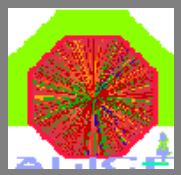
Geometrical Modeler News

Andrei Gheata

ALICE week - 20.03.02

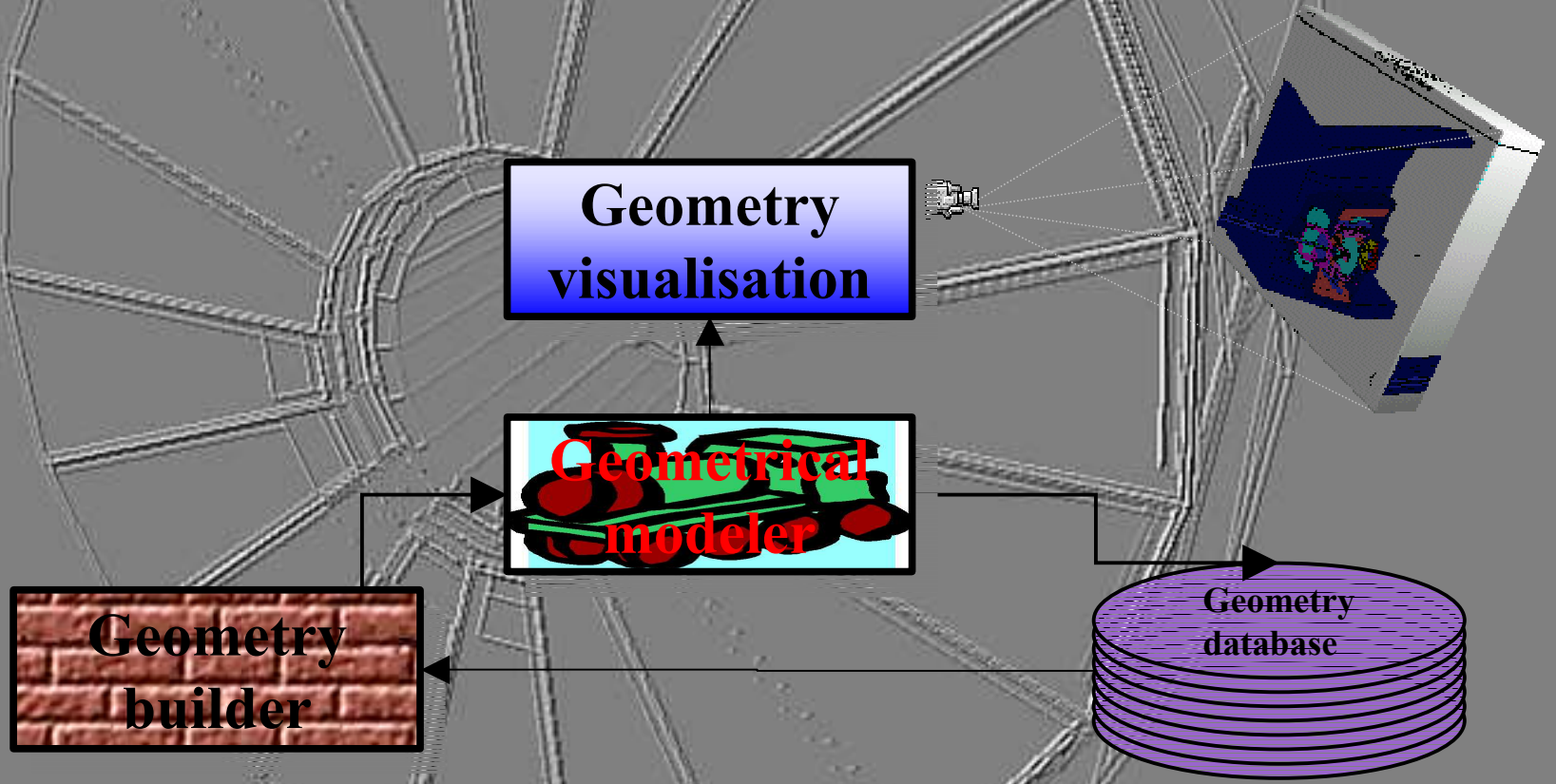


Outlines

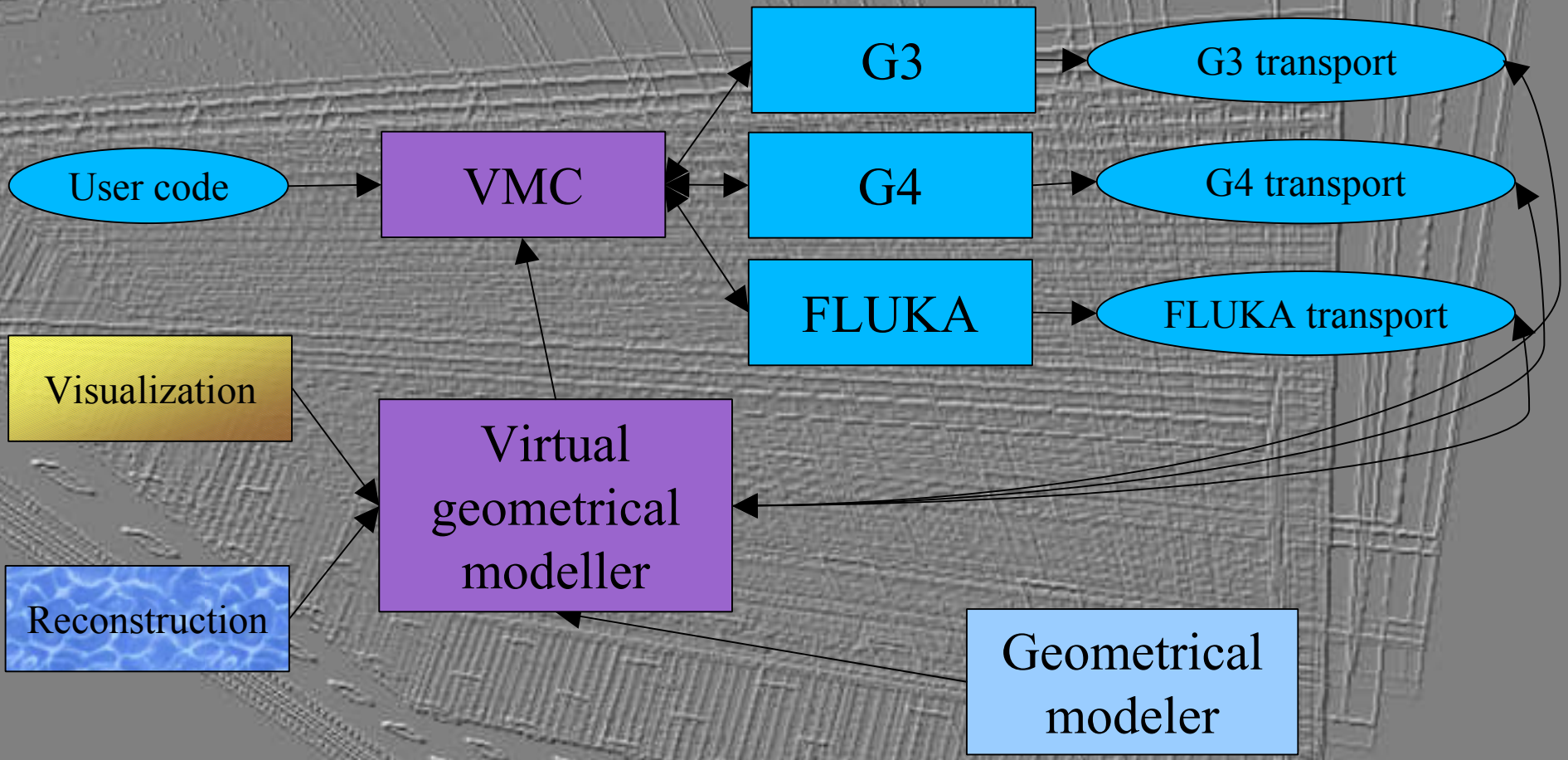
- **Mission statement**
- **Framework description**
- **Status of the code**
- **User interface**
- **Solid model representation**
- **"Where am I ?" implementation**
- **Speed optimisations**
- **Point classification benchmarks**
- **Visualization**
- **Milestones**

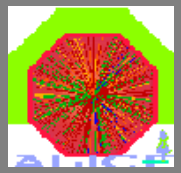
Mission statement

Development of an object oriented geometrical engine as an independent and coherent tool to provide all functionalities needed by simulation, reconstruction and event view.



Interfacing to AliRoot



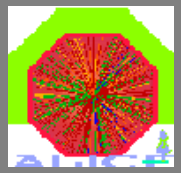


Framework description

- Features required
  - possibility to map GEANT geometry descriptions
  - possibility to handle tens of millions geometrical objects
  - speed performance comparable to G3
  - CSG features (boolean operations)

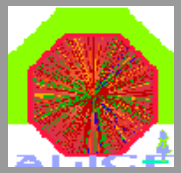
Not accomplished by any of the tested external modelers

- We have chosen to start a completely new development within ROOT framework, based on : the good ideas of the model representation from G3, the CSG features related to boolean operations and the advantages of the virtuality concept provided by C++ - everything taking fully advantage of ROOT I/O, containers, visualization, ...



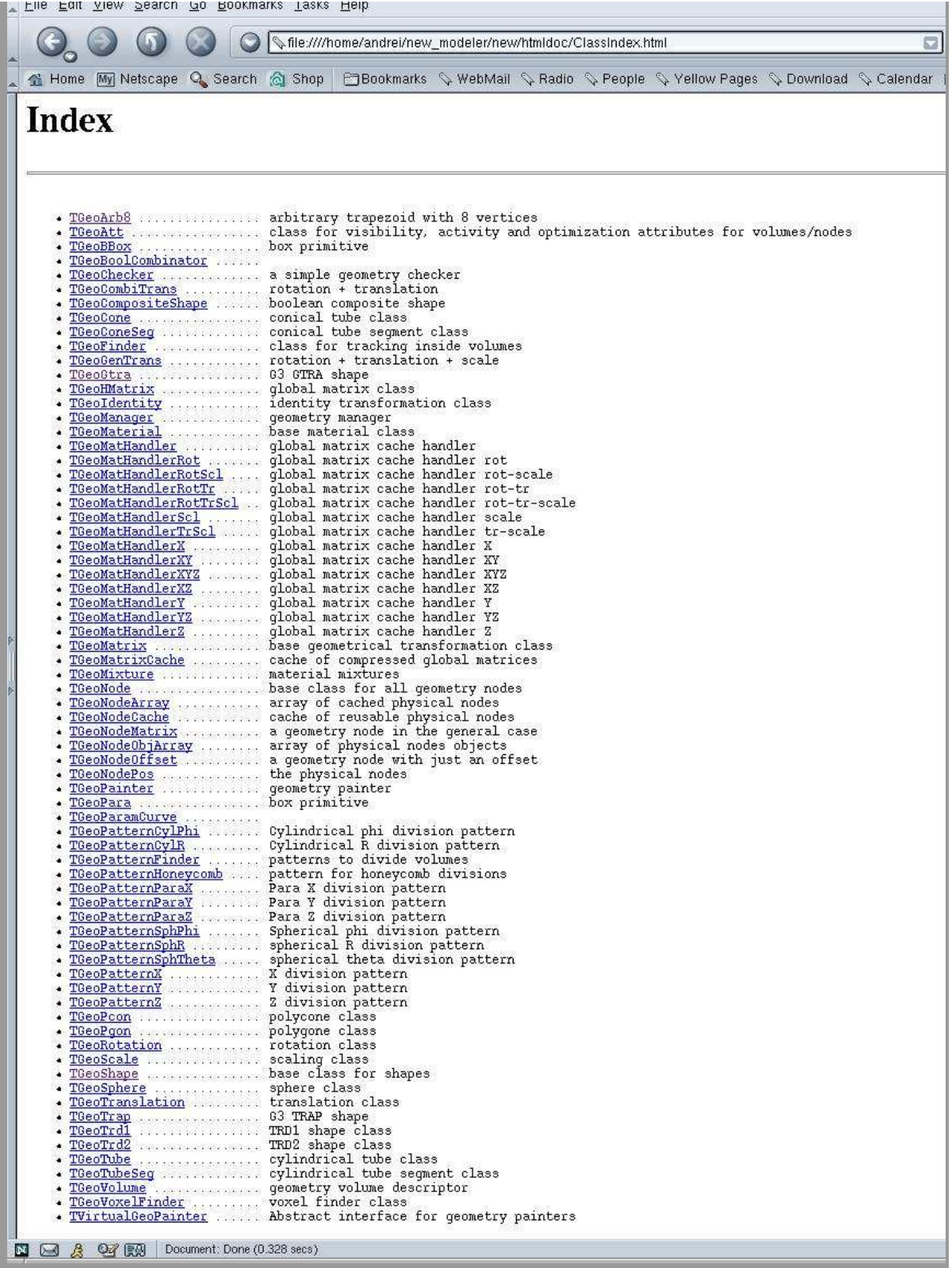
Status of the code

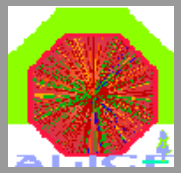
- class design started : Oct.2001
- core data structures (logical tree) + html documentation : Dec2001
- geometry convertor fron G3 ZEBRA banks : Jan.2002
- point classification with respect to shapes : Jan.2002
- first implementation of physical tree cache mechanism : Feb.2002
- implementation of bounding boxes and divisions: Feb.2002
- cache redesign and first "Where am I?" implementation : Feb.2002
- implementation of bounding box ordering mechanism and voxelization : Mar.2002
- mapping of G3 option MANY and division topologies Mar.2002
- benchmarking of "Where am I?" and visualisation ongoing



- ~70 classes, due to intensive usage of virtuality
- code well documented, class description (mostly outdated), UML diagrams, examples
- Most of G3 shapes supported (except HYPE, ELTU, CTUB)
- All division topologies in G3 supported
- MANY option supported (until implementation of composite shapes)
- Speed optimisations

Andrei Gheata





File Edit View Search Go Bookmarks Tasks Help

file:///home/andrei/new_modeler/new/htmldoc/TGeoManager.html

Search

Home Netscape Search Shop Bookmarks WebMail Radio People Yellow Pages Download Calendar Channels

TGeoManager

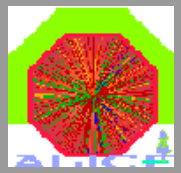
class description – *source file* – *inheritance tree*

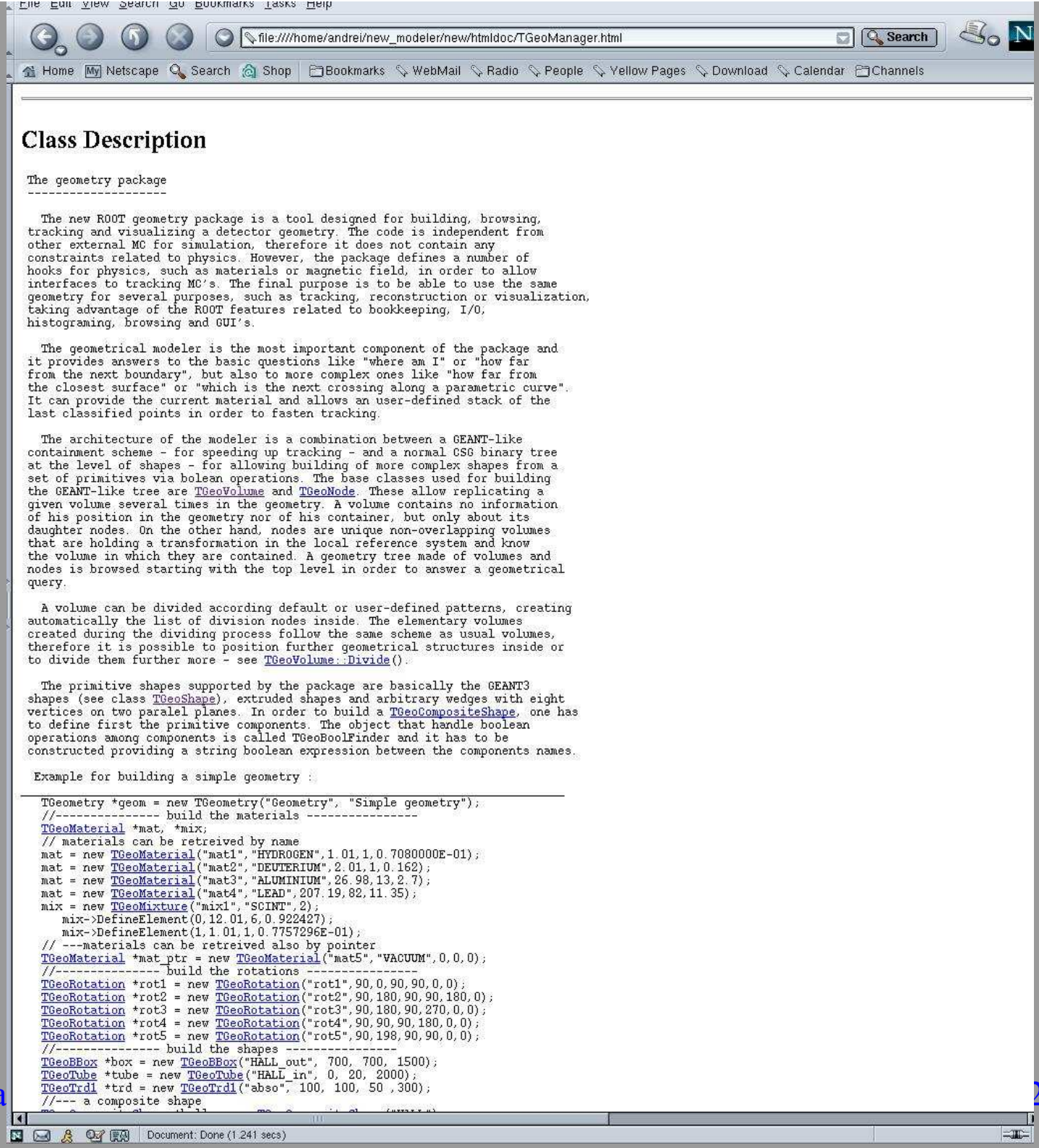
class TGeoManager : public TNamed

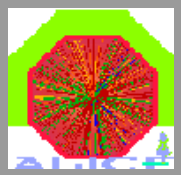
public:

```
          TGeoManager TGeoManager()
          TGeoManager TGeoManager(const char* name, const char* title)
          TGeoManager TGeoManager(TGeoManager&)
        virtual void ~TGeoManager()
                void AddCheckedNode(TGeoNode* node, Int_t level)
              Int_t AddMaterial(TGeoMaterial* material)
              Int_t AddShape(TGeoShape* shape)
              Int_t AddTransformation(TGeoMatrix* matrix)
              Int_t AddVolume(TGeoVolume* volume)
        virtual void Browse(TBrowser* b)
                void BuildCache()
                void BuildDefaultMaterials()
        virtual void cd(const char* path)
                void CdDown(Int_t index)
                void CdTop()
                void CdUp()
                void CheckGeometry(Option_t* option)
                void CheckPoint(Double_t box = 0.1, Option_t* option = "B")
      static TClass* Class()
                void CloseGeometry()
                void ComputeGlobalMatrices(Option_t* option = "0")
           Double_t CoSin(Double_t phi, Bool_t icos = kTRUE)
              Int_t CountNodes(TGeoVolume* vol = 0, Int_t nlevels = 1000)
                void Down(Int_t id)
                void DrawPoint(Double_t x, Double_t y, Double_t z)
                void DrawPoints(TGeoVolume* vol, Int_t npoints = 100000)
          TGeoNode* FindNode(Bool_t downwards = kFALSE, TGeoNode* skipnode = 0)
          TGeoNode* FindNodeOverlap()
       virtual Int_t GetByteCount(Option_t* option = "0")
     TGeoNodeCache* GetCache()
       TGeoHMatrix* GetCurrentMatrix()
          TGeoNode* GetCurrentNode()
          Double_t* GetCurrentPoint()
        TGeoVolume* GetCurrentVolume()
              Int_t GetLevel()
              TList* GetListOfMaterials()
              TList* GetListOfMatrices()
              TList* GetListOfShapes()
              TList* GetListOfVolumes()
TVirtualGeoPainter* GetMakeDefPainter()
        TGeoMaterial* GetMaterial(const char* matname)
        TGeoMaterial* GetMaterial(Int_t id)
              Int_t GetMaterialIndex(const char* matname)
              Int_t GetNNodes()
          TGeoNode* GetNode(Int_t level)
              Int_t GetNsegments()
         const char* GetPath() const
         TGeoShape* GetShape(const char* name)
          TGeoNode* GetTopNode()
        TGeoVolume* GetTopVolume()
              Int_t GetVisLevel()
              Int_t GetVisOption()
        TGeoVolume* GetVolume(const char* name)
      virtual TClass* IsA() const
       virtual Bool_t IsFolder() const
                void LocalToMaster(Double_t* local, Double_t* master)
                void LocalToMaster(Float_t* local, Float_t* master)
        TGeoVolume* MakeArb8(const char* name, const char* material, Double_t dz, Double_t* vertices = 0)
        TGeoVolume* MakeBox(const char* name, const char* material, Double_t dx, Double_t dy, Double_t dz)
        TGeoVolume* MakeCone(const char* name, const char* material, Double_t dz, Double_t rmin1, Double_t rmax1, Double_t rmin2, Double_t
        TGeoVolume* MakeCons(const char* name, const char* material, Double_t dz, Double_t rmin1, Double_t rmax1, Double_t rmin2, Double_t
        TGeoVolume* MakeGtra(const char* name, const char* material, Double_t dz, Double_t theta, Double_t phi, Double_t twist, Double_t h1,
```

Document: Done (1.241 secs)




File Edit View Search Go Bookmarks Tasks Help
file:///home/andrei/new_modeler/new/htmldoc/TGeoManager.html
Search
Home Netscape Search Shop Bookmarks WebMail Radio People Yellow Pages Download Calendar Channels
Class Description
The geometry package
The new ROOT geometry package is a tool designed for building, browsing, tracking and visualizing a detector geometry. The code is independent from other external MC for simulation, therefore it does not contain any constraints related to physics. However, the package defines a number of hooks for physics, such as materials or magnetic field, in order to allow interfaces to tracking MC's. The final purpose is to be able to use the same geometry for several purposes, such as tracking, reconstruction or visualization, taking advantage of the ROOT features related to bookkeeping, I/O, histograming, browsing and GUI's.
The geometrical modeler is the most important component of the package and it provides answers to the basic questions like "where am I" or "how far from the next boundary", but also to more complex ones like "how far from the closest surface" or "which is the next crossing along a parametric curve". It can provide the current material and allows an user-defined stack of the last classified points in order to fasten tracking.
The architecture of the modeler is a combination between a GEANT-like containment scheme - for speeding up tracking - and a normal CSG binary tree at the level of shapes - for allowing building of more complex shapes from a set of primitives via bolean operations. The base classes used for building the GEANT-like tree are TGeoVolume and TGeoNode. These allow replicating a given volume several times in the geometry. A volume contains no information of his position in the geometry nor of his container, but only about its daughter nodes. On the other hand, nodes are unique non-overlapping volumes that are holding a transformation in the local reference system and know the volume in which they are contained. A geometry tree made of volumes and nodes is browsed starting with the top level in order to answer a geometrical query.
A volume can be divided according default or user-defined patterns, creating automatically the list of division nodes inside. The elementary volumes created during the dividing process follow the same scheme as usual volumes, therefore it is possible to position further geometrical structures inside or to divide them further more - see TGeoVolume::Divide().
The primitive shapes supported by the package are basically the GEANT3 shapes (see class TGeoShape), extruded shapes and arbitrary wedges with eight vertices on two paralel planes. In order to build a TGeoCompositeShape, one has to define first the primitive components. The object that handle boolean operations among components is called TGeoBoolFinder and it has to be constructed providing a string boolean expression between the components names.
Example for building a simple geometry :
TGeometry *geom = new TGeometry("Geometry", "Simple geometry");
//--------------- build the materials ----------------
TGeoMaterial *mat, *mix;
// materials can be retreived by name
mat = new TGeoMaterial("mat1",1.01,1,0.7080000E-01);
mat = new TGeoMaterial("mat2","DEUTERIUM",2.01,1,0.162);
mat = new TGeoMaterial("mat3","ALUMINIUM",26.98,13,2.7);
mat = new TGeoMaterial("mat4","LEAD",207.19,82,11.35);
mix = new TGeoMixture("mix1","SCINT",2);
mix->DefineElement(0,12.01,6,0.922427);
mix->DefineElement(1,1.01,1,0.7757296E-01);
// ---materials can be retreived also by pointer
TGeoMaterial *mat_ptr = new TGeoMaterial("mat5","VACUUM",0,0,0);
//--------------- build the rotations ----------------
TGeoRotation *rot1 = new TGeoRotation("rot1",90,0,90,90,0,0);
TGeoRotation *rot2 = new TGeoRotation("rot2",90,180,90,90,180,0);
TGeoRotation *rot3 = new TGeoRotation("rot3",90,180,90,270,0,0);
TGeoRotation *rot4 = new TGeoRotation("rot4",90,90,90,180,0,0);
TGeoRotation *rot5 = new TGeoRotation("rot5",90,198,90,90,0,0);
//--------------- build the shapes ----------------
TGeoBBox *box = new TGeoBBox("HALL_out", 700, 700, 1500);
TGeoTube *tube = new TGeoTube("HALL_in", 0, 20, 2000);
TGeoTrd1 *trd = new TGeoTrd1("abso", 100, 100, 50 ,300);
//--- a composite shape
Document: Done (1.241 secs)



File Edit View Search Go Bookmarks Tasks Help

file:///home/andrei/new_modeler/new/htmldoc/TGeoManager.html

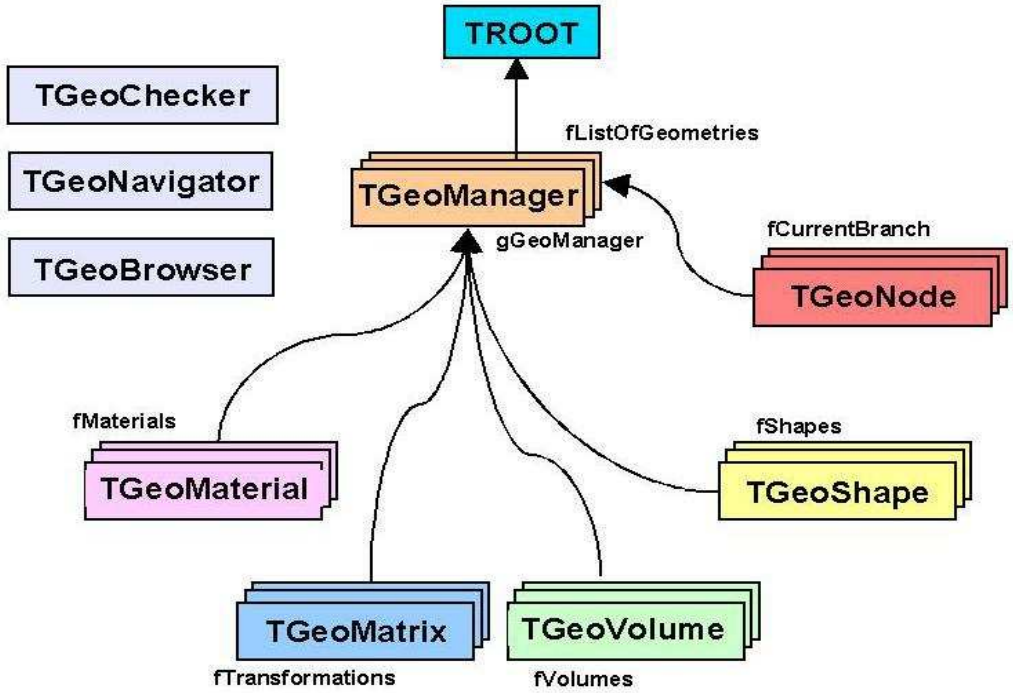
Search

Home Netscape Search Shop Bookmarks WebMail Radio People Yellow Pages Download Calendar Channels

TGeoManager::GetVolume(const char *name);
The top level volume must be specified with TGeoManager::SetTopVolume();
All objects like transformations, materials, shapes or volumes can be created
with new operator and they will register themselves to TGeoManager class.
The user does not have to take care of their deletion.

/*

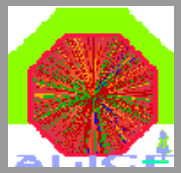
Geometry manager



*/

TGeoManager()

Document: Done (1.241 secs)



User Interface

ROOT Object Browser

File View Options Help

B076

All Folders

- root
- /home/andrei/new_modeler/new/macros
- ROOT Files
- ALIC
  - L3MO
  - ABSM
  - DDIP
  - DCM0
  - HEW1
  - B077
    - B076
      - B047
      - B045
      - BM49
        - B049
        - B051
      - B042

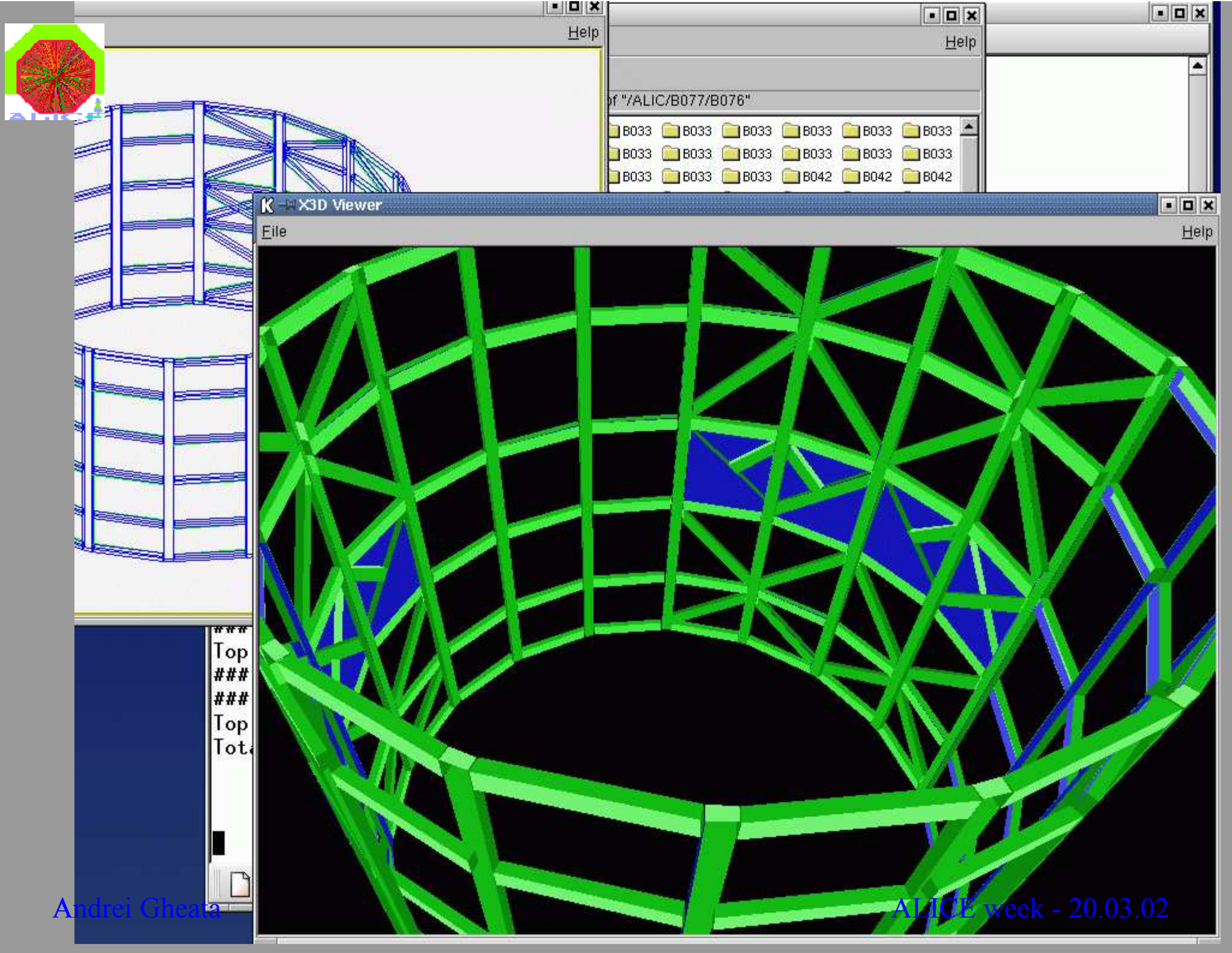
Contents of "/ALIC/B077/B076"

B033 B033 B033 B033 B033 B033 B033
B033 B033 B033 B033 B033 B033 B033
B033 B033 B033 B033 B042 B042 B042
B042 B042 B042 B042 B042 B042 B042
B042 B042 B042 B042 B042 B042 B042
B042 B042 B042 B042 B042 B042 B042
B042 B042 B042 B042 B042 B042 B042
B042 B042 B042 B042 B042 B042 B042
B042 B042 B042 B042 B042 B042 B042
B042 B042 B042 B042 B042 B042 B042
B042 B042 B042 B042 B042 B042 B042
B042 B042 B042 B042 B042 B042 B042
B042 B042 B042 B042 B042 B042 B042
B042 B042 B042 B042 B042 B042 B042
B042 B042 B042 B042 B042 B042 B042

166 Objects.

```
root [0]
Processing ALICE.C...
trap : dz=0.011250, h1=-1.000000, bl1=-1.000000, tl1= -1.000000, h2=-1.000000, bl
2=-1.000000, tl2=-1.000000
Top volume is ALIC. Master volume is ALIC
Counting nodes...
Voxelizing...
Building caches for nodes and matrices...
### nodes stored in cache 130000 ###
### matrix caches of size 1000 built ###
### nodes in ALICEgeometry : 1238495
----------------modeler ready----------------
root [1] new TBrowser
(class TBrowser*)0x86af930
```

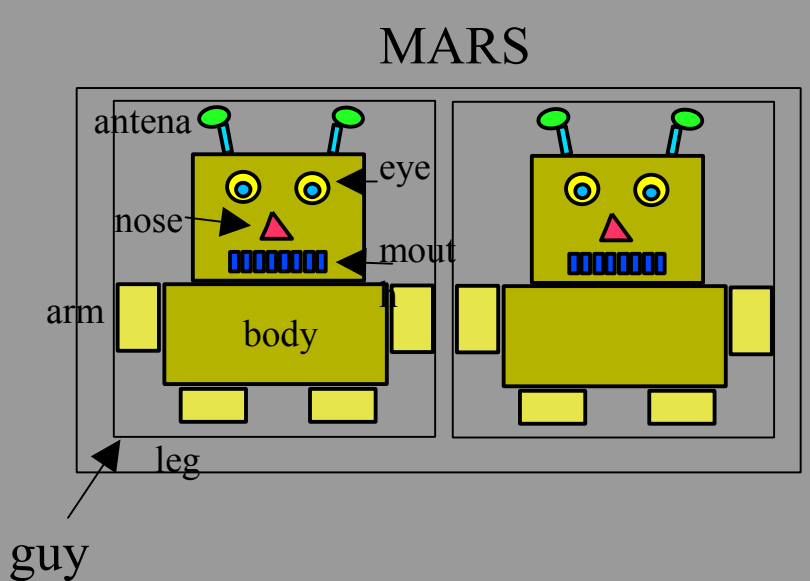
New Konsole No 1

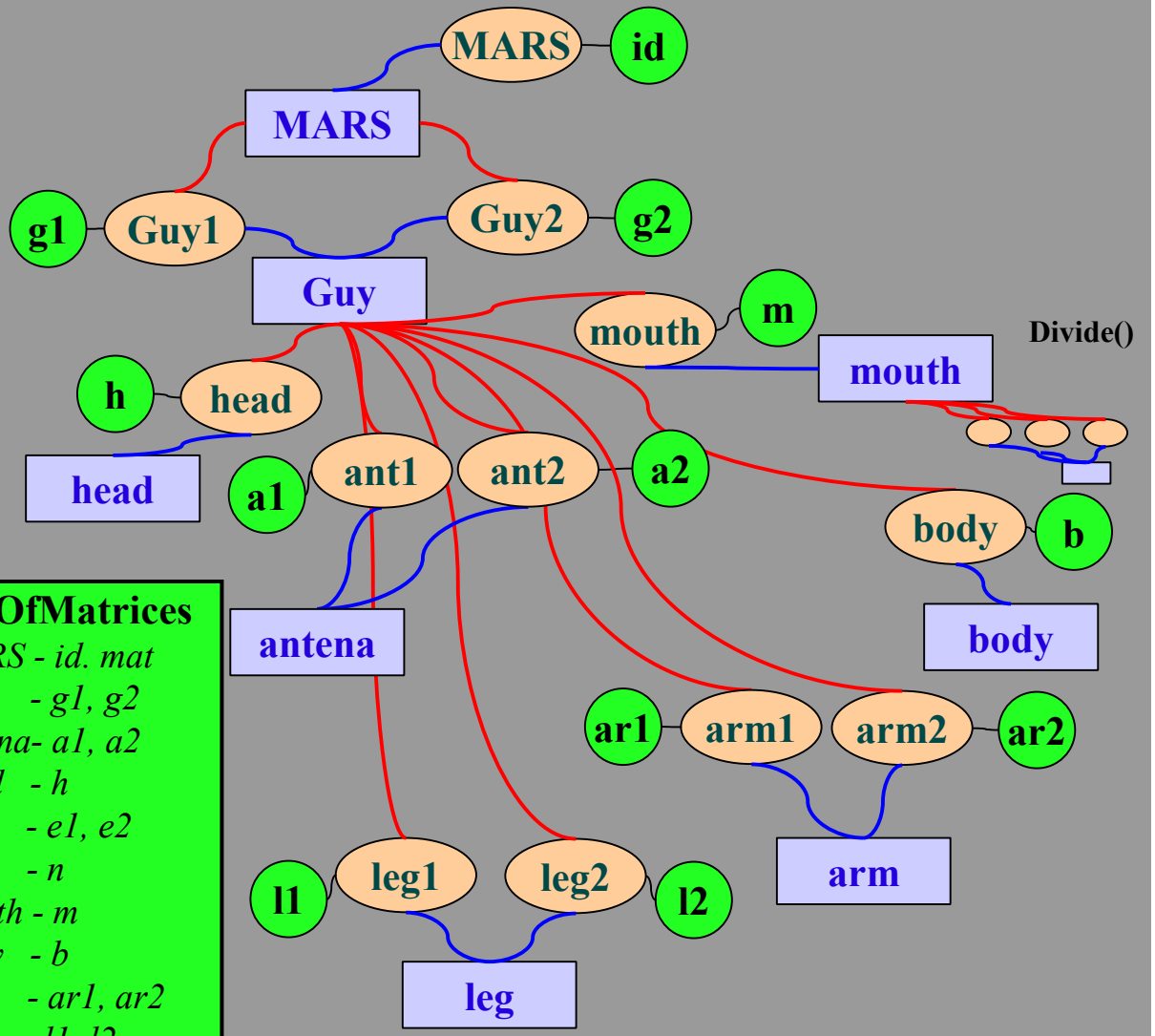

Help
Help
of "/ALIC/B077/B076"
B033 B033 B033 B033 B033 B033
B033 B033 B033 B033 B033 B033
B033 B033 B033 B042 B042 B042
X3D Viewer
File
Help
Top
###
###
Top
Tot

User interface (contd)

- User-friendly builder interface via the top level class TGeoManager
- G3 builder functions mapped
- Interaction with objects via context menu
- picking of volumes from the view (very soon)
- geometry visual checking tools (visualisation of point neighbourhoods, computation of distance to the closest boundary via sampling, test of FindNode() with random points and highlighting of "touched" nodes.

Solid model representation





ListOfMaterials

ListOfShapes
MARS box
guy box
antena - composite
- wire cylinder
- ball sphere
head box
eye - composite
...

ListOfVolumes
MARS
guy
antena
head
eye
nose
mouth
body
arm
leg

ListOfMatrices
MARS - id. mat
guy - g1, g2
antena- a1, a2
head - h
eye - e1, e2
nose - n
mouth - m
body - b
arm - ar1, ar2
leg - l1, l2

build materials and shapes ...

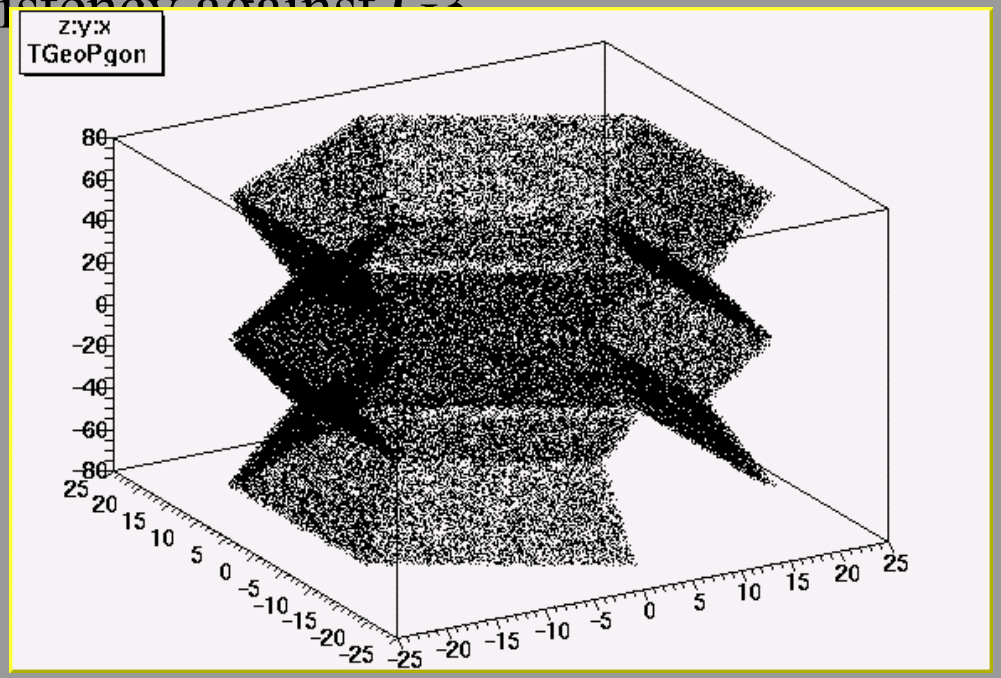
then volumes ...

then transformations ...

and finally nodes.

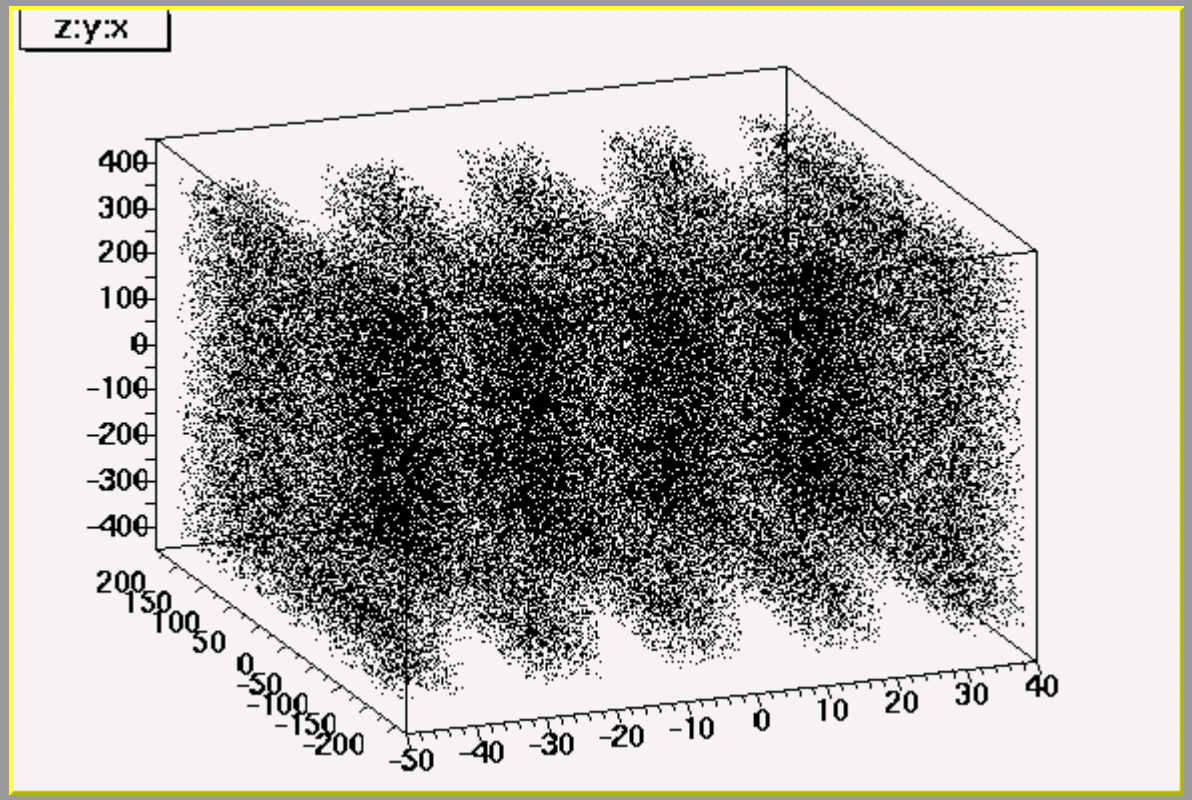
"Where am I" implementation

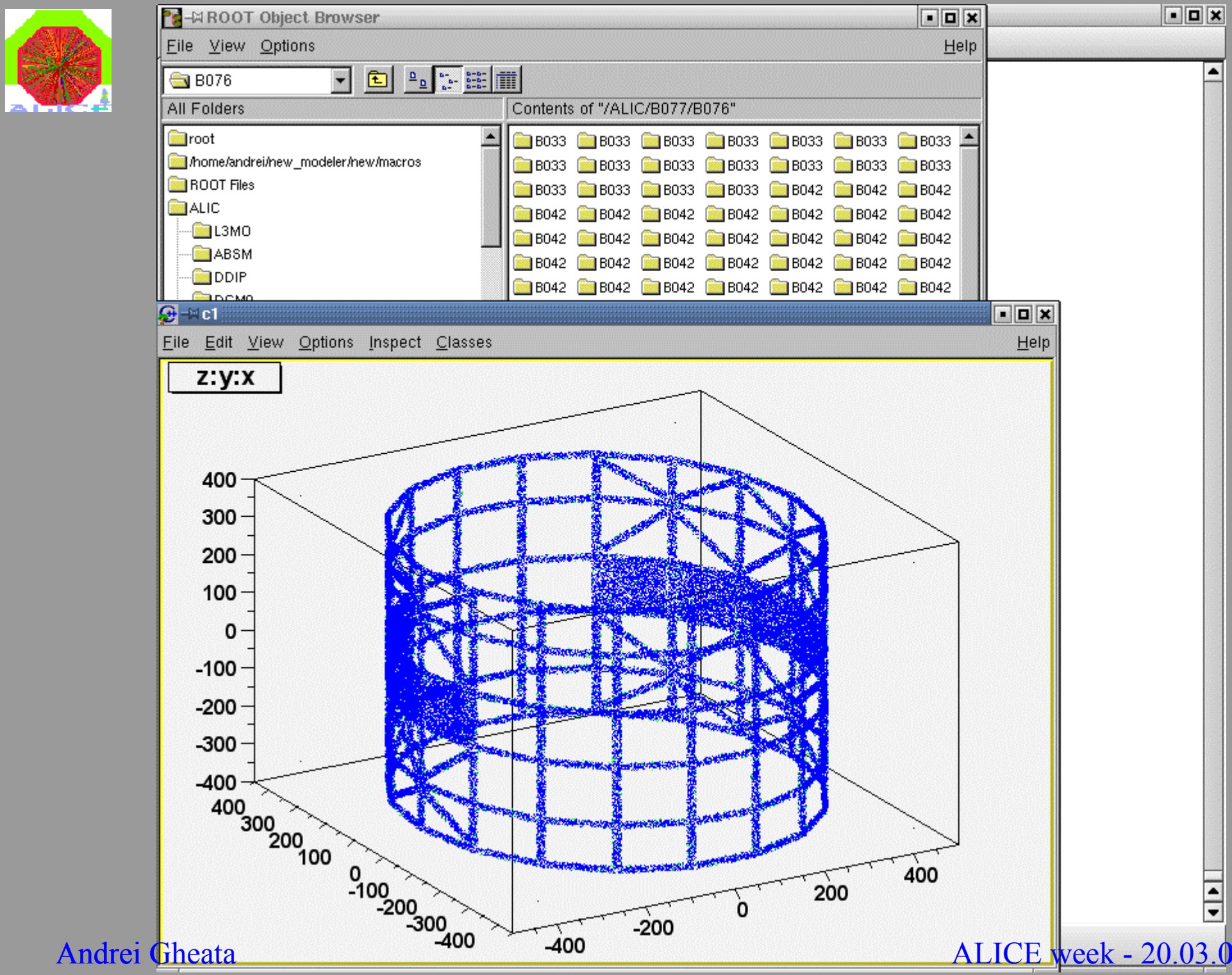
- The most basic (and important) feature provided by a geometrical modeler
- It evolved and it was benchmarked (thanks to Rene Brun) togeather with the addition of new features in the modeler.
- A conversion tool producing a C++ macro from G3 ZEBRA banks was implemented by Mihaela Gheata. This allowed benchmarking the algorithm speed and consistency against G3
- Algorithms for point classification with respect to a shape were mapped (most of them) from G3 by Mihaela.

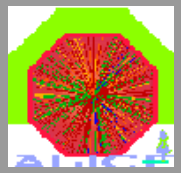


Divisions

- Divisions were tested as soon as they were implemented. Currently all division topologies from G3 are supported (M.Gheata)

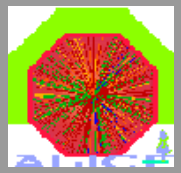



ROOT Object Browser
File View Options Help
B076
All Folders
Contents of "/ALIC/B077/B076"
root
/home/andrei/new_modeler/new/macros
ROOT Files
ALIC
L3MO
ABSM
DDIP
B033
B042
c1
File Edit View Options Inspect Classes Help
z:y:x
400
300
200
100
0
-100
-200
-300
-400



Speed optimisations

- One of the most important requirements - at least at the same level as G3
- Several levels of optimisation
  - bounding boxes
  - ordering of bounding boxes
  - voxelization
  - caching of most frequently used nodes with their global transformation matrix + persistency & garbage collection
  - abstraction level in geometrical transformations and implementation of specialized matrix handlers directly within cache in order to fasten local->master and master->local operations
  - improvements in solving MANY overlaps and restricting the problem to MANY-MANY overlap



Point classification benchmarks

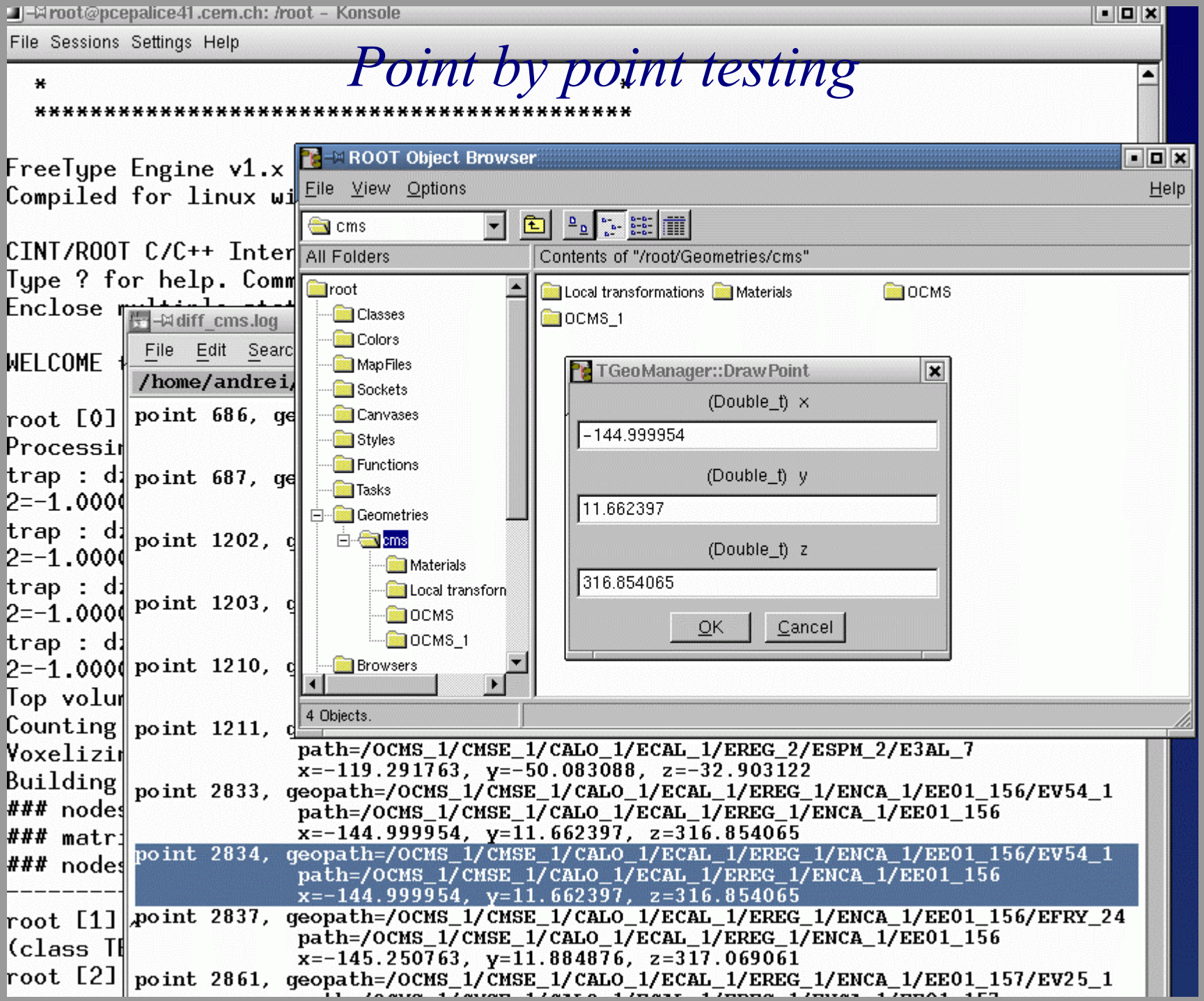
Geometries converted from G3 : G3 examples, ALICE, CMS, ATLAS

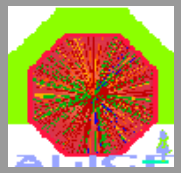
- diffs G3-TGeo ~4% of points, due to:
- floating point roundings (G3 I/O is in float)
- bugs still undetected

- Speed comparisons (PIII/600) [microseconds / point]

Geometry	G3 phys.	G3 random	TGeo phys.	TGeo random
gexam1	3.08	6.60	2.80	3.86
gexam3	2.87	3.47	2.21	1.74
gexam4	2.51	12.09	3.28	5.98
CMS	33.57	39.09	9.83	22.44
ATLAS	8.95	32.28	6.14	29.14

Point by point testing





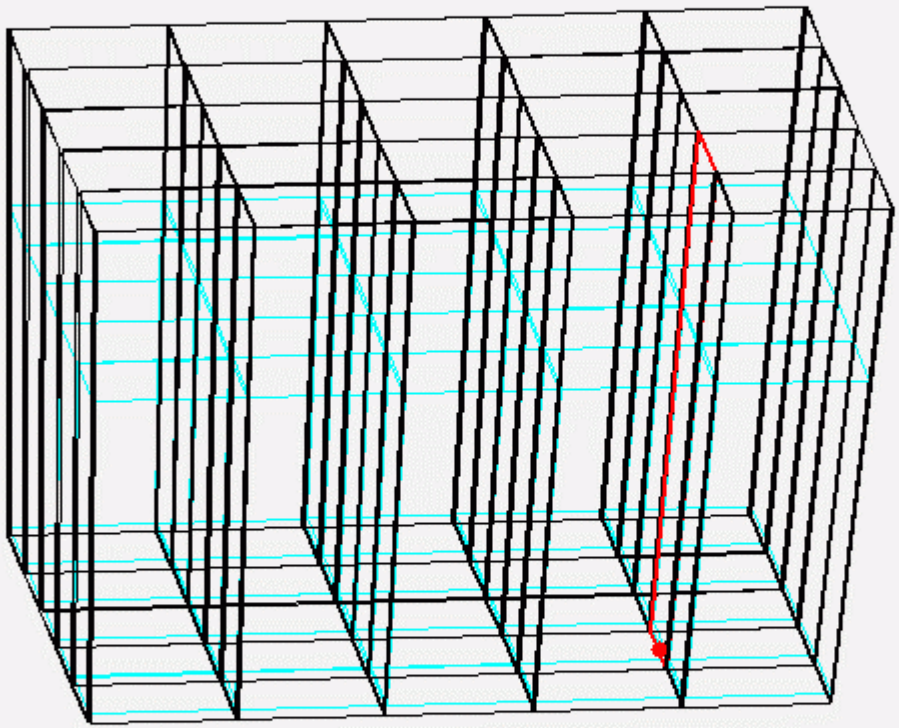
root@pcepalice41.cern.ch: /root - Konsole

File Sessions Settings Help

```
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
```

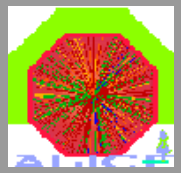
c1

File Edit View Options Inspect Classes Help



```
### matrix caches of size 1000 built ###
Top volume is EE01. Master volume is OCMS
<TCanvas::MakeDefCanvas>: created default TCanvas with name c1
Distance to boundary : 0.081312
```

New Konsole No 1



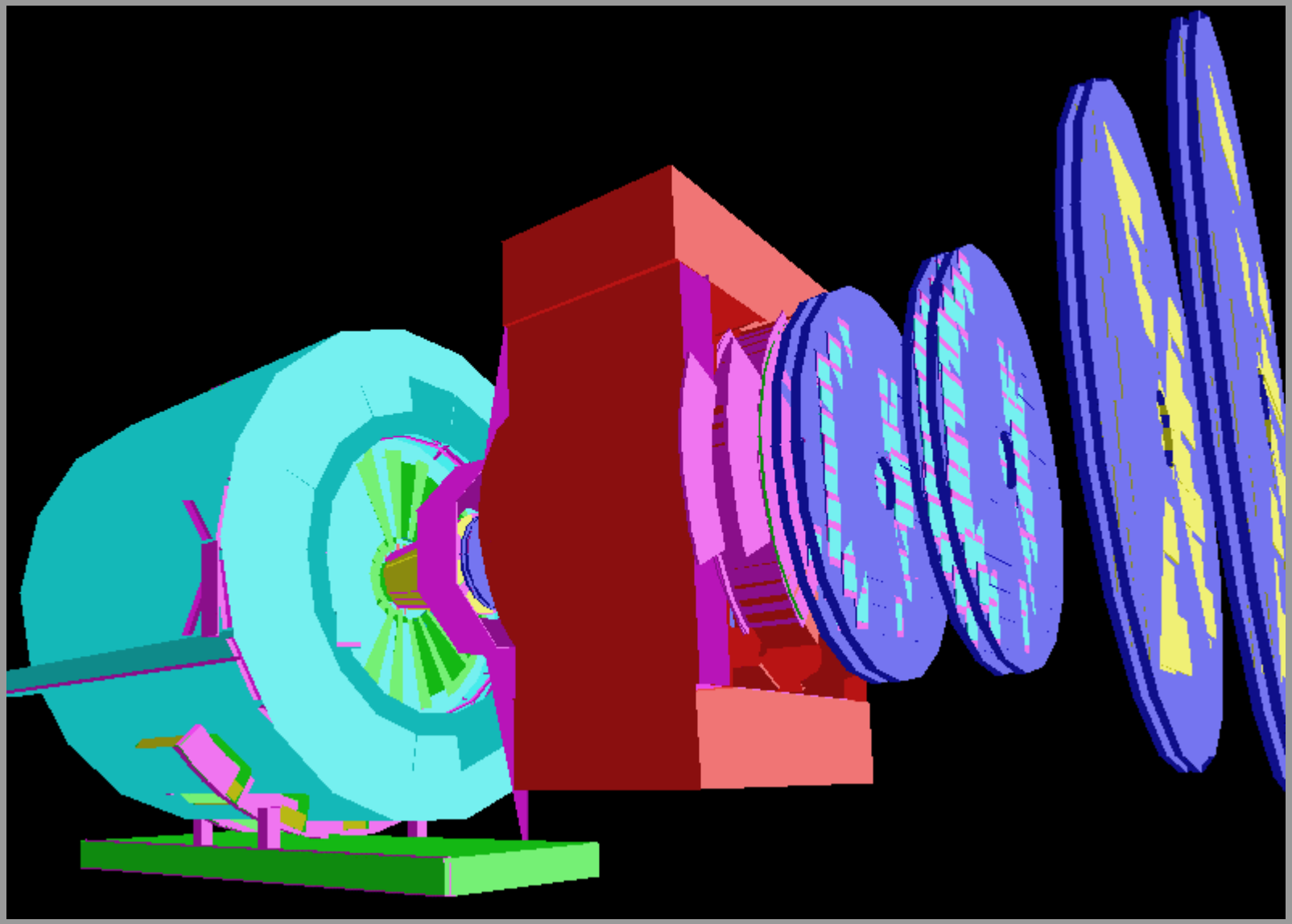
Visualization

- ROOT visualization in the pad and with x3d client.

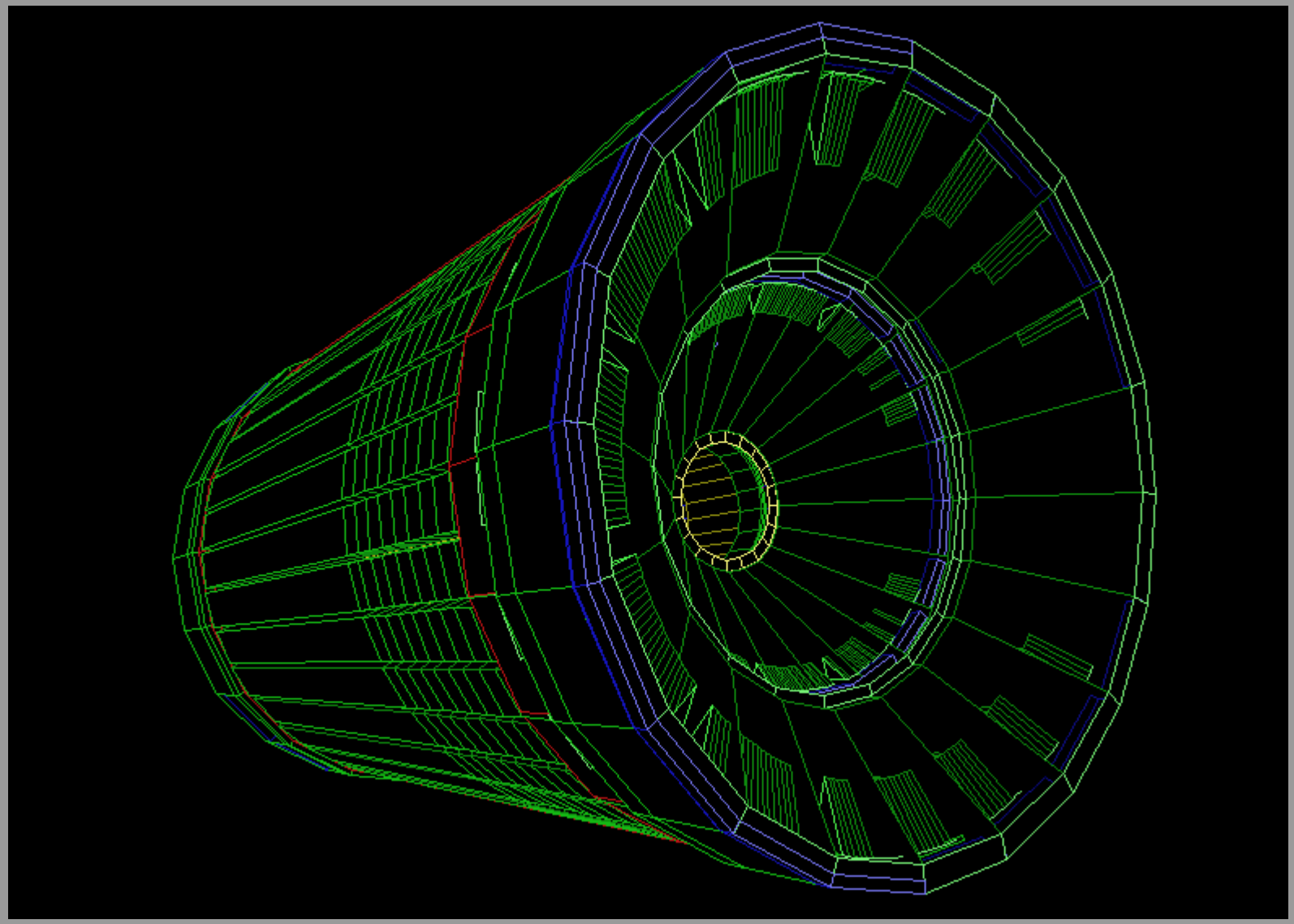
Next - OpenGL

- Done via the context menu of volumes within the browser
- Any volume is a 3-D object that has not only 3D attributes, but also visibility flags, colour, line and marker attributes to which the user may interact either at build or run time
- Global visibility options (drawing modes) can be set from the geometry manager class
  - mode 1: three levels down starting from the current volume
  - mode 2: only the leaves of the current branch are drawn.
- Tested not only on ALICE geometry, but also others

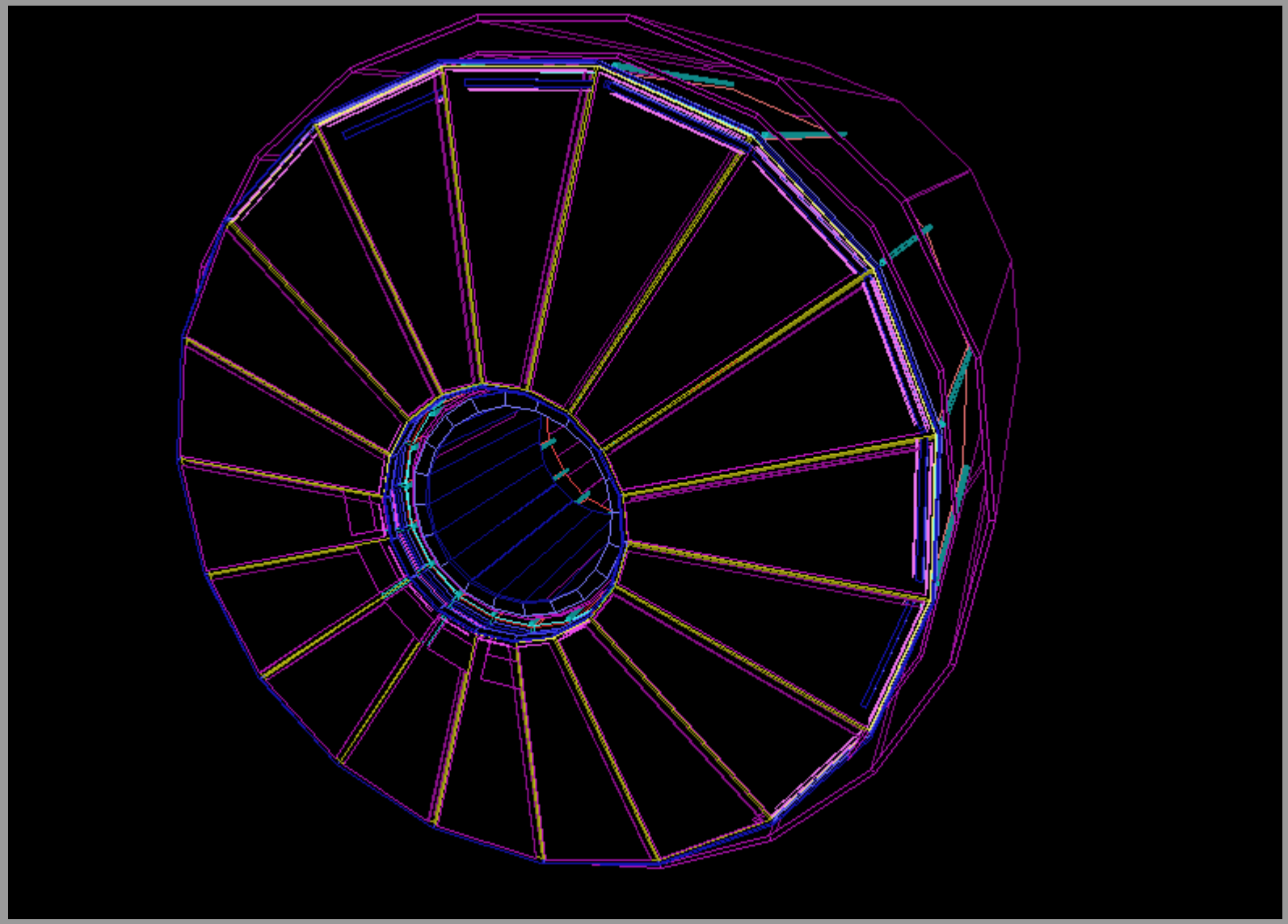
Most of ALICE...



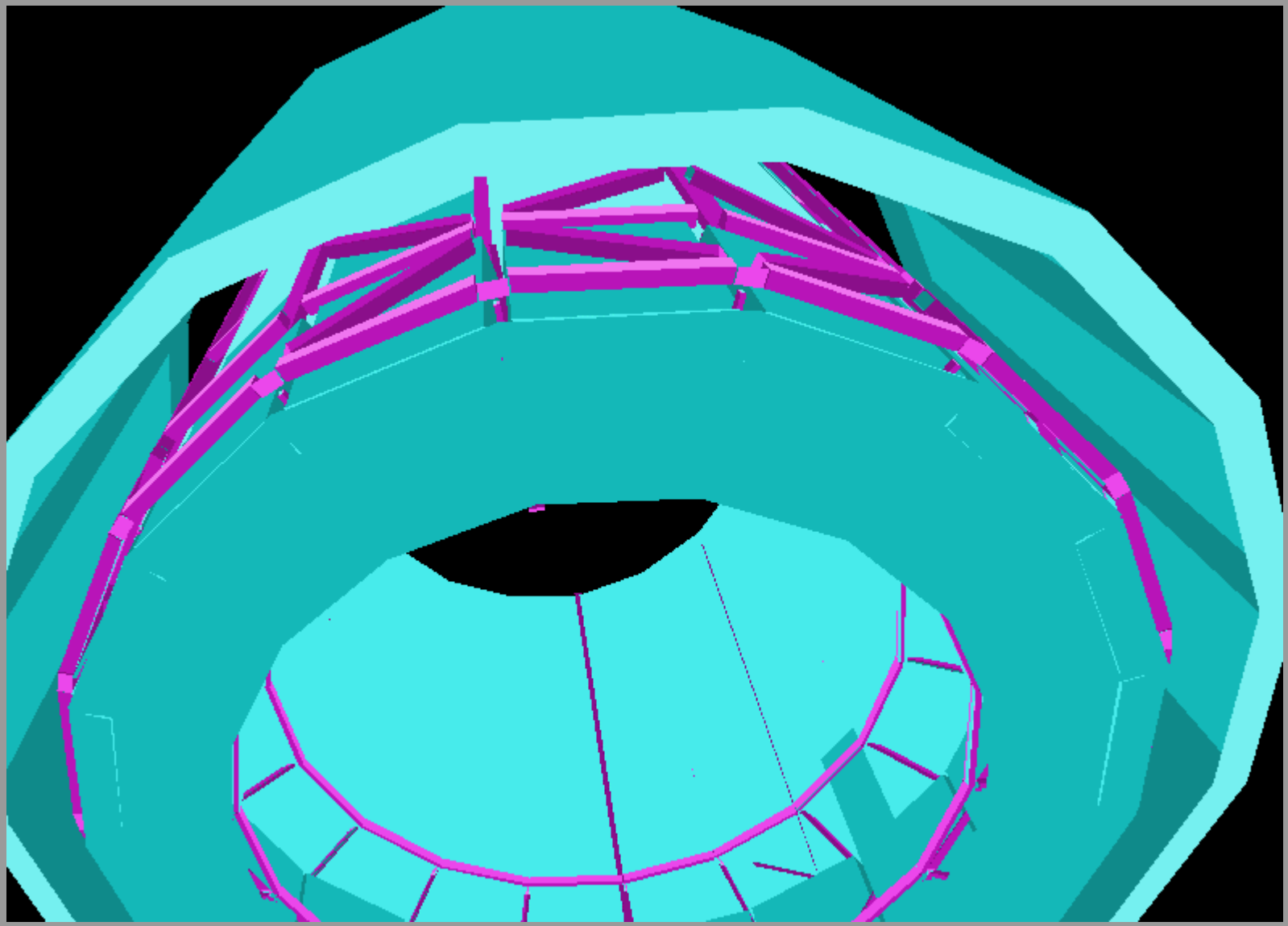
ITS



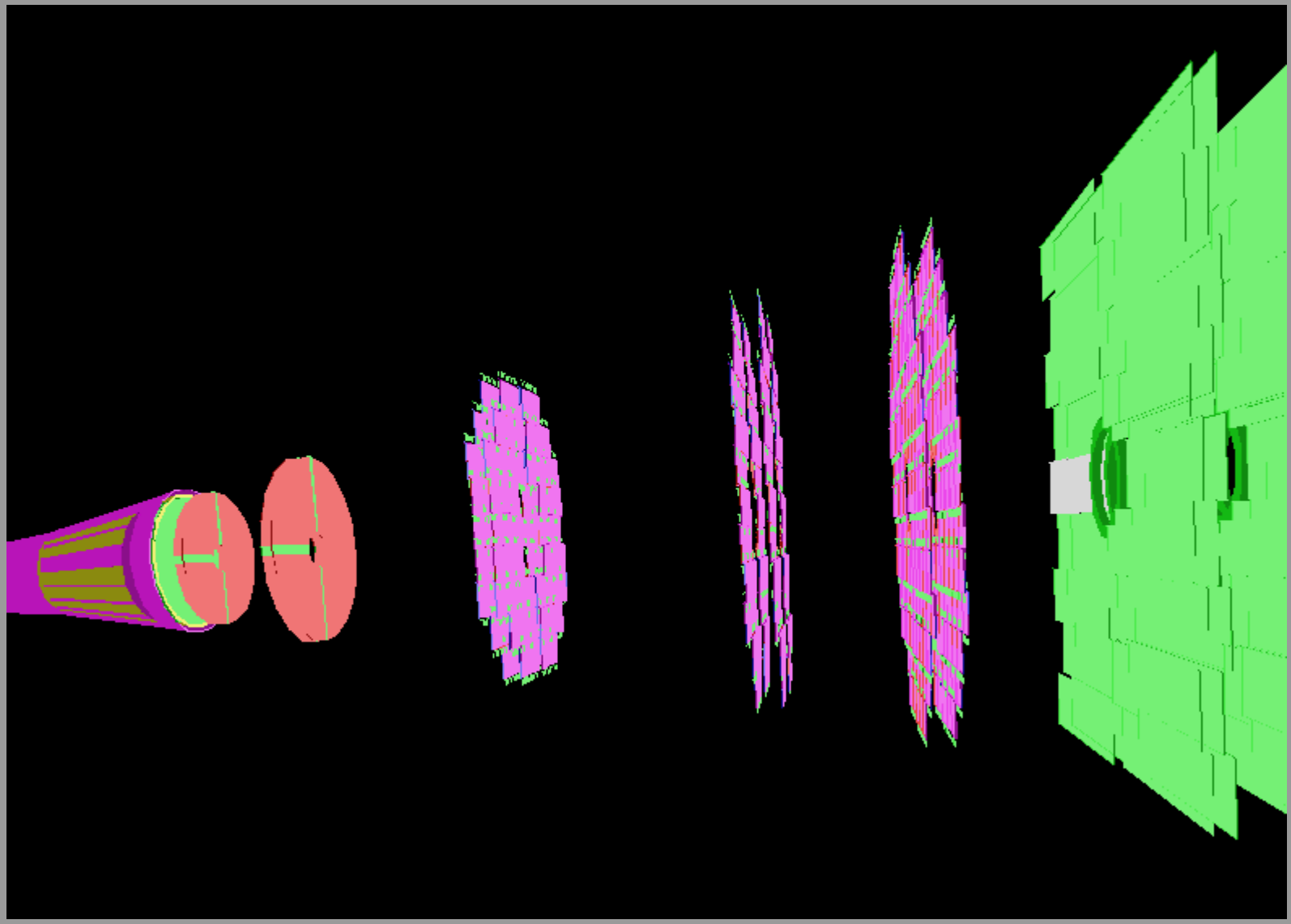
TPC



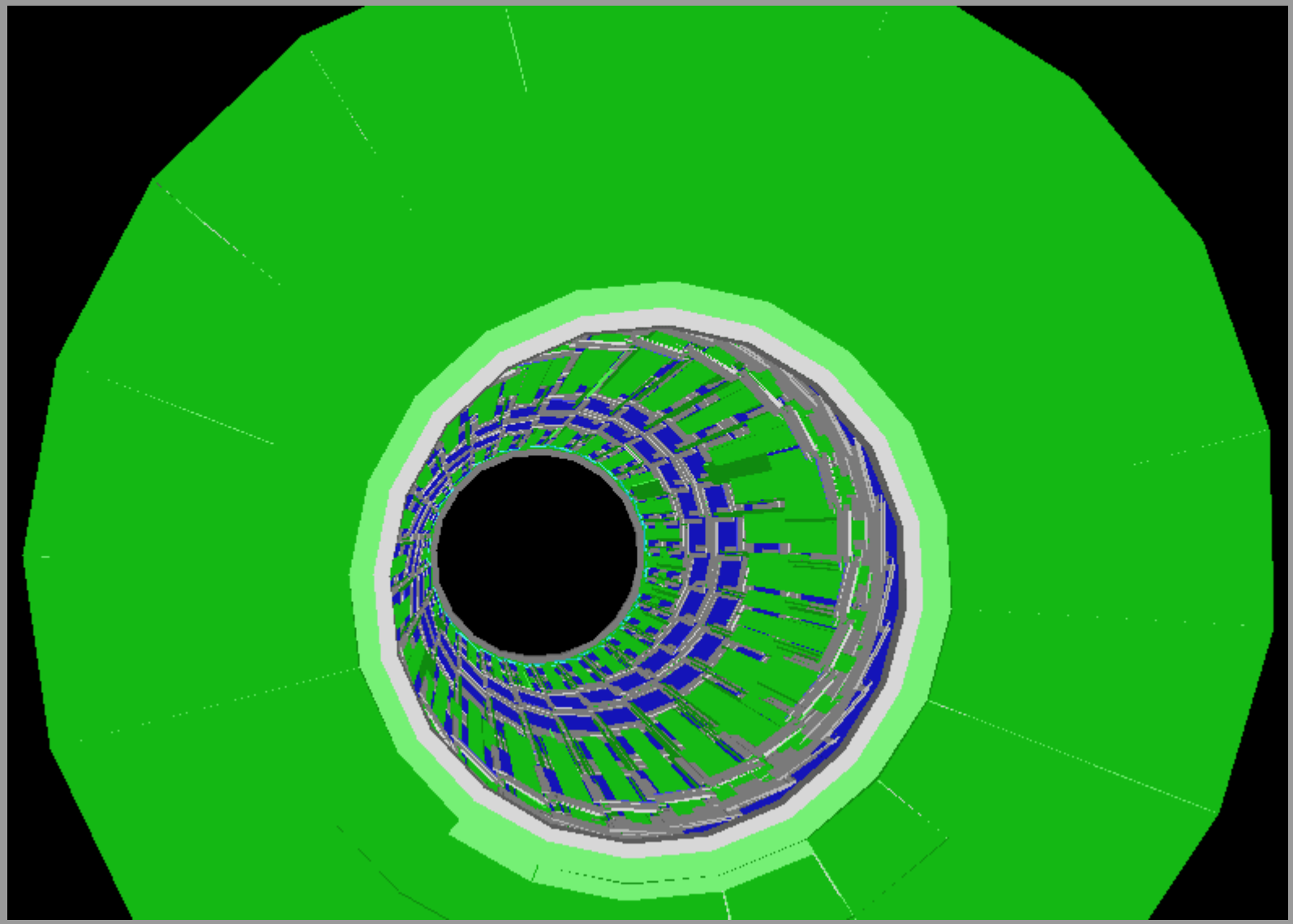
Frame



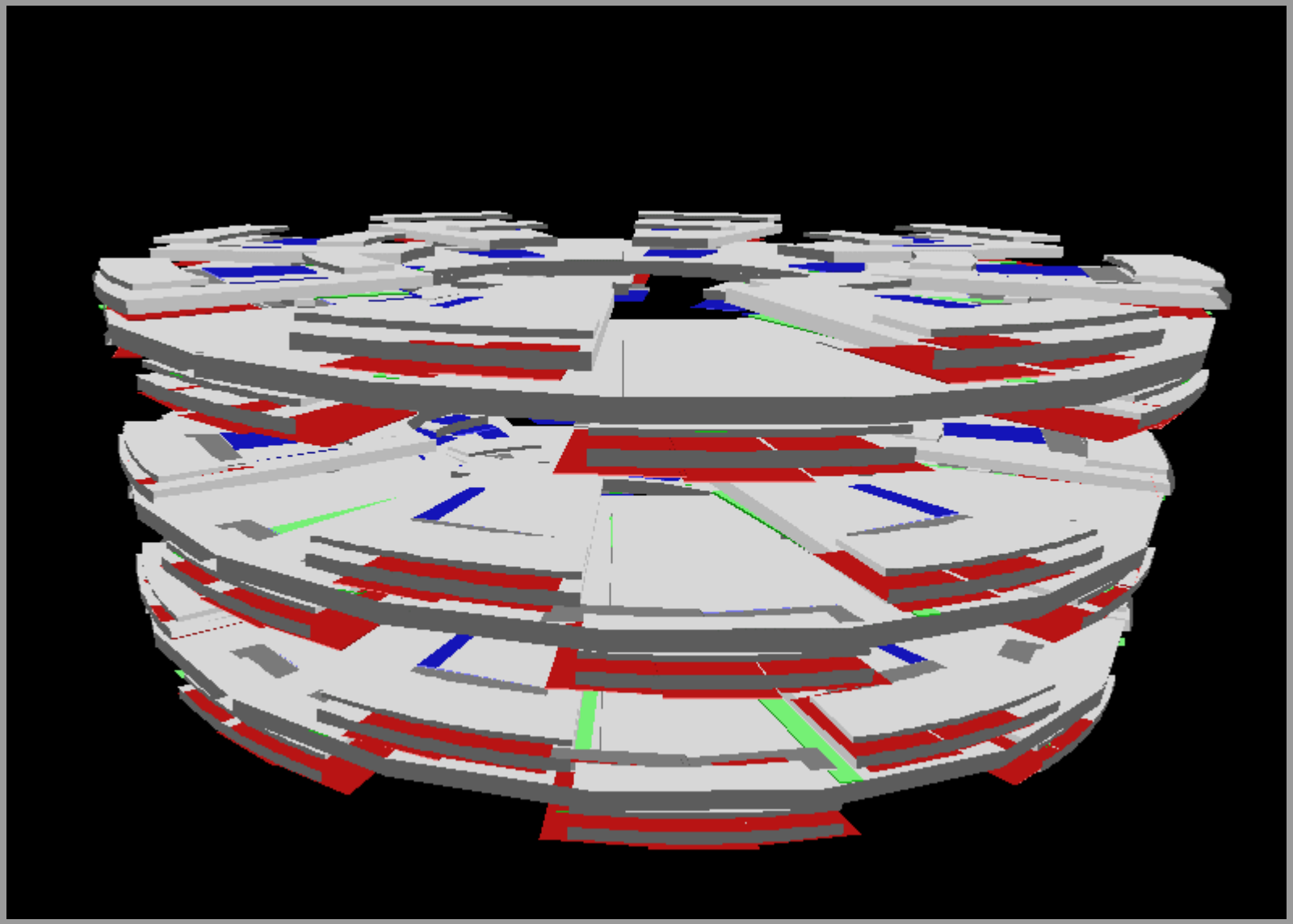
MUON

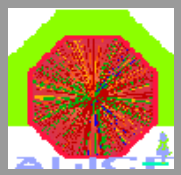


Structures in CMS

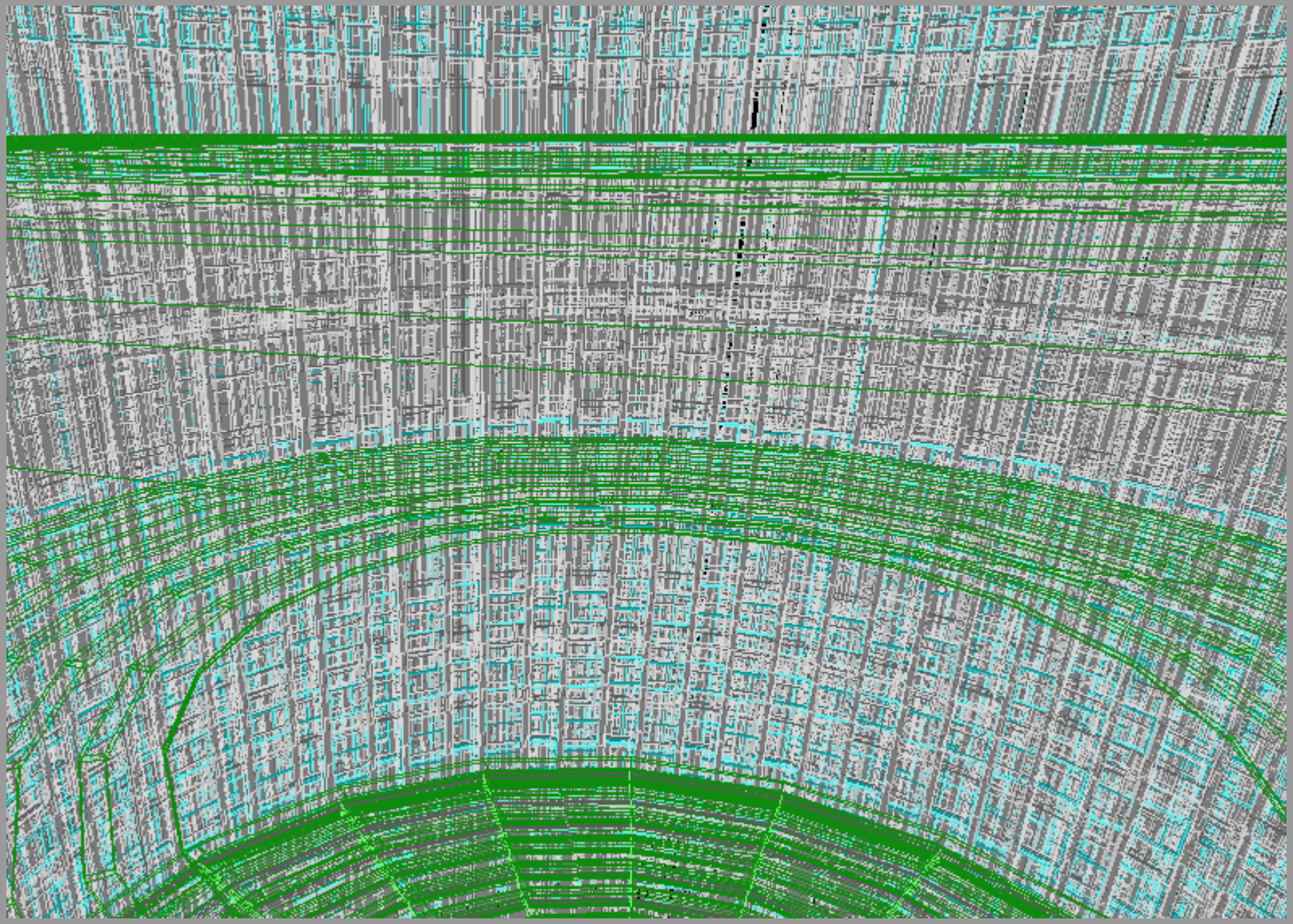


and others...

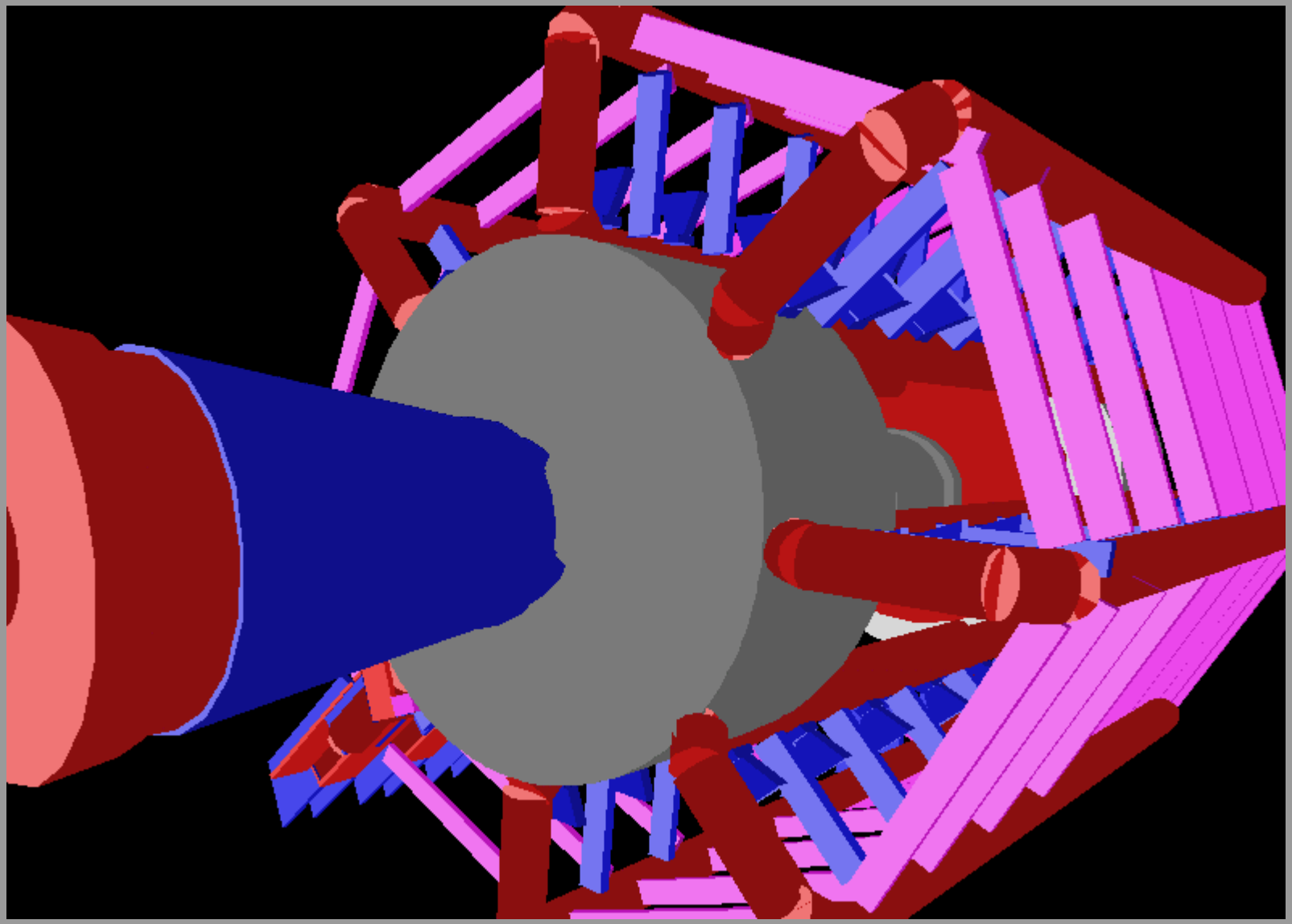


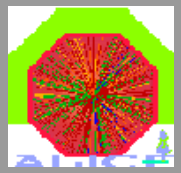


we can draw divisions



ATLAS geometry





Milestones

- Benchmarks for "Where am I?" in ALICE geometry. - 1 week
- Last iteration of debugging for point classification algorithm, code cleaning and restructuring in view of committing to ROOT CVS. ~ 1 month
- Distance to the next boundary implementation ~ 1 month
- Start working on composite shapes ~ 2 months
- Algorithms for computing normal to surface and safety ~ 3 months
- Implementation of the interface to AliRoot ~ 4 months
- Design an implementation of a geometrical GUI, including checking tools and builder features.
- Provide support for different I/O GDL formats.