Środowisko programistyczne GEANT4

Leszek Adamczyk

Wydział Fizyki i Informatyki Stosowanej
Akademia Górniczo-Hutnicza

Wykłady w semestrze zimowym 2013/2014

Wizualizacje : katalogi poleceń /vis

```
Idle> ls /vis
   /vis/ASCIITree/   Commands for ASCIITree control.
   /vis/heprep/      HepRep commands.
   /vis/rayTracer/   RayTracer commands.
   /vis/gMocren/     gMocren commands.
   /vis/ogl/         G4OpenGLViewer commands.
   /vis/modeling/    Modeling commands.
   /vis/filtering/   Filtering commands.
   /vis/geometry/    Operations on vis attributes of Geant4
   /vis/scene/       Operations on Geant4 scenes.
   /vis/sceneHandler/   Operations on Geant4 scene handlers
   /vis/viewer/      Operations on Geant4 viewers.
 Commands :
   verbose * Simple graded message scheme - digit or string
   initialize * Initialise visualisation manager.
   abortReviewKeptEvents * Abort review of kept events.
   enable * Enables/disables visualization system.
   disable * Disables visualization system.
   list * Lists visualization parameters.
```

Wizualizacje : żargon

- scene
  zbiór 3d. obiektów które chcemy wizualizować (format GEANT'a) ;
- scene handler
  obsługa sceny przygotowująca scenę do wizualizacji (format graficzny);
- viewer
  generuje obraz na podstawie danych produkowanych przez obsługę sceny;
- Interfejs wizualizacji = obsługa sceny + generator obrazu

Wizualizacje : ogólny schemat

1. Kreacja obsługi sceny i generatora obrazu
2. Kreacja pustej sceny
3. Dodanie obiektów 3D do pustej sceny
4. Przypisanie obsługi sceny do sceny
5. Ustalenie stylu grafiki (kąt patrzenia, przeźroczystość)
6. Generator obrazu dokonuje wizualizacji
7. Deklaracja zakończenia wizualizacji sceny

Wiele poleceń wykonuje kilka kroków na raz:

- `/vis/open` krok 1 i 2
- `/vis/drawVolume` krok 3,4,5 i 6

Wizualizacje: /vis/viewer/

- poniższe polecenia określają sposób wizualizacji i są wymagane tyko do pracy z OpenGL, w przypadku HepRep i DAWN odpowiednie polecenia wykonuje się później w programach zewnętrznych:
- część poleceń dostępna za pomocą skrótów klawiszowych, myszy i ich kombinacji.

1 Ustawienie kąta widzenia

```
/vis/viewer/set/viewpointThetaPhi <theta> <phi>
```

2 Powiększenie/pomniejszenie

```
/vis/viewer/zoom <scale_factor>
```

3 Powrót do początkowego sposobu wizualizacji

```
/vis/viewer/reset
```

4 Ustawienie stylu wizualizacji obiektów

```
/vis/viewer/set/style <style>
```

`<style>=wireframe, surface`
Polecenie nie działa jeśli styl wizualizacji został wymuszony w kodzie c++ za pomocą metod: `setForceWireframe` lub `setForceSolid`

Wizualizacje: /vis/viewer/ /vis/scene/add

5 Wizualizacja linii pomocniczych

```
/vis/viewer/set/auxiliaryEdge <bool>
```

6 Brak wizualizacji zasłoniętych linii

```
/vis/viewer/set/hiddenEdge <bool>
```

7 Dodanie prostoktnego układu osi

```
/vis/scene/add/axes <x0> <y0> <z0> <length> <unit>
```

8 Dodanie trajektorii czastek

```
/vis/scene/add/trajectories <smooth> <rich>
```

`smooth`
Dodanie pomocniczych punktów w celu polepszenia wizualizacji torów krzywoliniowych

Wizualizacje: Akumulacja przypadków

9 Aby wizualizować trajektorie cząstek należy je symulować

```
/run/beamOn <nevents>
```

Domyślnie zobaczymy wizualizacje każdego przypadku z osobna.

10 Akumulacja trajektorii dla wszystkich przypadków

```
/vis/scene/endOfEventAction accumulate
```

- Powrót do wizualizacji po każdym przypadku

```
/vis/scene/endOfEventAction refresh
```

- Przeglądanie zgromadzonych przypadków

```
/vis/reviewKeptEvents
```

polecenia `continue` lub `/vis/abortReviewKeptEvents`

- W kodzie c++ można użyć polecenia do gromadzenia ciekawych przypadków

```
G4EventManager->KeepTheCurrentEvent()
```

Takie rozwiązanie pozwala na symulację dużej liczby przypadków ale gromadzenie do wizualizacji tylko tych które są ciekawe.

Wizualizacje: Odświeżanie grafiki

- W OpenGl wyniki poleceń widoczne sa natychmiast w okienku graficznym. Jesli geometria, trajektorie sa bardzo skomplikowane to tak częste odświeżanie obrazu spowalnia wykonanie symulacji.

11 Aby tymczasowo wyłączyć odświeżanie:

```
/vis/viewer/set/autoRefresh false
```

11 Aby wymusić odświeżenie:

```
/vis/viewer/set/autoRefresh true
```

12 Ilością informacji o wizualizacji jaka pojawia się w okienku tekstowym sterujemy za pomocą polecenia:

```
/vis/verbose <level>
```

```
level> = quit,...,errors,..., all
```

13 W OpenGl tworzymy plik postscriptowy z grafiką za pomocą polecenia:

```
/vis/ogl/printEPS
```

W trakcie sesji powstaja pliki G4OpenGl_n.eps, gdzie n=0,1,...

Wizualizacje: pliki tymczasowe HepRep i DAWN

- Aby tworzyć pliki tymczasowe z grafiką korzystjąc z HepRep i DAWN należy utworzyć nową/kolejną obsługę sceny

```
/vis/open HepRepFile
/vis/open DAWNFILE
```

- W OpenGl wyniki poleceń widoczne sa natychmiast w okienku graficznym
- W HepRep i DAWN tymczasowy plik z wizualizacją tworzony jest w momencie wykonania polecenia

```
/run/beamOn <nevents>
```

14 Jesli chcemy utworzyć plik z wizualizacja w innym momencie

```
/vis/viewer/flush
```

W trakcie sesji powstaja pliki G4Datan.heprep oraz g4_nn.prim, gdzie n,nn=0,1,...

- trajektorie zawierają dodatkowe (niewizualizowane przez OpenGl) informacje:
  - początkowe wartości energii i pędu rodzaj cząstki, ...
  - te informacje sa dostepne poprzez kliknięcie na trajektorie w niektórych interfejsach np. HepRep
- Ta informacja może być rozszerzona poprzez polecenie:

```
/vis/scene/add/trajectories rich
```

Materiały

- GEANT4 umozliwia używanie nastepujących materiałów:
  - pierwiastków, związków chemicznych, mieszanin pierwiastków i/lub związków chemicznych;
  - gazów, cieczy i ciał stałych;
  - o różnych gęstościach, temperaturach i ciśnieniach;
- Ogólny schemat definicji materiałów w GEANT4:
  - pierwiastek jest instancją klasy G4Element;
  - związki definiujemy przypisując kilka pierwiastków do instancji klasy G4Material;
  - mieszaniny definiujemy przypisując kilka związków i/lub pierwiastków do instancji klasy G4Material;
  - instancja klasy G4Material może być zdefiniowana za pomocą jednego pierwiastka
  - można tworzyć własne pierwiastki z izotopów (instancji klasy G4Isotope)

Materiały, pierwiastki, izotopy

- klasy G4Isotope i G4Element określają własności jądra atomowego i atomu:
  - należy określić ich: liczbe i mase atomową;
  - mozna dodatkowo okreslić: liczbę masową,energie elektronów na orbitach, przekroje czynne na atom,...
- klasa G4Material opisuje własności makroskopowe materii
  - należy określić gęstość;
  - mozna określić: temperature, ciśnienie, stan skupienia, długość radiacyjną
- materiały muszą być zdefiniowane w klasie wyprowadzonej z klasy G4VUserDetectorConstruction

Materiały: Pojedyńczy pierwiastek

```
G4double a, z, density;
new G4Material(   "Titanium",       // nazwa
                        z=22.,      // liczba masowa
               a=47.90*g/mole,      // masa atomowa
          density=4.540*g/cm3);     // gęstość
```

Materiały: Związki chemiczne

```
G4int ncomp,natoms;
G4string symbol;

G4Material * SiO2 =
new G4Material(  "Quartz",          // nazwa
       density=2.65*g/cm3,          // gęstość
                   ncomp=2);        // ilość komponentów

G4Element * elSi =
new G4Element(  "Silicon",          // nazwa
              symbol="Si",          // symbol chem.
                    z=14.,          // liczba masowa
           a=28.09*g/mole);         // masa atomowa
G4Element * elO = .......

SiO2->AddElement(elSi, natoms=1);
SiO2->AddElement(elO, natoms=2);
```

Materiały: Mieszaniny pierwiastków i/lub materiałów

```
G4Material * Aero =
new G4Material( "Aerogel",          // nazwa
        density=0.2*g/cm3,          // gęstość
                  ncomp=3);         // ilość komp.


G4Element * elC = .....              // węgiel
G4Material * SiO2 = .....            // kwarc
G4Material * H2O = .....             // woda


G4double fracMass;
Aero->AddElement(elC,  fracMass= 0.1*perCent);
Aero->AddMaterial(H2O, fracMass=37.4*perCent);
Aero->AddMaterial(SiO2,fracMass=62.5*perCent);
```

Materiały: Pierwiastek o nienaturalnym składzie izotopowym

```
G4Element * elDH =
new G4Element( "DH",          // nazwa
         symbol="H",          // symbol chem.
            ncomp=2);         // ilość komp.


G4Isotope * isoH1 =
new G4Isotope( "H1",          // nazwa
             iz = 1,          // liczba atomowa
             ia = 1,          // liczba masowa
    a=1.008*g/mole);          // masa atomowa


G4Isotope * isoH2 =
new G4Isotope( "H2",          // nazwa
             iz = 1,          // liczba atomowa
             ia = 2,          // liczba masowa
    a=2.014*g/mole);          // masa atomowa


G4double abund;
elDH->AddIsotope(isoH1, abund= 50.*perCent);
elDH->AddIsotope(isoH2, abund= 50.*perCent);
```

Materiały: Stan skupienia

- Materiał o gęstości $< 10 mg/cm^3$ domyślnie jest gazem;
- często zachodzi konieczność określenia temperatury i ciśnienia gazu.

```
G4Material * CO2 =
new G4Material("CarbonicGas",          // nazwa
          density=27*mg/cm3,           // gęstość
                  ncomp=2,             // ilość komponentów
                kStateGas,             // stan gazowy
            temp=325*kelvin,           // temperatura
            pressure=50.0*atmosphere);// ciśnienie
```

- `kStateSolid` - ciało stałe
- `kStateLiquid` - płyn

Materiały: Baza materiałów NIST

- GEANT4 ma wbudowaną bazę materiałów NIST;
- baza zawiera definicje wszystkich pierwiastków o naturalnym składzie izotopowym;
- oraz szeregu materiałów z nich skonstruowanych; Aby korzystać z bazy danych należy uzyskać wskaźnik do menedżera bazy:

```
G4NistManager * nistman = G4NistManager::Instance();
```

uzyć którejś z metody:

- `FindOrBuildElement("O");`
- `FindOrBuildElement(G4int z = 8);`
- `FindOrBuildMaterial("Nist_name");`
- `ConstructNewMaterial(...)`
  tworzy związki chem. i mieszanki materiałów dostępnych w bazie
- `ConstructNewGasMaterial(...)`
  zmienia temperaturę i ciśnienie gazu zdefiniowanego w bazie

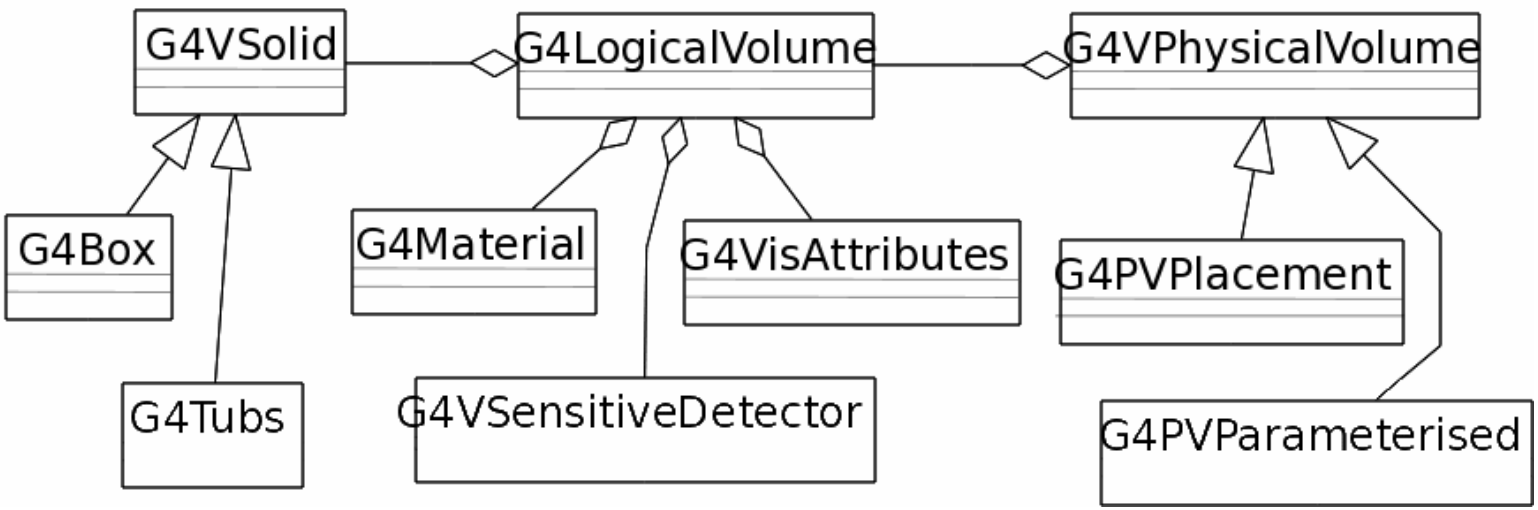
Materiały: Przydatne funkcje

- `/material/nist/printElement`
- `/material/nist/printMaterial`
- `G4cout << wskaźnik_do_materiału;`
- `G4cout << *(G4Material::GetMaterialTable()) << endl;`

Geometria detektora

Trzy poziomy definicji geomterii detektora

- `G4VSolid` - kształt, rozmiar
- `G4LogicalVolume` - materiał, wizualizacja
- `G4VPhysicalVolume` - umiejscowienie



Geometria: Typowy schemat

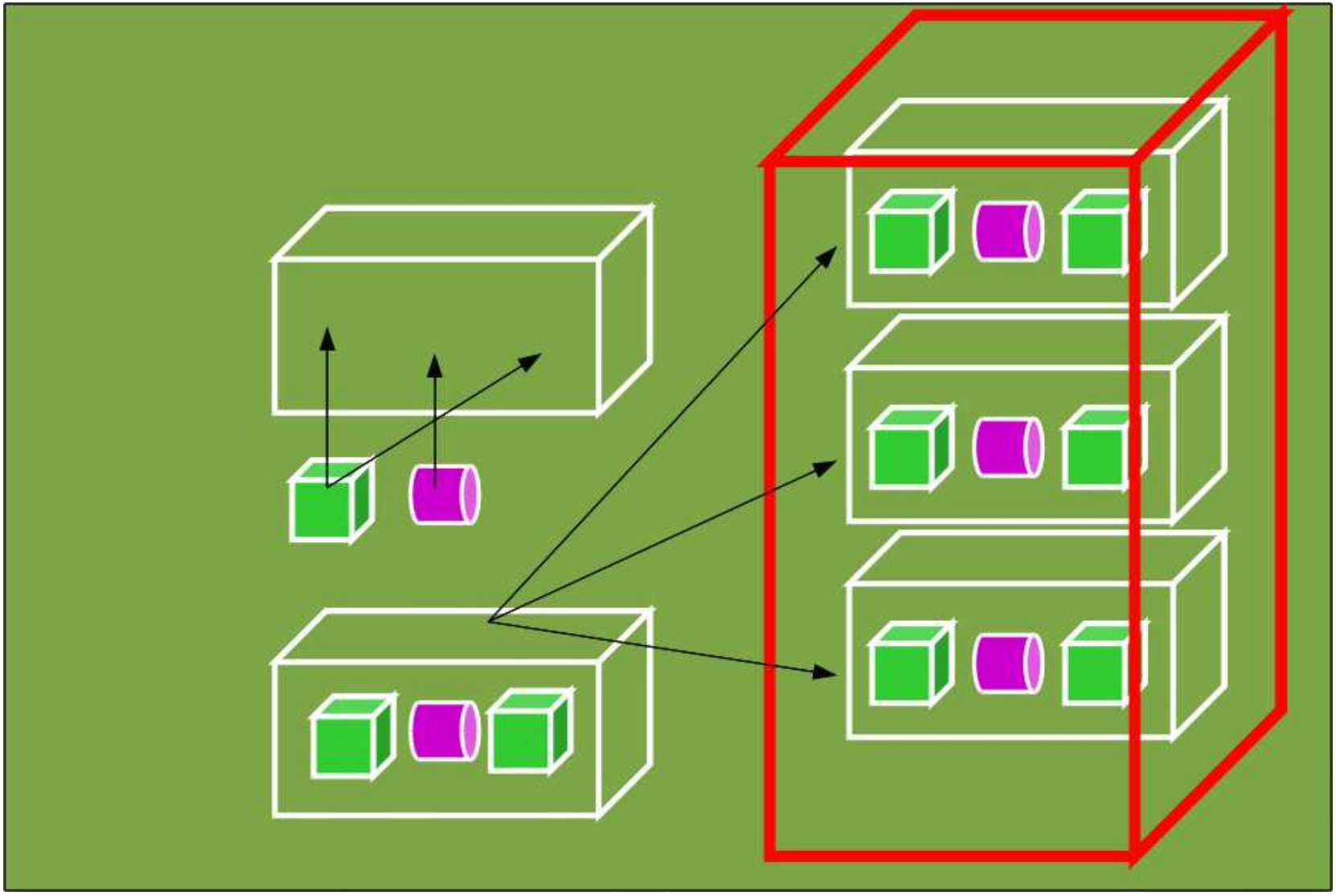
```
// World volume
G4Box* worldSolid =
   new G4Box("World_Solid",          // Name
      2.0*m, 2.0*m, 2.0*m);          // Half lengths


fpWorldLogical =
 new G4LogicalVolume(worldSolid,    // Solid
                             air,   // Material
                "World_Logical");   // Name


fpWorldPhysical =
 new G4PVPlacement(0,               // Rotation matrix pointer
     G4ThreeVector(),               // Translation vector
      fpWorldLogical,               // Logical volume
    "World_Physical",               // Name
                   0,               // Mother volume
               false,               // Unused boolean parameter
                   0);              // Copy number
```

- każdy obszar jest umieszczany wewnątrz swojego obszaru matki;
- pozycja i orientacja obszaru określana jest względem układu współrzędnych matki
- obszar musi być całkowicie zawarty wewnątrz obszaru matki.

Geometria detektora



Geometria detektora

- Obszar logiczny może zawierać wiele obszarów fizycznych;
- Obszar logiczny można umieścić w wielu różnych miejscach(wraz z cała zawartością);
- Obszar world może być tylko jeden i musi zawierać całkowicie pozostałe obszary;
- Obszar world definiuje globalny układ współrzędnych;
- Pozycje czastek, depozytów energii, ..., podawane są w globalnym układzie współrzędnych.