Chroma Ray Tracer

Motivation was reconstruction in Cherenkov detectors:

$$L(x,y,z,t,\vec{p},ID) \sim \prod_{i=1}^{N_{PMT}} p_i(t_i^{res},Q_i \mid x,y,z,t,\vec{p},ID)$$

Why not use complete MC simulation to generate p_is?

i.e., for a hypothesized x,y,z,t,p,ID simulate N complete events and tally up the p_i based on MC prediction

Then fitter moves along in the 7 dimensions (for a given ID) and tries a new point in that space and continues until L is maximized…

This should in principal be the best one could do.

But

- Waaaay too slow: 1 event would take ~few years
- Likelihood space is stochastic---how to find minimum?

Chroma Ray Tracer

Created by S. Seibert and A. LaTorre

Chroma is a GPU-accelerated raytracer and Monte Carlo photon simulation written in a mixture of Python and CUDA:

- Uses GEANT4 or your own software to create initial photon vertices. (Included generator does Cherenkov light; bolt on your own for other light production.)
- Up to 200x faster than GEANT4 at simulating 1 GeV electrons in an LBNE-sized detector. (millions of photons steps/sec)
- Designed to be used either as a library for other Python applications, a server for communication with existing C++ programs, or standalone use.

S. Seibert

Chroma Ray Tracer

Can be used for many things:

- An interactive 3D geometry exploration and event visualization application.
- A standalone Monte Carlo simulation of electrons, muons, and π_0 events.
- A “Monte Carlo photon propagation server” that can be used by existing GEANT4-based simulations.
- A PDF generation tool
- A time/hit likelihood calculator
- Hypothesis testing of hit patterns

S. Seibert

Chroma Ray Tracer

To win 200x in photon propagation, we had to combine two different lines of technological development:

- Better software: Computer graphics researchers have developed many techniques for efficiently tracing the paths of photon rays. These techniques focus on *surfaces*, rather than *solids*.

  Existing rendering libraries try to avoid physics whenever possible! Can't just download Blender/Maya/POV-Ray...

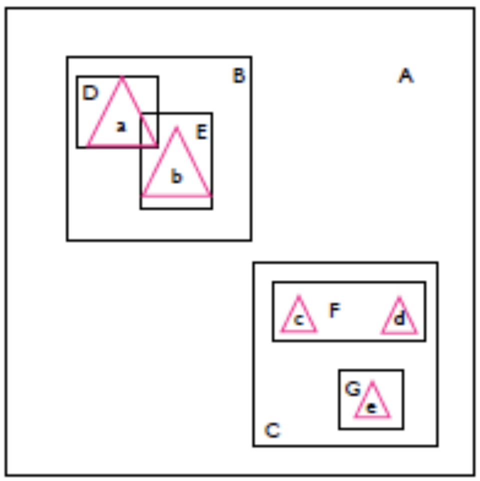
- Better hardware: Standard graphics processing units (GPUs) are now fully programmable, massively parallel vector machines. Moreover, the exponential growth of computing power in GPUs is currently faster than CPUs.

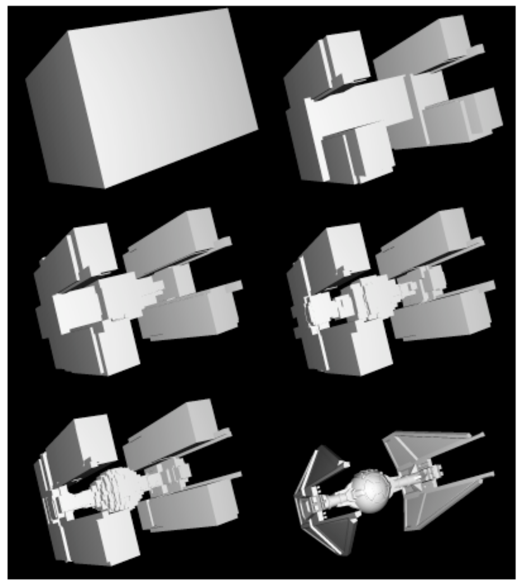
S. Seibert

Chroma Ray Tracer

<u>Very</u> fast photon ray-tracing using graphics cards (GPUs)

To make this work, Seibert and LaTorre had to re-invent the way geometry is handled in particle physics simulations

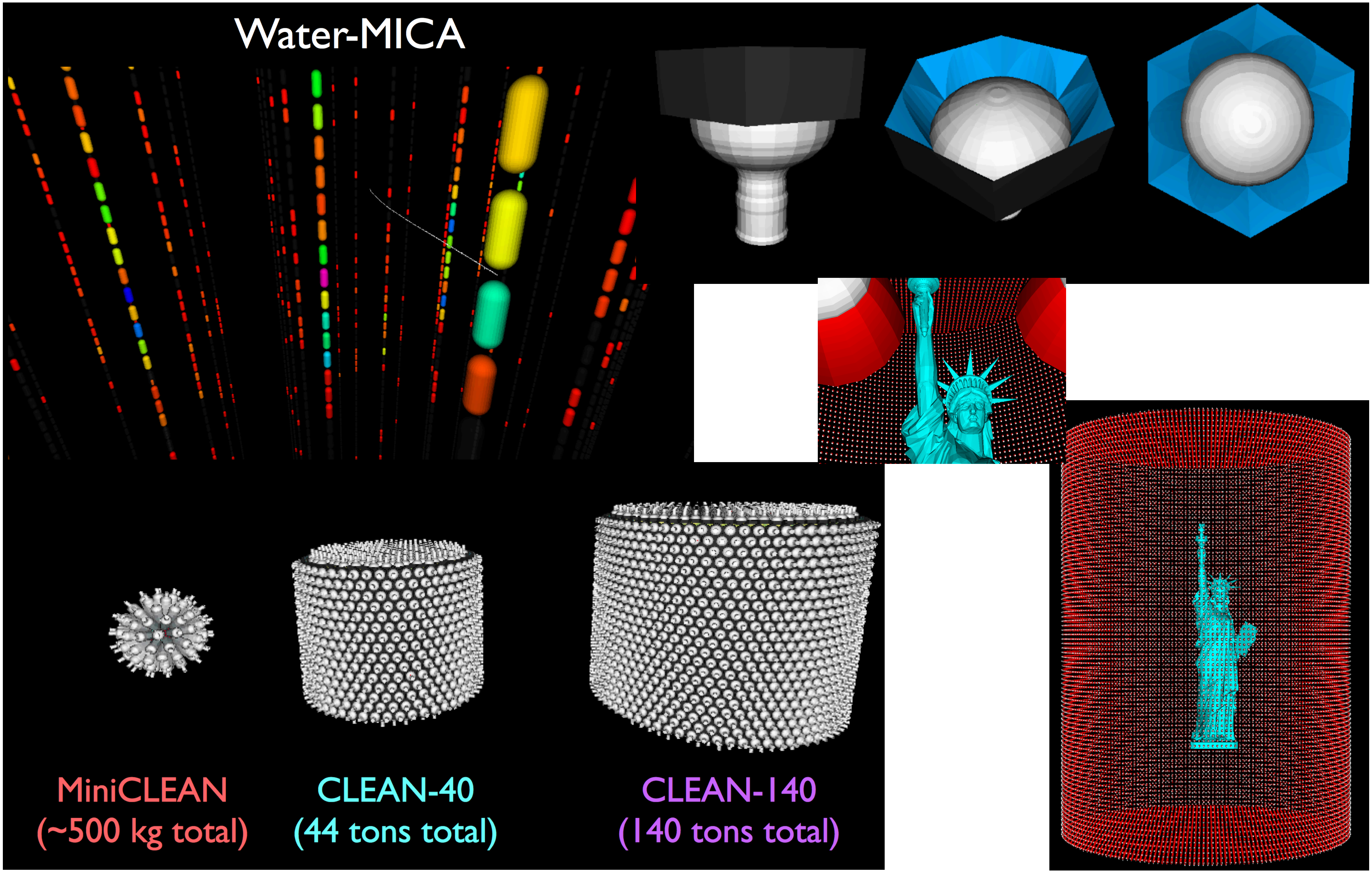




`Triangle mesh' surfaces (CAD) rather than volumes (GEANT4)

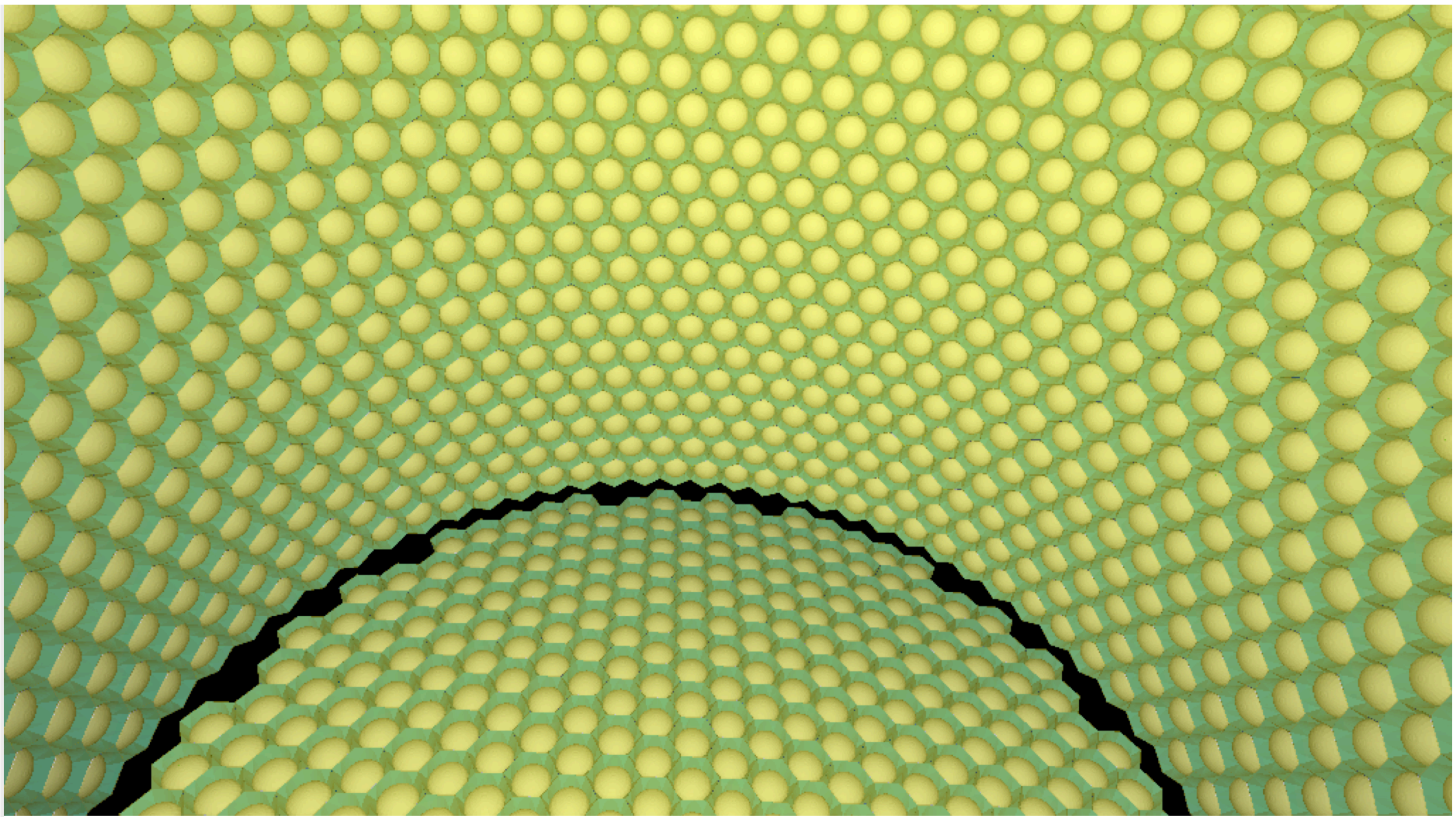
Chroma Ray Tracer

Describing detectors is relatively easy:



Chroma Ray Tracer

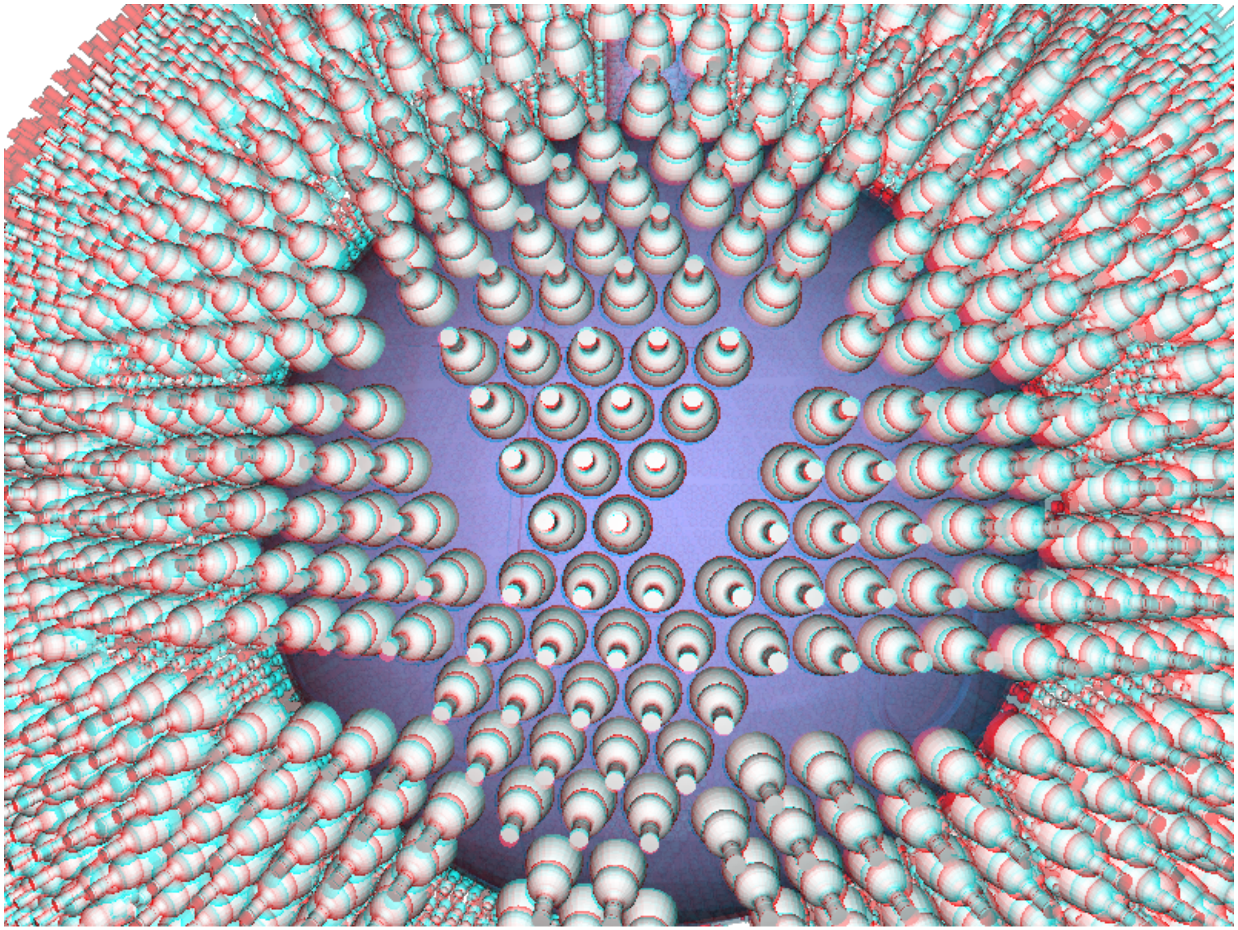
Describing detectors is relatively easy:

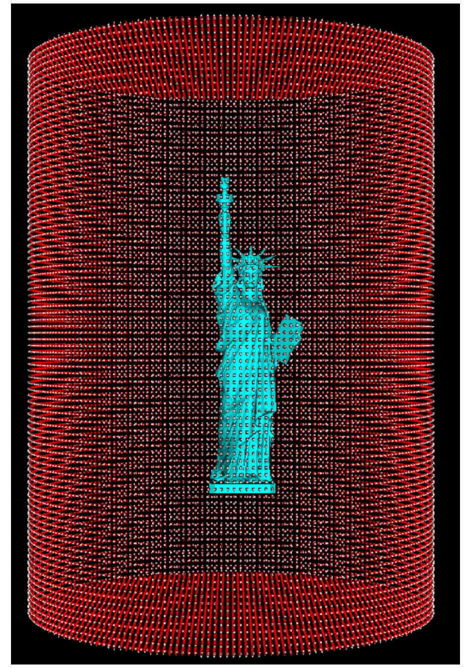


Inside 140 t dark matter detector

Chroma Ray Tracer

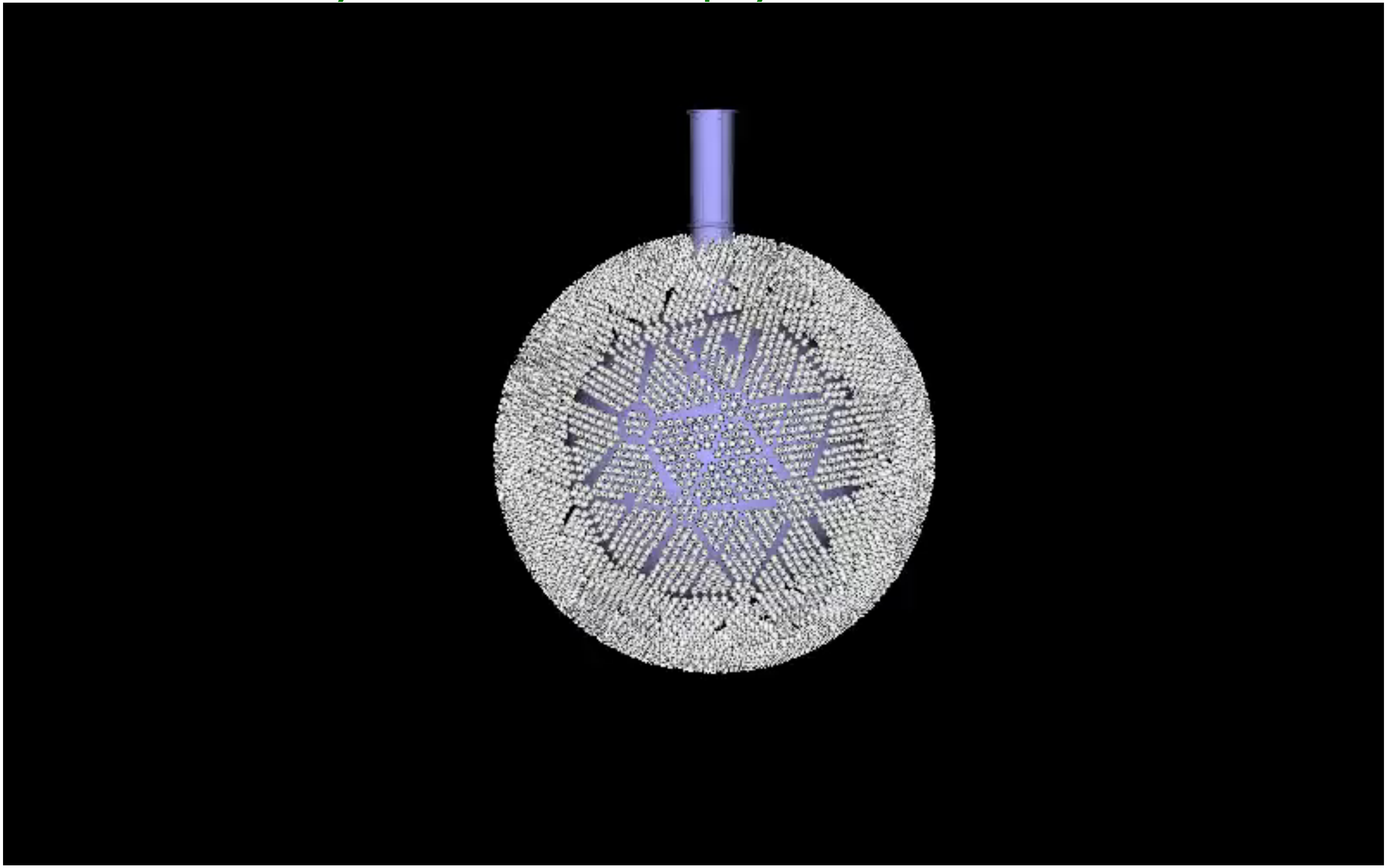
Side benefit: very nice detector displays….

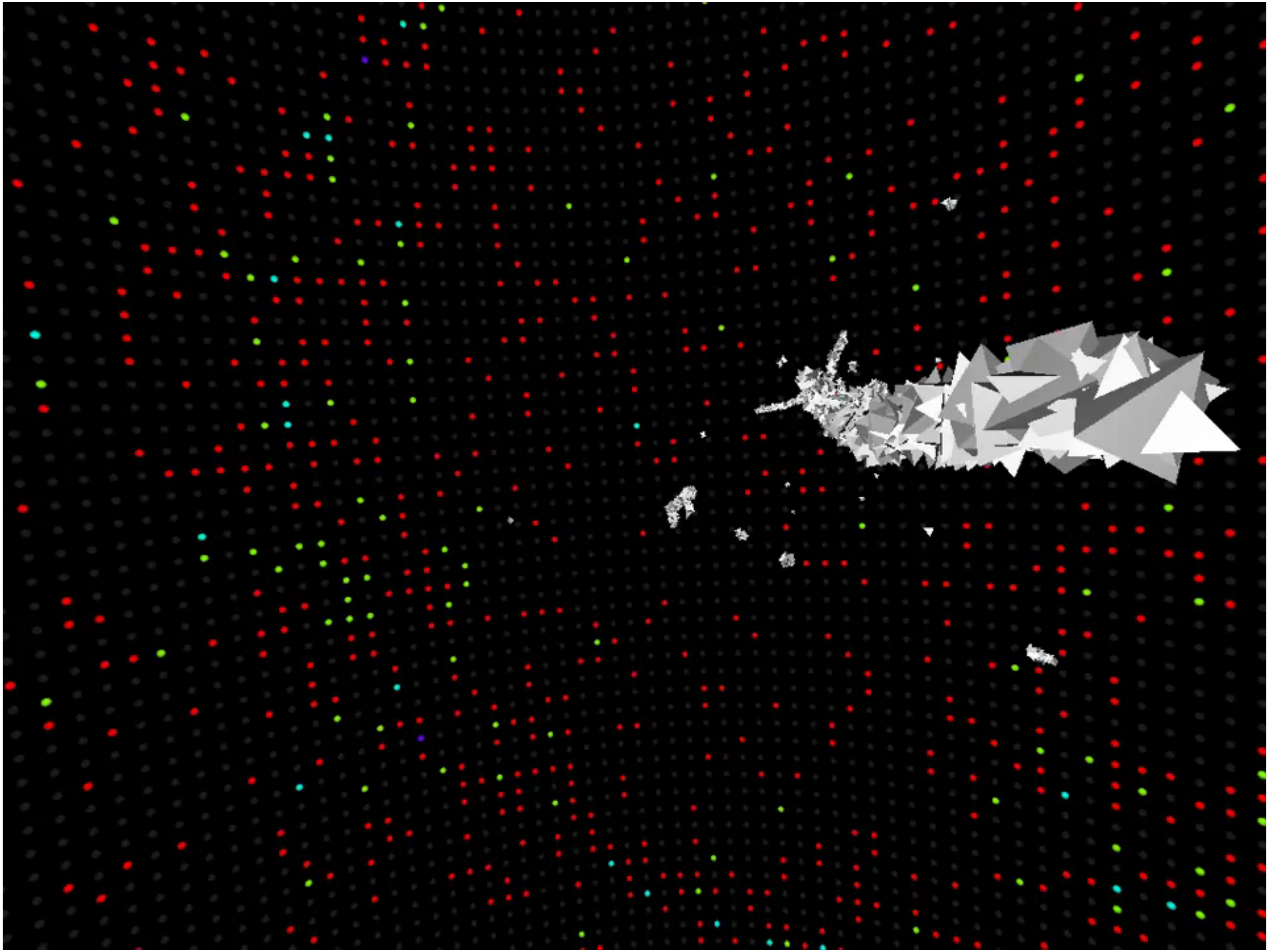




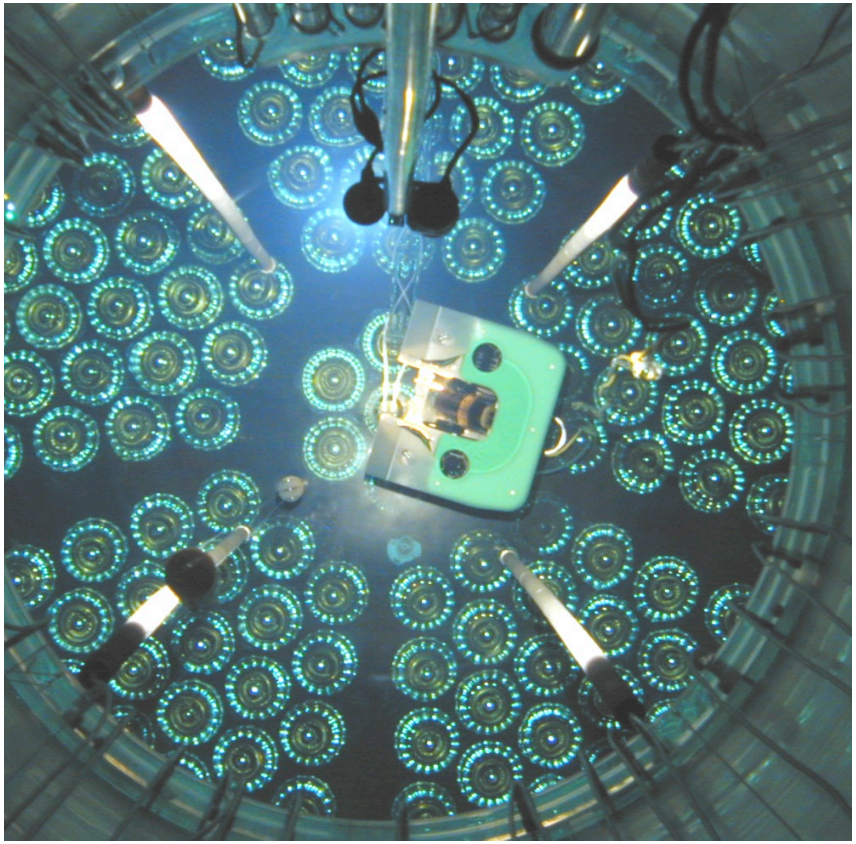
Chroma Ray Tracer

Side benefit: very nice detector displays….

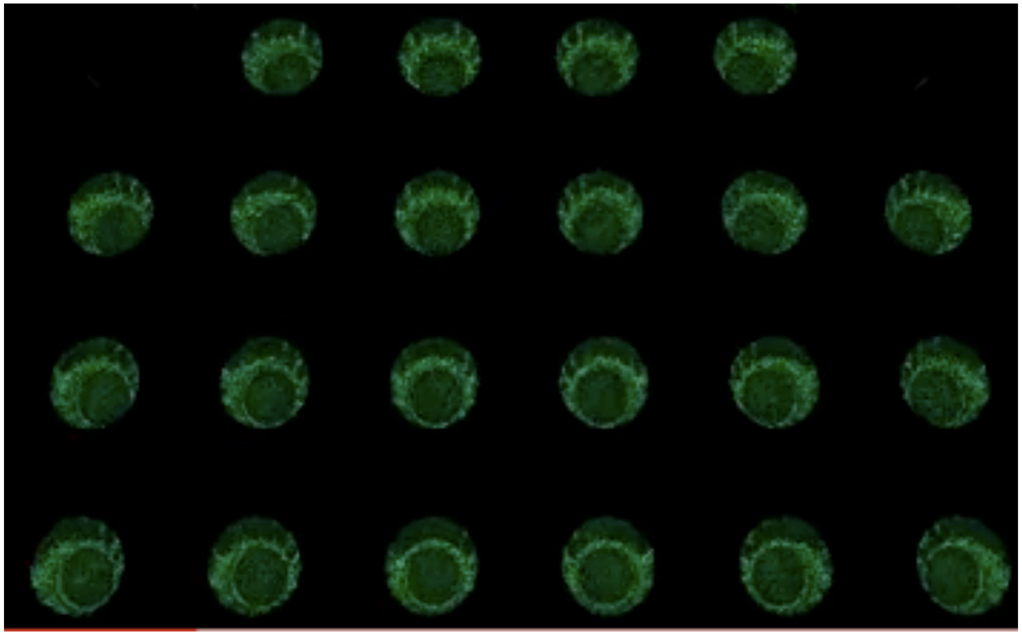




Chroma Ray Tracer



With `physics' turned on:



This is not what your videogame or (probably) CAD program does

Chroma Ray Tracer

- "Ultra-fast" ray-tracing opens up new possibilities for photon-based detectors:
  - Fast simulations
  - MC-based reconstruction
  - Interactive event displays
  - Very fast design optimizations
- Has been made to work in RAT (as a ray-tracing "engine")
- Other experiments already using this, including LAr-TPCs, LXe $0\nu\beta\beta$, LS
- Open source including documentation!

http://chroma.bitbucket.org

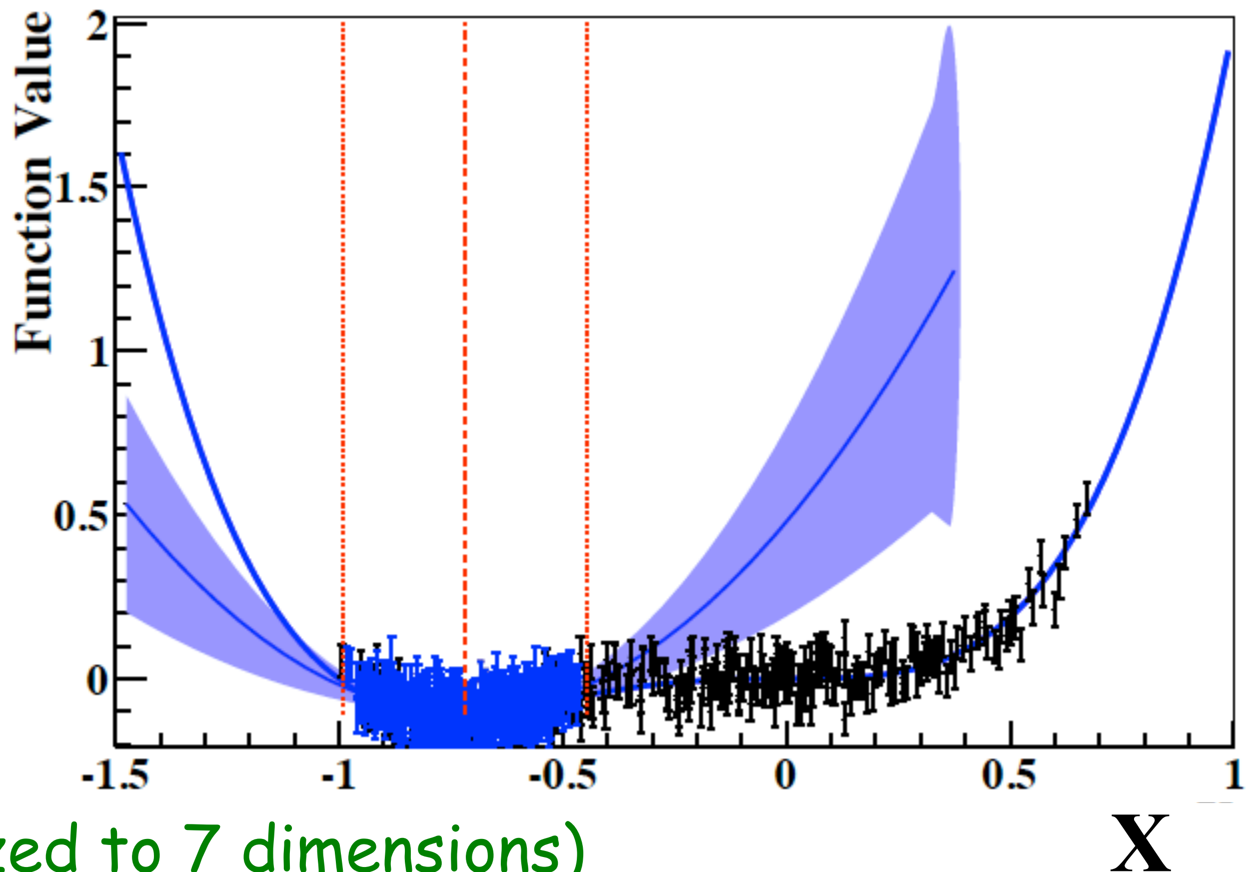
[redacted]

Reconstruction/Simulation

$$L(x,y,z,t,\vec{p},ID) \sim \prod_{i=1}^{N_{PMT}} p_i(t_i^{res},Q_i \mid x,y,z,t,\vec{p},ID)$$

But still:

Stochastic Gradient Descent (`Fuzzy Fitter') by Seibert



(generalized to 7 dimensions)