Summary

Summary of SiD Tracking Discussions, May 7-11, 2007 at SLAC

This is a summary of some issues that we decided or agreed to leave open. A separate page contains a list of work packages.

The summary is written by Rob Kutschke and me personal comments are in italics.

Persistency

- We will only persist the official LCIO objects: SimTrackerHit, RawTrackerHit, TrackerHit and the reconstructed track class (don't know its proper name).
- We will work with LCIO to get small changes made to these objects, but not big changes. We will shoehorn our objects into the defined persistent objects.
- We will not support persistency for other information. Most, but not all, of this information can easily be recomputed.
    - We will not persist an organized container of RawTrackerHits.
    - We will not persist clusters.
- We anticipate that a future evolution of the reconstructed track class (perhaps it will be spelled trajectory, not track), will allow us to persist residuals.

I view this as much too confining. We must be able to persist anything we put into the event, even if that information is not useful to the other collaborations. I do agree that track objects and all those downstream of them should be compatible with those used by other collaborations.

Treatment of Real Data

For now we are not considering the use case that we will (hopefully) someday use this framework to analyze data from a testbeam or from a real experiment. Therefore the hit and cluster classes all contain methods that return MC truth information. We did not define separate datahit classes and mchit classes in which the datahit classes are free from MC information.

New Geometry

The new geometry is well underway but not yet finished. This is the geometry that can describe the detector made of wafers. It can also describe a detector made of cylinders.

RawTrackerHit

We will get LCIO to approve several mods to the RawTrackerHit class and use it as the persistent representation of a single hit pixel or strip.

Needed Changes

- getDetectorElement(): Gives access to all geometry info.
- getCellID(): For us this will return a channel ID on the wafer, not a global ID, although the thing persisted will need to be a global ID.
- return a List of SimTrackerHits not a single SimTrackerHit.

Consequences:

- Proper merging of backgrounds can only be done at the SimTrackerHit Level.
- Merging can be done at the RawTrackerHit level but only using digitized pulse heights.
- The true pulse height (non-digitized) is never available to the user.
- The breakdown of which track contributed how much to the pulse height is never available to the user.
- The amount of electronic noise (+ or -) in the pulse height is not available to the user.

Recommendation:

People who write RawTrackerHit creators must be reminded to sum contributions before digitization. There is no support for this is in the class system we have specfied so they will need to do it themselves.

Clusters

- There are two cluster options on the table. We have not yet made a decision on this.
    - A class like Rob's or Rich's stripped down version.
    - Something like Dima suggested.
- We will add some organized container of clusters to the event. Perhaps along the model Rob suggested, perhaps a different one. If we go Dima's model, the container will hold ClusterData.
- If we go with Dima's model, should we also add a parallel container of TrackCluster objects instantiated with a null helix?
- There are no plans to persist clusters.
- We do not have a solution to the bookkeeping of which hits remain available for subsequent pattern recognition. What ever solution we choose, it will not modify clusters that are already in the event.

Track Fitters

There are a several track fitters in various states of readiness:

- Norman's TRF.
  - Still needs code to load its geometry from the org.lcsim geometry and code to load its hits from org.lcsim clusters.
- Nick's, weight matrix fitter.
  - Works on the cylindrical geometry, but not yet on the wafer geometry.
- Caroline's Kalman filter used for muon tracking and could be extended for vertex+tracker.
- Kfitter+SOD Tracker - there is a kalman filter in there but I don't know exactly where.
- We do not have proper output classes for any of these.

Track Classes

We need improved Track classes:

- Report track parameters and covarinace matrix valid in different regions
  - Example regions:
    - Innermost hit.
    - Outermost hit.
    - Point of closest approach to beam line.
    - Impact point on calorimeter.
    - Second innermost hit (useful if vertex pattern recognition says that the innermost hit is spurious).
- Access to residuals for hit rejection. (Need not be persisted).

TrackerHits

- We plan to persist TrackerHits.
  - They have ceased being useful objects for anything except graphics.
  - They are now the product of pattern recognition and track fitting, not the input to these codes, as originally envisioned.
- We intend to modify TrackerHit to point to cluster information, in order to provide access to the geometry.
  - However we do not plan to persist the clusters. Therefore it is not possible to access geometry from TrackerHits in persisted events. Therefore it is not possible to run one job to create the input to pattern recognition and run many subsequent jobs to develop pattern recognition codes.
- We plan to shoehorn clusters into TrackerHits? We plan to shoehorm TRF hits into TrackerHits.
  - Is this plan self consistent?
  - Maybe, if we add a type field to TrackerHit - which we probably need to do anyway.

Cheaters

A variety of pattern recognition cheaters exist. So far as I know they only work in the barrel and only work for the cylindrical tracker geometry.

Configuration and Metadata

- We have no mechanism for run time configuration. At many points in the discussion we noted that we need one.
- LCIO allows metadata to be associated with each object stored in the event header. We can and should make use of this to describe how the objects were made: versions of code, run time configuration information.

Old Style Geometries

There are many existing lcio files containing events simulated with the cylindrical vertex and tracking detectors. It will still be possible to read these files and instantiate the geometry representation. Old code will continue to work. Pattern recognition code developed for the planar detectors is unlikely to work with this old files.

How did we do with our Goals?

Our goals stated on Monday were:

1. Define classes for tracking infrastructure
   - Flesh out Rob's diagram: http://docdb.fnal.gov/ILC-public/DocDB/ShowDocument?docid=402
   - Make sure the relevant ideas from existing code are included.
2. Define a cheater for SimTrackHit to whatever it is that is used as input for pattern recognition.
   - Rob will do this once the endpoint is defined.
3. Re: links back and forth between Geometry and TrackerHits
   - Do we need full bidirectionality, as Jeremy has suggested, or is one way enough.
4. Re: Forward geometry.
   - Can Hans present a road map?
5. How does the above affect tracking algorithms?
   - *This was written in response to Rob's comments that we need to resolve:*
     - *Cylindrical vs polyhedral issue.*
     - *Version control.*

Here is how we did:

1. We have narrowed down the options considerably. The main remaining choice is whether to go with Dima's option for clusters or with something else.
2. This will be done as soon as 1) is implemented.
3. This was not decided. Most of us agree that there should be a clear distinction between event scope information and run scope or job scope information. So access to hits should be via the event no the geometry system. Jeremy feels that the correct method to access the events is via a hasReadOut() method implemented on elements of the geometry system.
    - *I strongly prefer the first option since I believe it is important to segregte event scope information from job scope information.*
4. Hans prepared material but we did not get to it. The material is attached.
5. We made progress on the cylindrical vs polyhedral issue by reducing the scope of disagreement: see Monday. We did not make progress on version control.