Using GPS (The General Particle Source)

This page describes how to use the GPS particle source provided by GEANT.

This particle source is incredibly useful, but extremely is idiosyncratic. It's also has relatively little documentation, and the documentation that exists is often misleading. This page attempts to cover the basics of using GPS to write GEANT macros that produce samples of events to be used for testing ND280 software (meaning I got annoyed enough at the GPS documentation make notes).

Introduction

The general particle source as developed for the ESA, so it tends to use astronomical conventions. This is particularly relevant with describing angles and directions. Astronomers (and apparently space scientists) tend to speak about where the particle is coming from, not where it is going. This makes for an odd match when dealing with a particle simulation which is concerned with tracking the particles, and not determining the position of the source. Once you get beyond the idiosyncratic definitions (which are at odds with the rest of GEANT), the GPS can be used to create almost any sort of particle distribution for testing programs.

Controlling the Simulated Particle

The particle being simulated is set using the "/gps/particle" command. The particles are set by name, and can be any of the ones simulated by GEANT. The most common particles are "mu-", "mu+", "pi+", "pi-, "e-", "e+", "gamma", and "proton". For example

```
/gps/particle mu-
/gps/particle pi+
/gps/particle gamma
/gps/particle proton
```

Controlling the Position Distribution

The vertex position is controlled using the commands in the /gps/pos/ subdirectory. The most useful distributions are the Point, Plane, and Volume. To generate a point source the following commands are used

Making a point source

```
/gps/position 0 0 -50 cm
/gps/pos/type Point
```

where the first line specifies that this is a point and the second line specifies the position. Notice that all three coordinates must have the same units. For a point source, it's not necessary to specify the type. [GEANT is very inconsistent about the spelling of "centre" vs "center" with 19 classes using "centre" and 33 using "center". You just have to check which version is being used by which class. For consistency in our code, prefer /gps/position and avoid /gps/pos/centre]

Making a rectangular or circular source

Since a point source can bias tests, vertices should usually be spread over a plane or a volume. The vertices can be distributed over a flat surface by setting

```
/gps/position 0 0 -50 cm
/gps/pos/type Plane
/gps/pos/shape Rectangle
/gps/pos/halfx 1 cm
/gps/pos/halfy 1 cm
/gps/pos/rot1 0 0 -1
/gps/pos/rot2 0 1 0
```

Making a volume source

To distribute the vertices over a volume the type is changed to Volume, and a volumetric shape must be chosen. The following

commands will generate a box

```
/gps/position 0 0 -50 cm
/gps/pos/type Volume
/gps/pos/shape Para
/gps/pos/halfx 1 cm
/gps/pos/halfy 1 cm
/gps/pos/halfz 1 cm
```

that is 100 by 100 by 10 cm. It's important that the /gps/position command comes first. The other shapes that might be useful are Sphere and Cylinder which are specified as

```
/gps/position 0 0 -50 cm
/gps/pos/type Volume
/gps/pos/shape Sphere
/gps/pos/radius 1 cm
```

and

```
/gps/position 0 0 -50 cm
/gps/pos/type Volume
/gps/pos/shape Cylinder
/gps/pos/radius 1 cm
/gps/pos/halfz 1 cm
```

Controlling the Direction Distribution

The direction distribution is set using the commands in the /gps/ang/ subdirectory. When specifying the direction, be aware that the code was written under contract to the ESA, and no effort was made to be consistent with the rest of the GEANT code base. As a result, the directions use a different angle convention when compared to the rest of GEANT. In fact, while the documentation says that the /gps/ang/ commands control the angular distribution of the source, they actually control the direction from a point along an initial particle direction ray back to where the source is found [note that the mathematics are very poorly defined]. To be honest, it's an odd and poorly thought out mix of definitions used by physicists and astronomers.

Specifically, to get a particle traveling in the positive Z direction, you specify a theta of 180 degree.

The coordinate system used to generate the direction can be changed by using the /gps/ang/rot1 and /gps/ang/rot2 commands. The X axis of the direction coordinate system is defined by the /gps/ang/rot1 command, and the XY plane is determined by the /gps/ang/rot2 command. Neither of the axes need to be unit vectors, and the rot2 vector doesn't need to be perpendicular to rot1 (the normal will be determined by rot1 cross rot2). The directions will then be distributed around the negative Z axis in the rotated coordinate system. This means that

```
/gps/ang/rot1 0 0 1
/gps/ang/rot2 0 1 0
```

will rotate the direction coordinate system so that the local Z axis points along to global negative X axis. This will cause the generated particles to travel along the positive X axis (i.e. along the local negative Z axis). Note: The default for /gps/ang/rot2 is [0 1 0], so it doesn't need to be specified.

Setting a fixed particle direction

A fixed particle direction can be specified using

```
/gps/direction 0 0 1
```

where this specifies the direction that the particle is traveling. In this case, the particle is traveling along the positive Z axis. Notice unlike the rest of the gps direction commands, this uses the same convention as the rest of GEANT.

Making directions with a Gaussian spread

The problem with specifying a fixed direction is that it may bias a test. A Gaussian direction spread can be created using a one dimensional beam traveling along the local negative Z axis (yes, it is a bizarre choice). For instance, to create a beam traveling along the Z axis with a 10 degree divergence (sigma) the following commands would be used.

```
/gps/ang/type beam1d
/gps/ang/sigma_r 10 deg
/gps/ang/rot1 0 0 1
/gps/ang/rot2 0 1 0
```

The first line specifies that a 1D beam should be generated where this means that the generated directions have a spread around the negative Z axis. The second line specifies a 10 degree spread around the negative Z axis. The third line fixes the screwball convention that the beam travels along the negative Z axis by reversing the coordinate system (this aligns the direction coordinate system so that it's negative Z axis points along the global Z axis). You should be aware that (for reasons completely beyond me, but I've verified it in the source code) the /gps/ang/maxtheta and /gps/ang/mintheta commands have no effect on the beam distributions.

Generating directions in a cone

To spread the directions isotropically over all direction, the following command is used

```
/gps/ang/type iso
```

The directions can then be limited to a cone using the max and min commands. By default, the maximum angle is 180 degree and the minimum is 0 degree. Bear in mind that 180 degrees points along the positive Z axis and 0 degrees points along the negative Z axis (these guys failed high school geometry). To create a 10 degree cone of directions along the Z axis use

```
/gps/ang/type iso
/gps/ang/mintheta 170 deg
```

or alternatively

```
/gps/ang/type iso
/gps/ang/maxtheta 10 deg
/gps/ang/rot1 0 0 1
/gps/ang/rot2 0 1 0
```

where the final line deals with the wacky angle convention.

Controlling the Energy Distribution

The energy distribution is set using the commands in the /gps/ene/ subdirectory. There are many options available, but for testing programs the most useful ones are to create a mono-energetic, Gaussian, or uniform energy source. By default the energy is specified as the kinetic energy of the created particle.

Making a monoenergetic source

To create a monoenergetic source, the energy is specified like this:

```
/gps/ene/mono 100 MeV
```

where you must specify both the energy value and the unit. This create a monoenergetic source with 100 MeV of kinetic energy.

Making a uniform energy distribution.

A uniform energy distribution can be created using:

```
/gps/ene/type Lin
/gps/ene/gradient 0
/gps/ene/intercept 1
/gps/ene/min 100 MeV
/gps/ene/max 500 MeV
```

where this creates a source with a uniform kinetic energy distribution between 100 MeV and 500 MeV of kinetic energy. The gradient is specified in MeV^{-1}. The intercept is a pure number, and the final distribution does not need to be normalized.

Making a Gaussian energy distribution

A Gaussian energy distribution can be created using

```
/gps/ene/type Gauss
/gps/ene/mono 400 MeV
/gps/ene/sigmae 30 MeV
```

where this create a source with a Gaussian spread of 30 MeV around 400 MeV.

Creating Multiple Particle Events

The GPS is very inflexible when it comes to handling multiple particles in a single event, so it's not possible to handle the general case with multiple particles coming from the same vertex. This is overcome in detSim by implementing the "/generator /combine" command which is discussed below.

For the simple case of generating multiple particles drawn from the same angle and energy distributions, but with the same position, use "/gps/number" command. For instance

```
/gps/number 2
```

would create two particles at the vertex. Note that both particles have the same energy and position distributions.

If you need control over the individual particles, then you have to use the /gps/source commands in combination with the /generator/combine command. For instance, to create an event with a mu- and a proton from the same vertex position (notice there will be two "primary vertexes" with the same position coordinates), you could create a macro:

```
# Clear the source and tell GPS to make multiple vertices
/gps/source/clear
/gps/source/multiplevertex true

# Create the first particle.  This can be done using any of the GPS macro
# commands.
/gps/source/add 1
/gps/particle mu-
/gps/energy 500 MeV
```

```
/gps/direction 0 1 1
/gps/position 0.0 0.0 -50 cm
/gps/pos/type Volume
/gps/pos/shape Para
/gps/pos/halfx 1 cm
/gps/pos/halfy 1 cm
/gps/pos/halfz 1 cm

# Create the second particle.  This can be done using any of the GPS macro
# commands.  The position will be overridden below.
/gps/source/add 2
/gps/particle proton
/gps/energy 300 MeV
/gps/direction 0 0 1
/gps/position 0 0 0

# Add the GPS generator.  It will create two primary vertices
# (G4PrimaryVertex objects).
/generator/add

# Copy the vertex position from the first G4PrimaryVertex object to the
# second.  Notice that the vertices in GPS are numbered from one, but that
# the G4PrimaryVertex objects are numbered from zero.

/generator/combine 0 1
```

The "/generator/combine" command copies the vertex position from the first vertex to the second, and takes three arguments. The first argument is the vertex number to copy the position, the second argument is the vertex number to copy the position into, and the optional third argument is a boolean flag specifying if the position of the second vertex should be relative to the first. If the argument is true, then the position is of the second particle is adjusted to be relative to the position of the first (default: false). This can be used to simulate displaced vertices. As an example, if the optional third argument is true, the first vertex position is [1,1,1], and the second vertex position is [0,1,0], then the simulated vertex positions will be [1,1,1] and [1,2,1].

Package Summary

Package Name: detSim

Package Version: master

Package Manager: Clark McGrew -- clark.mcgrew@sunysb.edu
