



Calorimeter Assisted Track Finder

Tracking Infrastructure

Dmitry Onoprienko

Kansas State University

Linear Collider Workshop 2007
May 30 - June 3, 2007. Hamburg, Germany

Why ?

Calorimeter assisted track finder package has been developed primarily for SiD concept studies → based on [org.lcsim framework](#)

[org.lcsim](#) event data model is LCIO-compatible (not identical)

Once complexity of detector options and realism of studies had increased, tracking infrastructure became a bottleneck

TrackerHit - measurement, input to pattern recognition

- no way to represent non-point-like object like silicon strip
- no access to geometry
- no well defined way to access MC truth
- global coordinates only
- no convenient way to introduce corrections once the hit is assigned to track

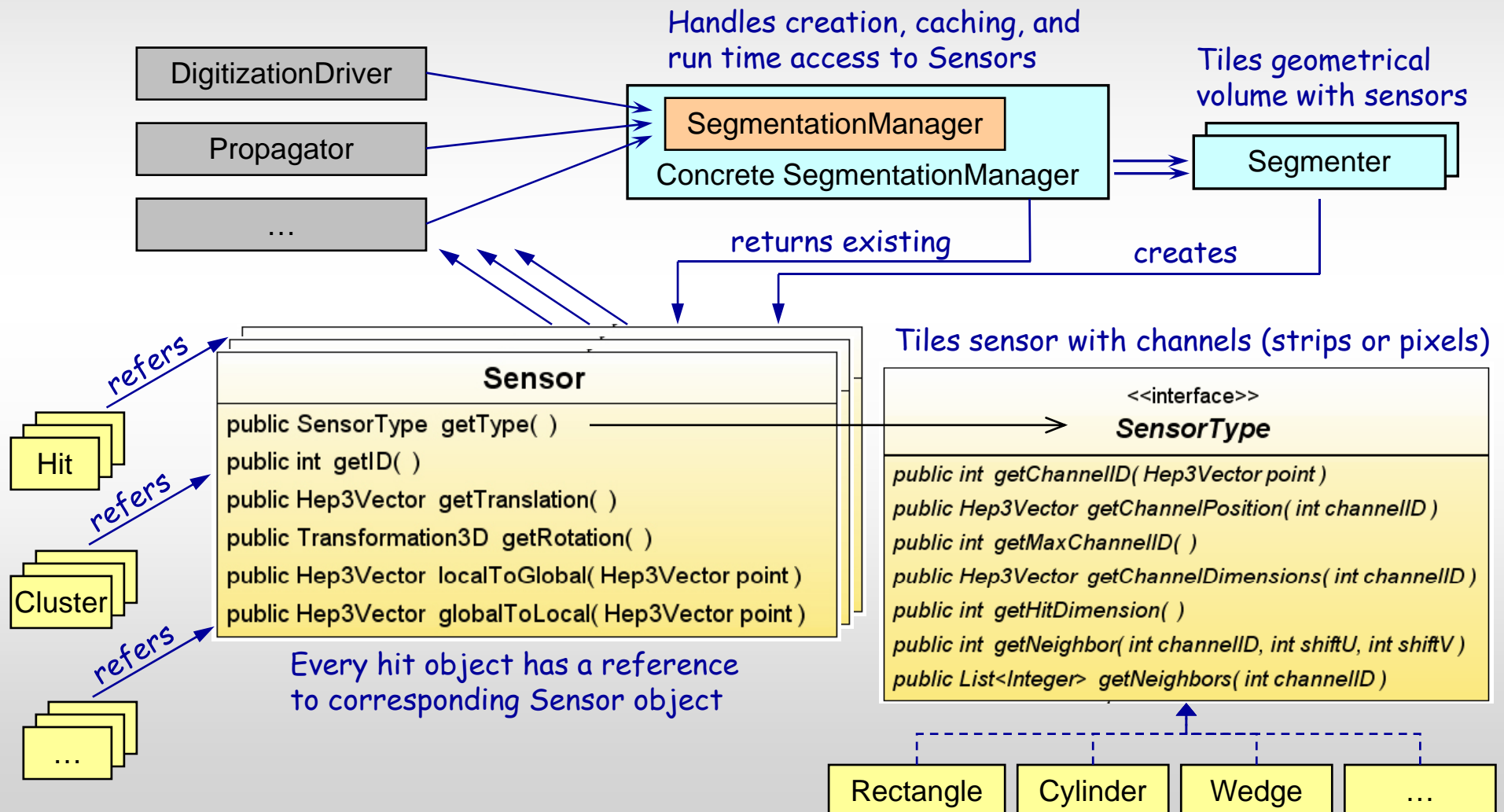
Track, RawTrackerHit, others have their own problems...

Multiple custom extensions have been introduced by algorithm developers, resulting in lots of incompatible packages.

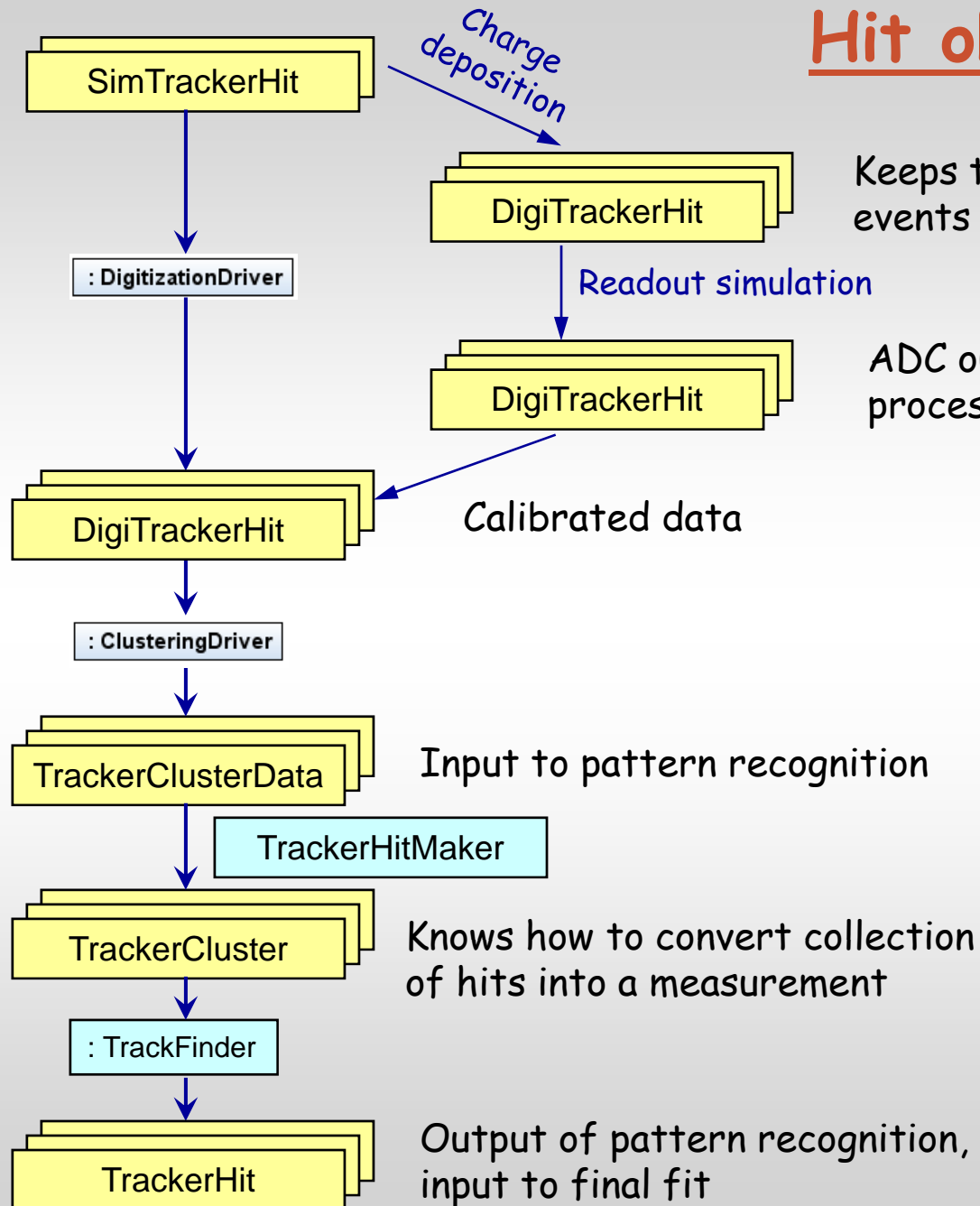
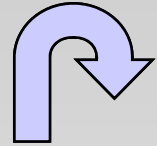
Lightweight interface to geometry

Designed to support virtual segmentation at run time.

Each readout channel is identified by a reference to Sensor and a channel ID.



Hit objects



Keeps true pulse height (collected charge) - events can be overlaid at this stage

ADC output - beam test data processing path merges at this stage

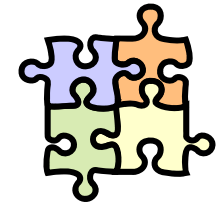
All intermediate objects between **SimTrackerHit** and **TrackerHit** are kept in **HashMap<Sensor,List<Object>>** collections accessible through **EventHeader** object

Sensors can be grouped to provide additional structure

Downstream from **SimTrackerHit**, access to MC truth is not built into interfaces, but provided by separate utility classes

Interface DigiTrackerHit

DigiTrackerHit



All Superinterfaces:

java.lang.Comparable<[DigiTrackerHit](#)>

All Known Implementing Classes:

[DigiTrackerHitAdapter](#), [DigiTrackerHitComposite](#), [DigiTrackerHitElemental](#)

Representation of signal from a single tracker channel (pixel or strip).

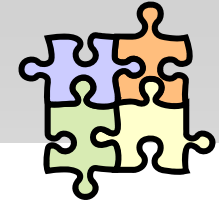
Method Summary

int	getChannel ()	Returns channel ID on the sensor.
java.util.List< DigiTrackerHit >	getElementalHits ()	Returns a list of underlying elemental hits.
MCParticle	getMCParticle ()	Returns MCParticle that produced the hit.
Sensor	getSensor ()	Returns Sensor object this hit belongs to.
double	getSignal ()	Returns signal in the channel.
double	getTime ()	Returns time associated with the hit.
boolean	isComposite ()	Returns true if the hit is a superposition of more than one elemental hit.

Methods inherited from interface java.lang.Comparable

compareTo

TrackerClusterData



org.lcsim.contrib.onoprien.tracking.hit

Interface TrackerClusterData

All Known Subinterfaces:

[TrackerCluster](#)

All Known Implementing Classes:

[TrackerClusterBasic](#), [TrackerClusterDataBasic](#)

```
public interface TrackerClusterData
```

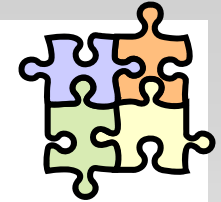
Collection of `DigiTrackerHits` that cannot be unambiguously separated. Clusters are independent in a sense that a track crossing the sensor only contributes to one cluster. But it is possible for several tracks to contribute to one cluster.

Method Summary

<code>java.util.List<DigiTrackerHit></code>	<code>getDigiHits()</code> Get list of <code>DigiTrackerHits</code> that compose the cluster, sorted by channel ID.
<code>Sensor</code>	<code>getSensor()</code> Returns the <code>Sensor</code> object associated with this cluster.
<code>double</code>	<code>getSignal()</code> Returns combined signal of all <code>DigiTrackerHits</code> in the cluster.
<code>double</code>	<code>getTime()</code> Returns signal-weighted average time for all <code>DigiTrackerHits</code> in the cluster.

Interface TrackerCluster

TrackerCluster



All Superinterfaces:

[TrackerClusterData](#)

All Known Implementing Classes:

[TrackerClusterBasic](#)Adds behavior to **TrackerClusterData**

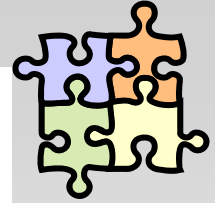
```
public interface TrackerCluster
  extends TrackerClusterData
```

Method Summary

<code>java.util.List<TrackerHit></code>	getTrackerHits () Returns a list of TrackerHits produced from this cluster.
<code>TrackerHit</code>	makeTrackerHit () Creates TrackerHit from this cluster without using any trajectory information.
<code>TrackerHit</code>	makeTrackerHit (TrackerHit hit) Updates an existing TrackerHit without using any trajectory information.
<code>TrackerHit</code>	makeTrackerHit (TrackPoint trackPoint) Creates TrackerHit from this cluster.
<code>TrackerHit</code>	makeTrackerHit (TrackPoint trackPoint, TrackerHit hit) Updates an existing TrackerHit.

Methods inherited from interface org.lcsim.contrib.onoprien.tracking.hit.[TrackerClusterData](#)[getDigiHits](#), [getSensor](#), [getSignal](#), [getTime](#)

TrackerHit



org.lcsim.contrib.onoprien.tracking.hit

Interface TrackerHit

All Known Implementing Classes:

[TrackerHitAdapter](#), [TrackerHitBasic](#), [TrackerHitPoint](#)

```
public interface TrackerHit
```

Tracker hit object to be used by a fitter. `TrackerHit` can represent either a point-like (pixel) or a segment-like (strip) object. Each hit has a local reference frame (u,v,w) associated with it. U is the measurement direction, V is along the length of the strip, $W = U \times V$.

Method Summary

TrackerCluster	getCluster() Points back to <code>TrackerCluster</code> that produced this hit.
hep.physics.matrix.SymmetricMatrix	getCovMatrix() Returns covariance matrix of the hit in global reference frame.
double	getLength() Returns length of the segment defining the hit.
hep.physics.matrix.SymmetricMatrix	getLocalCovMatrix() Returns covariance matrix in local frame of the Sensor .
hep.physics.vec.Hep3Vector	getLocalPosition() Returns position of the hit in local reference frame of the Sensor .
SpacePointVector	getLocalSegment() Returns <code>SpacePointVector</code> pointing from start to end of the segment defining the hit in the local reference frame.
hep.physics.vec.Hep3Vector	getPosition() Returns position of the hit in global reference frame.
SpacePointVector	getSegment() Returns <code>SpacePointVector</code> pointing from start to end of the segment defining the hit in the global reference frame.
Sensor	getSensor() Returns Sensor object for this hit.

The Rest

Many classes are tailored specifically to calorimeter assisted tracking algorithm.

Track - list of **TrackNodes**, each consisting of **TrackPoint** (local trajectory parameterization) and **TrackAnchor** (knows how to calculate residuals with respect to **TrackPoint**, used in fitting).

Both **TrackerHit** and **MipStub** implementing classes also implement **TrackAnchor**.

Given **TrackNode**, propagator can get a list of **TrackAnchor** candidates for attachment to the **Track** (currently based on layer numbers, the infrastructure allows more flexible grouping).

Package status - available functionality

In the contributed area of org.lcsim CVS repository:

`org.lcsim.contrib.onoprien.tracking`

Basic implementations available for all discussed elements. Some are untested.

Segmentation:

Segmenters based on DetectorElement

SensorType: Cylinder, Rectangle, Ring, Wedge, Hexagon.

Digitization:

SimTrackerHit smearing

Interface to Nick Sinev's pixel digitization package

Interface to Tim Nelson's strip digitization package

Clustering:

Nearest Neighbor

TrackerHitMaker:

SimpleCentroid

Track Finder - work in progress