Stealing Daya Bay MC and Other Software for DUSEL

Brett Viren

Physics Department

Neutrino Working Group @ BNL, August 2008

Contents

Overview of Daya Bay Software

- Based on Gaudi Software Framework of LHC fame.
  - User community mailing list started at BNL
- Completely flexible XML based detector description from LHCb's DetDesc.
- Custom pluggable kinematics generator subsystem, HepMC for data types
- Geant4 for detector simulation through LHCb's GiGa (Geant4/Gaudi) interface.
- Electronics and trigger simulation is Daya Bay specific but much could be taken there.
- ROOT based file I/O is general and flexible.
- Automatic build system for the external and internal packages.

An intentional effort was made to make the software as general as possible in order to be used to study future Water Cerenkov detectors. Other technology can also be accommodated.

Benefits to DUSEL

- Tried and tested by Daya Bay and much by LHC.
- Components well known to many in the field.
- Gaudi framework allows varied contributions without tight coupling of disparate work.
- Mostly a "turn-key" solution, main novel effort is in detector-specific geometry modeling.
- Much BNL-local expertise

DUSEL Specific Areas Needing Work

- Improve/agnostify the auto-install scripts. Some work going on in MINERνA.
- Define detector-specific data model - much can be simply taken from Daya Bay (a PMT hit is a PMT hit).
- Main effort is in detector geometry and materials modeling.

DetDesc: Geometry Modelling

- Unified geometry - generation, simulation and reconstruction all see identical geometry.
- Supports logical, physical and touchable volumes.
- Support for non-ideal alignment offsets to an ideal geometry.
- Set optical and material properties.
- Existing 3D viewer based on LHCb PANORAMIX, can also use Geant4 visualization.
- Very little coupling between geometry modeling and detector simulation
  - Just need to edit XML to simulate drastically different geometries without recompiling.

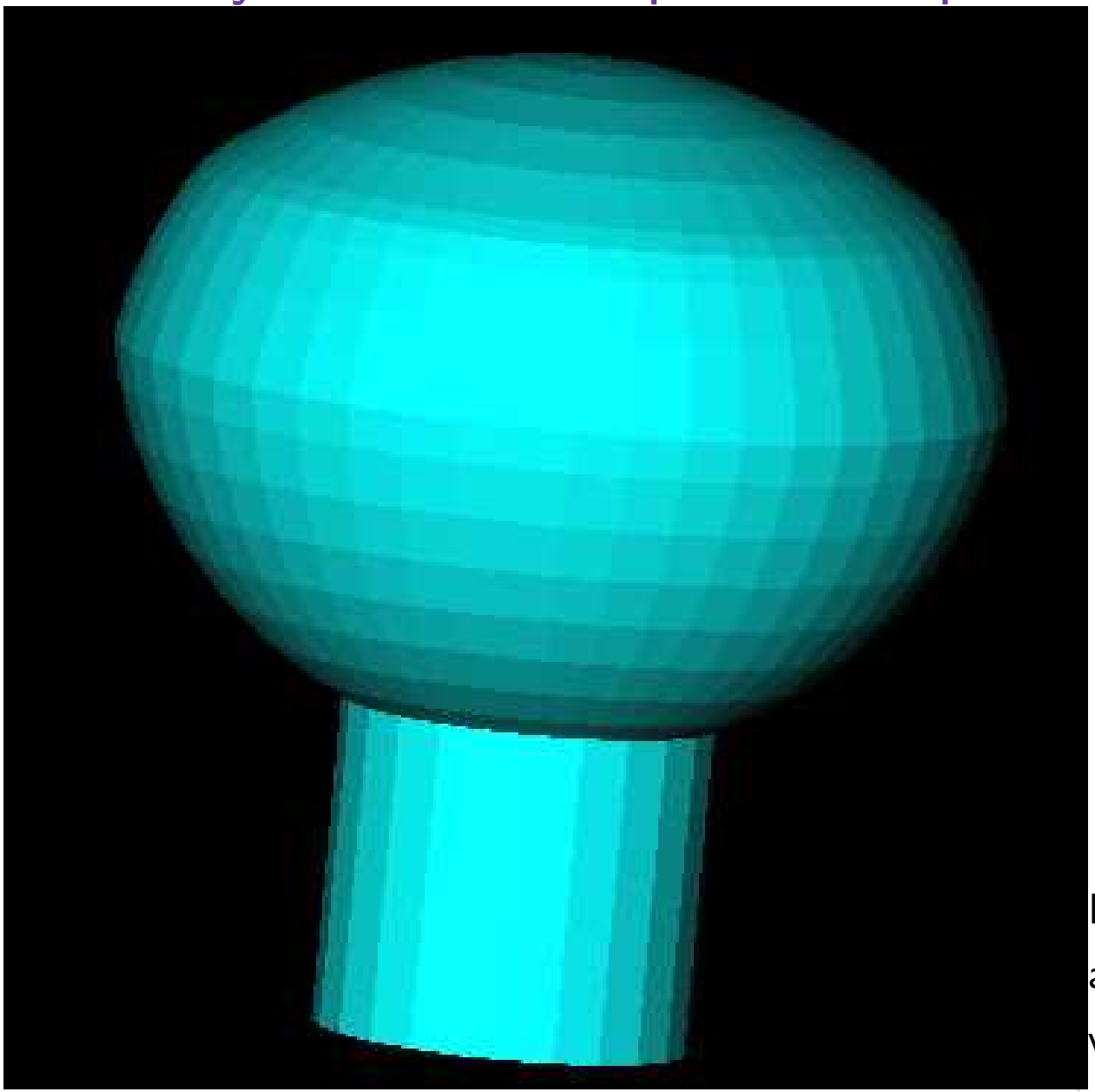
Geometry XML - Example 1: Daya Bay AD

Daya Bay Antineutrino Detector = an oil cylinder logical volume with two daughters: an outer acrylic cylinder and an array of PMTs.

```
<logvol name="lvOIL" material="MineralOil">
  <!-- the shape: -->
  <tubs name="oil" sizeZ="ADoilHeight"
              outerRadius="ADoilRadius" />

  <!-- any physical daughter volumes: -->
  <physvol name="pvOAV"
           logvol="/dd/Geometry/AD/lvOAV"/>
  <physvol name="pvAdPmtArray"
           logvol="/dd/Geometry/AdPmts/lvAdPmtArray"/>
</logvol>
```

Geometry XML - Example 2: Simple PMT

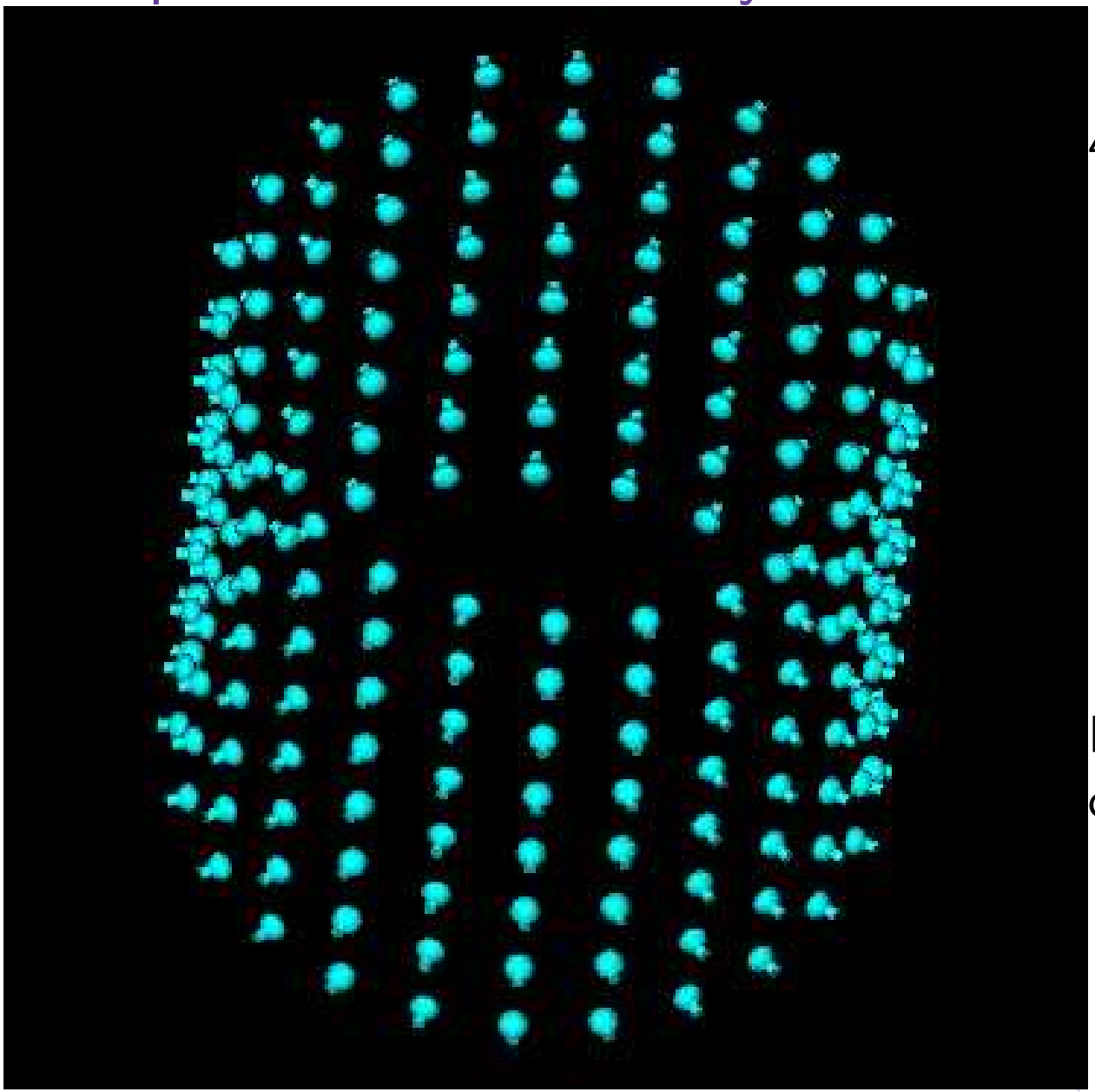


- Simple union of three hemispheres
- Pyrex, photocathode and vacuum volumes
- Photocathode / sensitive detector connection given in XML
- Center @ face sphere's center

Note: polygons are just an artifact of this particular visualization program.

```
<logvol name="lvPmtHemi" material="Pyrex">
  <union name="pmt-hemi">
    <intersection name="pmt-hemi-glass-bulb">
      <sphere name="pmt-hemi-face-glass"
              outerRadius="PmtHemiFaceROC"/>
      <sphere name="pmt-hemi-top-glass"
              outerRadius="PmtHemiBellyROC"/>
      <posXYZ z="PmtHemiFaceOff-PmtHemiBellyOff"/>
     <sphere name="pmt-hemi-bot-glass"
              outerRadius="PmtHemiBellyROC"/>
      <posXYZ z="PmtHemiFaceOff+PmtHemiBellyOff"/>
    </intersection>
    <tubs name="pmt-hemi-base"
          sizeZ="PmtHemiGlassBaseLength"
          outerRadius="PmtHemiGlassBaseRadius"/>
    <posXYZ z="-0.5*PmtHemiGlassBaseLength"/>
  </union>
  <physvol name="pvPmtHemiCathode"
           logvol="/dd/Geometry/PMT/lvPmtHemiCathode"/>
  <physvol name="pvPmtHemiVacuum"
           logvol="/dd/Geometry/PMT/lvPmtHemiVacuum"/>
</logvol>
```

Example 3: AD PMT Array



4 steps:

1. Position one PMT at bottom of tank
2. Copy to make ring of 24 PMTs
3. Copy ring 8 times
4. Rotate everything 1/2 angular period.

Result can be placed as one volume.

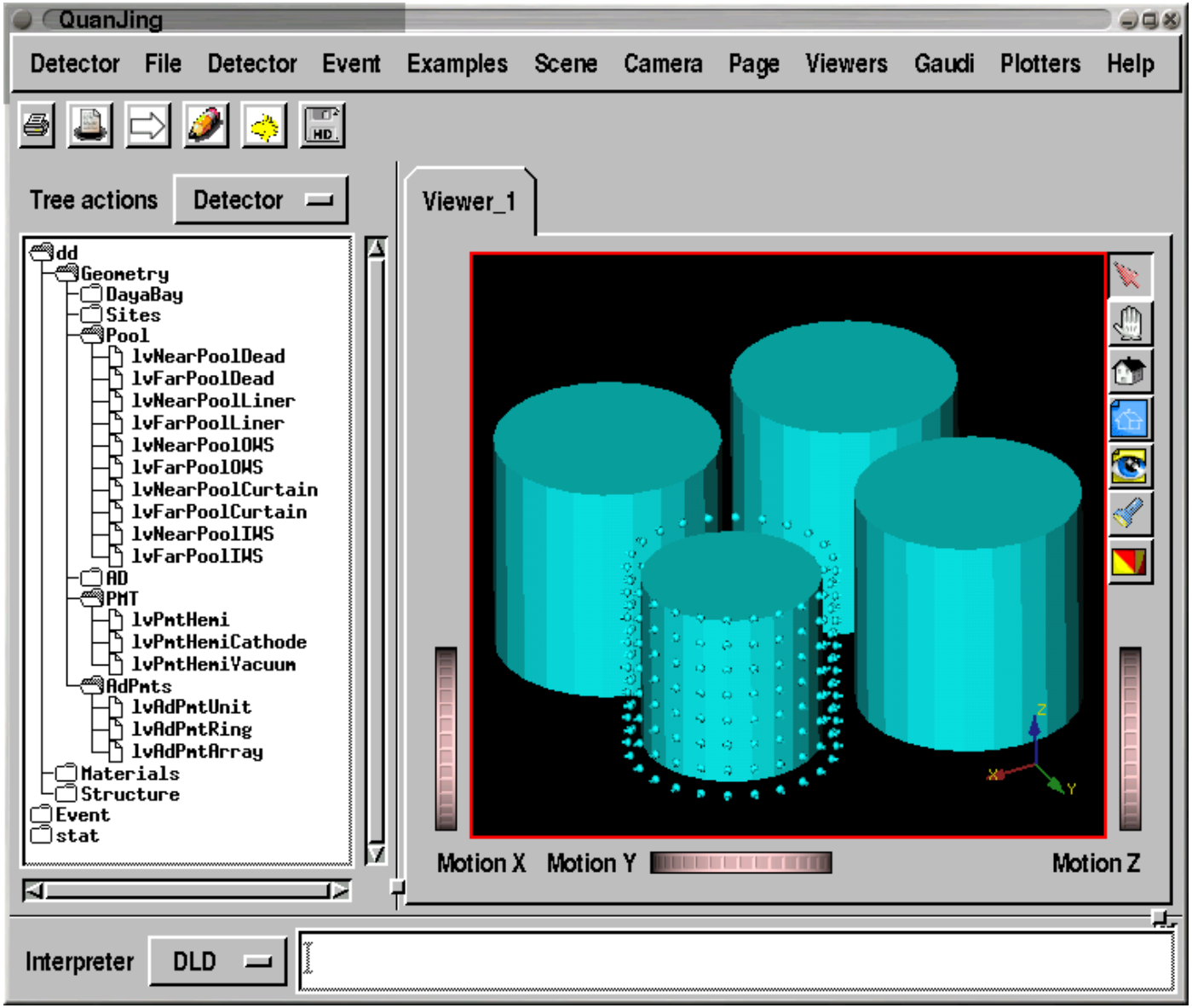
Building the AD PMT array with parameterized placement

```
1  <logvol name="lvAdPmtUnit">              <!-- step 1 -->
2    <physvol name="pvAdPmtUnit" logvol="/dd/Geometry/PMT/lvPmtHemi">
3      <posXYZ x="AdPmtRadialPos" z="-0.5*(AdPmtNrings-1)*AdPmtZsep"/>
4      <rotXYZ rotY="-90*degree" />
5    </physvol>
6  </logvol>
7
8  <logvol name="lvAdPmtRing">              <!-- step 2 -->
9    <paramphysvol number="AdPmtNperRing">
10     <physvol name="pvAdPmtInRing:1" logvol="/dd/Geometry/AdPmts/lvAdPmtUnit" />
11     <posXYZ/>
12     <rotXYZ rotZ="AdPmtAngularSep" />
13   </paramphysvol>
14 </logvol>
15
16 <logvol name="lvAdPmtArrayZero">         <!-- step 3 -->
17   <paramphysvol number="AdPmtNrings">
18     <physvol name="pvAdPmtRingInCyl:1" logvol="/dd/Geometry/AdPmts/lvAdPmtRing"/>
19     <posXYZ z="AdPmtZsep"/>
20   </paramphysvol>
21 </logvol>
22
23 <logvol name="lvAdPmtArray">             <!-- step 4 -->
24   <physvol name="pvAdPmtArray" logvol="/dd/Geometry/AdPmts/lvAdPmtArrayZero">
25     <posXYZ/>
26     <rotXYZ rotZ="0.5*AdPmtAngularSep"/>
27   </physvol>
28 </logvol>
```

Fully parameter driven, no hard coded values!

Set initial copy numbers, eg "pvAdPmtInRing:1".

Direct visualization: `quanjing.py`



- Inspect TDS (and TES)
- Rotate, pan, zoom
- Interrogate volumes
- Delete outer to see inner volumes
- Transparency & color effects
- GUI built from simple XML
- Easy to plug in user code
- Can call C++ or Python from interpreter
- Display event data

High Minded: Philosophy of Software Development

An important first step is to agree on our philosophy of software design.

- DUSEL is a long term project.
- Covers a large R&D parameter space
- Need to vet/reject some underlying technology ASAP (LAr)
- Some basic design parameters need nailing down with realistic sim/reco (Water-Cerenkov needs a veto? If so, how thick? What number/size of PMTs).
- Everyone has day jobs.
- Developers are spread around the world.

So, do we take a myopic view of the here and now or organize and develop the software for the coming decade(s)?

Technicals: Shared Software Infrastructure

Need to organize a shared software infrastructure:

- DUSEL software specific mailing list
  - Suggest leveraging lists.bnl.gov
- Agree on software repository mechanism, centralized vs. decentralized, but definitely need to be inter-operating.
  - Suggest standardizing on GIT but can live with central SVN
- Web presence
  - Suggest leveraging wiki.bnl.gov
- Bug/issue tracking
  - Some local expertise exists using Trac.