VXD Simulation

Peter Kvasnicka
Peter.Kvasnicka@mff.cuni.cz

Charles University in Prague

Belle II F2F Tracking group meeting,
12-14 May 2014, INFN Pisa, Italy

Outline

PXD and SVD SimHits

2 kinds of SimHit-like objects produced for silicons:

SimHits : Data for Geant4 steps (set to 5 μm in Si)

- Start and end positions, start momentum, deposited energy, time, particle pdg.
- tens per reconstructed hit

TrueHits Data for a piece of MC track entering and/or leaving a Si detector.

Problem:

- Too detailed data stored for VXD simulation. The detailed information on energy deposition in space are only needed for digitization; for all other purposes, summary TrueHit-like data would be OK.
- Without TrueHits in background, occupancy studies in SVD require a cloned digitizer.
- The amount of simulation data is large.

PXD and SVD SimHits - cont'd

Solution:

- Economize SimHits
  - Piecewise linear approximation of track in Si
  - Aggregate SimHits over linear segments.
  - Approximate energy deposition along segments with a small set of discrete deposits.

Plan:

- Martin Ritter undertook the task.
- Planned with the general update of `basf2` SimHits last week of May.

Merging of SVDDigits

SVD digit is an APV sample

- Several consecutive samples are read out from each strip at trigger.
- As the number of samples had been unclear when the digitization scheme was implemented, individual APV readings are stored in independent objects.
- 2D clustering (sensitive coordinate + time) is then used to reconstruct clusters.

Problem:

- Now the SVD readout scheme is becoming fixed, so we can fix the number of APV samples
- Thus we can get rid of 2D clustering and data overhead.

Proposal:

- This was waiting for implementation since fall.
- Implement this with the SimHit general update by end of May.

VXD (mis-)alignment

Alignment implementation

- Hierarchy of alignable structures, corresponding to subdetector - (half-)layer - ladder - sensor structure. linked with transformation matrices
- Properties of alignable structures, such as whether they are rigid bodies or deformable structures, and how they calculate adjustments to spatial transformations based on alignment parameters (and position).
- Code to read/write (mis-)alignment data files.
- System to shift coordinates of clusters based on misalignment information
- System to apply geometry modificatons to coordinate transforms to reflect alignment information.

The `AlignmentGeometry` class

- The `AlignmentGeometry` class provides access to data on alignment hierarchy and on individual alignables.
- It is implemented as a singleton and is built together with VXD geometry.

Alignment Hierarchy

- The alignment hierarchy is implemented as a `std::map<AlignableID, AlignmentLink>`
- Alignable IDs *are not* the VxdIDs, since the structure may deviate from the layer/ladder/sensor scheme. Strings are used as IDs instead, and only for sensors the IDs are guaranteed to be string representations of the VxdIDs.
- The alignment link contains data on mother of the alignable, including the baseline transform, and IDs of daughters, if any.
- The structure is easily browsable in both directions.
- Initialization takes place in the geometry creator.

Alignables

- The list of alignables is a `std::map<AlignableID, Alignable>`.
- The `Alignable` class holds current displacement/alignment/misalignment parameters, and methods to construct a 3D tranform based on alignable type.
- The class features transforms from local to global coordinates (and v.v.), so that position-dependent transformations for deformable structures can be supported.
- The list is initialized in the geometry creator based on the displacement xml file.
- Alignment or misalignment modules fill the corresponding data.

Misalignment

I will spend a second on misalignment implementation.

The Misalignment module

- The module's basic task is to provide input of misalignment data, either from an xml file or randomly simulated.
- These data are input to Alignables.
- A method providing a (possibly position-dependent) 3D transform) will be provided for each sensitive plane and retrieved by sensor VxdID via GeoCache.

RecoHit construtor

- The RecoHit constructor will adapt the position of the generating cluster based on the misalignment information in the GeoCache and direction of the generating MC track (taken from a TrueHit).

(Mis-)alignment - outlook

Outlook:

- The code is ready for submitting.
- Potential use cases not clear:
  - Misalignment scenarios
  - Requirements from alignment (support for calculation of global derivatives etc.)

  ... need to crystallize once the basic version is available.

VXD conditions database

Testbeam lesson:

- We need a database to manage run data.
  - There are a lot of data to keep for each run file.
  - The existing basf2 doesn't support handling of large amount of run-dependent data.
- The database can also be used to record processing tasks and manage processing settings.
- We need to tune the data organization:
  - Unclear performance issues with bulky data (pixel noises etc.)
  - Unclear feasibility of some use-cases (is it practical to retrieve settings for steering files?)

VXD conditions database (cont'd)

Current status:

- Not only still missing, but still a long way to go.
- We have database interfaces to basf2 from Jasmine - currently close to unsupported
- The database group people are more occupied with gross database system design than with providing usable tools
- There is a proposal by PNNL group to provide "pocket" database implementation on a VM.
- However, they will make the implementation and want data models and all information to understand the working of a subdetector and its data.
- This seems to be currently the only way to go.
  - It will be only usable if the turnaround cycle is reasonably fast.
  - It takes a lot of work to assemble all the data they need, though we do understand how things should work.
- Alternative: Store things in ROOT files.
  - Object-level storage.
  - Data files easily shareable.

Conclusions

- VXD dataobjects will undergo a series of important changes by end of May. Think about your data!
- Basic implementation of VXD support for (mis-)alignment is ready.
- Missing database support is becoming a serious issue, not only for the testbeam, but also for Belle II operation.

Thanks

Thank you for attention.