Введение в GEANT4

Иванов Артем Викторович
E-mail: arivanov@jinr.ru

Содержание

Мы уже знаем как создать детектор, материал и добавить магнитное поле. Но если запустить Geant4, то он будет выполнять моделирование физики «в тишине». Нам нужно добавить немного кода, чтобы извлечь полезную для нас информацию (выделенная энергия, координаты и т.д.)

Разные методы:

- Sensitive Detector
- User Actions
- Scoring

Содержание

Мы уже знаем как создать детектор, материал и добавить магнитное поле. Но если запустить Geant4, то он будет выполнять моделирование физики «в тишине». Нам нужно добавить немного кода, чтобы извлечь полезную для нас информацию (выделенная энергия, координаты и т.д.)

Разные методы:

- **Sensitive Detector**
- User Actions
- Scoring

Что такое Sensitive Detector?

Sensitive Detector (SD) — это пользовательский класс в Geant4, который «ловит» взаимодействия частиц в заданном объёме (логическом объёме — logical volume) и сохраняет информацию о них.

Основная задача зарегистрировать прохождение частиц через детектор и собрать данные: энергия, переданная частицей, координаты взаимодействия, время, тип частицы и др.

Как создать свой Sensitive Detector?

Шаг 1

Создаем два файла

include/SensitiveDetector.hh
src/SensitiveDetector.cc

Как создать свой Sensitive Detector?

Шаг 2 В файле include/SensitiveDetector.hh

```
#ifndef SensitiveDetector_h
#define SensitiveDetector_h 1

#include "G4VSensitiveDetector.hh"

class G4Step;

class SensitiveDetector : public G4VSensitiveDetector
{
  public:
    SensitiveDetector(const G4String& name);

   G4bool ProcessHits(G4Step* step, G4TouchableHistory* history);

};

#endif
```

Добавить include

Наследуемся от класса G4VSensitiveDetector

Конструктор класса

Метод будет вызывается каждый раз, когда частица делает шаг внутри объёма, привязанного к этому SD.

Как создать свой Sensitive Detector?

Шаг 3 В файле src/SensitiveDetector.cc

```
SensitiveDetector::SensitiveDetector(const G4String& name):G4VSensitiveDetector(name){
}

G4bool SensitiveDetector::ProcessHits(G4Step* step, G4TouchableHistory* history)
{
    вот тут мы будет сталкерить частицу

    return true;
}
```

Как создать свой Sensitive Detector?

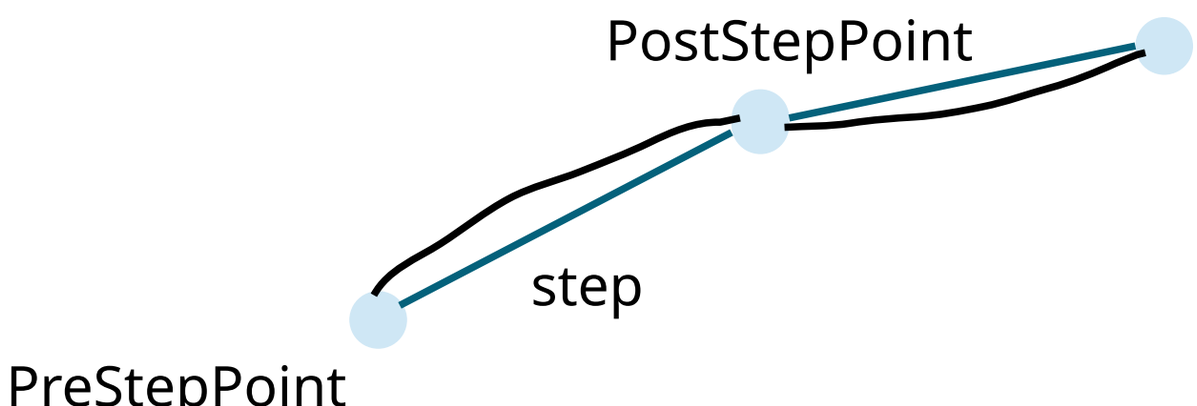
Шаг 4 В файле src/DetectorConstruction.cc

```
G4VPhysicalVolume *DetectorConstruction::Construct(){

  logic_detector = new G4LogicalVolume(solid_detector, GeneralStrawGas, "detector");

}

void DetectorConstruction::ConstructSDandField(){
    SensitiveDetector* SD = new SensitiveDetector("SD");
    G4SDManager::GetSDMpointer()->AddNewDetector(SD);
    SetSensitiveDetector(logic_detector, SD);
}
```

G4Step

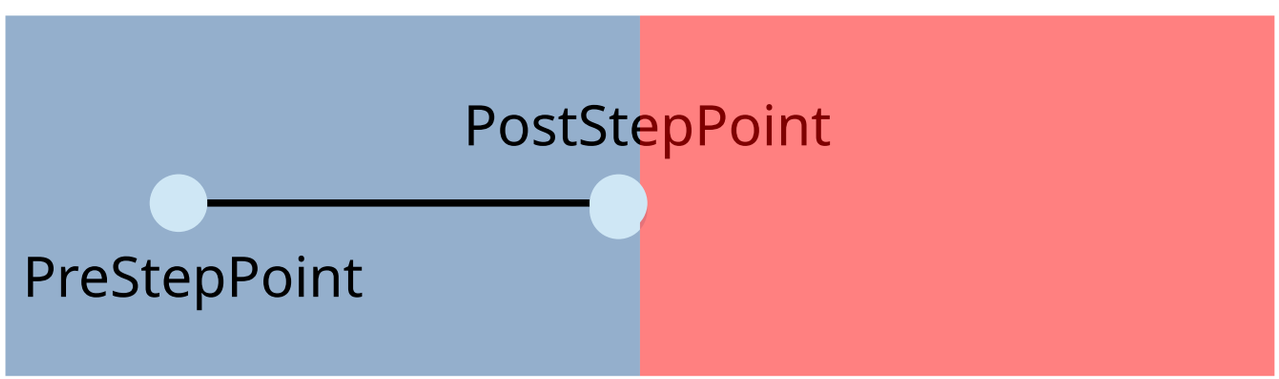
Шаг (G4Step)

- Описывает минимальное продвижение частицы через вещество с учетом различных физических процессов.
- На каждом шаге частица может терять энергию, менять направление, порождать вторичные частицы
- Шаг имеет две точки, а также «дельта»-информацию о частице (потеря энергии, время пролёта, затраченное на шаг, и т. д.).



G4Step

Для каждой точки траектории известен объём (и материал этого объёма), в котором она находится. Если шаг траектории упирается в границу между объёмами, то конечная точка шага физически останавливается на этой границе, но логически считается принадлежащей уже следующему объёму. Получайте информацию об объеме из PreStepPoin.



G4TouchableHistory

G4TouchableHistory

Это класс, содержащий полную иерархическую информацию о положении шага в геометрии: какие объёмы (и в каком порядке) были пройдены, чтобы добраться до текущего места.

Методы G4Point

Эти объекты содержат информацию о положении, энергии, материале и физическом состоянии частицы в точках.

*G4StepPoint *prepoint = aStep->GetPreStepPoint();* начало шага

*G4StepPoint *postpoint = aStep->GetPostStepPoint();* конец шага

Тип G4ThreeVector — это вектор с компонентами (x, y, z).

G4ThreeVector pos_pre = prepoint->GetPosition(); координата в начале шага

G4ThreeVector pos_post = postpoint->GetPosition(); координата в конце шага

G4double enery_pre = prepoint->GetKineticEnergy (); энергия в начале шага

G4double enery_post = postpoint->GetKineticEnergy (); энергия в конце шага

общее изменение энергии частицы за шаг (разница между enery_pre и enery_post).

G4double delta_energy = step->GetDeltaEnergy();

Частица в Geant4

Частица в Geant4 представлена тремя уровнями классов:

G4Track:

- Позиция, геометрическая информация и т.д.
- Это класс, представляющий частицу для отслеживания (трекинга).

G4DynamicParticle:

- "Динамические" физические свойства частицы, такие как импульс, энергия, спин и т.д.
- Это класс, представляющий конкретную (отдельную) частицу.

G4ParticleDefinition:

- "Статические" свойства частицы, такие как заряд, масса, время жизни, каналы распада и т.д.
- Все объекты G4DynamicParticle одного и того же типа частиц используют один и тот же (разделяют общий) G4ParticleDefinition.

G4Track

```
G4Track *track = aStep->GetTrack();
```

хранит динамические свойства частицы: энергия, импульс, спин и тп.

```
G4DynamicParticle *dparticle= track->GetDynamicParticle ();
```

хранит статические свойства типа частицы: масса, заряд, PDG-код, имя и тп

```
G4ParticleDefinition *pardef= track->GetParticleDefinition ();
```

это PDG-код частицы (по стандарту Particle Data Group)

```
G4int pdgcode = pardef->GetPDGEncoding();
```

Это импульс частицы

```
G4ThreeVector p = dparticle->GetMomentum();
```

Содержание

уникальный идентификатор (ID) частицы (1 это первичных частица)

```
G4int trackID = track->GetTrackID();
```

ID родительской частицы, из которой она родилась (0 для первичных частиц).

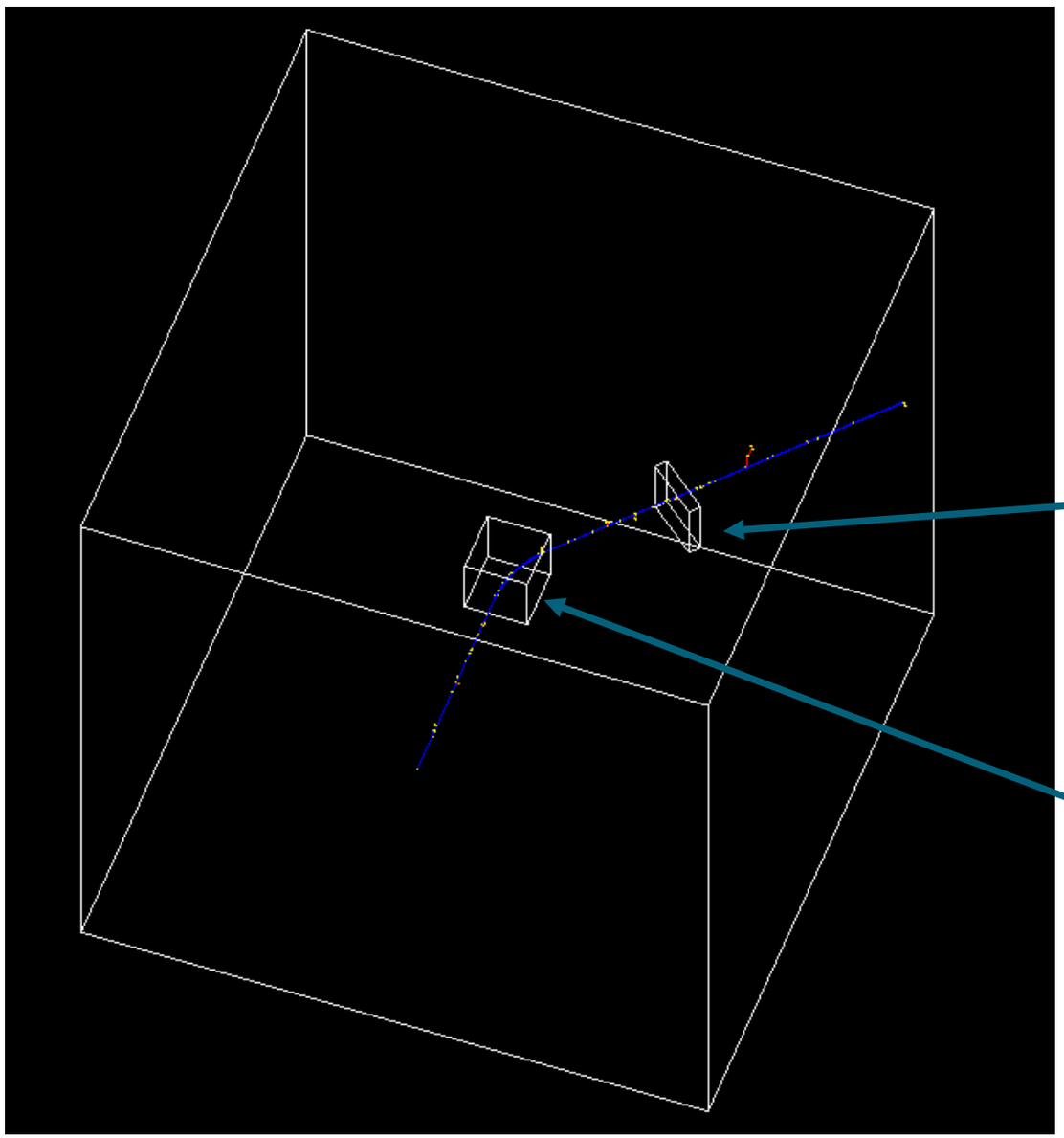
```
G4int parentID = track->GetParentID();
```

```
G4TrackStatus status = track->GetTrackStatus();
```

G4TrackStatus

fAlive	Continue the tracking
fStopButAlive	Invoke active rest physics processes and kill the current track afterward
fStopAndKill	Kill the current track
fKillTrackAndSecondaries	Kill the current track and also associated secondaries.
fSuspend	Suspend the current track
fPostponeToNextEvent	Postpones the tracking of the current track to the next event.

Практическое задание



https://github.com/Arivanoviktor/geant4.git

Директория p2

Детектор

Материал: газ Ar-CO2 70%/30%

Sensitive (Вывести координаты)

Магнитное поле