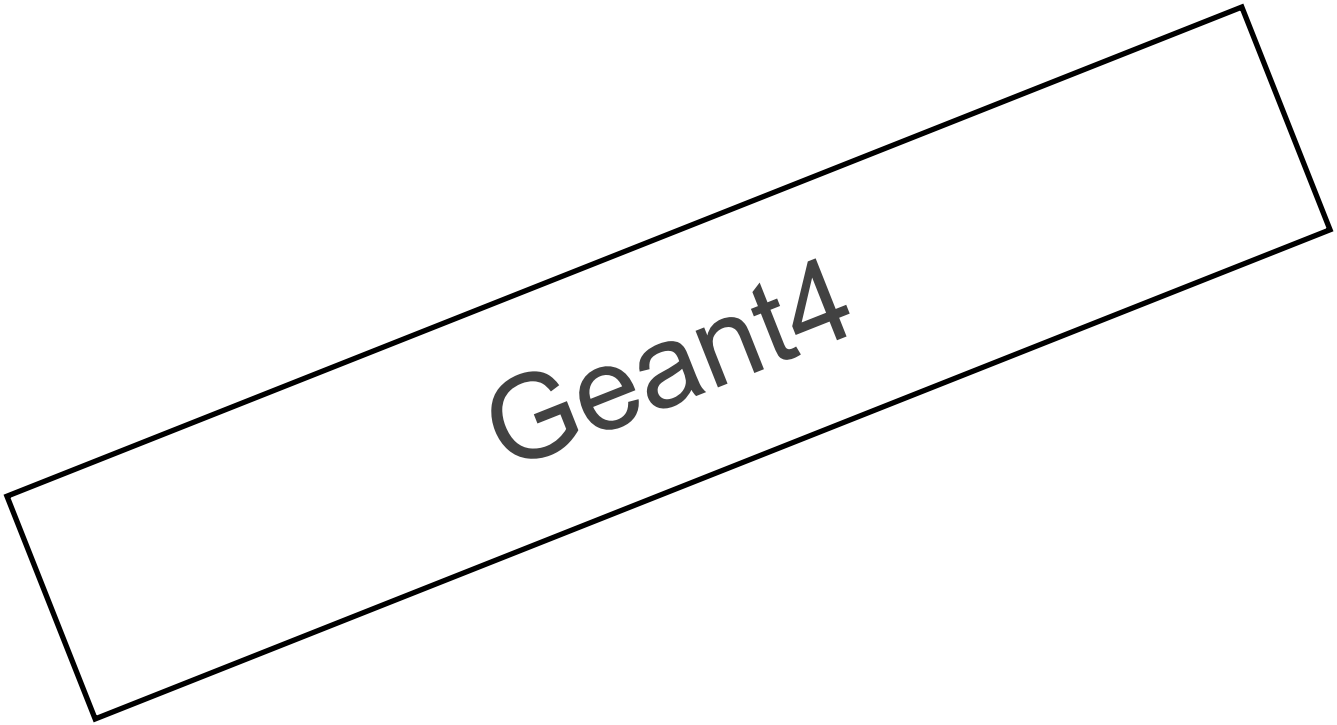
Geant4 Interface In Shoe


Geant4

Introduction (i)

- Documentation: https://geant4.web.cern.ch/docs/
- Download last version: https://geant4.web.cern.ch/download/11.3.0.html
- CMake options (in build folder):
  - cmake ../geant4-v11.3.0 -DGEANT4_USE_OPENGL_X11=true -DGEANT4_USE_QT*=true -DGEANT4_BUILD_MULTITHREADED=OFF** -DCMAKE_BUILD_TYPE=Debug -DGEANT4_INSTALL_DATA***=ON

* need to install Qt5 package

** to be compliant with Shoe

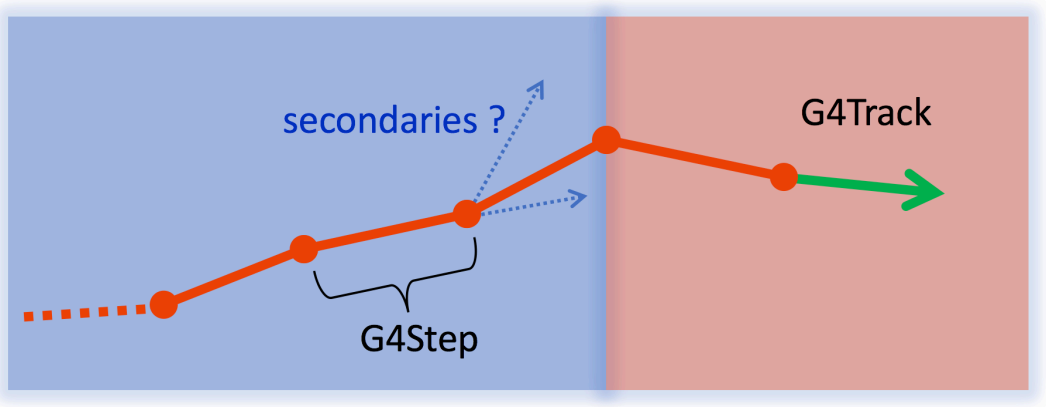
*** will download database set in build/data folder

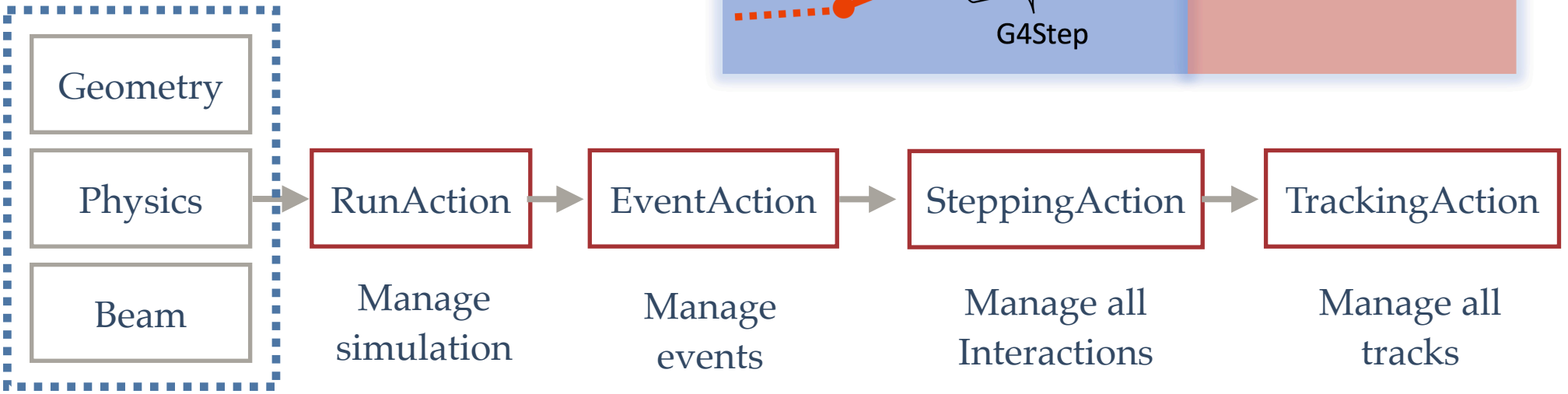
Introduction (ii)

- Units:
  - Default wise:
    - millimeter (mm), nanosecond (ns)
    - Mega electron Volt (MeV)
    - degree Kelvin (kelvin)
    - the amount of substance (mole)
    - radian (radian)
    - steradian (steradian)
  - Conversion factors (namespace CLHEP)
    - micrometer, picosecond, …
    - joule, …
    - degree,
    - …

Introduction (iii)

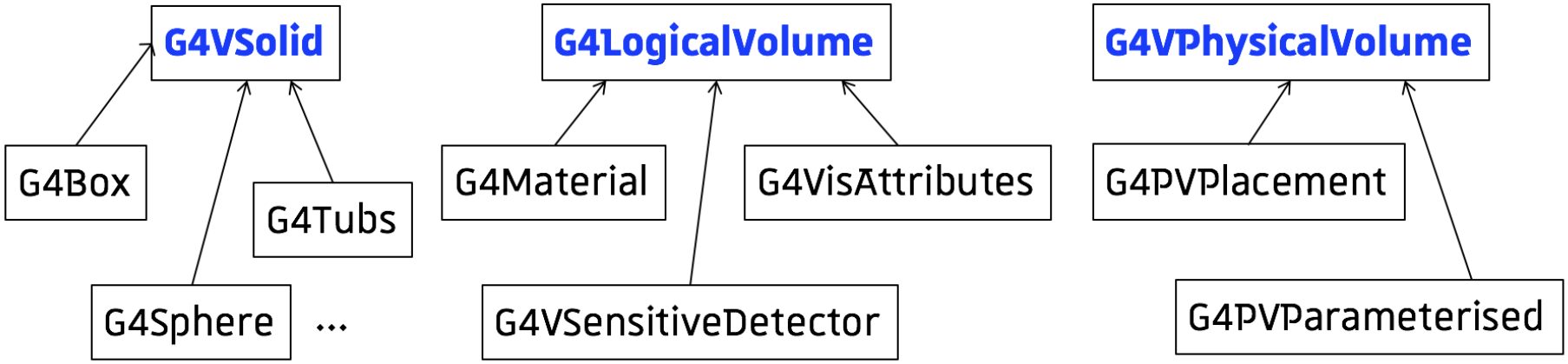
- Point of entry: RunManager
  - Geant4 works through a set of Actions
  - Most of what the user can customise need to be registered through the G4RunManager





Geometry in G4 (i)

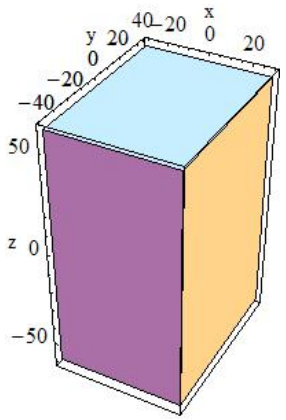
- Volumes:
  - Geometry in Geant4 is defined through three levels:
    - The solid volume, which defines the shape and dimensions of the volume.
    - The logical volume, which defines the material, whether the volume is sensitive or not, etc.
    - The physical volume, which positions the volume.



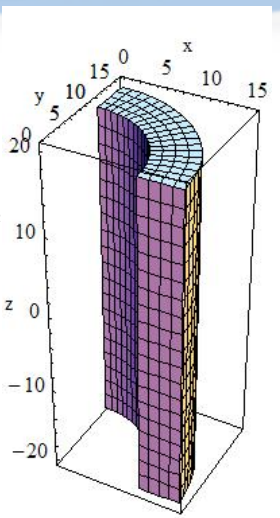
Geometry in G4 (ii)

- Shapes:
  - Examples:

```
G4Box( const G4String& pName,
       G4double pX,
       G4double pY,
       G4double pZ );
```



```
G4Tubs( const G4String& pName
        G4double pRMin,
        G4double pRMax,
        G4double pDz,
        G4double pSPhi,
        G4double pDPhi );
```



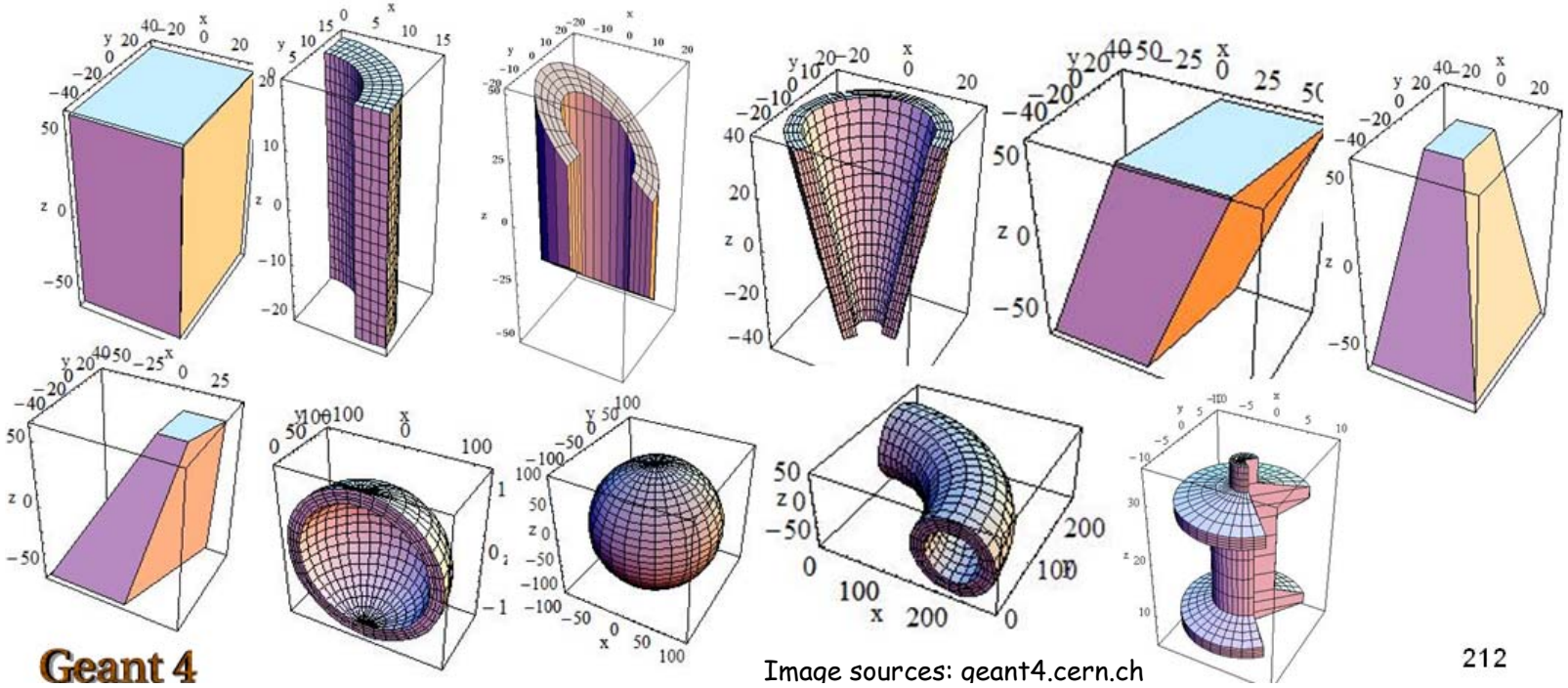


Image sources: geant4.cern.ch

212

Geometry in G4 (iii)

- Implementation:
  - runManager->SetUserInitialization(new DetectorConstruction);

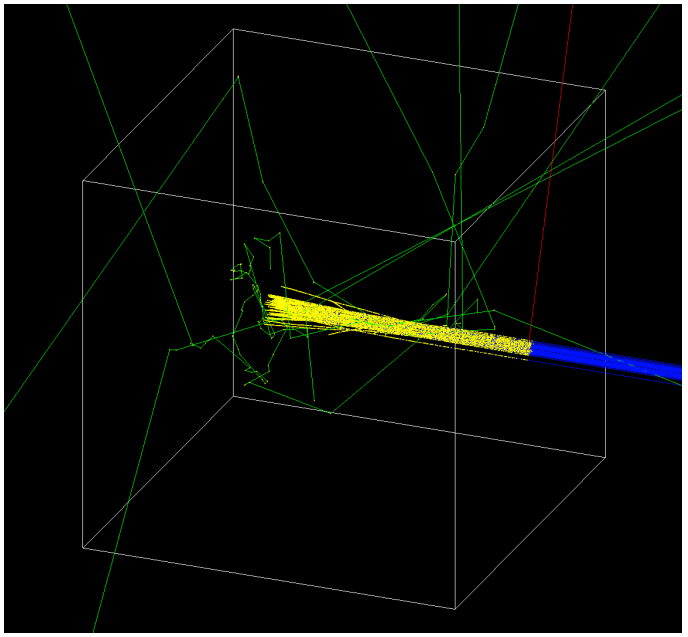
```
class DetectorConstruction : public G4VUserDetectorConstruction
{
public:

   DetectorConstruction();
   ~DetectorConstruction();

public:
   . . .
   G4VPhysicalVolume* Construct();
. . .
};
```

must inherits from

Need to be implemented

Beam in G4 (i)

- G4ParticleGun:
  - Shouting particles with a given:
    - Type
    - Energy, momentum
    - Polarisation
    - Atomic charge
    - Number of impinging particles
  - Shape and position of beam



Beam in G4 (ii)

- Implementation:
  - runManager->SetUserAction(**new** PrimaryGeneratorAction);

```
class PrimaryGeneratorAction : public G4VUserPrimaryGeneratorAction
{
  public:
    PrimaryGeneratorAction();
   ~PrimaryGeneratorAction();
 private:
   G4ParticleGun*            fParticleGun;
   G4GeneralParticleSource* fGeneralParticleSource;

  public:
   void GeneratePrimaries(G4Event*);



. . .
};
```

must inherits from

Need to be implemented

Physics in G4 (i)

- PhysicsList:
  - Several families of process are present:
    - **Electromagnetic physics**
    - **Ion nuclear interactions**
    - Transport
    - Extra physics processes for gamma and leptons
    - Decay
    - Hadron elastic
    - Hadron inelastic
    - Stopping particles capture processes
    - Photo-leptonic
    - Optic
  - Electromagnetic Model (i):
    - EM processes applied for:
      - Electrons and positrons
      - X-Ray & γ-Ray
      - Muons
      - Charged hadrons
      - Ions

- processes
  - Multi-scattering
  - Bremsstrahlung
  - Ionisation
  - Annihilation
  - Photoelectric effect
  - Compton scattering
  - Rayleigh scattering
  - Gamma conversion
  - Pair creation
  - Synchrotron radiation
  - Cherenkov radiation
  - Refraction
  - Reflection
  - Absorption
  - Scintillation
  - Fluorescence
  - Auger emission

Energy Loss

Physics in G4 (ii)

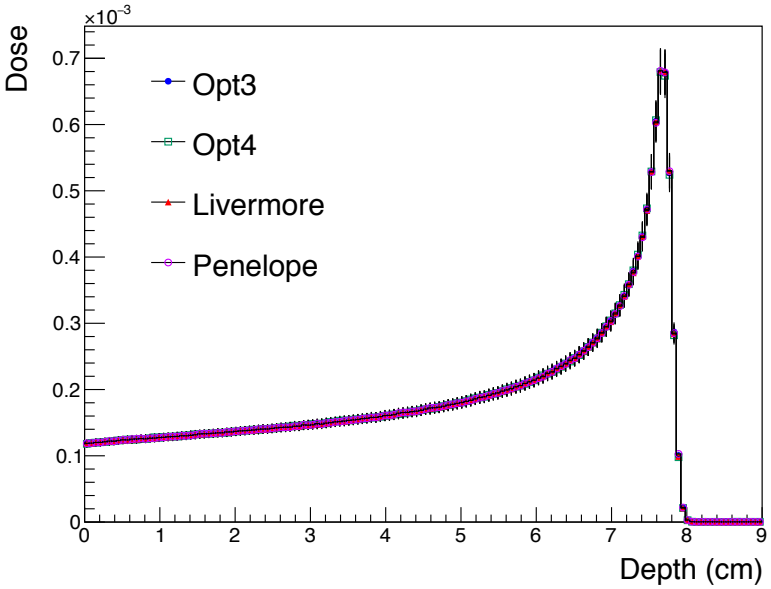
- Electromagnetic Model (ii):
  - Excerpt of available models

Constructor	Comments	Energies
G4EmStandardPhysics	Default	< 100 TeV
G4EmStandardPhysics_option1	Fast but less precise	< 100 TeV
G4EmStandardPhysics_option2	Experimental	< 100 TeV
G4EmStandardPhysics_option3	Medical and space applications	< 100 TeV
G4EmStandardPhysics_option4	Most precise	< 100 TeV
G4EmLivermore	Opt. 3 + Livermore (γ, e-)	250 eV - 100 GeV
G4EmPenelope	Opt. 3 + Penelope (γ, e-, e+)	100 eV - 1 GeV
G4EmLivermorePolarized	Opt. 3 + Livermore + polarisation (γ)	250 eV - 100 GeV
G4EmDNA	Opt. 3 + DNA	7 eV - 100 MeV
G4EmLowEPPhysics	Livermore + improved Compton	250 eV - 100 GeV

- More Electromagnetic physics constructors

Physics in G4 (iii)

- Electromagnetic Model (iii):
  - Examples:
    - Energy deposition of protons in water @ 100 MeV



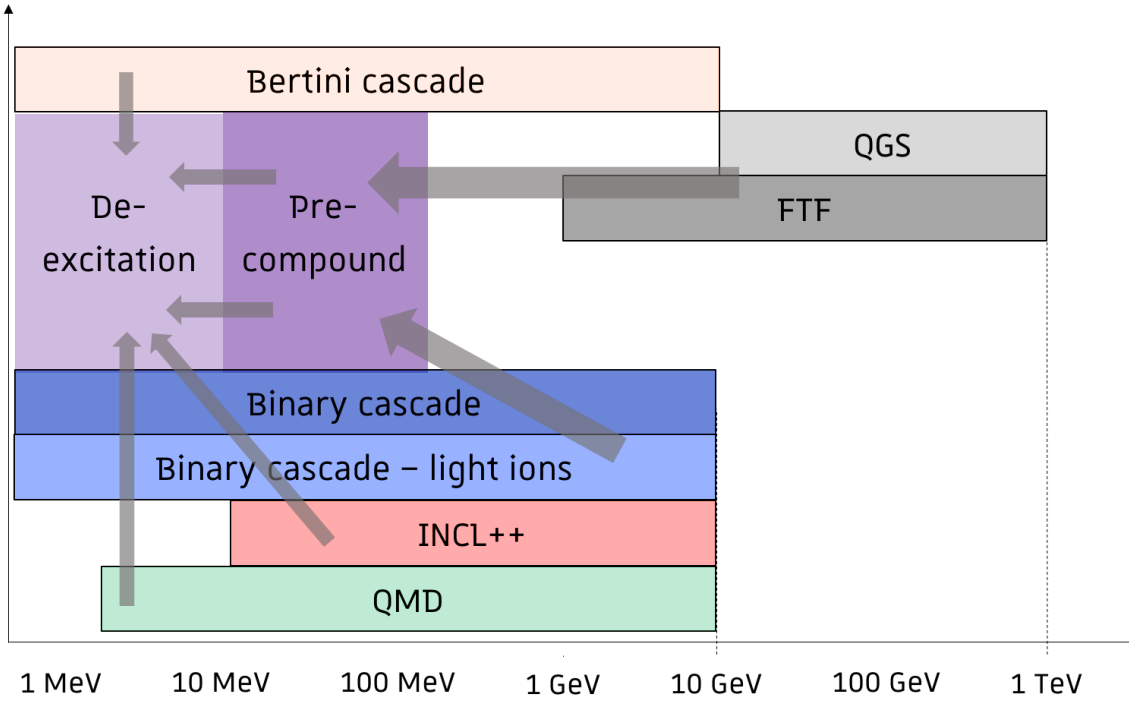
    - Comparison stopping power for protons tabled (PSTAR) and simulated

Physics in G4 (iv)

- Hadronic and ions Models (i):
  - Hadronic model manage (in)-elastic processes hadron with the atomic nucleus (up to 100 TeV)
  - Models are more accurate depending on the energy range
  - Cross section are computed or loaded independently (need to load data files)

  - Several ready-to-use physics lists are provided by the Geant4 developers

  - Different models can be used on a same energy range



Physics in G4 (v)

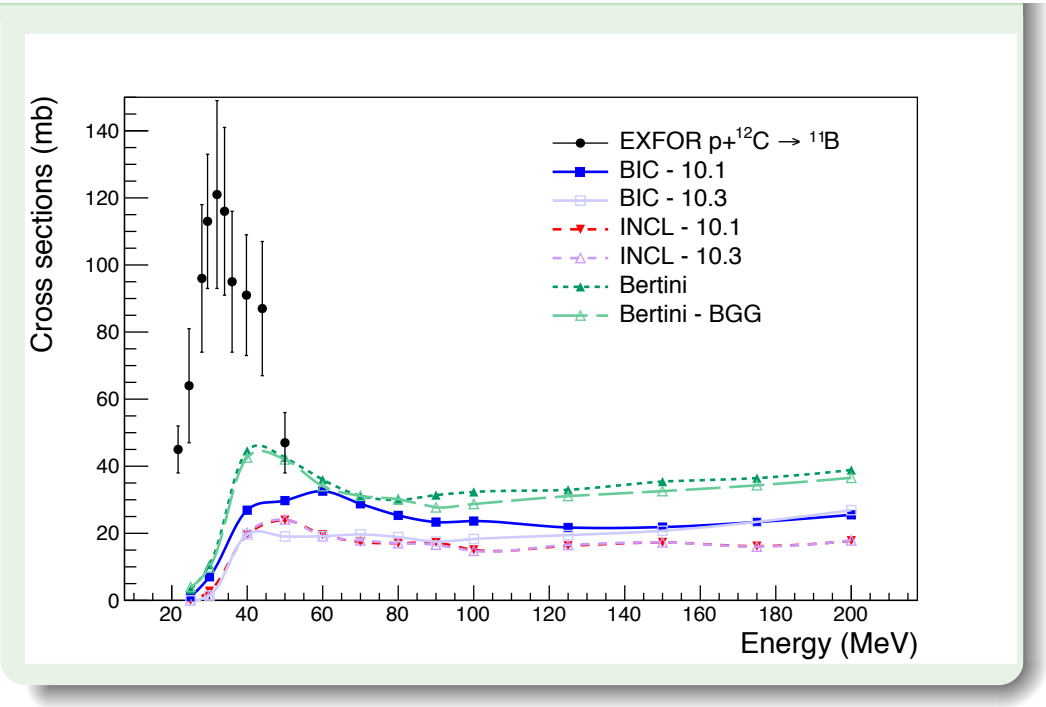
- Hadronic and Ion Physics Lists/Models (ii):
  - Excerpt of physics lists available
    - FTFP_BERT,
    - FTFP_BERT_TRV,
    - FTFP_BERT_ATL,
    - FTFP_BERT_HP,
    - FTFQGSP_BERT,
    - FTFP_INCLXX,
    - FTFP_INCLXX_HP,
    - FTF_BIC,
    - LBE,
    - QBBC,
    - QGSP_BERT,
    - Shielding,
    - ShieldingLEND,
    - ShieldingLLXX_HPT,
    - Shielding_HPT,
    - ShieldingLIQMD_HPT,
    - ShieldingM_HPT,
    - QGSP_BERT_HP,
    - QGSP_BIC,
    - QGSP_BIC_HP,
    - QGSP_BIC_AllHP,
    - QGSP_FTFP_BERT,
    - QGSP_INCLXX,
    - QGSP_INCLXX_HP,
    - QGS_BIC,
    - IQMD,
    - ShieldingM,
    - NuBeam,
    - Shielding_HP,
    - ShieldingLIQMD_HP,
    - ShieldingM_HP,
    - FTFP_BERT_HPT,
    - FTFP_INCLXX_HPT,
    - QGSP_BERT_HPT,
    - QGSP_BIC_HPT,
    - QGSP_BIC_AllHPT,
    - QGSP_INC
  - Excerpt of ion models available
    - G4IonBinaryCascadePhysics,
    - G4IonPhysics,
    - G4IonPhysicsXS,
    - G4LightIonQMDPhysics,
    - G4IonINCLXXPhysics,
    - G4IonPhysicsPHP,
    - G4IonQMDPhysics.cc

Physics in G4 (vi)

- Hadronic and ions Models (iii):
  - High-energy models (string models)
    - **QGS**: Quark-Gluon String, QCD model, valid between 20 GeV and 1 TeV.
    - **FTF**: Fritiof Fragmentation, an alternative model to QGS, valid between 3 GeV and 1 TeV.
  - Cascades:
    - **Bertini**: classical model, widely used for LHC applications, for incident particles: p, n, π, K^+, K^-, K_L^0, K_s^0, Λ, Σ^+, Σ^-, Ξ^-, Ξ^+, Ξ^0, Ω^-, γ, e–.
    - **BIC**: similar to Bertini, but uses the Pre-Compound model for nuclear de-excitation after the cascade. Incident particles: p, n from 0 to 10 GeV, π from 0 to 1.3 GeV.
      The G4BinaryLightIonReaction class contains a binary cascade model for light ions.
    - **INCL++**: Liege intra-nuclear cascade, used in nuclear physics, particularly for spallation studies, valid for p, n, π up to 3 GeV/u.
    - **QMD**: Quantum Molecular Dynamics, applies only to particles heavier than p, n.
  - Pre-equilibrium model: G4PrecompoundModel.
  - De-excitation model: G4ExcitationHandler.

Physics in G4 (vii)

- Hadronic and ions Model (iv):
  - Examples:
    - Cross section of ^{11}B production in (p, ^{12}C) reaction



Physics in G4 (viii)

- Implementation:
  - runManager->SetUserInitialization(new PhysicsList);

```
class PhysicsList: public G4VModularPhysicsList
{
public:

  PhysicsList();
 ~PhysicsList();

  virtual void ConstructParticle();
  virtual void ConstructProcess();




. . .
};
```

must inherits from

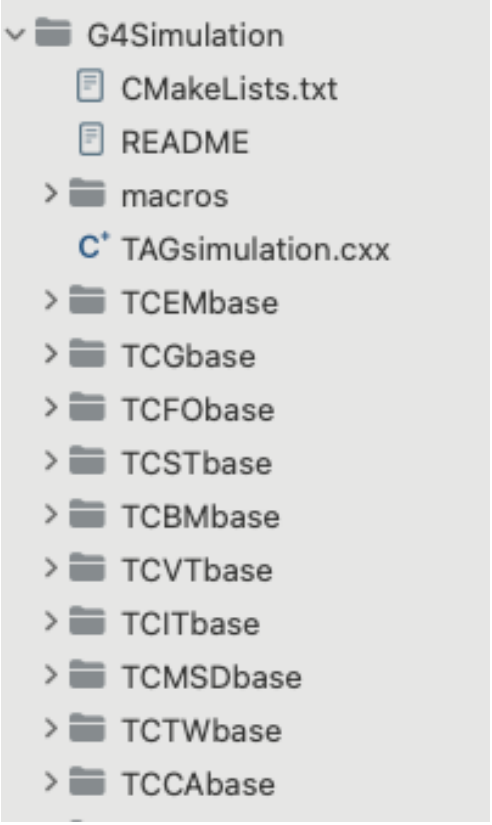
Need to be implemented

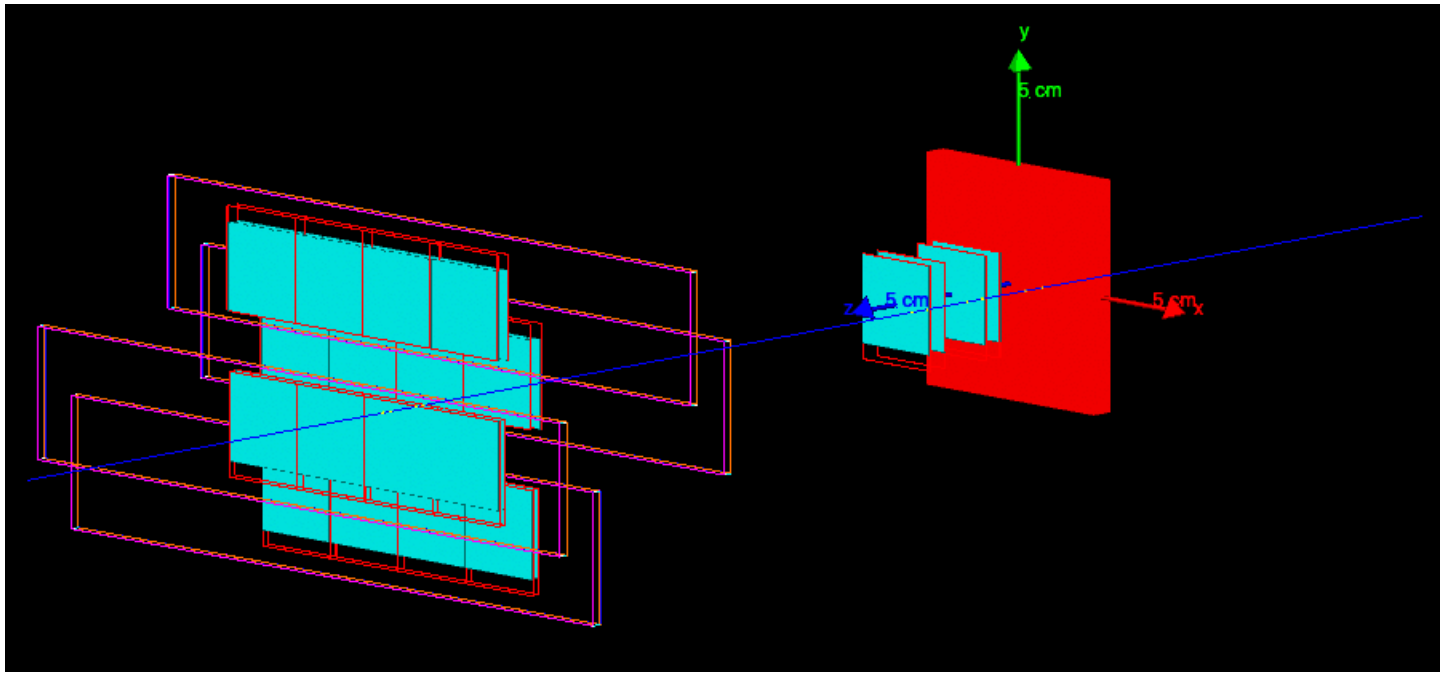
  - Pre-existing physics list initialisation:
    - runManager->SetUserInitialization(new QGSP_BERT());
    - runManager->SetUserInitialization(new QGSP_BIC());
    - runManager->SetUserInitialization(new QGSP_INCLXX());
    - More reference physics lists

Geant4 in Shoe

Folder in Shoe

- G4Simulation:





- Dedicated classes in G4Simulation folder, same structure as for Libraries folder
- Activate at cmake level: cmake -D**Geant4_DIR=$G4_BUILD11** /path/to/foot/src (need Geant4 version ≥ 11.2.0)

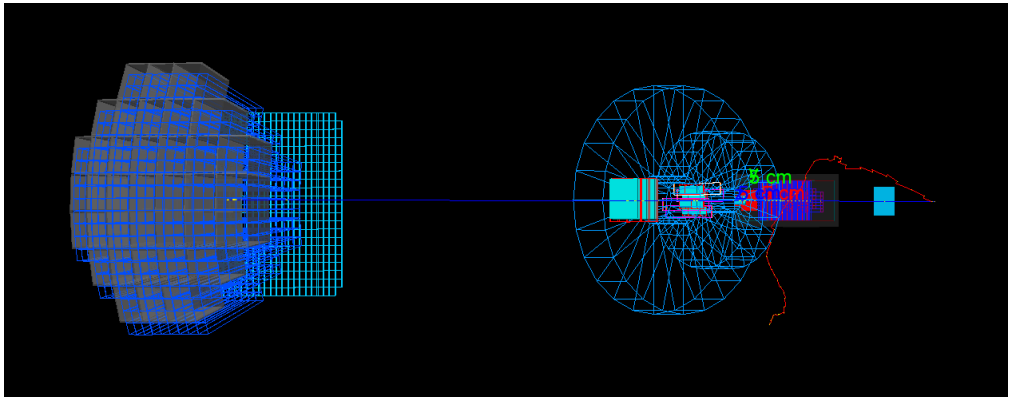
Geometry in Shoe (i)

- Classes:
  - TCGtargetConstructor: construct target geometry with TAGparGeo interface
  - TC*geometryConstructor: construct ST, BM, VT, IT, EM*, MSD, TW and CA geometry with TA*parGeo interfaces (*EM = Dipole)
  - TCFOgeometryConstructor: construct and placed detectors/target with TAGgeoTrafo interface.

- Interfaces with Shoe:
  - Geometry files call via the TAGcampaignManager interface
  - Detectors switch off/on respect to TAGrecoManager interface
  - All configuration and geometry files copy in G4Simulation folder (build)

➡ Still some corrections in BM and TW geometry

➡ Need to add passive materials (boxes)



Geometry in Shoe (ii)

- Implementation:
  - Example: TCVTgeometryConstructor

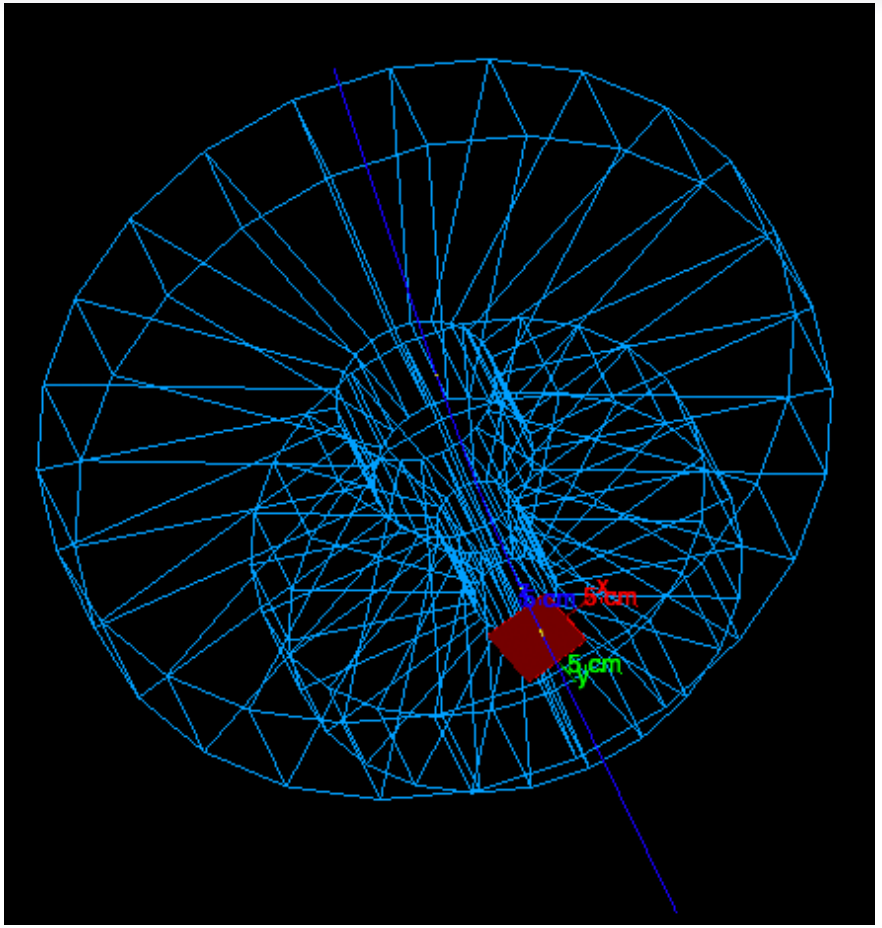
Interface with shoe geometry

```
class TCVTgeometryConstructor : public TCGbaseConstructor
{
public:
   TCVTgeometryConstructor(TAVTbaseParGeo* pParGeo);
   virtual ~TCVTgeometryConstructor();

   // Method in which the physical volume is constructed
   virtual G4LogicalVolume* Construct();
. . .
};
```

Need to be implemented

Magnetic Field in Shoe

- Classes:
  - TCEMgeometryConstructor: construct dipole geometry with TADIparGeo interface
  - TCEMfield: get back the field map values with TADIgeoField interface
  - TCEMfieldSetup: set propagation model (helix, Euler, Runge-Kutta, etc…)



- Printout info when including dipoles:

```
Info in <TCEMgeometryConstructor::Construct()>: Construct Magnets
Info in <TADIgeoField::TADIgeoField()>: Going to open ./data/MagneticMap_2023.table
Info in <TCEMfieldSetup::SetStepper()>: G4ClassicalRK4 (default) is called
Info in <TCEMfieldSetup::CreateStepperAndChordFinder()>: The minimal step is equal to 1.000000 mm
```

Beam in Shoe

- Classes:
  - TCGprimaryGeneratorAction: interface with TAGparGeo class

- Implementation:

Interface with shoe geometry

```
class TCGprimaryGeneratorAction : public G4VUserPrimaryGeneratorAction
{
public:
   TCGprimaryGeneratorAction(TAGparGeo* pParGeo);
   ~TCGprimaryGeneratorAction();

public:
   // Generate primaries
   void GeneratePrimaries(G4Event* anEvent);

private:
   G4ParticleGun* fParticleGun;        ///< Particle gun
   TAGparGeo*     fpParGeo;            ///< Geometry parameters for beam
   Bool_t         fHasPartDefined;     ///< Flag for defined particle
};
```

Need to be implemented

Physics in Shoe (i)

- TCGbasePhysics:
  - Standard progresses: ion/hadron elastic, decay and EM

```
TCGbasePhysics::TCGbasePhysics()
:  G4VModularPhysicsList(),
. . .
{
    // Ion Elastic scattering
    fIonElastic = new G4IonElasticPhysics(0);
    RegisterPhysics(fIonElastic);
  . . .
    // Particle decays
    fDecay = new G4DecayPhysics();
    fRadioactiveDecay = new G4RadioactiveDecayPhysics(0);
    RegisterPhysics(fDecay);
    RegisterPhysics(fRadioactiveDecay);
. . .
    // EM
    fExtraEmPhysics = new G4EmExtraPhysics(0);
    RegisterPhysics(fExtraEmPhysics);
    fElectromagnetic = new G4EmStandardPhysics_option3(0);
    RegisterPhysics(fElectromagnetic);
. . .
    // Hadron elastic physics
    fHadronElastic = new G4HadronElasticPhysicsHP(0);
    RegisterPhysics( fHadronElastic);
}
```

Physics in Shoe (ii)

- TCGphysicsINCLXX:
  - Related physics processes for ion/hadron inelastic and stopping

```
TCGphysicsINCLXX::TCGphysicsINCLXX()
:  TCGbasePhysics()
{
    // Stopping Physics
    fStoppingPhysics = new G4StoppingPhysicsWithINCLXX(0) ;
    RegisterPhysics(fStoppingPhysics);
. . .
    // Ion Inelastic physics
    fIonInelastic = new G4IonINCLXXPhysics(0);
    RegisterPhysics(fIonInelastic);
. . .
    // Hadron Inelastic physics
    fHadronInelastic = new G4HadronPhysicsINCLXX(0);
    RegisterPhysics(fHadronInelastic);
. . .
}
```

Physics in Shoe (iii)

- TCGphysicsQMD:
  - Related physics processes for ion/hadron inelastic and stopping (flag for heavy ion model)

```
TCGphysicsQMD::TCGphysicsQMD(Bool_t heavy)
:  TCGbasePhysics()
{
    // Stopping Physics
    fStoppingPhysics =  new G4StoppingPhysics(0);
    RegisterPhysics(fStoppingPhysics);
. . .
    // Ion Inelastic physics
    fIonInelastic = 0x0;
    if (heavy) {
      fIonInelastic = new G4IonQMDPhysics(0);
. .
    } else {
      fIonInelastic = new G4LightIonQMDPhysics(0);
. . .
    }
    RegisterPhysics(fIonInelastic);

    // Hadron Inelastic physics
    fHadronInelastic = new G4HadronPhysicsQGSP_BIC_HP(0);
    RegisterPhysics(fHadronInelastic);
. . .
}
```

Physics in Shoe (iv)

- TCGphysicsBIC:
  - Related physics processes for ion/hadron inelastic and stopping (flag for HP*)

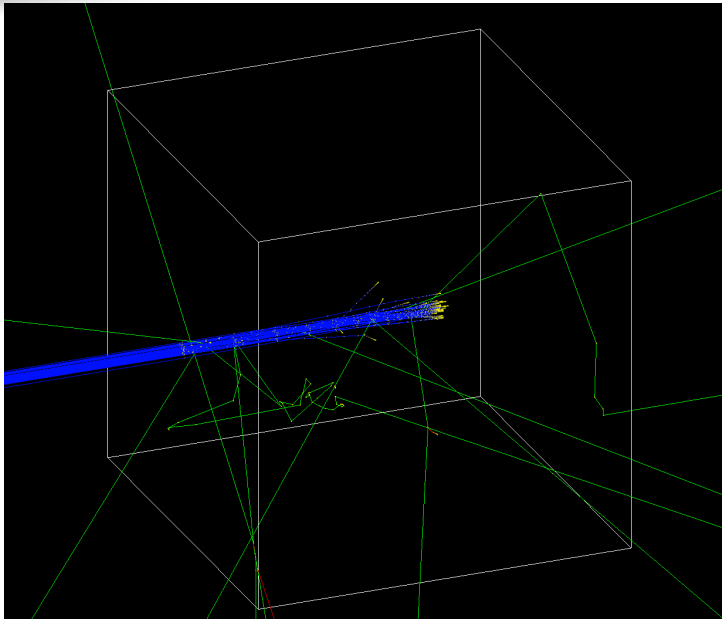
```
TCGphysicsBIC::TCGphysicsBIC(Bool_t hp)
:  TCGbasePhysics()
{
    // Stopping Physics
    fStoppingPhysics = new G4StoppingPhysicsFritiofWithBinaryCascade(0);
    RegisterPhysics(fStoppingPhysics);
 . . .
    // Ion Inelastic physics
    fIonInelastic = 0x0;
    if (hp) {
       fIonInelastic = new G4IonPhysicsPHP(0);
   } else {
       fIonInelastic = new G4IonPhysics(0);
   }
    RegisterPhysics(fIonInelastic);
. . .
    // Hadron Inelastic physics
    fHadronInelastic = 0x0;
    if (hp) {
       fHadronInelastic = new G4HadronPhysicsQGSP_BIC_AllHP(0);
   } else {
       fHadronInelastic = new G4HadronPhysicsQGSP_BIC(0);
   }
   RegisterPhysics(fHadronInelastic);
. . .
}
```

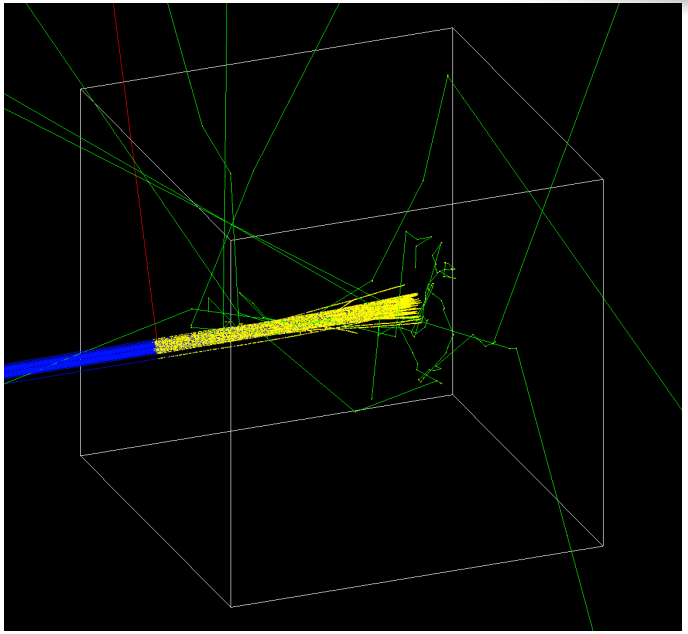
* Need an additional database set (TANDLE)

Physics in Shoe (v)

- EM cuts:
  - Default wise cuts put on e+, e- and γ tracking

```
if (cutEM) {
    physics->SetCutValue(1.*CLHEP::m, "gamma");
    physics->SetCutValue(1.*CLHEP::m, "e-");
    physics->SetCutValue(1.*CLHEP::m, "e+");
}
```





Physics in Shoe (vi)

- Main class: TAGsimulation

```
#if G4VERSION_NUMBER < 1102
    printf("Geant4 v11.01 and below not supported");
    exit(0);
#else
    if (physListName == "bic")
      physics = new TCGphysicsBIC();
   else if (physListName == "bichp")
      physics = new TCGphysicsBIC(true);
   else if (physListName == "incl")
      physics = new TCGphysicsINCLXX();
   else if (physListName == "qmd")
      physics = new TCGphysicsQMD();
   else if (physListName == "qmdhi")
      physics = new TCGphysicsQMD(true);
   else
      printf("\n\n No physics list defined !!\n\n”);
#endif
```

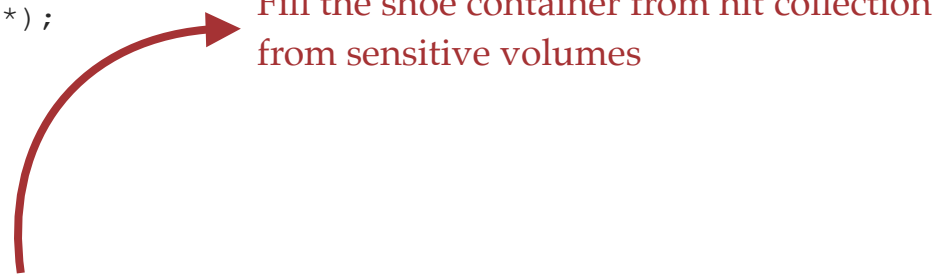
- Pre-implemented physics list
  - BIC (HP)
  - QMD (HI)
  - INCL

Event Action

- Class:
  - TCFOeventAction: fill the TAMCntuHit container via the help of TAMCevent

- Implementation:

```
class TCFOeventAction : public TCFObaseEventAction
{
public:
    TCFOeventAction(TCFOrunAction* runAction,
                    TCGbaseGeometryConstructor* hitGeomConstructor);
    ~TCFOeventAction();

public:
    // Begin event action
    void   EndOfEventAction(const G4Event*);
    // Collect hits
    void   Collect(const G4Event* evt);

private:
    // Fill track (particle) containers
    void   FillTrack();
   // Fill shoe hits from G4 hits
    void   FillHits(TAMCevent* mcHit, TCGmcHit* hit);
```



Tracking Action

- Class:
  - TCFOtrackingAction: fill the TAMCntuPart container via the help of TAMCevent

- Implementation:

```
class TCFOtrackingAction : public G4UserTrackingAction
{

private :
    TCFObaseEventAction* fEventAction;            ///< event action

public :
   TCFOtrackingAction(TCFObaseEventAction* aEventAction);
   ~TCFOtrackingAction();

   // Pre-tracking action
   void PreUserTrackingAction(const G4Track*);
   // Post-tracking action
   void PostUserTrackingAction(const G4Track*);
};
```

Executable

- Command line:
  - Default with mandatory options:

    TAGsimulation -**exp** CNAO23PS_MC -**run** 200

  - Additional options:

    -**out** fileName : output root file name (default wise: “exp_run.root”)

    -**nev** number : number of events to process (otherwise take the one in TAGparGeo interface)

    -**frag** : option to record only fragmented events

    -**phys** [bic(hp) qmd(hi) incl] : option to choose a given physics list (default wise: incl)

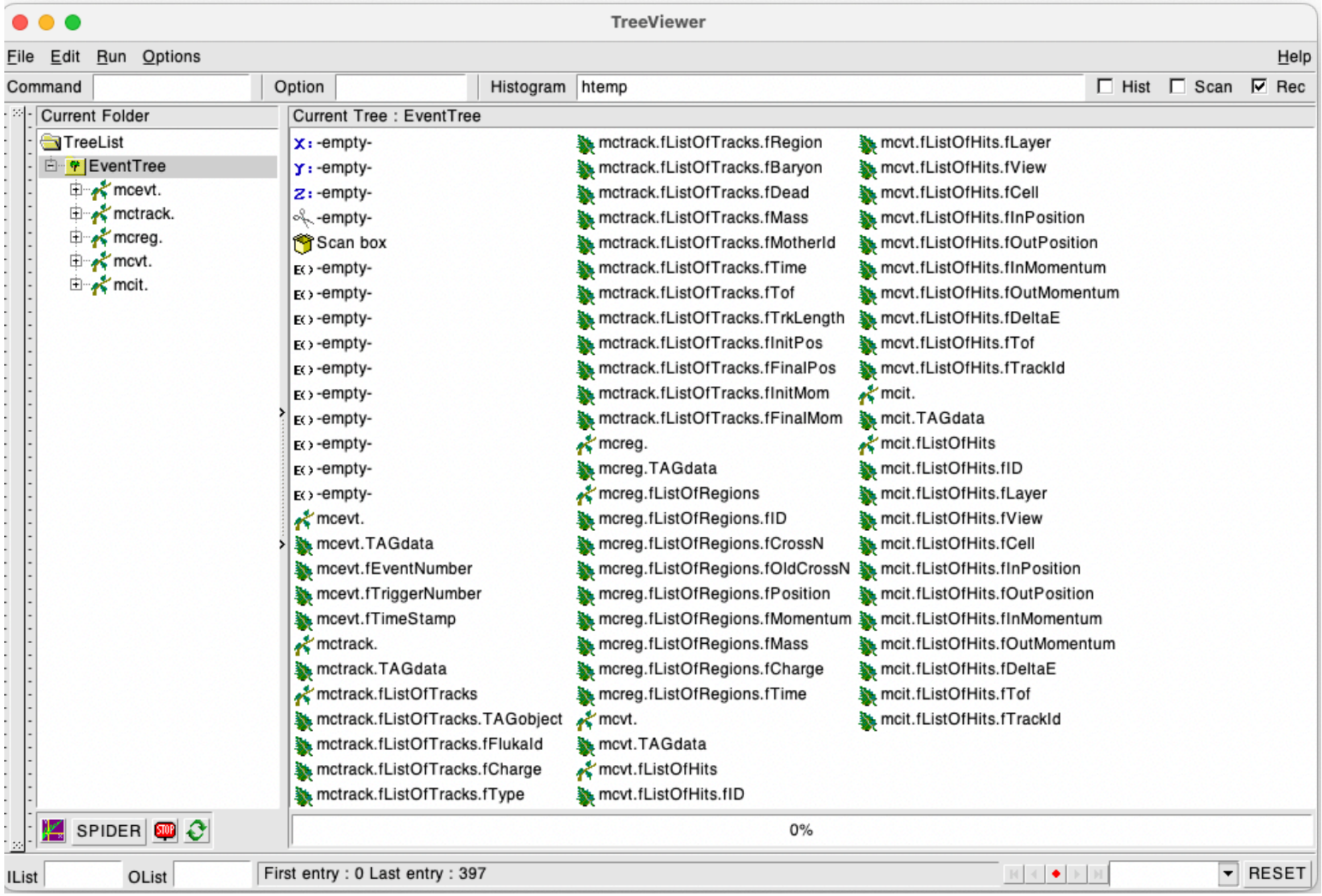
    -**seed** number : option to choose a given seed

    -**nocuts** : option to remove EM cuts

    -**i** : activate the interactive mode in Geant4

Outputs

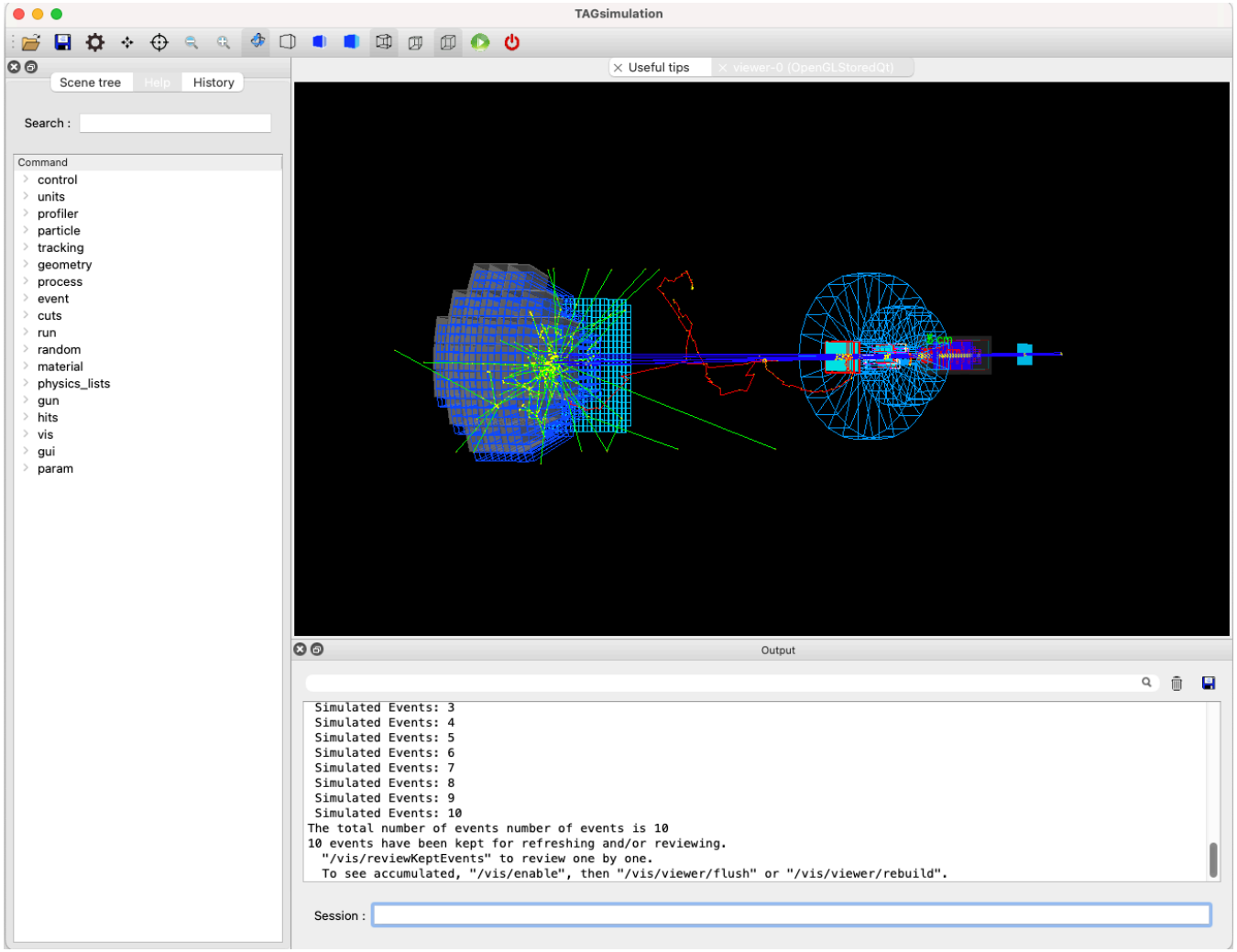
- Geant4 in SHOE creates an output rootfile containing a TTree (EventTree):
  - With the TAMCntuPart, TAMCntuHit, TAMCntuEvent and TAMCntuRegion* containers:



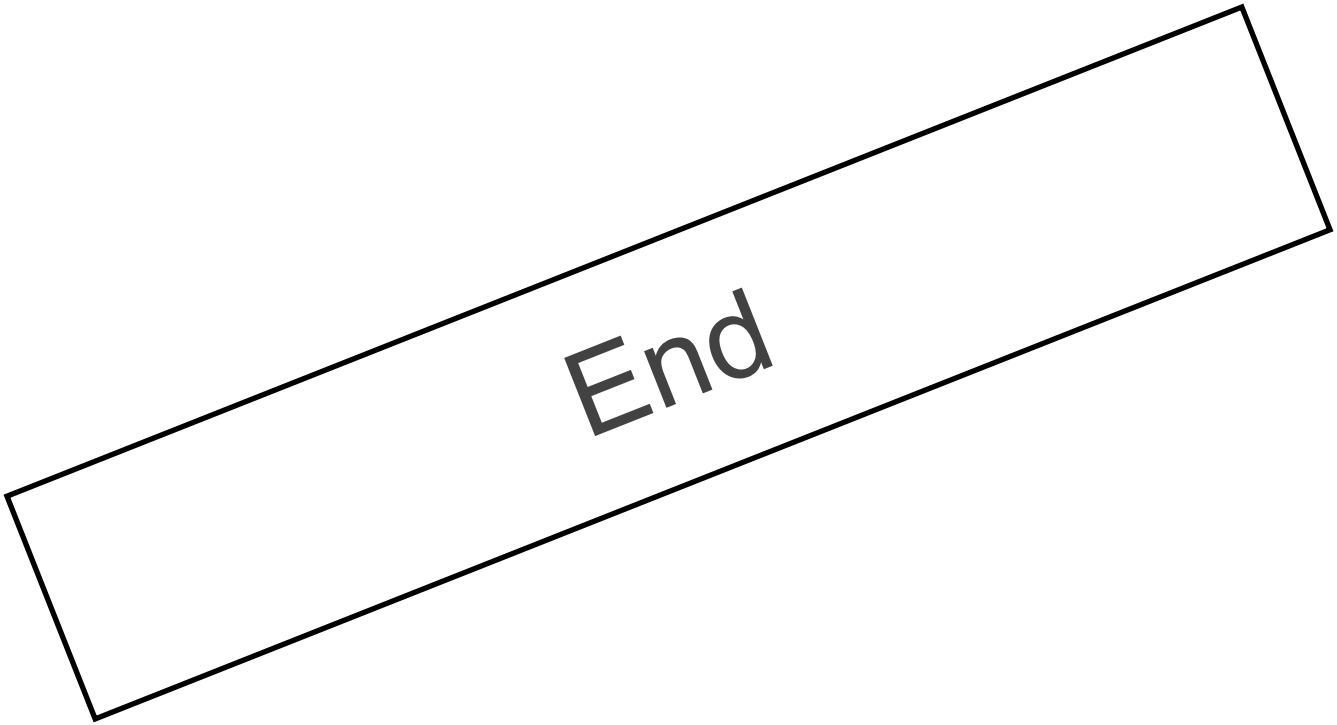
*Only Sensitive regions saved

Conclusions

- Geant4 in SHOE (with some corrections) is able to produce MC data



➡ Still some debugging in geometry (e.g.: BM fix with the help of Yun)


End

Backup