geant4_10_3_package Documentation

Release 1.0

Maximilian Burkart

Feb 02, 2017

TABLE OF CONTENTS:

CHAPTER

ONE

INTRODUCTION

The `geant4_10_3_package` *Python* package is designed to simulate particle interactions in matter. It uses the Geant4Py *Python* bridge to make the Geant4 toolkit available to the *Python* scripting language. It is intended to be used in courses on detector physics. The package enables an unexperienced user to execute simulations with the Geant4 toolkit. It aims to demonstrate the usage of the Geant4 toolkit. Therefore it introduces the class `ApplicationManager` to control the workflow of a Geant4 application.

The package contains modules used by all possible simulations. These include the visualization of the detector and simulated events, three action classes intended to store information on events and a primary generator. The changeable parts can be chosen by methods of the `ApplicationManager` class. The changeable parts include the geometry and the physics list. A set of ready to use geometries is provided in the `geometry` subpackage. Some of the reference physics lists as well as two additionally defined physics lists can be used. The properties of the primary particle can also be changed easily. This documentation gives an overview of the methods provided by the `ApplicationManager` and their usage.

The first section presents the methods of the `ApplicationManager` class. The second section gives an overview of the defined geometries and their interfaces. The materials used in the detector geometries are discussed in the third section. The fourth and last section introduces the additionally defined physics lists.

CHAPTER

TWO

CONTROLLING AN APPLICATION

Control Geant4.

This package provides the functionalities needed to control the Geant4 kernel. In order to do so the `ApplicationManager` class is defined. Auto-completion is also activated for use in interactive mode.

Subpackages

geometry : Define detector geometries.

Modules

actions : Define action classes and readout functions.

libphyslist : Define additional physics lists.

materials : Define detector materials.

vis : Visualize the detector and events.

Classes

ApplicationManager : Control the Geant4 application.

class `geant4_10_3_package.`**`ApplicationManager`**

Control a Geant4 application.

This class provides an easy way to control the workflow of a Geant4 application and to change parameters of the simulation. It allows the analysis of simulated events.

`calo_readout``()`

Count the number of charged tracks in the active material.

Count the number of charged particles crossing active layers of a calorimeter. First determine the start and the end point of the track. Next, use those points to calculate the number of crossed active layers for each track. Add up the result for all tracks. To insure correct functionality the volumes in the detector have to be numbered in increasing order and the active layers assigned odd numbers. The parameter for the world volume should be -1.

Returns The number of charged tracks in all active layers.

Return type int

change_material()

Change the material of the `pbbox` geometry.

This method enables switching between lead and iron as the material the `pbbox`'s Target volume is built of.

get_edep_in_volume(*volumeName*)

Calculate the energy deposited in a volume.

Parameters **volumeName** (*str*) – Name of the volume.

Returns The energy deposited in the volume in GeV.

Return type float

get_xOfFirstVertex()

Calculate the x coordinate of the first vertex.

Loop over all secondary particles and find the vertex where the first secondary particle is created.

Returns The x coordinate of the creation vertex of the first secondary particle.

Return type float

initialize()

Initialize the Geant4 kernel.

This method creates and passes the three mandatory and the additional user action classes to the `RunManager`. The `Initialize()` method of the `RunManager` is then invoked to initialize the detector. This method must be called before starting the first run. Depending on the state the `isInitialized` flag is in, all action classes are either assigned to the `RunManager` instance or only the new geometry is initialized. The re-initialization of the geometry takes place if the flag holds true.

set_energy(*energy*)

Set the energy of the primary particle.

Parameters **energy** (*float*) – Kinetic energy of the primary particle in GeV.

set_geometry(*geometryName*)

Set the geometry.

Specify the `G4VUserDetectorConstruction` class used to describe the detector geometry during the simulation. This method has to be invoked before calling the *initialize()* method. The method is also used to change the geometry between two runs. Additional arguments may be passed to this method like one would call the object itself, e.g.:

```
set_geometry('pbbox(length=40)')
```

Parameters **geometryName** (*str*) – Name of the `G4VUserDetectorConstruction` class that should be used during the simulation.

set_numberOfParticles(*nOfParticles*)

Set the number of particles.

Parameters **nOfParticles** (*int*) – Number of primary particles simultanously generated in one event.

set_particle(*particle*)

Set the primary particle.

Specify the primary particle the ParticleGun is going to use in the following events. If the particle is referenced to by an integer, this integer is the representation of the particle in terms of the Monte Carlo particle numbering scheme.

Parameters **`particle`** (*`str or int`*) – Identifier of the particle. If the particle is referenced to by an integer, this integer is the representation of the Monte Carlo particle numbering scheme. Some string and integer representations of common particles are given in the table below.

string representation	PDG code
e-	11
e+	-11
mu-	13
mu+	-13
pi+	211
gamma	22

`set_physics_list` (*physList*)

Set the physics list.

Specify the physics list used during the simulation.

Parameters **`physList`** (*`str`*) – Name of the physics list. Possible physics lists include the lists defined in the `libphyslist` submodule and the lists provided by the Geant4Py module.

`start_run` (*numberOfEvents=1*, *vis=1*)

Start a run.

Parameters

- **`numberOfEvents`** (*`int`*) – Number of events simulated in the started run.

  Default is 1.

- **`vis`** (*`int`*) – Draw option. If the value is `0`, visualization is disabled.

  Default is `1`.

CHAPTER

THREE

GEOMETRIES

Construct the detector geometries.

This subpackage provides a set of geometries that can be used during simulations. Each geometry is defined in its own module. The definition of the geometries is done by defining a class inherited from the `G4VUserDetectorConstruction` class.

Classes

The available geometry classes are:

class `geant4_10_3_package.geometry.example_geometry.`**`example_geometry`**(*halfX=20.0*)

A simple geometry example geometry.

This example geometry introduces the first user to the Geant4 way of defining geometries.

Parameters **`halfX`** (*`float`*) – Half length of the created box in cm. The created box ranges from -halfX to halfX.

Default is 20.

class `geant4_10_3_package.geometry.pbbox.`**`pbbox`**(*length=20.0*, *height=10.0*)

Create a simple lead box to simulate elementary interactions.

The material of the box can be changed to iron via the `change_material()` method of the `ApplicationManager` class.

Parameters

- **`length`** (*`float`*) – Length of the lead box in cm.

  Default is 20.

- **`height`** (*`float`*) – Height and width of the lead box.

  Default is 10.

class `geant4_10_3_package.geometry.samplingcalo.`**`samplingcalo`**(*absLen=2.0*, *actLen=1.0*, *layerNum=16*)

Define a sampling calorimeter made of lead and liquid argon layers.

The width of the absorber, the active layers and the number of layers may be set by the user. The length of the calorimeter is calculated from these values.

Parameters

- **absLen** (*float*) – Width of the absorber layer in cm.

  Default is 2.
- **actLen** (*float*) – Width of the active layer in cm.

  Default is 1.
- **layerNum** (*int*) – Number of active layers in the calorimeter.

  Default is 16.

class `geant4_10_3_package.geometry.febox_with_tailcatch.`**`febox_with_tailcatch`**(*length=20.0*, *height=10.0*)

Create a volume for leakage studies.

The volume consists of an iron box followed by a smaller lead box. The lead box is used to determine if a part of the energy of the incoming particle is not deposited in the iron box.

Parameters

- **length** (*float*) – Length of the iron box in cm.

  Default is 20.
- **height** (*float*) – The y and z dimension of the box in cm.

  Default is 10.

class `geant4_10_3_package.geometry.compcalor.`**`compcalor`**(*absLen=3.0*, *actLen=1.0*, *feLen=1.6*)

Construct a sampling calorimeter used for compensation studies.

The calorimeter consists of passive iron layers and active scintillator layers. Its length is determined by the total length of iron that should be used. The number of layers is calculated from the given parameters.

Parameters

- **absLen** (*float*) – Width of the absorber layer in cm.

  Default is 3.0.
- **actLen** (*float*) – Width of the active layer in cm.

  Default is 1.0.
- **feLen** (*float*) – Total amount of iron given in length in m.

  Default is 1.6.

class `geant4_10_3_package.geometry.tumortherapy.`**`tumortherapy`**

Construct the geometry used for studies on radiation therapy.

The geometry represents the head of a patient. It consists of a spherical shell made out of the bone material. Inside this shell are two water-filled spheres. The bigger one represents the brain of the patient and the small one the tumor.

CHAPTER FOUR

MATERIALS

Create the materials used in the geometries.

This module provides the materials being used by the geometries. These materials can also be used to define new `G4VUserDetectorConstruction` classes. The properties of materials not created through the `gNistManager` instance are taken from the review *Atomic and Nuclear properties of materials* provided by the Particle Data Group.[Pat16]_ All available materials are listed below.

vac [G4Material] The representation of the vacuum.

air [G4Material] The representation of air at standard conditions.

water [G4Material] The representation of water.

Pb [G4Material] The representation of lead.

Fe [G4Material] The representation of iron.

liquidAr [G4Material] The representation of liquid argon for use in calorimeters.

scint [G4Material] The scintillator material used in the compensating calorimeter. The properties of this material are copied from the old application.

pet [G4Material] The representation of polyethylene.

bone [G4Material] The bone material used to build the skull.

CHAPTER

FIVE

PHYSICS LISTS

Provide two additional physics lists.

The two physics lists inherit from reference physics lists and add a tracking cut on neutrons with energies below 0.5 MeV to those physics lists. They also allow the use of the `G4UserLimits` class during a simulation.

class `geant4_10_3_package.libphyslist.`**`MyQGSP_BERT`**`(`*verbose=1*`)`

Create the physics list MyQGSP_BERT.

This list inherits from the reference physics list QGSP_BERT and is used when cuts on the step size or tracking cuts are needed.

Parameters **`verbose`** (*`int`*) – Verbosity during the construction of the list.

class `geant4_10_3_package.libphyslist.`**`TherapyPhysics`**`(`*verbose=1*`)`

Create the physics list used for the radiation therapy simulation.

This physics lists inherits from the QGSP_BIC reference physics list.

Parameters **`verbose`** (*`int`*) – Verbosity during the construction of the list.

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search

[Pat16] Patrigiani, C. et al (Particle Data Group): 2016 Review of Particle Physics. Chinese Physics C, 40(10001), 2016. avaiable at: www-pdg.lbl.gov/2016/reviews/ rpp2016-rev-atomic-nuclear-prop.pdf