Daya Bay Detector Description

Brett Viren

Physics Department



Daya Bay NuWa Offline Software Tutorial

Overview

1. Basics: Logical vs. Physical (Placed) Volumes
2. Detector Elements
3. Transient Detector Store Organization
4. Job Configuration
5. Accessing Detector Description from an Algorithm
6. DetDesc XML Files
7. Visualization

Caveats and Requests

Caveats:

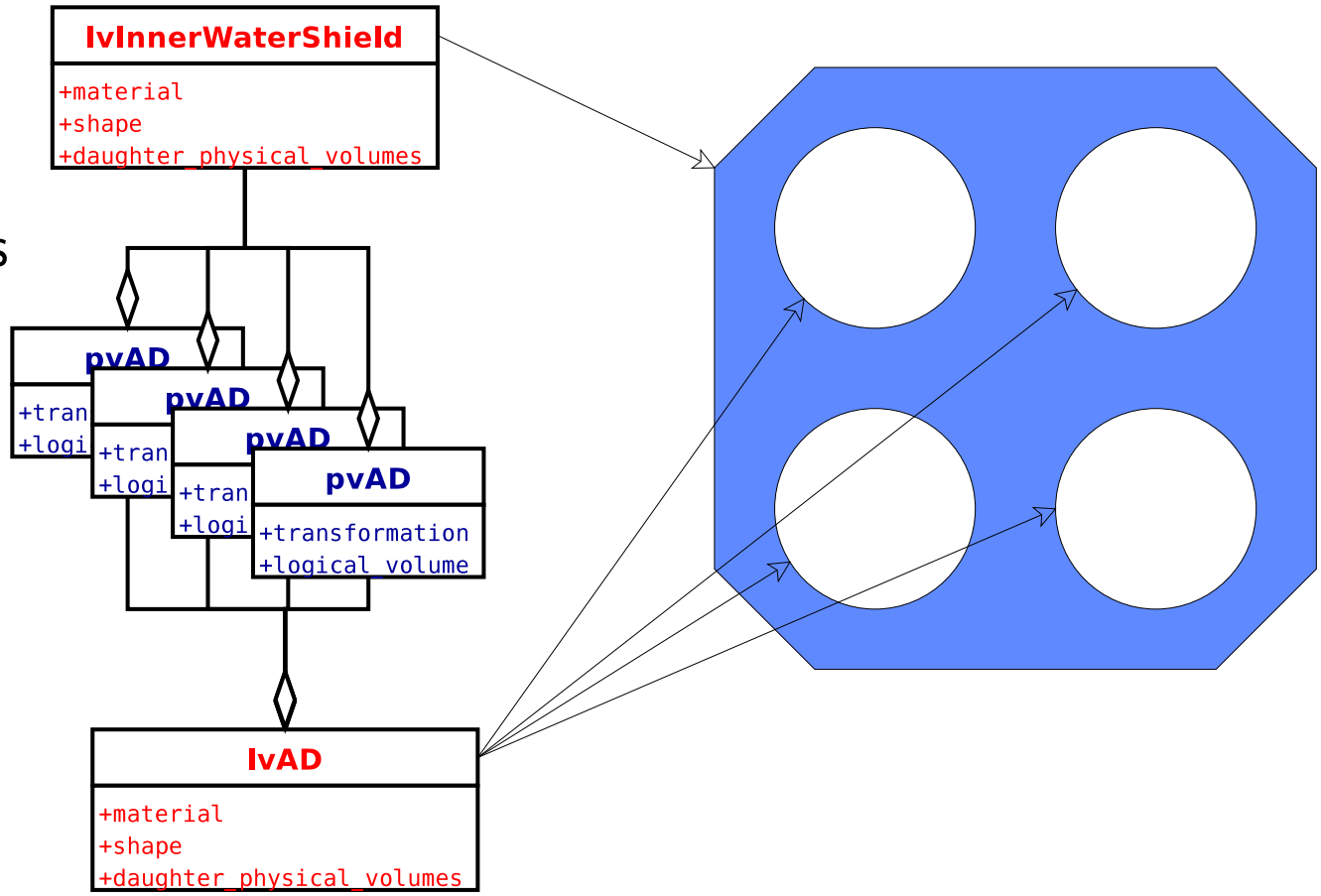
- The geometry is not yet integrated into the detector simulation.
- No exercises in this lesson.

Requests:

- So far, only basic pool geometry entered
- Need detector subsystem experts to write detailed detector description files.

Logical vs. Physical Volumes

- Logical volumes have shape, material and daughter volumes
- Daughters are placed via physical volumes
- Physical volumes have a logical volume and placement transformation

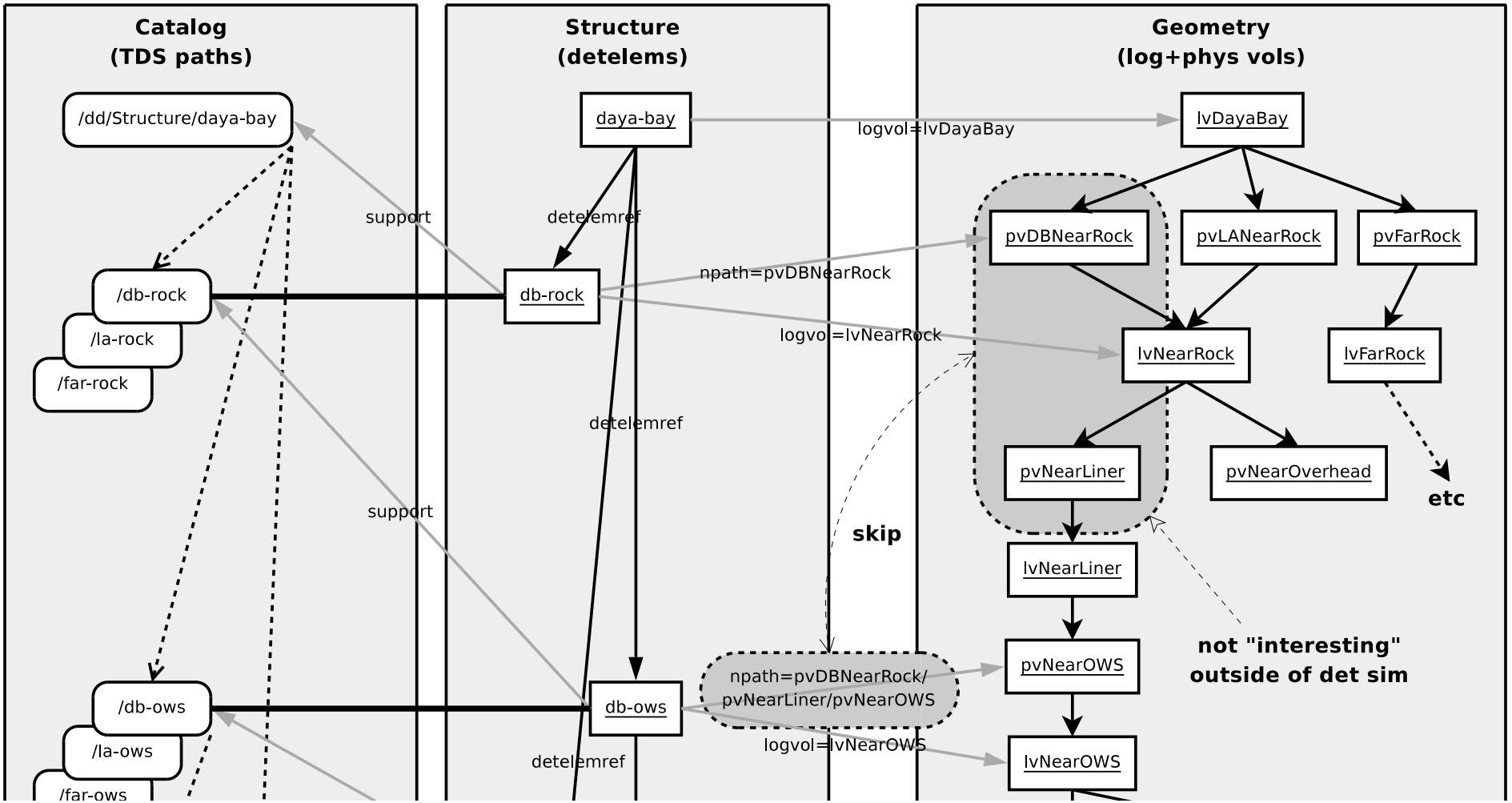


Detector Elements: important volumes

A third type of volume: Detector Element:

- Detector Simulations need all volumes carefully listed
- Humans and reconstruction code cares only about "important" volumes
- Detector Elements define a hierarchy of these volumes
- Connect to geometry via:
  - The logical volume of importance
  - A physical volume path rooted in the parent Detector Element's logical volume
- Also used to apply offsets/alignments to the ideal geometry.

Detector Elements, Logical and Physical Volumes



Transient DetectorStore Organization

Three areas:

`/dd/Materials` access materials by name

`/dd/Geometry` access logical/physical volume hierarchy

`/dd/Structure` access Detector Element hierarchy

Select Materials in the TDS

/dd/Materials/Oxygen
/dd/Materials/Nitrogen
/dd/Materials/Hydrogen
/dd/Materials/Carbon
/dd/Materials/Fluorine
/dd/Materials/Silicon
/dd/Materials/Argon
/dd/Materials/Iron
/dd/Materials/Vacuum
/dd/Materials/Water
/dd/Materials/Air
/dd/Materials/Rock
/dd/Materials/Tyvek

- Elements
  - A, Z, density
- Isotopes
  - A, Z, density, mass fractions
- Materials
  - density, temperature, pressure
  - component element counts

Select Geometry in the TDS

/dd/Geometry/DayaBay/lvDayaBay
/dd/Geometry/Sites/lvNearSiteRock
/dd/Geometry/Sites/lvFarSiteRock
/dd/Geometry/Sites/lvNearHallTop
/dd/Geometry/Sites/lvNearHallBot
/dd/Geometry/Sites/lvFarHallTop
/dd/Geometry/Sites/lvFarHallBot
/dd/Geometry/Pool/lvNearPoolDead
/dd/Geometry/Pool/lvNearPoolLiner
/dd/Geometry/Pool/lvNearPoolOWS
/dd/Geometry/Pool/lvNearPoolCurtain
/dd/Geometry/Pool/lvNearPoolIWS
/dd/Geometry/AD/lvADTank
/dd/Geometry/AD/lvADOil
/dd/Geometry/AD/lvADOuterAV
/dd/Geometry/AD/lvADLS
/dd/Geometry/AD/lvADInnerAV
/dd/Geometry/AD/lvADGdLS

- Access logical volumes through TDS
- Access physical volumes through logical volume
- Organize with subdir for each major component

Select Structure in the TDS

/dd/Structure/daya-bay
/dd/Structure/daya-bay/far-rock
/dd/Structure/daya-bay/far-rock/far-ows
/dd/Structure/daya-bay/far-rock/far-ows/far-curtain
/dd/Structure/daya-bay/far-rock/far-ows/far-curtain/far-iws
/dd/Structure/daya-bay/far-ad1-tank
/dd/Structure/daya-bay/far-ad2-tank/far-ad2-oil
/dd/Structure/daya-bay/far-ad2-tank/far-ad2-oil/far-ad2-oav/far-ad2-ls

- Must explicitly enumerate all "important" volumes
- Paths formed from Detector Element names
- All names must be globally unique! (kind of a pain)
- Logical volumes access from Detector Elements
- Do not need to express complete geometry hierarchy in paths

DetDesc Specific Job Configuration

```
import gaudimodule as gaudi
app = gaudi.AppMgr()
app.ExtSvc += [ "XmlCnvSvc" , "XmlParserSvc" ]

detper = app.service('DetectorPersistencySvc')
detper.CnvServices = ["XmlCnvSvc"]

xmlsvc = app.service('XmlCnvSvc')
xmlsvc.AllowGenericConversion = True

detdat = app.service('DetectorDataSvc')
detdat.UsePersistency = True
detdat.DetDbRootName  = "dd"
detdat.DetStorageType = 7  # magic number meaning 'use XML'
detdat.DetDbLocation  = \
            '/path/to/dybgaudi/XmlDetDesc/DDDB/dayabay.xml'
# test:
det = app.detSvc()
det.dir('/dd')
det.dump()
```

Accessing Detector Description from an Algorithm

Example: `tutorial/Detector/DetWalker/src/DetWalkerAlg.*`

```
StatusCode DetWalkerAlg::initialize()
{
  //...
  // m_volumePath is a property of this algorithm
  DataObject *d = getDet<DataObject>(m_volumePath.value());

  // Get it if it's a logical volume
  if (ILVolume* lv = dynamic_cast<ILVolume*>(d)) {
    return this->dumpLVolume(lv);
  }
  // Get it if it's a physical volume
  if (IPVolume* pv = dynamic_cast<IPVolume*>(d)) {
    return this->dumpPVolume(pv);
  }
  // ...
}
```

DetDesc XML Files, overview

XML files:

- Hold **all** description of Geometry, Structure and Materials. No C++ required!
- Are organized in a modular way: each subsystem expert need to only be responsible for just their XML
  - Require well defined interface volumes and conventions between subsystems.
- Follow well defined schema (DTD files)

  http://lhcb-comp.web.cern.ch/lhcb-comp/Frameworks/DetDesc/Documents/lhcbdtd.pdf
- Located in `dybgaudi/Detector/XmlDetDesc` package.
- Files may be hand written or generated via scripts
  - Still need to add CMT support to auto-run the scripts.

The rest goes through some example XML files.

Main file: DDDB/dayabay.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE DDDB SYSTEM "DTD/structure.dtd">
<DDDB>
  <catalog name="dd">
    <catalogref href="geometry.xml#Geometry" />
    <catalogref href="materials/materials.xml#Materials" />
    <catalogref href="structure.xml#Structure" />
  </catalog>
</DDDB>
```

- Each file sub directory has `geometry.xml` and `structure.xml` as main entries points
- TDS directories are built up with catalogs.
- Reference a (sub)catalog at some URL with catalogref.

Example Materials XML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE materials SYSTEM "../DTD/materials.dtd">
<materials>
  <catalog name="Materials">
    <element density="0.70800E-01*g/cm3" name="Hydrogen"
             symbol="H">
      <atom A="1.00794*g/mole" Zeff="1.0"/>
    </element>
    <element density="0.14300e-02*g/cm3" name="Oxygen"
             symbol="O">
      <atom A="15.999*g/mole" Zeff="8.0000" />
    </element>
    <material density="1.0000*g/cm3" name="Water">
      <component name="Hydrogen" natoms="2"/>
      <component name="Oxygen" natoms="1"/>
    </material>
  </catalog>
</materials>
```

Example Geometry XML, main geometry.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE DDDB SYSTEM "DTD/geometry.dtd">
<DDDB>
  <catalog name="Geometry">
    <catalogref href="DayaBay/geometry.xml#DayaBay" />
    <catalogref href="Sites/geometry.xml#Sites" />
    <catalogref href="Pool/geometry.xml#Pool" />
  </catalog>
</DDDB>
```

- Each subsystem has files in its own subdirectory named after the subsystem
- A `geometry.xml` with a catalog named after the subsystem is main entry point

Ex Geom. cont: parameters and Sites/geometry.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE DDDB SYSTEM "../DTD/geometry.dtd"
[<!ENTITY SiteParameters SYSTEM "parameters.xml">]>
<DDDB>
  &SiteParameters;
  <catalog name="Sites">
    <logvolref href="#lvNearSiteRock"/>
    ...
  <logvol name="lvNearSiteRock" material="Rock">
    <box name="near_rock"
         sizeX="NearSiteRockSize"
         sizeY="NearSiteRockSize"
         sizeZ="NearSiteRockSize"/>
    <physvol name="pvNearHallTop"
             logvol="/dd/Geometry/Sites/lvNearHallTop">
      <posXYZ
        x="NearHallTopOffsetX"
        y="NearHallTopOffsetY"
        z="0.5*NearHallTopSizeZ"/>
    </physvol>
```

Example Structure XML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE DDDB SYSTEM "../DTD/structure.dtd">
<DDDB>
  <catalog name="Sites">
    <detelem name="db-rock">
      <geometryinfo lvname="/dd/Geometry/Sites/lvNearSiteRock"
                    npath="pvDBSiteRock"
                    support="/dd/Structure/daya-bay"/>
      <detelemref href="../Pool/structure.xml#db-ows"/>
    </detelem>
    ...
```

- Also organized into catalogs
- Must specify, geometryinfo:
  - lvname: name in TDS of the "important" logical volume
  - support: detector element that this one is logically held by
  - npath: physical volume path from support's logical volume to this particular detector element
- Can specify a sub-detelem if one exists with a detelemref.

Visualization

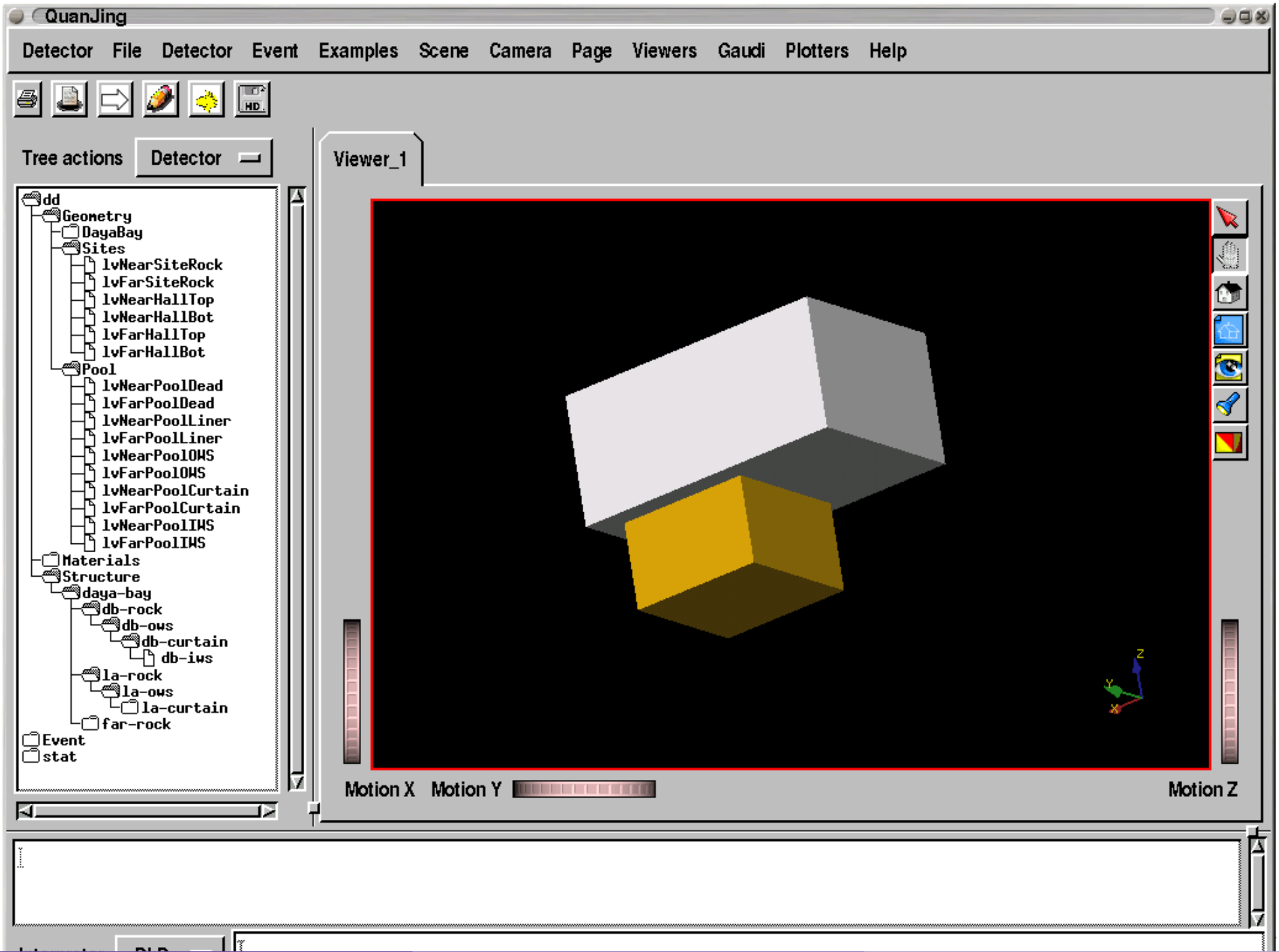
Options:

quanjing.py ("panorama") based on LHCb's Panoramix in `Detector/DetDescVis/python`

- Extensible, GUI built from XML, can run user code
- Can be used to display events, MC Truth
- Doesn't render Boolean volumes properly

Geant4 based visualization

- Requires DetDesc → Geant4 conversion, (not yet done)
- GiGa can provide the conversion, but need to make G4dyb GiGa based.


QuanJing
Detector File Detector Event Examples Scene Camera Page Viewers Gaudi Plotters Help
Tree actions
Detector
dd
Geometry
DayaBay
Sites
lvNearSiteRock
lvFarSiteRock
lvNearHallTop
lvNearHallBot
lvFarHallTop
lvFarHallBot
Pool
lvNearPoolDead
lvFarPoolDead
lvNearPoolLiner
lvFarPoolLiner
lvNearPoolOWS
lvFarPoolOWS
lvNearPoolCurtain
lvFarPoolCurtain
lvNearPoolIWS
lvFarPoolIWS
Materials
Structure
daya-bay
db-rock
db-ows
db-curtain
db-iws
la-rock
la-ows
la-curtain
far-rock
Event
stat
Viewer_1
Motion X
Motion Y
Motion Z

