacks from the GEANT4 kernel.

Illustration

- As the primary and secondary particles move through the geometry, sphere locations are computed dynamically in the grids needed to respond to the function calls of a solid. (e.g. distanceToIn, distanceToOut)

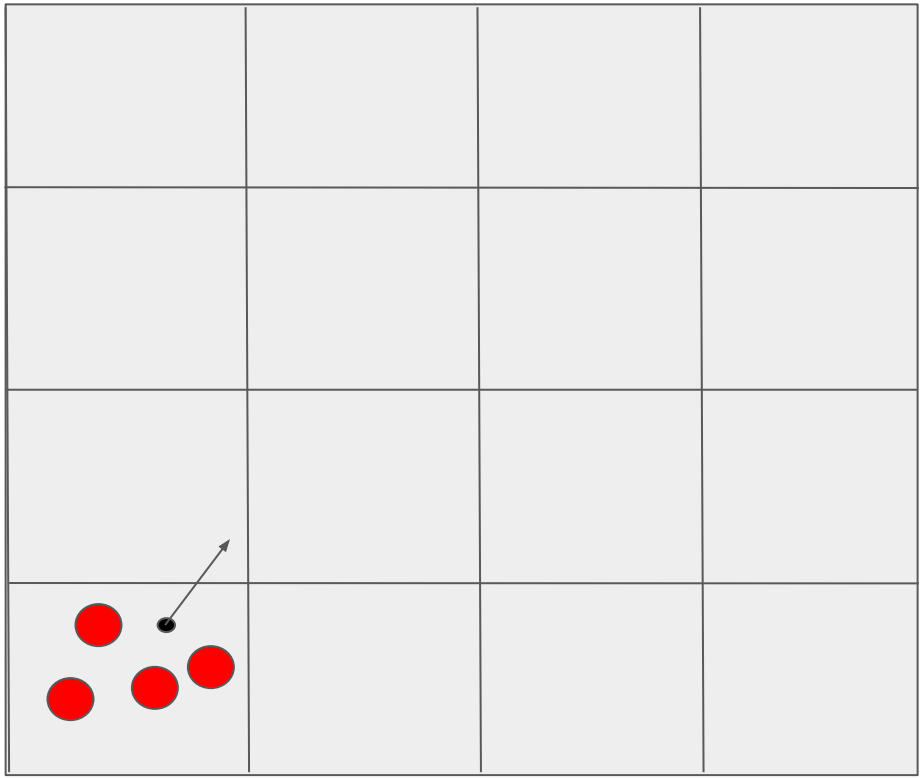
- The code shoots a Poisson random to get the number of spheres to add to a given grid, then shoots Uniform randoms to get coordinates for each. In a sphere collision, a new point is tried until successful placement.



Illustration

- As the primary and secondary particles move through the geometry, sphere locations are computed dynamically in the grids needed to respond to the function calls of a solid. (e.g. distanceToIn, distanceToOut)

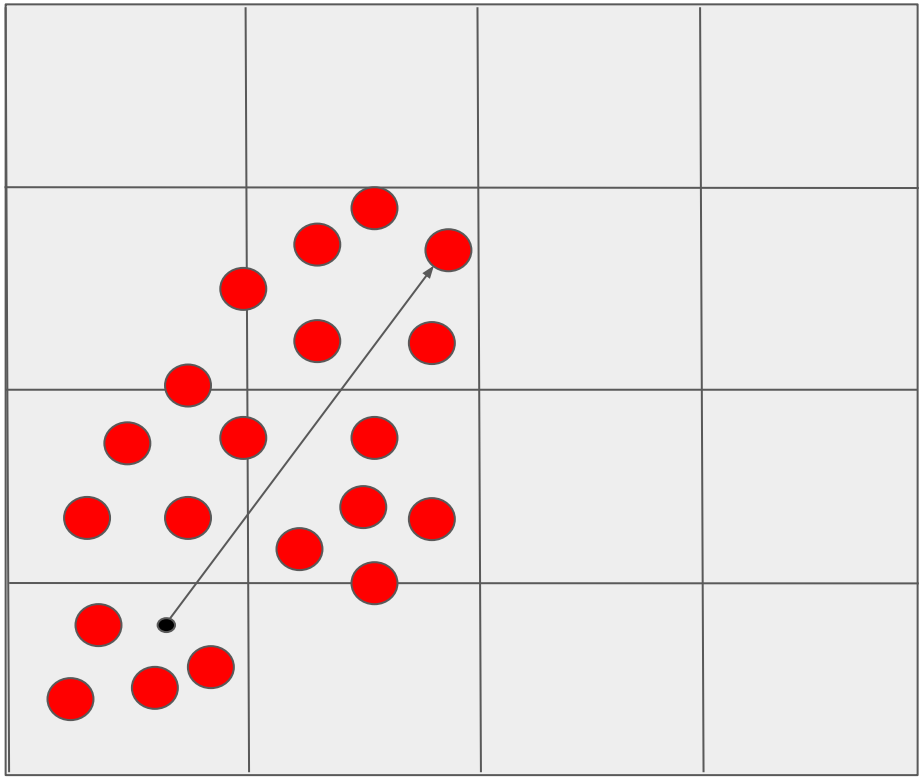
- The code shoots a Poisson random to get the number of spheres to add to a given grid, then shoots Uniform randoms to get coordinates for each. In a sphere collision, a new point is tried until successful placement.



Illustration

- As the primary and secondary particles move through the geometry, sphere locations are computed dynamically in the grids needed to respond to the function calls of a solid. (e.g. distanceToIn, distanceToOut)

- The code shoots a Poisson random to get the number of spheres to add to a given grid, then shoots Uniform randoms to get coordinates for each. In a sphere collision, a new point is tried until successful placement.



CloudOfSpheresSolid

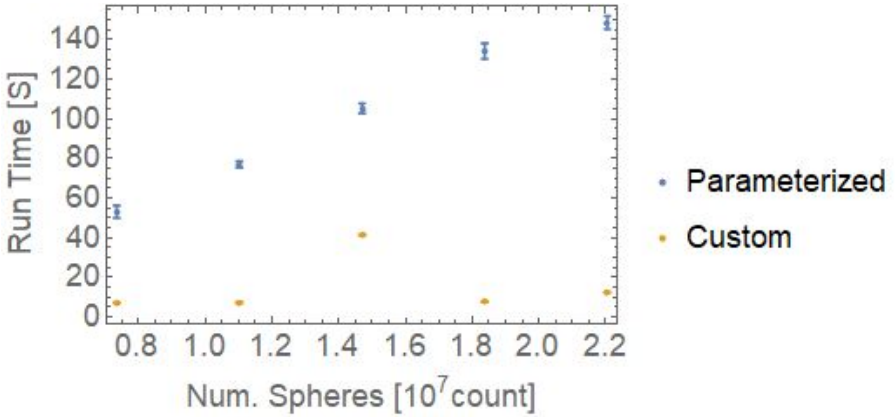
- Single geometry object represents a cloud of randomly-placed solid spheres.
- User specifies average volume fraction and sphere diameter.
- Overall shape of the cloud is a box.
- Supports event-level multithreading.
- Can handle giant clouds of tiny particles; we regularly run simulations involving hundreds of billions of solid dust particles using GEANT4. These simulations run in a few seconds per event on a 72-core AWS c5.18xlarge.

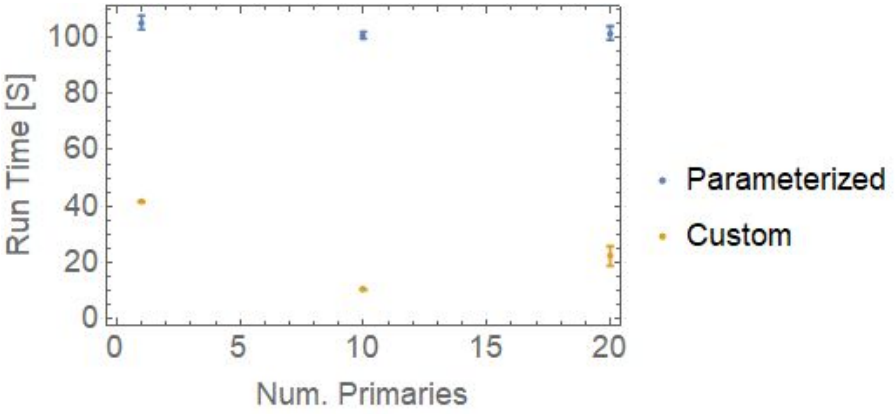
Usage Example

```
CloudOfSpheres* cloud = new CloudOfSpheres(outerRadius, minSpacing, gridPitch,
                        -0.5*cloud_size_XY, 0.5*cloud_size_XY,
                        -0.5*cloud_size_XY, 0.5*cloud_size_XY,
                        -0.5*cloud_size_Z, 0.5*cloud_size_Z,
                        numSpheres, maxVectorFollowingDistance,
                        firstX, firstY, firstZ);

CloudOfSpheresSolid* solidShell = new CloudOfSpheresSolid("shell", cloud, outerRadius,
                                                          innerRadius);

CloudOfSpheresSolid* solidCore = new CloudOfSpheresSolid("core", cloud, innerRadius,
                                                         0);
```

Performance Test Data

- Results from shooting a 3 GeV proton through a cube of dust 10cm in side length, using modified exampleB1
- Each time trial was run 3 times on a basic laptop
- For N spheres, runtime appears O(1) for cloudOfSpheresSolid and O(N) using a parameterized geometry.





Questions

- What are the generic applications of dusty environment modelling capability to the GEANT4 community? (Particle physics, medical, space, industrial, etc.)
- What features should we add?
- What tests should we perform?