Środowisko programistyczne GEANT4

Leszek Adamczyk

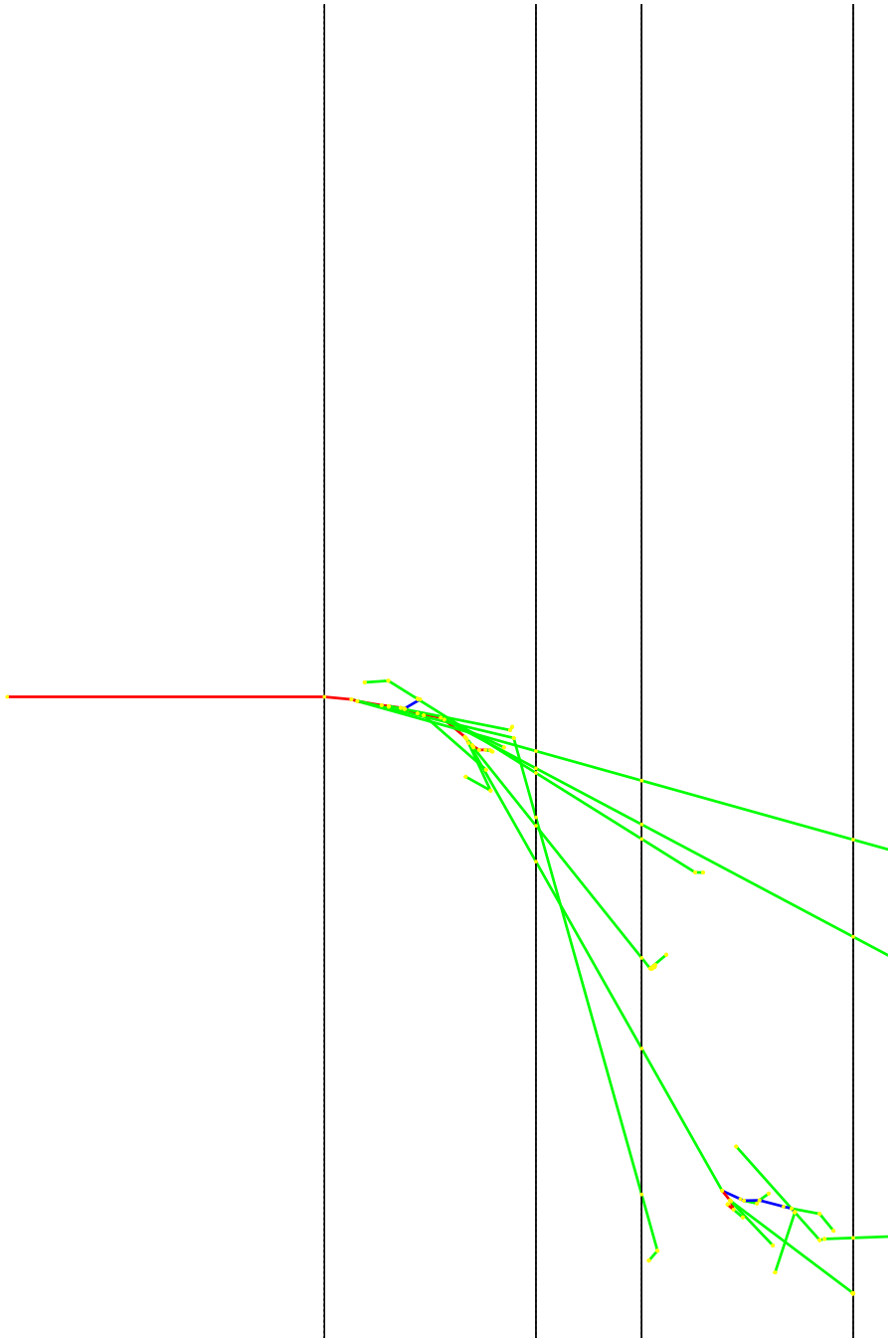
Wydział Fizyki i Informatyki Stosowanej
Akademia Górniczo-Hutnicza

Wykłady w semestrze zimowym 2013/2014

GEANT4: Symulacje przejścia cząstek przez materię

- proces przejścia cząstek przez materię jest zbyt złożony aby dało się jego wynik przewidzieć analitycznie;
- GEANT4 symuluje taki proces metodą Monte Carlo;
- z zadanych rozkładów prawdopodobieństwa losowane są wszystkie charakterystyki zderzeń elementarnych:
  - odległości pomiędzy zderzeniami;
  - rodzaju oddziaływania;
  - liczby cząstek wtórnych;
  - ich rodzaju, kąta emisji, energii, ...
- domyślnie symulowany jest los każdej z cząstek wtórnych;
- po wyczerpaniu cząstek otrzymujemy pełny opis ”sztucznie generowanego” procesu;

GEANT4: przejście elektronu przez materię



- przejście 50 *MeV* elektronu przez trzy warstwy materii;
- wtórnie powstają pozytony (kreacja pary elektron-pozyton) oraz fotony (promieniowania hamowania, anihilacja pozytonu).

GEANT4: Jądro

Jądro GEANT4 zapewnia:

- transport cząstek przez materię uwzględniając ich oddziaływanie z materiałem ośrodka oraz wpływ statycznych pól elektromagnetycznych aż do ich:
  - zatrzymania;
  - dezintegracji (rozpad lub absorpcje);
  - opuszczenia symulowanego obszaru;
- dostęp do informacji na temat procesu transportu oraz pełnych wyników symulacji:
  - na początku i końcu transportu cząstki;
  - na końcu każdego elementarnego kroku;
  - w momencie gdy cząstka wchodzi w interesujący nas obszar przestrzeni (np. obszar aktywny detektora, konkretny organ w ciele człowieka, tranzystor w układzie sterowania satelitą)

GEANT4: Twórca aplikacji

- użytkownik musi dostarczyć informacji o:
  - kształcie i składzie symulowanego ośrodka materialnego (Detector);
  - symulowanych procesach oddziaływania;
  - kinematyce cząstek pierwotnych.
- użytkownik może dostarczyć dodatkowych informacji:
  - o zewnętrznych polach elektromagnetycznych;
  - o działaniach jakie chcemy podjąć na końcu każdego elementarnego kroku podczas transportu lub w momencie gdy cząstka wchodzi w interesujący nas obszar detektora (metody UserAction).

GEANT4 : Event (przypadek)

- Event jest podstawową jednostką symulacji;
- na początku pierwotne cząstki umieszczane są na stosie;
- następnie cząstki jedna po drugiej pobierane są ze stosu i transportowane przez detektor;
- ewentualne cząstki wtórne odkładane są na stos;
- procesowanie przypadku kończy się w momencie opróżnienia stosu;
- przypadek reprezentowany jest przez klase G4Event składającą się się z następujących pól:
  - dane wejściowe: pierwotne cząstki;
  - dane wyjściowe: depozyty energii w obszarach aktywnych detektora oraz trajektorii cząstek;
- Procesowaniem przypadku kieruje klasa G4EventManager.

GEANT4 : Run

- Koncepcyjnie Run to zbiór Event'ów procesowanych przez ten sam detektor w identycznych warunkach fizycznych;
- w czasie Run'u użytkownik nie może zmienić:
  - detektora;
  - symulowanych procesów fizycznych;

  może zmienić kinematykę i rodzaj cząstek pierwotnych;
- Run reprezentowany jest przez klasę G4Run;
- procesowaniem Run'u steruje klasa G4RunManager.

GEANT4 : Track (Ślad)

- Track reprezentuje aktualny stan aktualnie transportowanej cząstki;
- Track nie ”pamięta” stanu cząstki z poprzedniego kroku transportu;
- Track nie jest ciągiem kroków, raczej Track jest aktualizowany po każdym kroku;
- do ”zapamiętania” kolejnych śladów cząstki używamy obiektu zwanego trajektorią;
- ślad reprezentowany jest przez klasę G4Track;
- procesowaniem śladu steruje klasa G4TrackingManager.

GEANT4 : Step (Krok)

- Krok składa się z punktu początkowego (PreStepPoint) oraz końcowego (PostStepPoint);
- oba punkty posiadają informacje o obszarze detektora (i materiale) w którym się znajdują, czy jest to pierwszy(ostatni) krok w obecnym obszarze detektora;
- w przypadku gdy krok ograniczony jest przez granicę ośrodków to punkt końcowy znajduje się fizycznie na granicy obszarów ale logicznie należy do następnego obszaru;
- krok posiada informacje o zmianie stanu cząstki podczas wykonywania kroku (strata energii, czas trwania kroku, ...)
- krok reprezentowany jest przez klase G4Step;
- klasa G4SteppingManager steruje procesowaniem kroku

GEANT4 : Trajectory (Trajektoria)

- W trakcie procesowania przypadku dostępna jest informacja tylko o aktualnie procesowanym śladzie oraz aktualnym kroku;
- okrojona informacja o historii śladów i kroków kopiowana jest dla każdej cząstki do obiektu zwanego trajektorią;
- klasa G4Trajectory zawiera okrojoną informacji o śladach;
- klasa G4TrajectoryPoint zawiera okrojoną informacji o krokach;
- w przypadku gdy klasy G4Trajectory lub G4TrajectoryPoint zawierają zbyt skąpe informacje, użytkownik może wyprowadzić własne klasy z klas bazowych G4VTrajectory oraz G4VTrajectoryPoint.

GEANT4 : Particle (Cząstki)

Cząstki w GEANT4 reprezentowane są przez trzy klasy:

- G4ParticleDefinition
  - określa ”statyczne” wielkości (ładunek, masa, czas życia, kanały rozpadu, ...);
- G4DynamicParticle
  - określa ”dynamiczne” wielkości fizyczne (pęd, energie, polaryzacje,...);
  - wszystkie obiekty klasy G4DynamicParticle tego samego typu stowarzyszone są z jednym obiektem klasy G4ParticleDefinition;
- G4Track
  - informacje geometryczne (pozycja, obszar detektora, materiał , straty energii, ...)
  - każdy G4Track stowarzyszony jest ze swoim własnym unikalnym obiektem klasy G4DynamicParticle

GEANT4 : Procesy fizyczne

- każdy typ cząstki ma własną listę procesów oddziaływania z materią;
- podczas każdego kroku wszystkie procesy z listy mają wpływ na:
  - wybór długości kroku;
  - zmianę wielkości fizycznych cząstki;
  - produkcję cząstek wtórnych;
  - zmianę stanu śladu.
- każdy proces charakteryzuje się jedną lub kombinacją kilku natur:
  - AtRest
    np. rozpad cząstki spoczywającej;
  - AlongStep (proces ciągły)
    np. promieniowanie Czerenkowa;
  - PostStep (proces dyskretny)
    np. rozpad cząstki w locie.

GEANT4: Algorytm wykonywania kroków

- dla każdego procesu dyskretnego losuje się odległość do następnego oddziaływania;
- najmniejszą z tych odległości wybiera się jako krok fizyczny;
- następnie oblicza się krok geometryczny jako odległość do granicy ośrodków;
- mniejszy z kroków fizycznego i geometrycznego decyduje o aktualnej długości kroku;
- zmiana energii kinetycznej w trakcie wykonywania kroku jest sumą wkładów od wszystkich procesów ciągłych;
- jeśli cząstka nie została zatrzymana przez procesy ciągłe to symuluje się proces dyskretny, który zdecydował o długości kroku.

GEANT4: Wyniki symulacji

- GEANT4 nie wie jakie informacje są nam potrzebne;
- użytkownik może mieć dostęp do interesujących go informacji na dwa sposoby:
  - korzystając z klas typu UserAction G4UserTrackingAction, G4UserSteppingAction,...
    - mamy dostęp do pełnej informacji po zakończeniu procesowania kazdego śladu, kroku, ...;
    - musimy sami przechować tę informację;
  - korzystając z funcjonalności GEANT4 zwanej obszarem aktywnym detektora (SensitiveDetector):
    - interesującemu nas obszarowi detektora przypisujemy obiekt klasy G4VSensitiveDetector;
    - GEANT4 automatycznie przechowuje wszystkie kroki wykonane w obszarze aktywnym detektora (Hits)
    - użytkownik ma do nich dostęp po zakończeniu procesowania przypadku korzystając z klasy G4UserEventAction

GEANT4: Układ jednostek

- GEANT4 nie ma domyślnych jednostek;
- aby nadać zmiennej wartość numeryczną, należy ją mnożyć przez jednostkę w której jest wyrażona np:

```
G4double length = 10.0*cm;
G4double energy = 10.0*GeV;
```

- prawie wszystkie powszechnie używane jednostki są zdefiniowane;
- można definiować własne jednostki;
- lista zdefiniowanych jednostek znajduje się w pliku nagłówkowym CLHEP:SystemOfUnits.h;
- aby uzyskać wartość:

```
G4cout << energy/MeV << "[MeV]" <<  G4endl;
G4cout << G4BestUnit(energy,"Energy") << G4endl;
```

GEANT4: Budowa aplikacji

Do komunikacji użytkownka z jądrem GEANT4 służą klasy bazowe (G4) :

- użytkownik tworzy klasy pochodne wyprowadzając je z klas bazowych;
- jądro GEANT4 operuje obiektami z klas pochodnych poprzez ich interfejs publiczny dziedziczony z klas bazowych;
- abstrakcyjne klasy bazowe (G4V) wymagają od użytkownika wyprowadzenia klas pochodnych oraz implementację metod czysto wirtualnych;
- nieabstrakcyjne klasy bazowe nie wymagają wyprowadzania klas pochodnych, ale ich wyprowadzenie umożliwia zmianę metod wirtualnych.

GEANT4: Klasy użytkownika

- main()
  GEANT4 jest środowiskiem, wymaga napisania programu głównego main();
- klasy inicjalizujące:
  - G4VUserDetectorConstruction
  - G4VUserPhysicsList
- klasy operacyjne
  - G4VUserPrimaryGeneratorAction
  - G4UserRunAction
  - G4UserEventAction
  - G4UserStackingAction
  - G4UserTrackingAction
  - G4UserSteppingAction

Klasy niebieskie sa opcjonalne, czerwone konieczne.

GEANT4: Katalog główny

- Proszę uruchomić przeglądarkę i otworzyć link:

  `http://home.agh.edu.pl/leszekad/dydaktyka/wfiis_geant4_2013/warsztaty_1/`
- zawartość katalogu głównego:

```
GNUmakefile  include  My.cc  src

./include:
MyDetectorConstruction.hh  MyPhysicsList.hh  MyPrimaryGeneratorAction.hh

./src:
MyDetectorConstruction.cc  MyPhysicsList.cc  MyPrimaryGeneratorAction.cc
```

- nazwy plików są jednocześnie nazwami klas które deklarują lub definiują;
- nazwy klas pochodnych zawierają nazwy klas bazowych poprzedzone wspólnym przedrostkiem charakterystycznym dla danego projektu.

GEANT4: Program główny

Program główny musi zawierać:

- utworzenie obiektu klasy G4RunManager

```
G4RunManager * runManager = new G4RunManager;
```

- utworzenie i rejestracje do RunManager'a obiektów obligatoryjnych klas:

  G4VUserDetectorConstruction
  G4VUserPhysicsList
  G4VUserPrimaryGeneratorAction

```
runManager->SetUserInitialization(new MyDetectorConstruction);
runManager->SetUserInitialization(new MyPhysicsList);
runManager->SetUserAction(new MyPrimaryGeneratorAction());
```

- inicjalizacja jądra GEANT4

```
runManager->Initialize();
```

- wykonanie petli po przypadkach

```
runManager->BeamOn(numberOfEvents);
```

GEANT4: Opis detektora

- należy wyprowadzić z abstrakcyjnej klasy G4VUserDetectorConstruction własną klase pochodną;
- zaimplementować wirtualną funkcje Construct()
  - definicje potrzebnych materiałów
  - definicje brył geometrycznych
  - opis umiejscowienia detektora
  - definicje obszarów aktywnych detektora
  - definicje pól elektro-magnetycznych
  - definicje sposobu wizualizacji detektora

GEANT4: Procesy fizyczne

- należy wyprowadzić z abstrakcyjnej klasy G4VUserPhysicsList własną klase pochodną ;
- zaimplementować metody wirtualne:
  - ConstructParticles()
    definicje wszystkich potrzebnych cząstek
  - ConstructProcesses()
    definicje wszystkich wymaganych procesów oddziaływania cząstek
  - SetCuts()
    określenie warunków na produkcję cząstek wtórnych (w postaci zasięgu)

GEANT4: Cząstki pierwotne

- należy wyprowadzić z abstrakcyjnej klasy G4VUserPrimaryGeneratorAction własną klase pochodną
- zdefiniować cząstki pierwotne (rodzaj, pozycje, kierunek, energie).
  Można wyprowadzić klase pochodną z abstrakcyjnej klasy bazowej G4VPrimaryParticle lub skorzystać z gotowych klas:
  - G4PartcleGun
    każdy przypadek to identyczna cząstka
  - G4HEPEvtInterface
    interfejs do standardowego formatu HEPEVT
  - G4GeneralParticleSource
    modeluje źródła promieniowania o dowolnym widmie energetycznym, rozkładzie kątowym i przestrzennym
- zaimplementować wirtualną funkcje GeneratePrimeries()

GEANT4: GNUmakefile

- Plik reguł dla programu make, służącego do automatyzacji kompilacji programów:

```
name := My
G4TARGET := $(name)
G4EXLIB := true
G4WORKDIR :=.

.PHONY: all
all:  lib bin

include $(G4INSTALL)/config/binmake.gmk
```

GEANT4: Opcjonalne klasy użytkownika I

Użytkownik może przedefiniować wirtualne funkcje klas bazowych aby zwiększyć kontrole lub dostęp do informacji na różnych etapach symulacji.

- G4UserRunAction
  - BeginOfRunAction
    - np. utworzenie histogramów
  - EndOfRunAction
    - np. zapis histogramów do pliku
- G4UserEventAction
  - BeginOfEventAction
    - np. selekcja przypadków
  - EndOfEventAction
    - np. analiza przypadku, zapis informacji

GEANT4: Opcjonalne klasy użytkownika II

- G4UserStackingAction
  - ClassifyNewTrack
    - wywoływana dla każdego nowego śladu
    - dokonuje klasyfikacji śladu do różnych stosów:
    - Urgent, Waiting, PostponeToNextEvent
  - NewStage
    - wywoływana po opróżnieniu stosu Urgent
    - np. zakończenie przypadku
  - PrepareNewEvent
    - inicjalizacja nowego przypadku
    - zmiana kryteriów klasyfikacji
- G4UserTrackingAction
  - PreUserTrackingAction
    - definiowanie własnych trajektorii
  - PostUserTrackingAction
    - usuwanie niepotrzebnych trajektorii
- G4UserSteppingAction
  - UserSteppingAction
    - niszczenie śladu w trakcie jego procesowania
    - zapisać pełną informacje o śladach

GEANT4: Pomoc i dokumentacja

- Przewodnik dla twórców aplikacji
  `http://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/ForApplicationDeveloper/html/`
  - wprowadzenie dla nowych użytkowników
  - opis najbardziej przydatnych narzędzi
  - opis procesu tworzenia aplikacji
  - ogólny przegląd środowiska.
- Przykładowe aplikacji zawarte w dystrybucji
  `$G4INSTALL/examples`
- Opis wszystkich klas
  `http://geant4.cern.ch/bin/SRM/G4GenDoc.csh`
- Przeglądarka kodu źrodłowego
  `http://www-geant4.kek.jp/LXR/`
  możliwość przeszukiwania całego kodu źródłowego