Geant4 introduction: Materials, Geometry, Sensistive detector

R. Pestotnik
Jožef Stefan Institute, Ljubljana

- Materials
- Solids
- Volumes
- Sensitive Volumes

Describe Your Detector

- Derive your own concrete class from G4VUserDetectorConstruction
- In the virtual method Construct(),
  - assemble all necessary materials
  - build the volumes of your detector geometry
  - construct sensitive detector classes and assign them to the detector volumes
- Optionally you may define:
  - regions for any part of your detector (for production ranges)
  - visualization attributes of detector elements
  - magnetic (or other) fields

Materials in Geant4

→ Three main classes in the Geant4 design

- Isotopes ←→ G4Isotope
- Elements ←→ G4Element
- Molecules, compounds, mixtures ←→ G4Material

- G4Isotope and G4Element describe the properties of the atoms:
  - Atomic number, number of nucleons, mass of a mole, shell energies
  - Cross-sections per atoms, etc…

- G4Material class
  - The only one visible to the rest of the toolkit: Used by tracking, geometry, physics
  - Contains all the information relative to the eventual elements and isotopes of which it is made (at the same time hiding implementation details)

- G4Material describes the macroscopic properties of the matter:
  - temperature, pressure, state, density
  - Radiation length, absorption length, etc…

G4SUPERB :

Use NIST materials as much as possible.

Some materials are also defined in B4Master_MaterialFactory

Otherwise you can define your materials in B4XXX_MaterialFactory

NIST materials in Geant4

=====================================
Elementary Materials from the NIST Data Base
=====================================

Z	Name	ChFormula	density(g/cm^3)	I(eV)
1	G4_H	H_2	8.3748e-05	19.2
2	G4_He		0.000166322	41.8
3	G4_Li		0.534	40
4	G4_Be		1.848	63.7
5	G4_B		2.37	76
6	G4_C		2	81
7	G4_N	N_2	0.0011652	82
8	G4_O	O_2	0.00133151	95
9	G4_F		0.00158029	115

=====================================
Compound Materials from the NIST Data Base
=====================================

N	Name	ChFormula	density(g/cm^3)	I(eV)
4	G4_Air		0.00120479	85.7
		6	0.000124	
		7	0.755268	
		8	0.231781	
		18	0.012827	
2	G4_CsI		4.51	553.1
		53	0.47692	
		55	0.52308	

- NIST Elementary Materials
  - H → Cf (Z=1→98)
- NIST compounds
  - E.g. “G4_ADIPOSE_TISSUE_ICRP”
- HEP and Nuclear Materials
  - E.g. liquid Ar, $PbWO_4$

- It is possible to build mixtures of NIST and user-defined materials

- Natural isotope compositions
- More than 3000 isotope masses

are used for definition of NIST elements

How to use materials in Geant4

- Do not need anymore to predefine elements and materials
- Main new user interfaces:

```
G4NistManager* manager = G4NistManager::GetPointer();
 G4Element* elm = manager->FindOrBuildElement("symb", G4bool iso);
 G4Element* elm = manager->FindOrBuildElement(G4int Z, G4bool iso);

G4Material* mat = manager->FindOrBuildMaterial("name", G4bool iso);
G4Material* mat = manager->ConstructNewMaterial("name",
                            const std::vector<G4int>& Z,
                            const std::vector<G4double>& weight,
                            G4double density, G4bool iso);

 G4double isotopeMass = manager->GetMass(G4int Z, G4int N);
 G4Material* mymaterial =
    new G4Material("mymaterial",density,ncomponents=2);
 Aerog->AddMaterial(elm,fractionmass=60*perCent);
 Aerog->AddElement (mat ,fractionmass= 40*perCent);
```

- UI commands

`/material/nist/printElement` ← print defined elements
`/material/nist/listMaterials` ← print defined materials

Detector geometry

- A detector geometry in Geant4 is made of a number of volumes.
- The largest volume is called the World volume(tsukuba_hall in g4superb) . It must contain all other volumes in the detector geometry
- The other volumes are created and placed inside previous volumes, including the World.
- Each volume is created by describing its shape and its physical characteristics and then placing it inside a containing volume.
- The coordinate system used to specify where the daughter volume is placed is the one of the mother.

G4SUPERB: Define your detector/component region(boundary) in

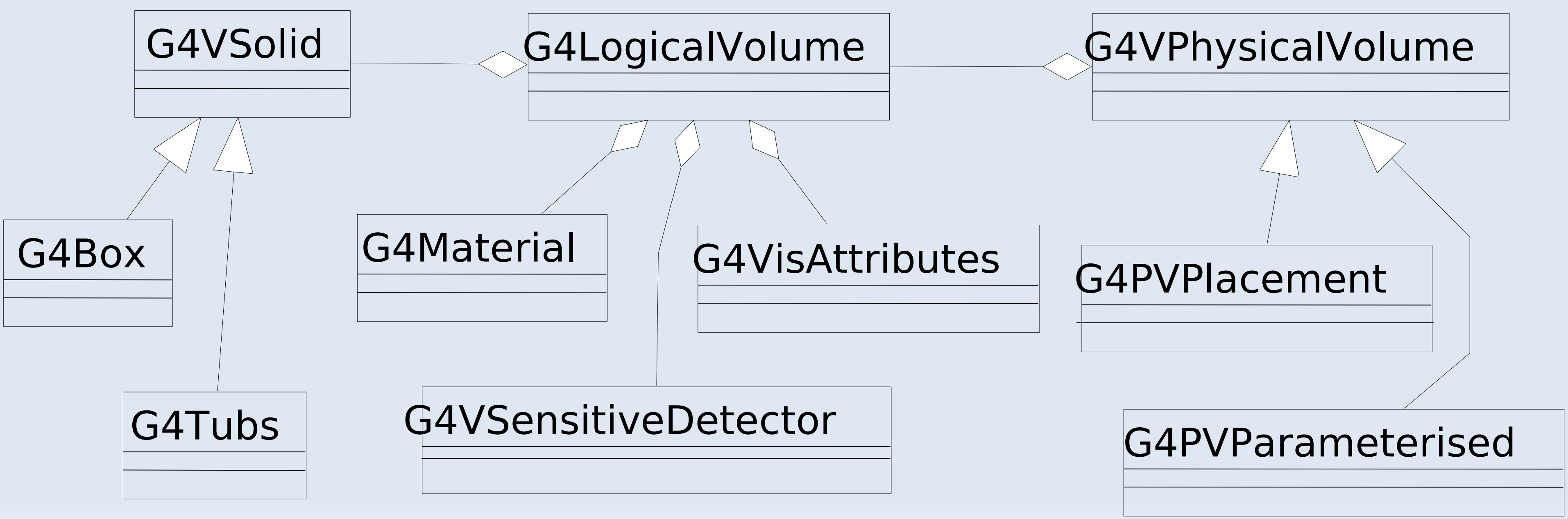
B4XXX::buildVolume().

Volumes

- A Solid is used to describe a volume’s shape. A solid is a geometrical object that has a shape and specific values for each of that shape’s dimensions
- A Logical Volume is use for describing a volume’s full properties. It starts from its geometrical properties (the solid) and adds physical characteristics, like the material, the sensitivity, the magnetic field, the color…
- What remains to describe is the position of the volume. For doing that, one creates a Physical volumes, which places a copy of the logical volume inside a larger, containing volume.

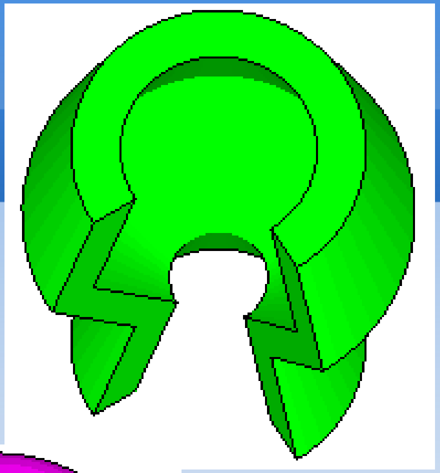
Define detector geometry

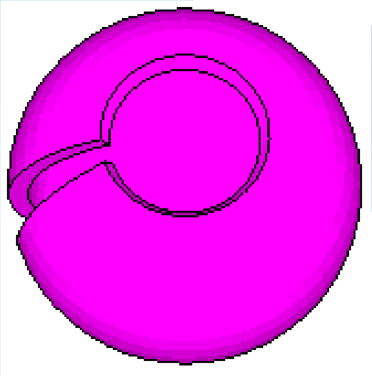
- Three conceptual layers
  - **Solid** -- *shape, size*
  - **Logical Volume** -- *material, sensitivity, user limits, etc.*
  - **Physical Volume** -- *position, rotation*

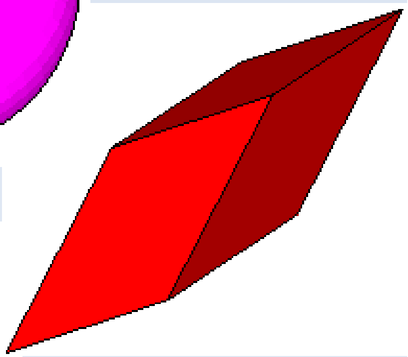


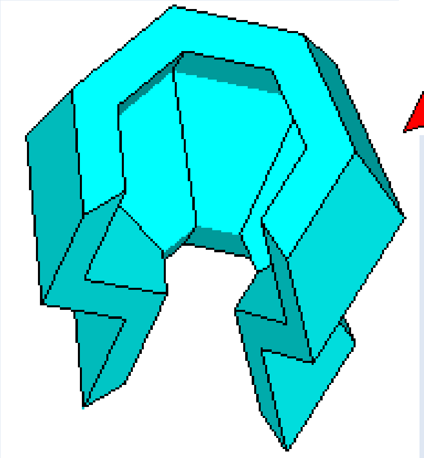
Solids

- Solids defined in Geant4:
  - CSG (Constructed Solid Geometry) solids
    - G4Box, G4Tubs, G4Cons, G4Trd, …
    - Analogous to simple GEANT3 CSG solids
  - Specific solids (CSG like)
    - G4Polycone, G4Polyhedra, G4Hype, …
  - BREP (Boundary REPresented) solids
    - G4BREPSolidPolycone, G4BSplineSurface,
    - Any order surface
  - Boolean solids
    - G4UnionSolid, G4SubtractionSolid, …









```
G4Box* box  = new G4Box(“BoxName”, x, y,z);
```

G4SUPERB:
Define your solids in B4XXX_MaterialFactory

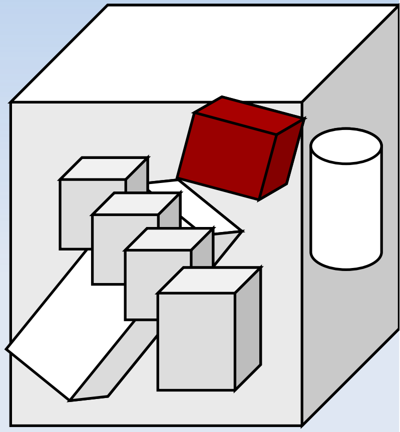
G4LogicalVolume

```
G4LogicalVolume(G4VSolid* pSolid, G4Material* pMaterial,
                const G4String& name, G4FieldManager* pFieldMgr=0,
                G4VSensitiveDetector* pSDetector=0,
                G4UserLimits* pULimits=0,
                G4bool optimise=true);
```

- Contains all information of volume except position:
  - Shape and dimension (G4VSolid)
  - Material, sensitivity, visualization attributes
  - Position of daughter volumes
  - Magnetic field, User limits
  - Shower parameterisation
- Physical volumes of same type can share a logical volume.
- The pointers to solid and material must be NOT null
- Once created it is automatically entered in the LV store
- It is not meant to act as a base class

G4VPhysicalVolume

- G4PVPlacement 1 Placement = One Volume
  - A volume instance positioned once in a mother volume
- G4PVParameterised 1 Parameterised = Many Volumes
  - Parameterised by the copy number
    - Shape, size, material, position and rotation can be parameterised, by implementing a concrete class of `G4VPVParameterisation`.
  - Reduction of memory consumption
    - Currently: parameterisation can be used only for volumes that either a) have no further daughters <u>or</u> b) are identical in size & shape.
- G4PVReplica 1 Replica = Many Volumes
  - Slicing a volume into smaller pieces (if it has a symmetry)



How to make you detector sensitive: Sensitive detector

- Each Logical Volume can have a pointer to a sensitive detector.
  - Then this volume becomes sensitive
  - Sensitive detector is a user-defined class derived from G4VSensitiveDetector
- Hit is a snapshot of the physical interaction of a track or an accumulation of interactions of tracks in the sensitive region of your detector
- A sensitive detector creates hit(s) using the information given in `G4Step` object. The user has to provide his/her own implementation of the detector response.
  - UserSteppingAction class should NOT do this
- Hit objects, which are still the user's class objects, are collected in a `G4Event` object at the end of an event

G4SUPERB: Define your sensitive detector in B4XXX_SensitiveDetector

Making a detecor sensitive

- Basic strategy

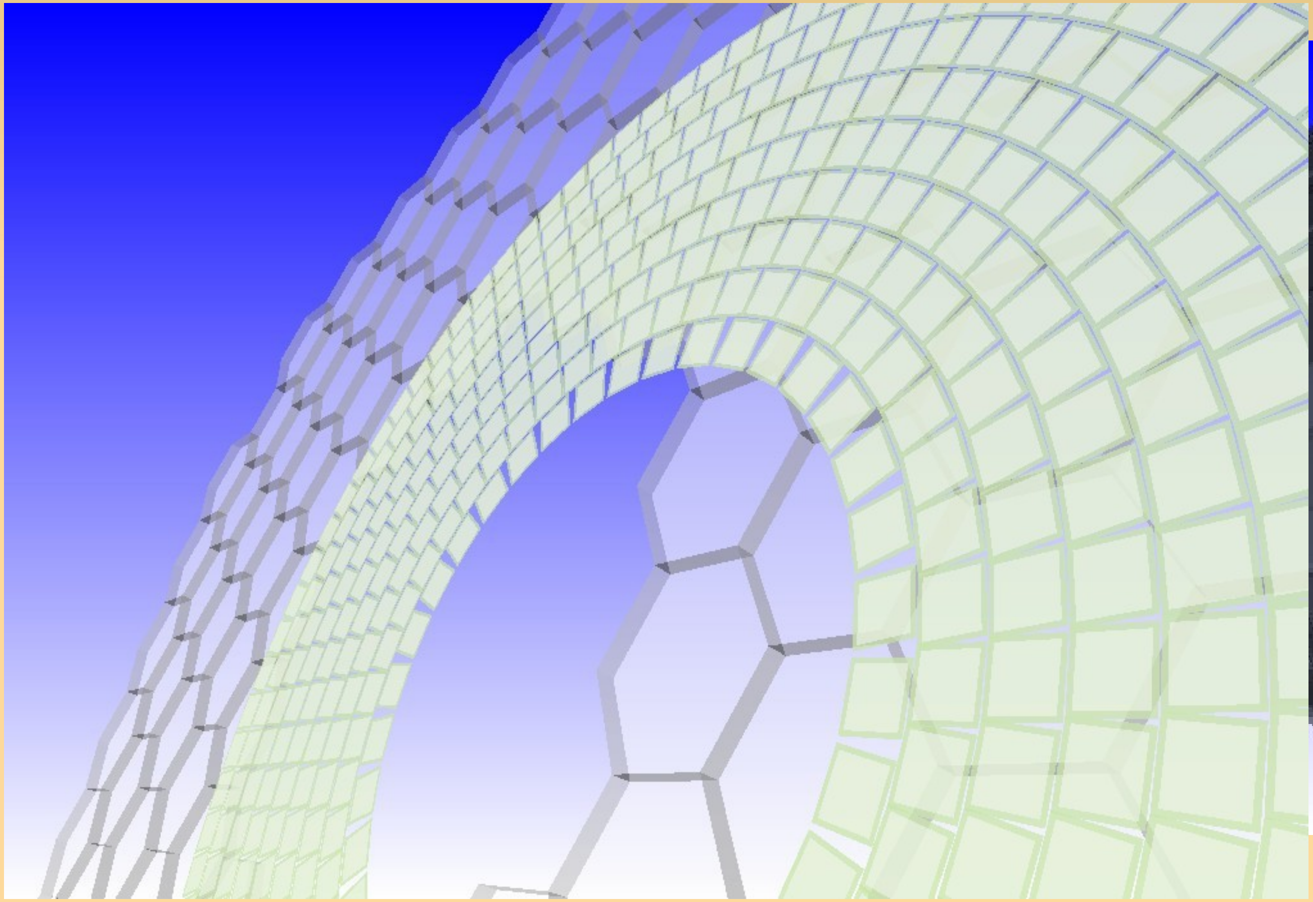
```
G4LogicalVolume* myLogCalor = ……;
G4VSensitiveDetector* pSensitivePart =
   new MyCalorimeter("/mydet/calorimeter1");
G4SDManager* SDMan = G4SDManager::GetSDMpointer();
SDMan->AddNewDetector(pSensitivePart);
myLogCalor->SetSensitiveDetector(pSensitivePart);
```

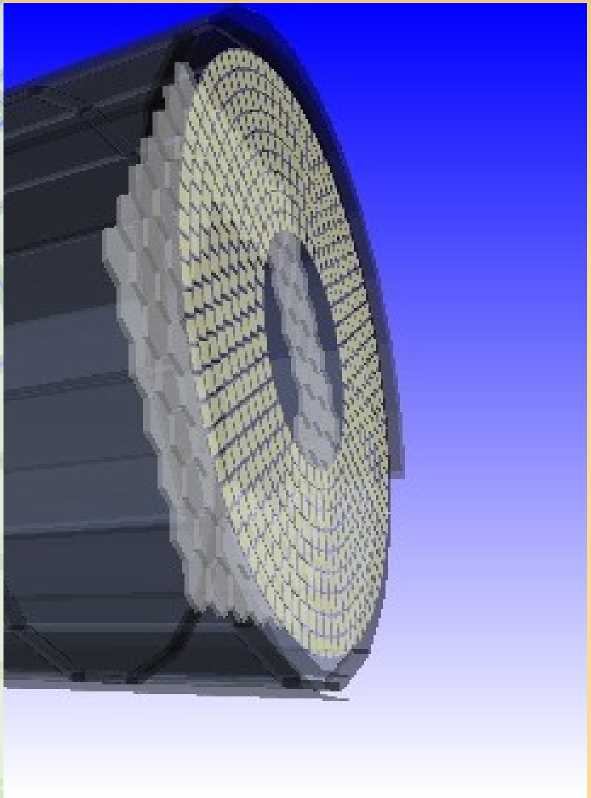
- Each detector object must have a unique name.
  - Some logical volumes can share one detector object
  - More than one detector objects can be made from one detector class with different detector name
  - One logical volume cannot have more than one detector objects. But, one detector object can generate more than one kinds of hits
    - e.g. a drift chamber class may generate anode and cathode hits separately

Summary

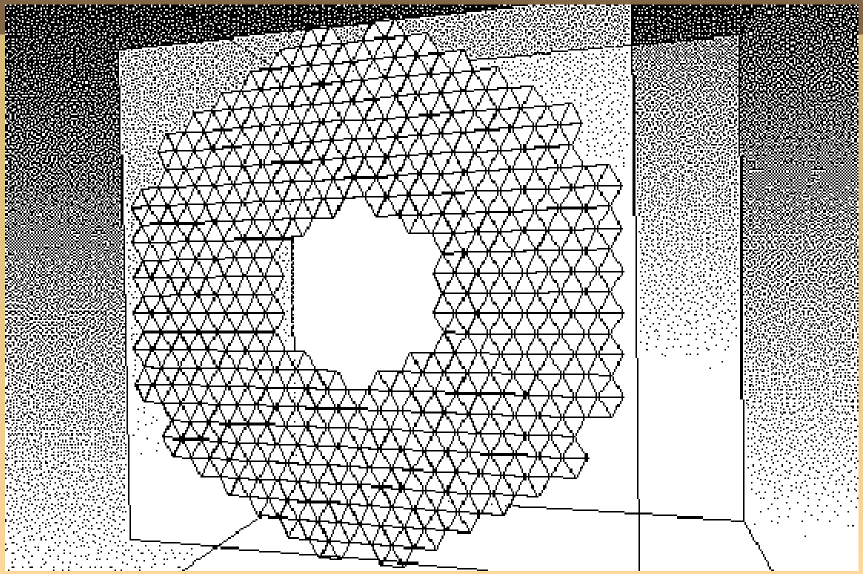
- Your (sub)detector is now build and sensitive.
- You can start using it;)

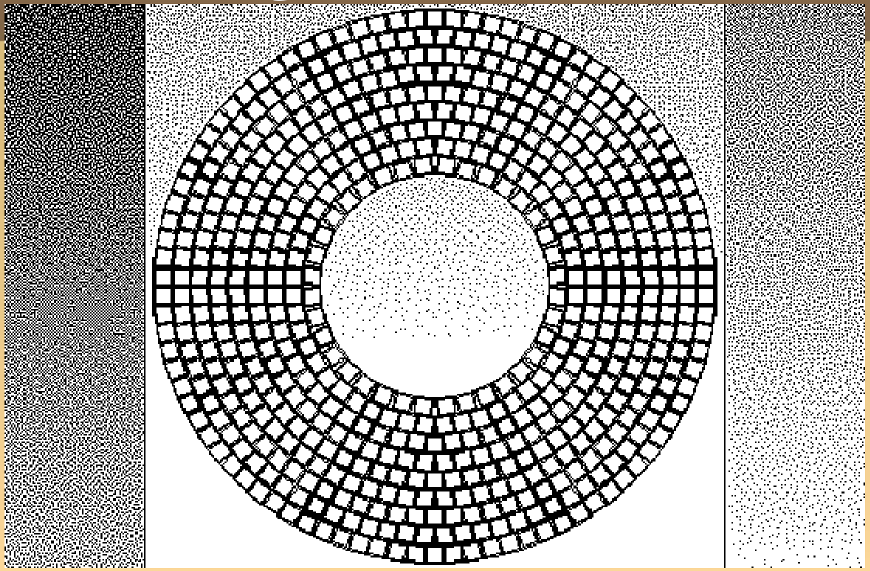
G4superb RICH – status and plan





Status Geometry





Geometry - basic functionality is included

- aerogel: hexagon tiles, missing rombs and pentagons on the borders
- photon detector: still in search for the best detector candidate,
  - the 144ch HAPD module block (quartz window+ Si layer+ PCB plate)
- The support frames not fixed yet
- Specific Processes in RICH:
  - Cherenkov photon generation
  - Rayleigh scattering in aerogel
  - Optical photon transport

Status Hits and Reconstruction

Hits

- Hits in the photon detector are generated and detected.
-

Reconstruction:

- standalone reconstruction on the simplified geometry:

http://kds.kek.jp/getFile.py/access?contribId=2&resId=0&materialId=slides&confId=204

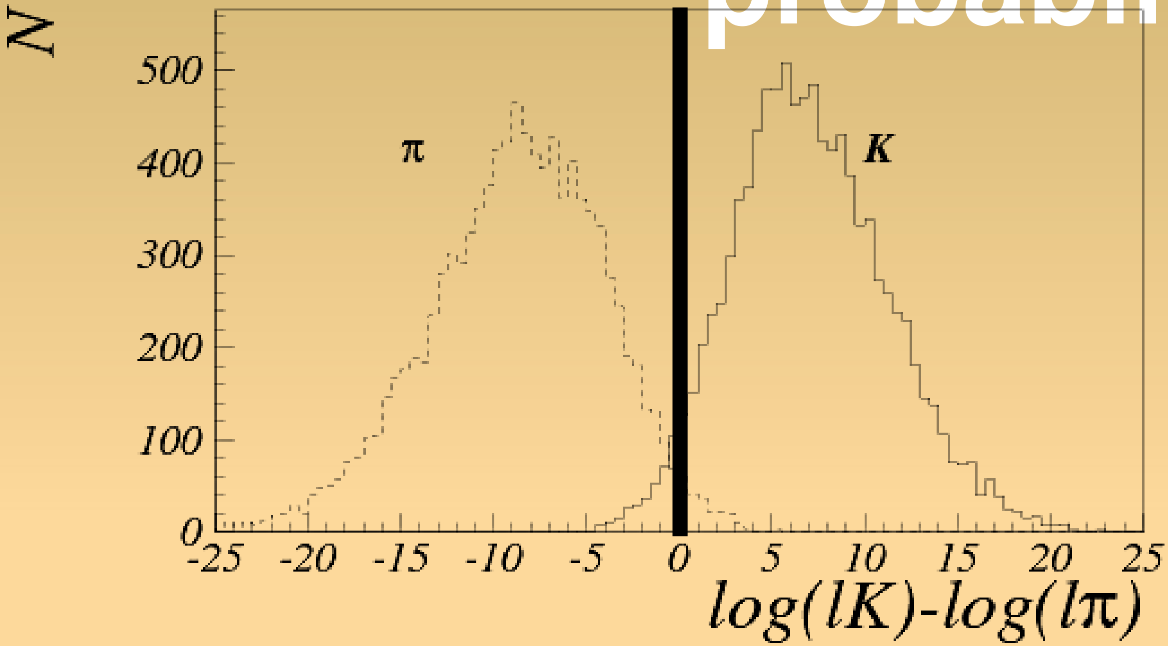
- based on the likelihood calculation:

$$ln\, L = -N + \sum_{\text{hit } i} n_i + \sum_{\text{hit } i} ln\left(1 - e^{-n_i}\right)$$

n_i expected number of photons in the pad i

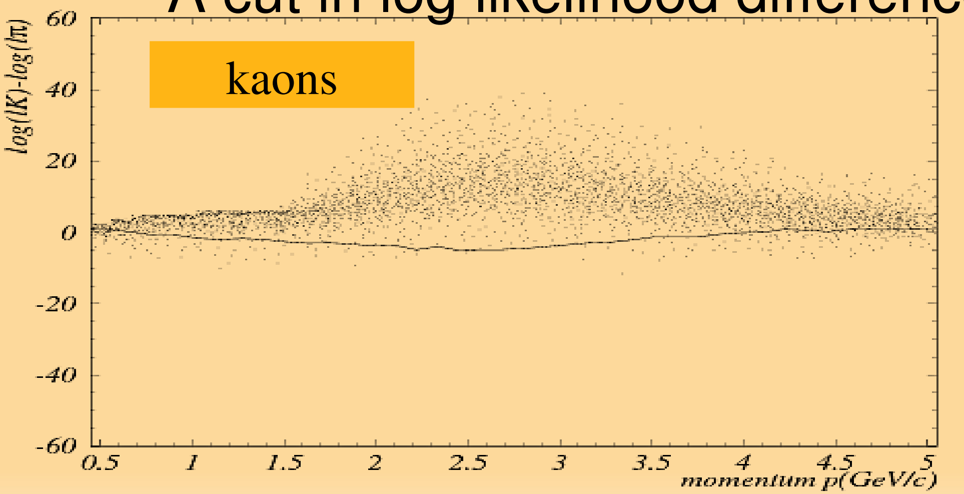
- I am moving the code to g4superb

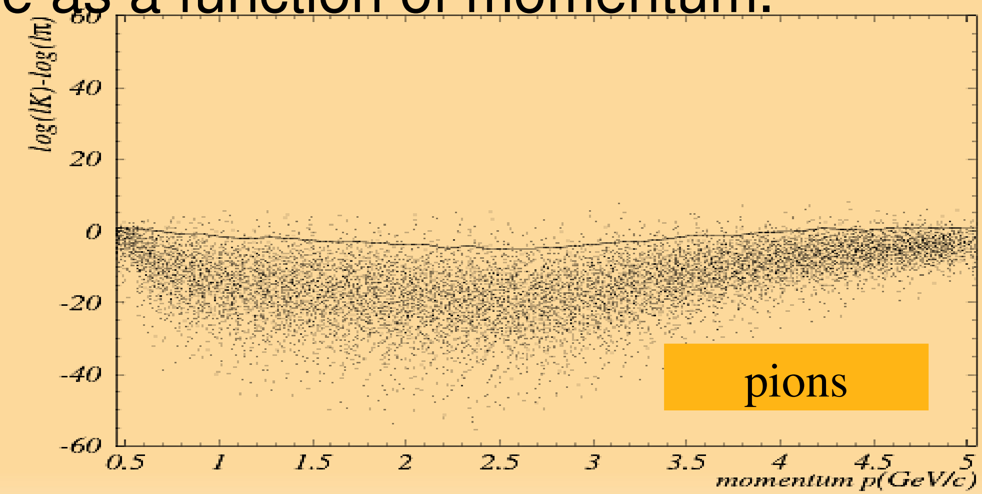
Identification efficiency and fake probabilities



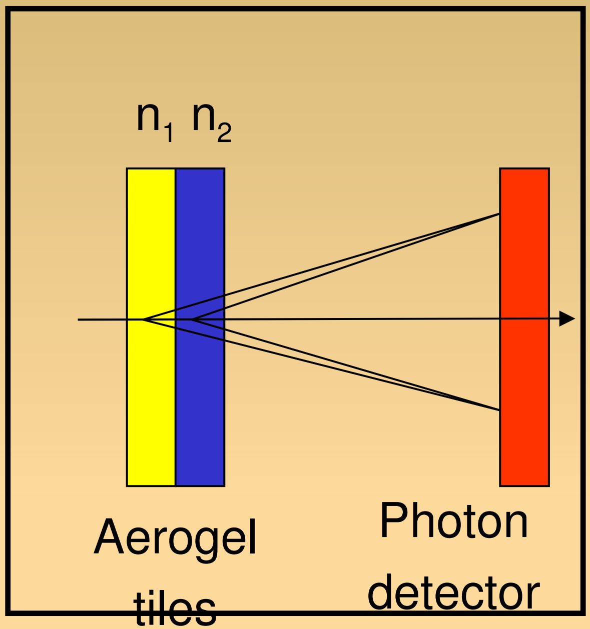
- fix fake rate probability π→K for each momentum bin
- calculate efficiency for K for each momentum bin

- A cut in log likelihood difference as a function of momentum.

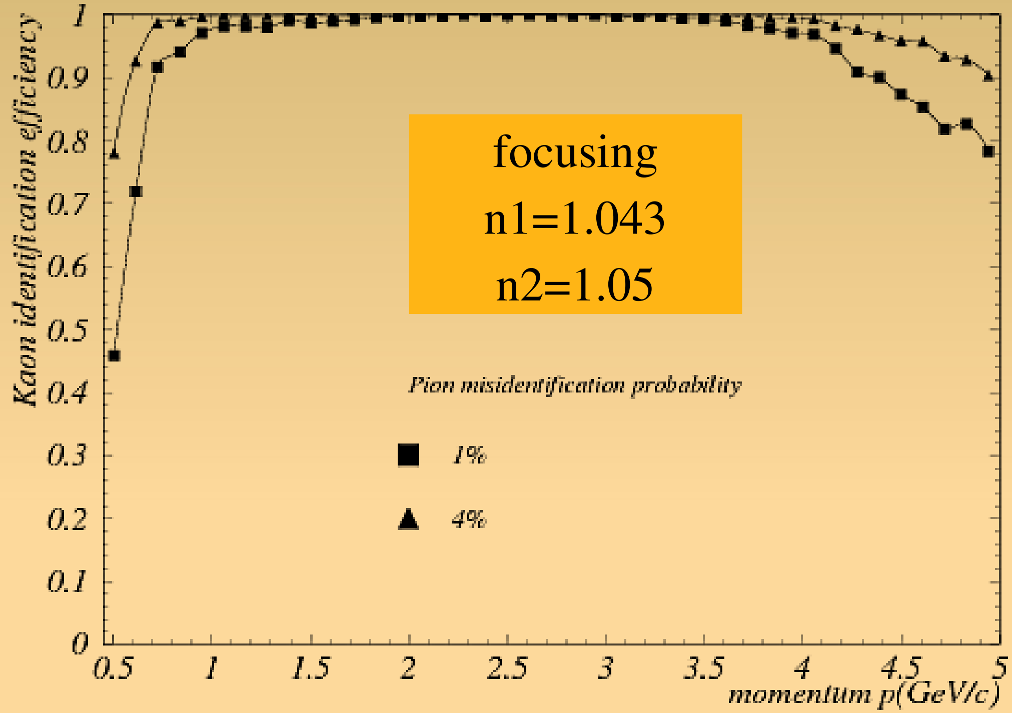




K identification efficiency focusing configuration



Focusing type



- Resulting kaon identification efficiency as a function of momentum for two pion misidentification probabilities: 1% and 4%

Summary

Status of the Aerogel RICH code in the G4superb

- geometry done
- hits   done
- reconstruction under move to G4Belle ; was waiting for g4ext module