Środowisko programistyczne GEANT4

Leszek Adamczyk

Wydział Fizyki i Informatyki Stosowanej
Akademia Górniczo-Hutnicza

Wykłady w semestrze zimowym 2013/2014

Geometria: G4VUserDetectorConstruction

- Definicje materiałów, geometrii, wizualizacji powinny znajdować się w klasie wyprowadzonej z klasy G4VUserDetectorConstruction;
- Użytkownik musi zaimplementować czysto wirtualna metode Construct()
- Opis klasy G4VUserDetectorConstruction

```
This is the abstract base class of the user's mandatory initialization class
for detector setup. It has only one pure virtual method Construct() which is
invoked by G4RunManager when it's Initialize() method is invoked.
The Construct() method must return the G4VPhysicalVolume pointer which represents
the world volume.
```

- Program główny My.CC

```
runManager->SetUserInitialization(new MyDetectorConstruction);
runManager->Initialize();
```

- Kod możemy umieścić w konstruktorze klasy MyDetectorConstruction lub w metodzie MyDetectorConstruction::Construct()

Geometria: MyDetectorConstruction.hh

```
#ifndef MYDETECTORCONSTRUCTION_HH
#define MYDETECTORCONSTRUCTION_HH

#include "G4VUserDetectorConstruction.hh"
#include "globals.hh"

class G4LogicalVolume;
class G4VPhysicalVolume;

class MyDetectorConstruction : public G4VUserDetectorConstruction {

public:

  // Constructor
  MyDetectorConstruction();

  // Destructor
  virtual ~MyDetectorConstruction();

  // Method
  virtual G4VPhysicalVolume* Construct();

private:

  // Helper methods
  void DefineMaterials();
  void SetupGeometry();

  // World logical and physical volumes
  G4LogicalVolume*   fpWorldLogical;
  G4VPhysicalVolume* fpWorldPhysical;

};
#endif
```

Geometria: MyDetectorConstruction.cc

```
MyDetectorConstruction::MyDetectorConstruction()
  :fpWorldLogical(0),fpWorldPhysical(0)
{.........}


G4VPhysicalVolume* MyDetectorConstruction::Construct()
{
  // Material Definition
  DefineMaterials();

  // Geometry Definition
  SetupGeometry();

  // Return world volume
  return fpWorldPhysical;
}


void MyDetectorConstruction::DefineMaterials()
{......}

void MyDetectorConstruction::SetupGeometry()
{......}
```

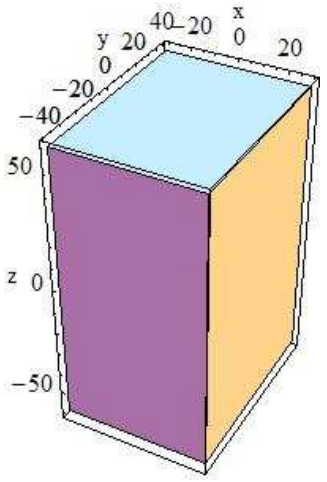
Geometria: Kształt i rozmiar (Bryły)

- Wszystkie bryły dostępne w GEANT4 sa wyprowadzone z klasy G4VSolid.
- Klasa bazowa wymaga implementacji metod potrzebnych do:
  - Obliczenia odległosci brzegu bryły do dowolnego punktu
    DistanceToOut, DistanceToIn
  - Obliczenia wektora normalnego do powierzchni bryły w dowolnym punktcie
    SurfaceNormal
  - Sprawdzenia czy dowolny punkt znajduje się wewnątrz bryły
    Inside
  - Obliczania objętości, pola powierzchni, ...
    GetSurfaceArea, GetCubicVolume, ...
- Użytkownik może definiować własne bryły.

Geometria: Bryły elementarne

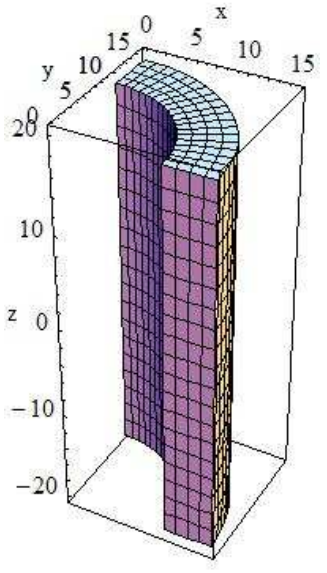
Prostopadłościan

```
G4Box(const G4String& Name,       // name
                G4double  X,      // half length in X
                G4double  Y,      // half length in Y
                G4double  Z)      // half length in Z
```



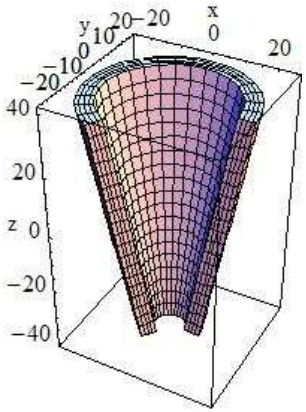
Wycinek powłoki cylindrycznej

```
G4Tubs(const G4String& Name,      // name
                G4double  RMin,   // inner radius
                G4double  RMax,   // outer radius
                G4double  Dz,     // half length in Z
                G4double  SPhi,   // the starting phi angle
                G4double  DPhi)   // the angle of the segment
```

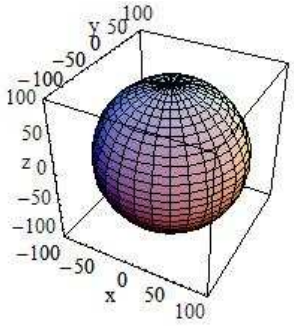


Geometria: Pozostałe bryły elementarne

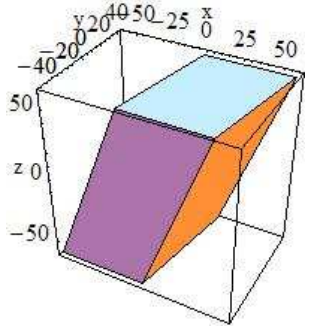
G4Cons
Powłoka stożkowa



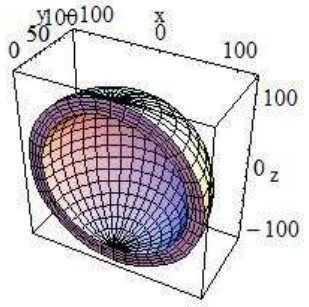
G4Orb
Pełna kula



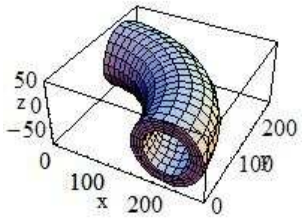
G4Para
Graniastosłup



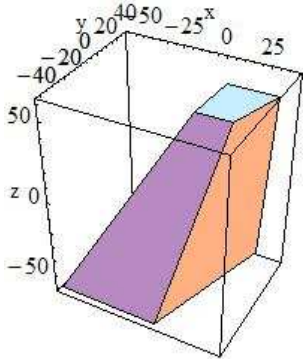
G4Sphere
Powłoka sferyczna



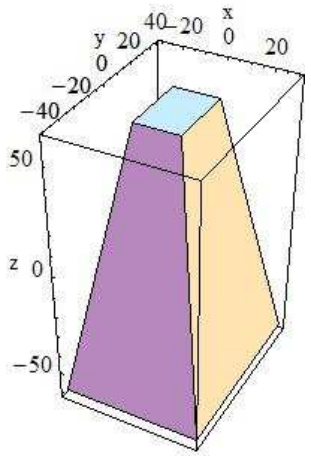
G4Torus
Torus



G4Trap
Ostrosłup

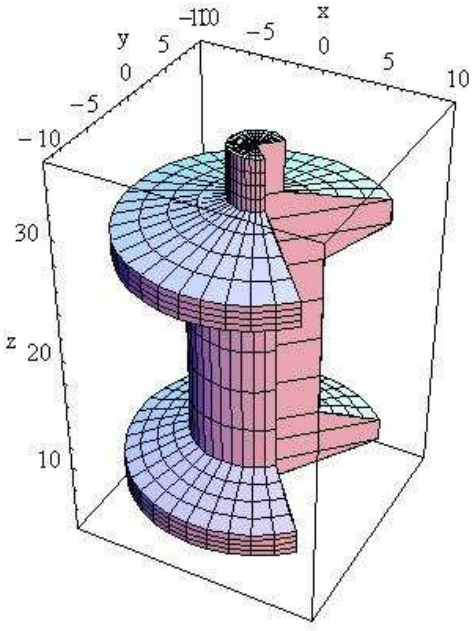


G4Trd
Ostrosłup foremny



Opis wszystkich brył elementarnych w rozdziale 4.1.2 Przewodnika dla twórców aplikacji.

Geometria: Bryły specjalne

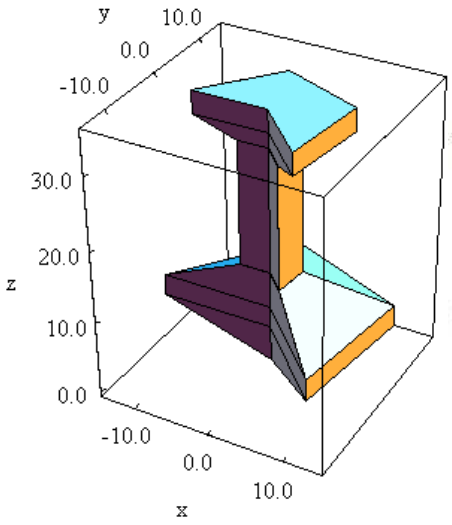


Wycinek wielostożka

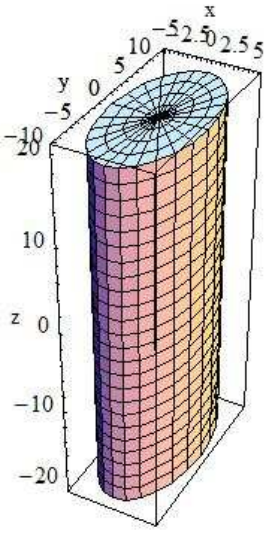
```
G4Polycone(const G4String& Name,               // name
                   G4double   hiStart,         // starting phi angle
                   G4double   hiTotal,         // total phi angle
                   G4int      numRZ,            // number of corners in R-Z space
                   const G4double  r[],         // r coordinate of corners
                   const G4double  z[])         // z coordinate of corners
```

Geometria: Pozostałe bryły specjalne

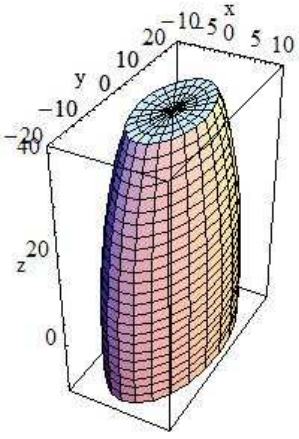
G4Polyhedra



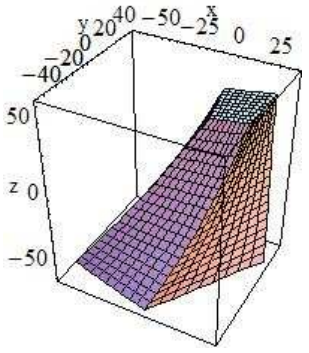
G4EllipticalTube



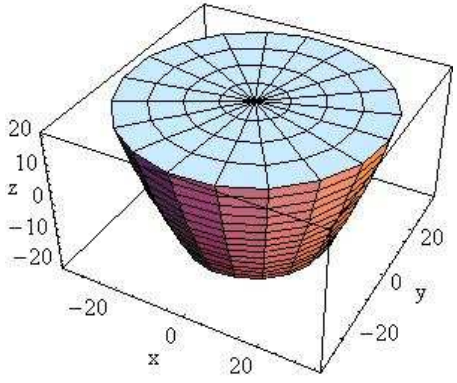
G4Ellipsoid



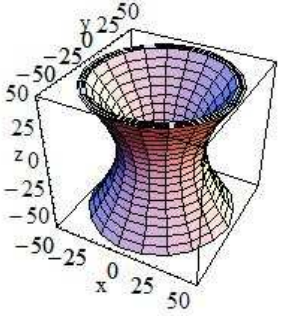
G4TwistedTrap



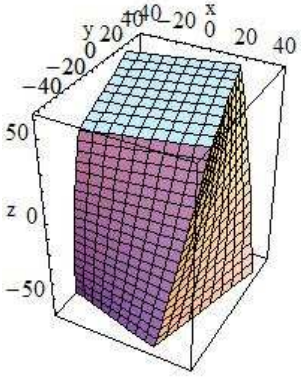
G4Paraboloid



G4Hype



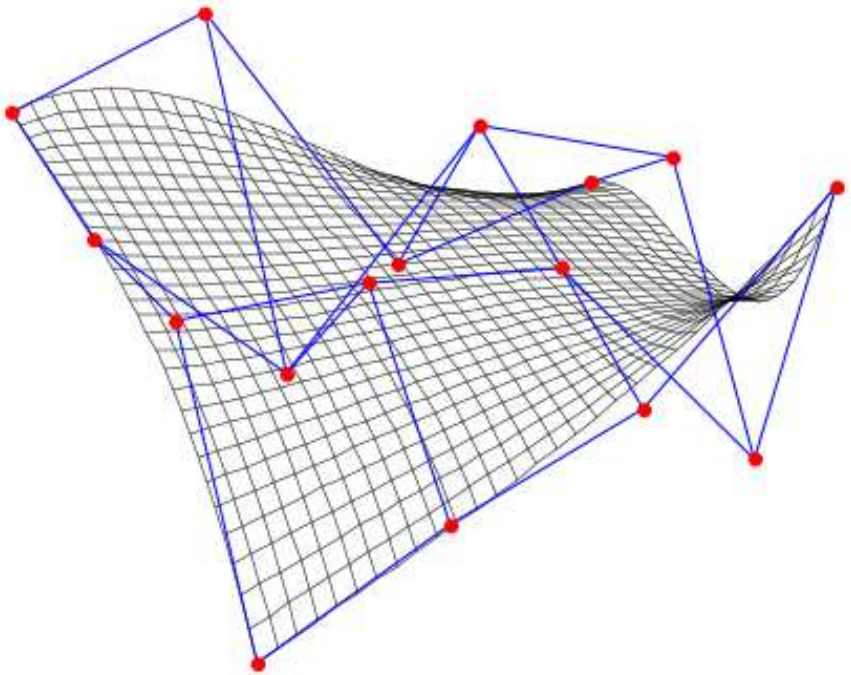
G4TwistedBox



Opis wszystkich brył specjalnych w rozdziale 4.1.2 Przewodnika dla twórców aplikacji.

Geometria: Bryły BREP (Boundary REPresenented solid)

- można definiować bryłę zadając wszystkie płaszczyzny które ja ograniczają;
- płaszczyzny mogą być: płaskie, zakrzywione, płaty Beziera.



Geometria: Operacje logiczne na bryłach

- można definiować bryły poprzez operacje logiczne na istniejących już bryłach;
- G4UnionSolid - suma brył;
- G4SubtractionSolid - różnica brył;
- G4IntersectionSolid - część wspólna brył;

Geometria: Obszary logiczne : G4LogicalVolume

```
G4LogicalVolume( G4VSolid*                  Solid,
                 G4Material*                Material,
                 const G4String&            Name,
                 G4FieldManager*            FieldMgr=0,
                 G4VSensitiveDetector*      SDetector=0,
                 G4UserLimits*              ULimits=0);
```

- obszar logiczny posiada pełna informację o obszarze oprócz jego położenia i orientacji w przestrzeni;
- określa kształt i rozmiar - G4VSolid;
- materiał -G4Material;
- pole magnetyczne - G4FieldManager;
- przypisuje atrybut obszaru czułego detektora -G4VSensitiveDetector;
- warunki produkcji czastek wtórnych - G4UserLimits;
- atrybutów wizualizacji SetVisAttributes
- pozycje i orientacje obszarów córek.

Geometria: Obszary fizyczne : G4PhysicalVolume

Obszary fizyczne dzielimy na:

- jednokrotne: obszar umieszczony jednokrotnie
  jeden fizyczny obszar = jeden obszar rzeczywisty
- wielokrotne: obszar umieszczany wielokrotnie
  jeden fizyczny obszar = wiele obszarów rzeczywistych
  Obszary wielokrotne ze względu na sposób powielania dzielimy na:
  - parametryzowane numerem kopii;
  - powielane wzdłuż jednej osi

Obszar ”matka” może zawierać:

- albo wiele obszarów jednokrotnych;
- albo jeden obszar wielokrotny;

Geometria: Obszary fizyczne : Przegląd

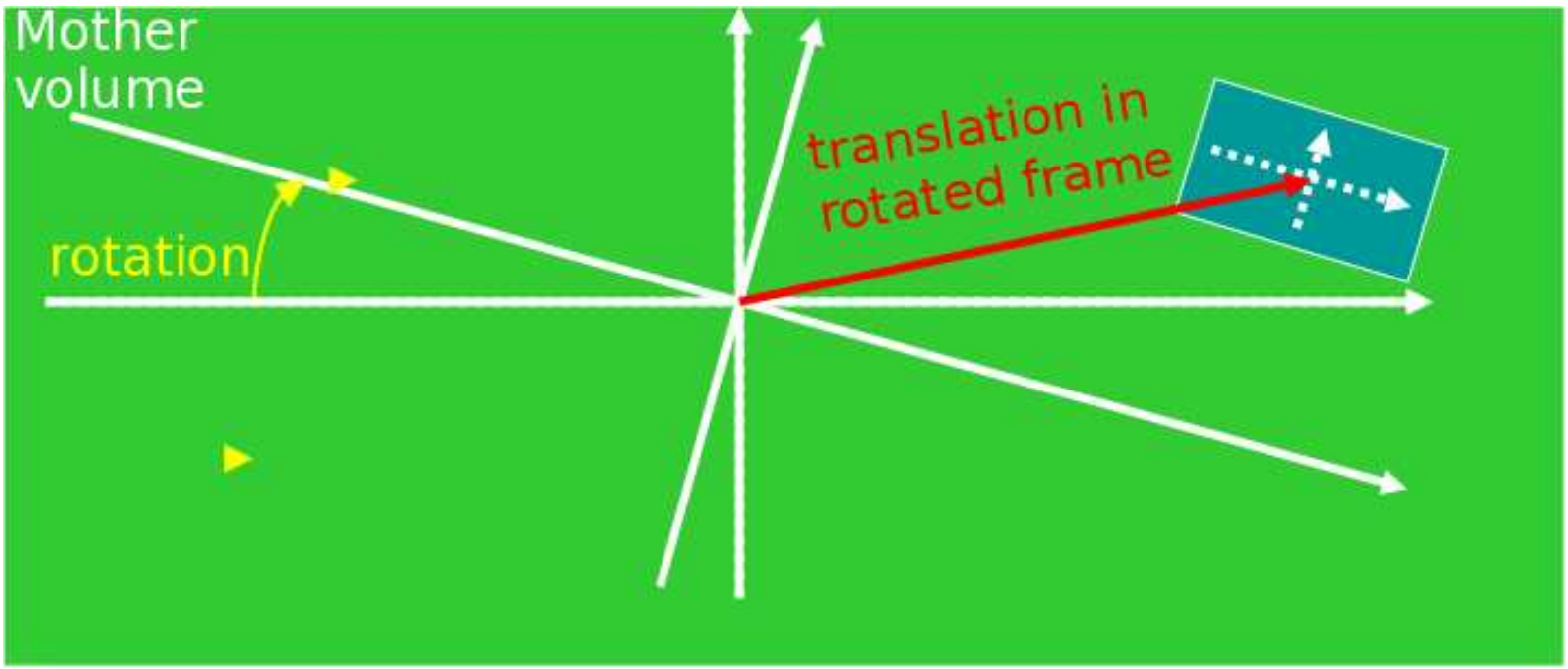
- G4Placement
  reprezentuje obszar jednokrotny;
- G4Parameterised
  reprezentuje obszar wielokrotny parametryzowany numerem kopii
  - kształt, rozmiar, materiał,... mogą być parametryzowane numerem kopii;
  - powielane obszary moga zawierać obszary ”wnuczki” o ile ”wnuczki” mają identyczne rozmiary i kształty;
  - wymagana jest implementacja klasy G4PVParameterisation
  - implementacja klasy G4PVNestedParameterisation umożliwia parametryzacje materiału,... także numerem kopii swojego ”przodka”

Geometria: Obszary fizyczne : Przegląd cd

- G4PVReplica (obszar wielokrotny)
  obszary ”córki” mające ten sam kształt ustawiane są wzdłuż jednej osi i wypełniają cały obszar ”matki” bez żadnych przerw;
- G4PVDivision (obszar wielokrotny)
  tak jak G4PVReplica ale dopuszcza istnienie przerwy między brzegiem obszaru ”matki” a najbardziej zewnętrznymi ”córkami”, brak przerw między ”córkami”
- G4ReflectionFactory (para obszarów)
  umożliwia umieszczenie obszaru i jego odbicia;
- G4AssemblyVolume (wiele obszarów)
  umożliwia umieszczenie jednokrotnych obszarów połączonych w grupę

Geometria: G4PVPlacement I

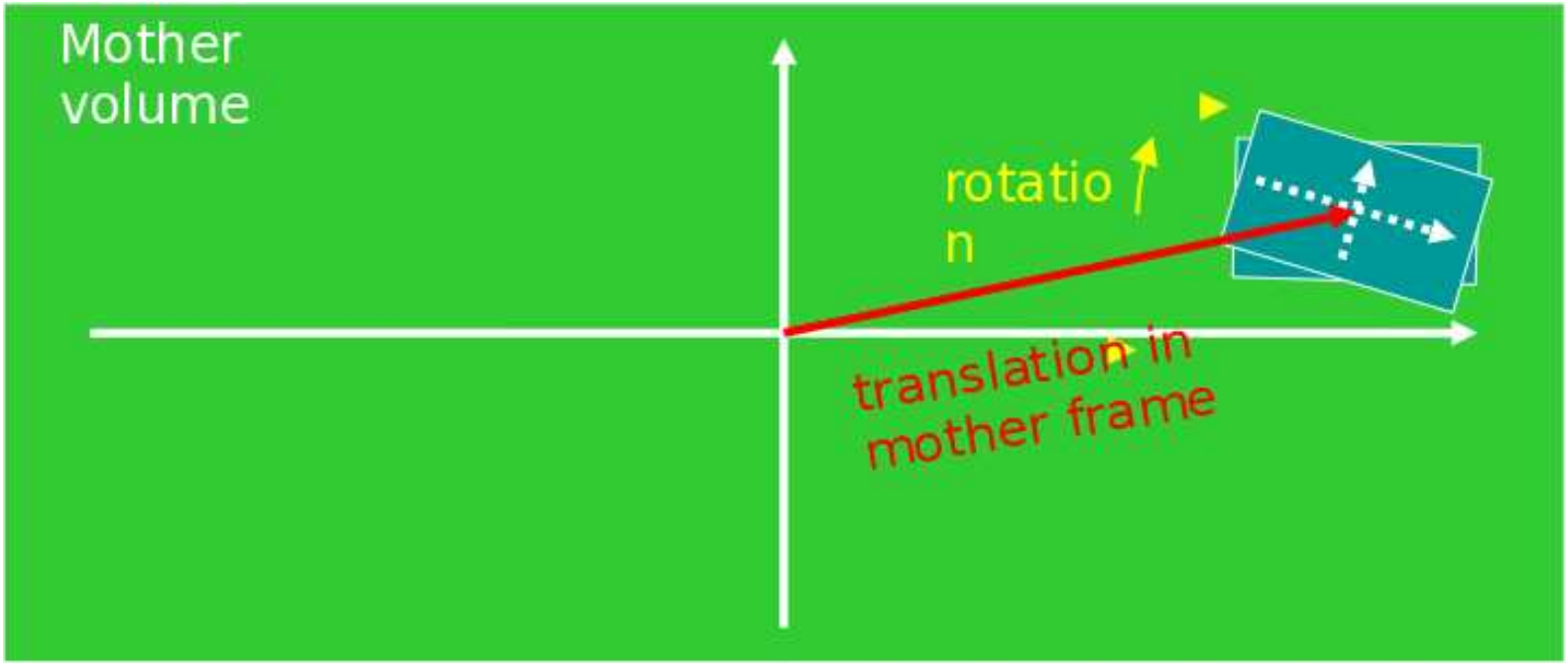
```
G4PVPlacement( G4RotationMatrix*  Rot,    // obrót układu matki
        const G4ThreeVector&      trans,  // pozycja w obróconym układzie
                        G4LogicalVolume*   CurrentLogical,
                const G4String&            Name,
                        G4LogicalVolume*   MotherLogical,
                        G4bool             Many,    // false
                        G4int              CopyNo, // dowolna unikalna liczba całk.
                        G4bool             SurfChk=false )
```



Geometria: G4PVPlacement II

```
Tra = G4Transform3D (
                      G4RotationMatrix &pRot,    // obrót układu  ”córki”
                const G4ThreeVector   &trans);   // pozycja w układzie ”matki”


G4PVPlacement( G4Transform3D  Tra , // transformacja córki
                           ..............................
```

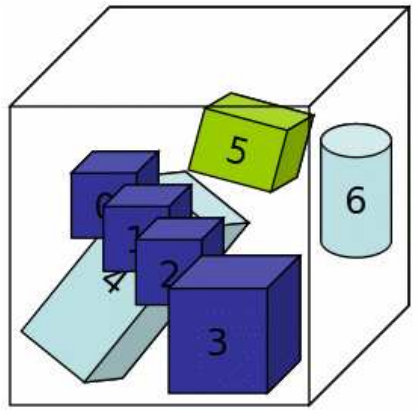


G4PVParameterised

```
G4PVParameterised(
   const G4String&              Name,
   G4LogicalVolume*         DLogical,
   G4LogicalVolume*         MLogical,
   const EAxis                  Axis,
   const G4int                     n,
      G4VPVParameterisation* Param)
```

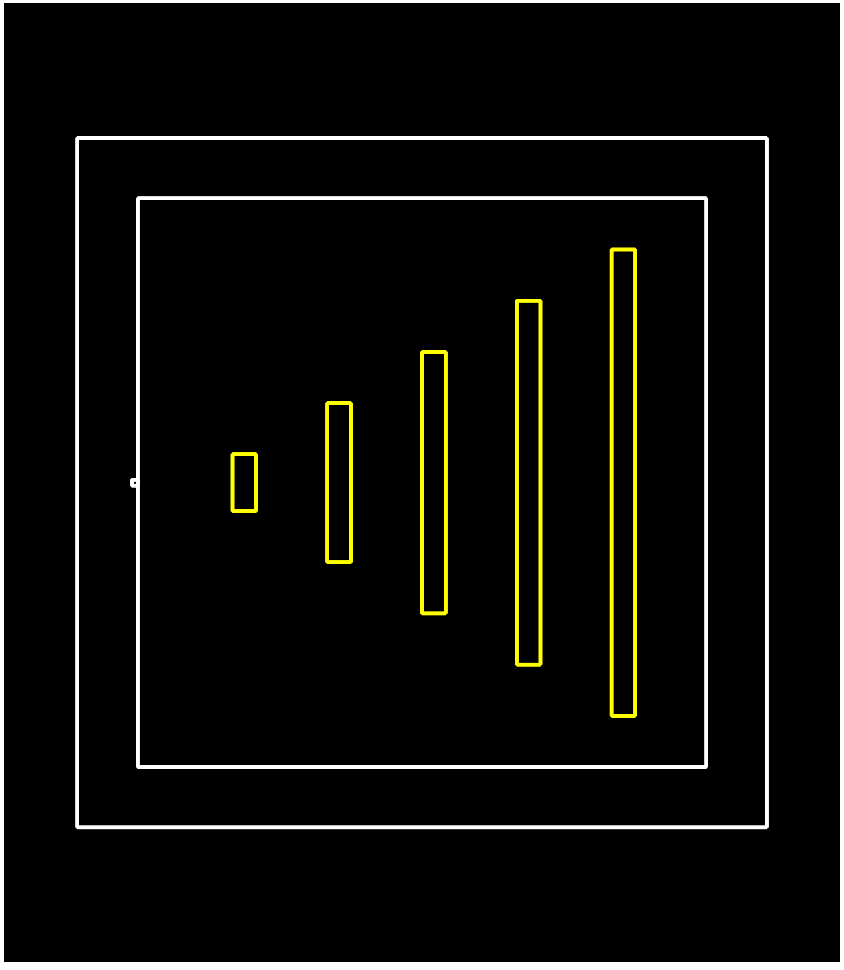
- Umieszcza obszar logiczny DLogical wewnątrz obszaru ”matki” MLogical n razy używając parametryzacji Param;
- Axis sugeruje wzdłuż której osi będą powielane obszary: kXAxis, kYAxis, kZAxis, kUndefined

G4PVParameterised



- Użytkownik musi zaimplementować klasę wyprowadzoną z klasy G4VPVParameterisation definiując następujące wielkości w funkcji numeru kopii:
  - położenie i orientacje ComputeTransformation
  - (opcjonalnie) rozmiar ComputeDimensions
  - (opcjonalnie) kształt ComputeSolid
  - (opcjonalnie) materiał ComputeMaterial
- wszystkie kopie muszą zawierać się w obszarze ”matce” i nie nachodzić na siebie
- nie wszystkie bryły można w ten sposób powielać

Geometria: G4PVParameterised przykład exampleN02



Układ pięciu komór śladowych o rosnących rozmiarach

G4PVParameterised: ExN02DetectorConstruction.cc

```
solidChamber = new G4Box(
    "chamber", 100*cm, 100*cm, 10*cm);
```

```
logicChamber = new G4LogicalVolume(
    solidChamber,ChamberMater,"Chamber",0,0,0);
```

```
chamberParam = new ExN02ChamberParameterisation(
    NbOfChambers,           // NoChambers
    firstPosition,          // Z of center of first
    ChamberSpacing,         // Z spacing of centers
    ChamberWidth,           // Width Chamber
    firstLength,            // lengthInitial
     lastLength);           // lengthFinal
```

```
physiChamber = new G4PVParameterised(
    "Chamber",              // their name
    logicChamber,           // their logical volume
    LogicTracker,           // Mother logical volume
    kZAxis,                 // Are placed along this axis
    NbOfChambers,           // Number of chambers
    chamberParam);          // The parametrisation
```

G4PVParameterised: ExN02ChamberParameterisation.hh

```
.....................................
class ExN02ChamberParameterisation : public
                            G4VPVParameterisation{
```

```
public:
  ExN02ChamberParameterisation(
          G4int    NoChambers,
          G4double startZ,
          G4double spacing,
          G4double widthChamber,
          G4double lengthInitial,
          G4double lengthFinal );
```

```
  void ComputeTransformation (
          const G4int copyNo,
          G4VPhysicalVolume* physVol) const;
```

```
  void ComputeDimensions (
           G4Box & trackerLayer,
           const G4int copyNo,
           const G4VPhysicalVolume* physVol) const;
.....................................
```

G4PVParameterised: ExN02ChamberParameterisation.cc

```
ExN02ChamberParameterisation::
  ExN02ChamberParameterisation(
   G4int    NoChambers,
   G4double startZ,
   G4double spacingZ,
   G4double widthChamber,
   G4double lengthInitial,
   G4double lengthFinal )
{
   fNoChambers =  NoChambers;
   fStartZ     =  startZ;
   fHalfWidth  =  widthChamber*0.5;
   fSpacing    =  spacingZ;
   fHalfLengthFirst = 0.5 * lengthInitial;
.....................................................
}
```

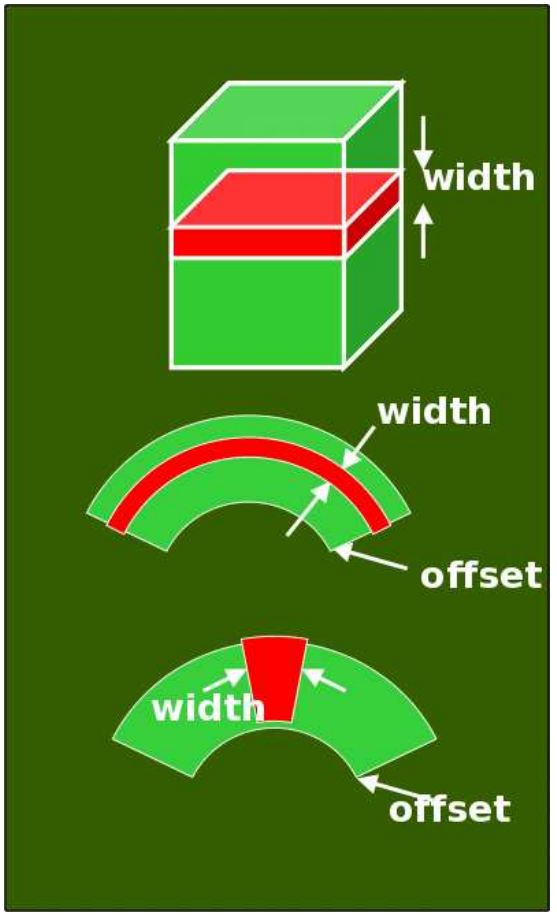
G4PVParameterised: ExN02ChamberParameterisation.cc

```
void ExN02ChamberParameterisation::ComputeTransformation(
  const G4int copyNo,
  G4VPhysicalVolume* physVol) const
{
  G4double      Zposition= fStartZ + (copyNo+1) * fSpacing;
  G4ThreeVector origin(0,0,Zposition);
  physVol->SetTranslation(origin);
  physVol->SetRotation(0);
}
```

```
 void ExN02ChamberParameterisation::ComputeDimensions(
   G4Box& trackerChamber,
   const G4int copyNo,
   const G4VPhysicalVolume*) const
{
  G4double  halfLength= fHalfLengthFirst+copyNo*fHalfLengthIncr;
  trackerChamber.SetXHalfLength(halfLength);
  trackerChamber.SetYHalfLength(halfLength);
  trackerChamber.SetZHalfLength(fHalfWidth);
}
```

G4PVReplica

- Obszar ”matki” musi być całkowicie wypełniony kopiami o takich samych grubościach i kształtach.
- Powielanie może odbywać się wzdłuż:
  - osi (X,Y,Z), plastry są prostopadłe do osi a lokalny układ wsp. związany jest ze środkiem plastra;
  - osi radialnej (R), powielane wycinki stożka/cylindra są współosiowe i nieobrócone a lokalne układy wsp. są takie same jak obszaru ”matki”
  - osi kąta azymutalnego (Phi), powielane wycinki stożka/cylindra są współosiowe i obracane o stały kąt a lokalny układ wsp. jest obrócony względem układu ”matki” w taki sposób, że oś X dzieli na pół każdy klin;



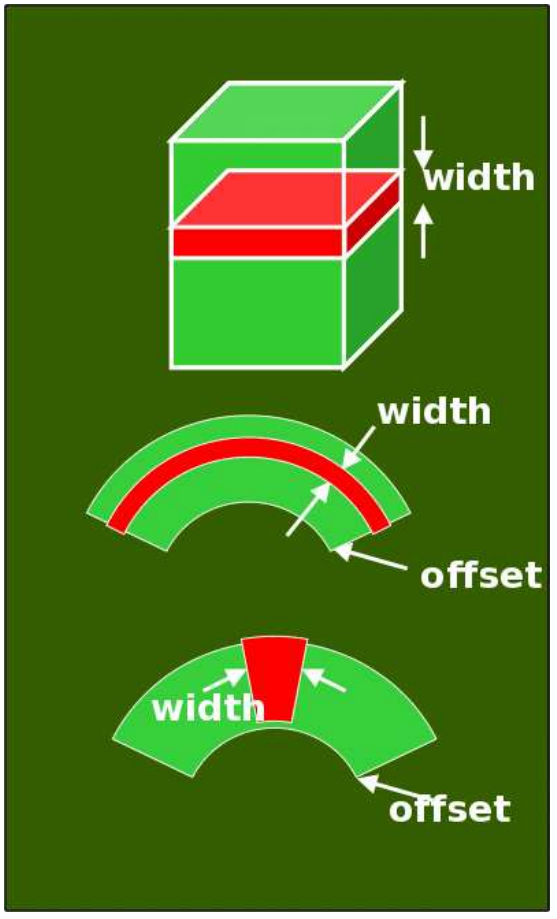
G4PVReplica

```
G4PVReplica( const G4String&     Name,
             G4LogicalVolume*    CurrentLogical,
             G4LogicalVolume*    MotherLogical,
             const EAxis         Axis,
             const G4int         nReplicas,
             const G4double      width,
             const G4double      offset=0 )
}
```

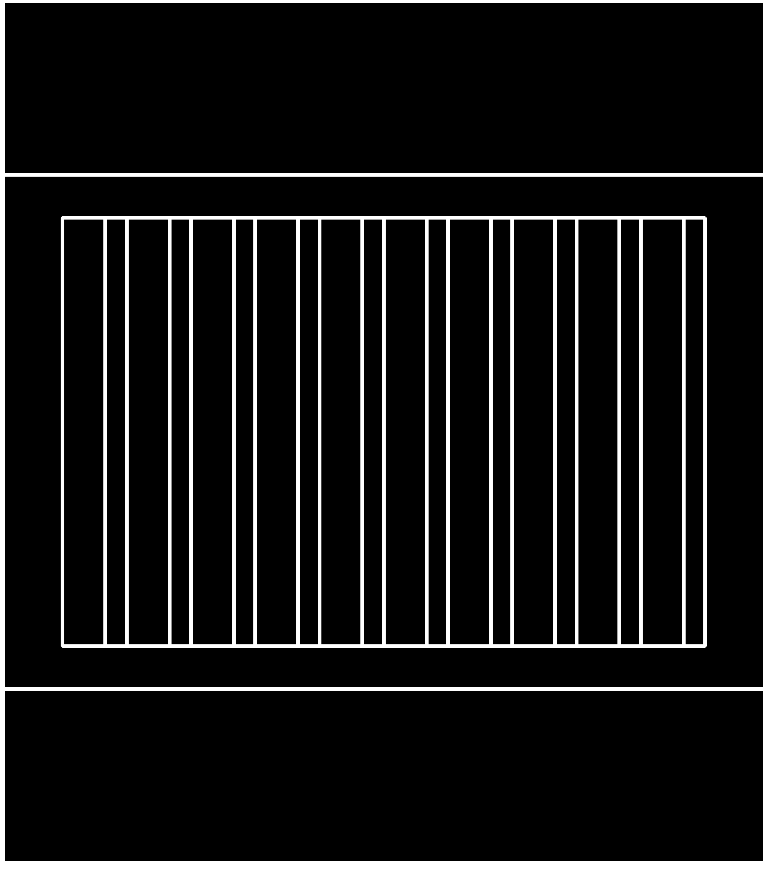
- Obszar ”matka” może sam być klasy G4PVReplica;
- Obszar klasy G4Placement może znajdować się w powielanym obszarze o ile powielanie nie następuje wzdłuż osi radialnej;
- Obszar klasy G4Parameterised nie może znajdować się w powielanym obszarze

G4PVReplica

- Osie kartezjańskie pAxis=kXaxis (YZ) offset musi mieć wartość 0
- Oś radialna pAxis=kRaxis offset musi być równy wewnętrznemu promieniowi obszaru ”matki”
- Oś kąta azymutalnego pAxis=kPhi offset musi być równy początkowemu kątowi obszaru ”matki”



Geometria: G4PVReplica przykład exampleN03



Kalorymetr zbudowany z 10 kopii warstwy absorbera oraz warstwy czynnej

G4PVReplica: ExN02DetectorConstruction.cc

```
solidLayer = new G4Box(
       "Layer"                                          //its name
       LayerThickness/2,CalorSizeYZ/2,CalorSizeYZ/2); //size
```

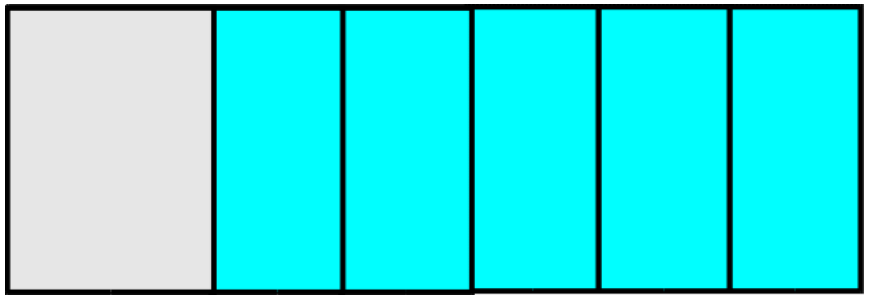
```
logicLayer = new G4LogicalVolume(
       solidLayer,                      // its solid
       defaultMaterial,                 // its material
              "Layer");                 // its name
```

```
physiLayer = new G4PVReplica(
        "Layer",                        // its name
        logicLayer,                     // its logical volume
        logicCalor,                     // its mother
        kXAxis,                         // axis of replication
        NbOfLayers,                     // number of replica
        LayerThickness);                // witdth of replica
```

```
physiAbsorber = new G4PVPlacement(0,      // no rotation
   G4ThreeVector(-GapThickness/2,0.,0.), // its position
         logicAbsorber,                  // its logical volume
   AbsorberMaterial->GetName(),          // its name
         logicLayer,                     // its mother
         false,                          // no boulean operat
            0);                          // copy number
```

G4PVDivision

- G4PVDivision=G4PVReplica+G4PVParameterised;
- Umożliwia podział obszaru matki na szereg identycznych kopii wzdłuż jakiejś z osi;
- G4PVParameterisation jest automatycznie generowana na podstawie parametrów konstruktora klasy G4PVDivision.
- Dopuszcza istnienie przerw między skrajnymi córkami a granicami obszaru matki.

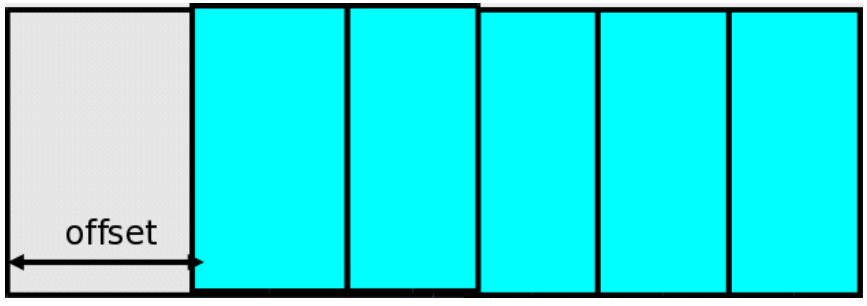


G4PVDivision

```
G4PVDivision( const G4String& Name,
  G4LogicalVolume* CurrentLogical,
  G4LogicalVolume*  MotherLogical,
                  const EAxis Axis,

            const G4int nDivisions,
            const G4double offset )
```

Szerokość kopii obliczona automatycznie tak aby wypełnić obszar ”matki” do końca.

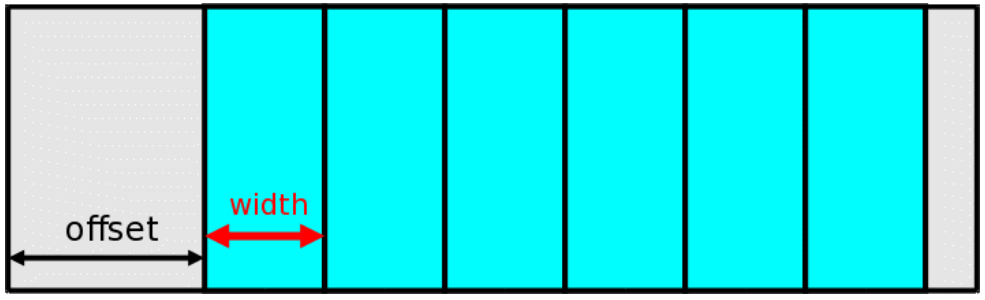


G4PVDivision

```
G4PVDivision( const G4String& Name,
  G4LogicalVolume* CurrentLogical,
  G4LogicalVolume*  MotherLogical,
                  const EAxis Axis,

              const G4double width,
              const G4double offset )
```

Ilość kopii obliczona automatycznie tak aby wypełnić obszar ”matki” tak bardzo jak się da.

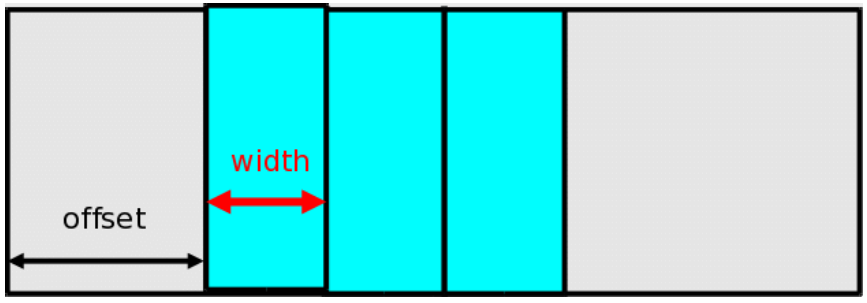


G4PVDivision

```
G4PVDivision( const G4String& Name,
  G4LogicalVolume* CurrentLogical,
  G4LogicalVolume*  MotherLogical,
                  const EAxis Axis,

              const G4int nDivisions,
              const G4double width,
              const G4double offset )
```

Ustalona zarówno ilość jak i szerokość kopii



G4PVDivision

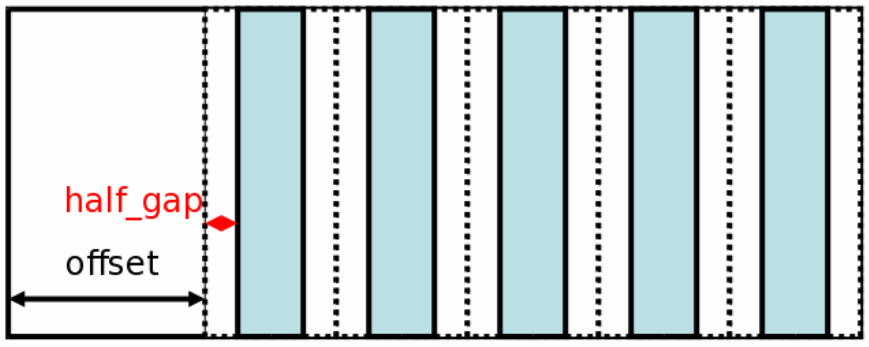
Obecnie można podzielić następujące bryły:

G4Box	Axis = kXAxis (Y,Z)
G4Tubs	Axiz = kRho (Phi,Z)
G4Cons	Axiz = kRho (Phi,Z)
G4Trd	Axis = kXAxis (Y,Z)
G4Para	Axis = kXAxis (Y,Z)
Polycone	Axiz = kRho (Phi,Z)
Polyhedra	Axiz = kRho (Phi,Z)

Ostatnie dwie bryły mozna dzielić wzdłuż osi Z tylko na taka ilość warstw z jakich same się składają.

G4PVReplicatedSlices

- Rozszerzenie klasy G4PVDivision;
- Umożliwia pozostawienie przerw między kolejnymi córkami;



G4PVNestedParameterisation

- NestedParameterisation umożliwia uzależnienie materiału córki nie tylko od swojego numeru kopii ale również od numeru kopii przodka bądź przodków w przypadku gdy obszary przodków są obszarami powielanymi: (Parameterised, Replica, Division, ReplicatedSlices).
- Wymaga implementacji trzech dodatkowych funkcji:
  - ComputeMaterials() - zwraca wskaźnik do materiału w zależności od numeru kopii własnej oraz przodków
  - GetNumberOfMaterials() - zwraca całkowitą liczbę materiałów jakie mogą być zwrócone przez ComputeMaterials()
  - GetMaterial(G4int i) – zwraca wskaźnik do i-tego materiału