ORCA Tutorial

CERN, Friday 30 January 2004

Muon Reconstruction with ORCA

Stefano Lacaprara

Stefano.Lacaprara@pd.infn.it

INFN and Padova University

Outline

- General introduction,
  - Goal of this tutorial,
  - Muon related packages,
  - Who and where,
  - Muon general BuildFile,
- Ex. 1: how to access and use Muon tracks,
  - which reconstructor?,
  - access track,
  - access track info,
  - apply vertex constraint,
  - homework,
- Ex. 2: access to Geometry, SimHits, Digis and RecHits ,
  - access DT geometry information,
  - access RPC SimHits,
  - access CSC Digis,
  - access RecHits for all 3 systems,
  - homework,

Introduction: Goals

- Goals of this tutorial
  - How to access and use muon reconstructed tracks
  - how to apply a vertex (beam-spot) constraint to muon tracks
  - how to access geometry information of DT, CSC and RPC
  - how to access SimHits, Digis and Rechits for the 3 systems
- This tutorial will not cover:
  - Algorithmic details about how the reconstruction is performed
  - Muon isolation for offline analysis (work in progress)

    (will give an HLT example as appendix)
  - L1 simulation and HLT reconstruction (refer to HLT package)

Introduction: Muon packages

There are 3 packages related to Muon reconstruction:

- Muon
  - Responsible for detector description of DT, CSC and RPC (reading from DDD), OO modelling of detectors
  - access to SimHits, simulation and access to Digis
  - local reconstruction (hits and segments)
  - stand alone reconstruction, using only muon detectors
- MuonReco
  - Responsible for global reconstruction (with tracker)
  - HLT reconstruction (L2, L3), using L1 trigger input
  - Muon isolation (using cal, pixel and tracker) for HLT (and off-line)
- MuonAnalysis
  - Framework for analysis for the PRS-μ group

Introduction: People

Who can answer to your questions

- Stefano Lacaprara all `Muon`, DT, reconstruction;
- Nicola Amapane DT, isolation;
- Tim Cox, Rick Wilkinson CSC;
- Artur Kalinosky, Giacomo Bruno RPC;
- Norbert Neumeister global muon reconstruction;

Don't forget the general ORCA mailing list:

`cms-orca-feedback@cern.ch`

Documentation:

- ORCA UserGuide large effort to write and update the DT part
- for CSC CMS Note 2001/013
- for local reconstruction CMS Note 2002/043 (DT part obsolete since december...)
- for track reconstruction DAQ TDR vol II (mostly HLT, but useful)
- for class interface ORCA Reference Manual (doxygen)

Introduction: BuildFile

What to put into your BuildFile

MuonRecHitReader : For RecHit reconstruction on DT, CSC and RPC;

MuonDigiReader : For reading on the 3 sub-detectors: no RecHit reconstruction is guaranteed

MuonSimHitReader : For SimHit reading on the 3 sub-detectors: no RecHit reconstruction and/or Digi simulation is guaranteed

⋆ Identical groups exist separately for each sub-system, replacing `Muon` with `MB, ME, MRpc`, respectively for DT (barrel), CSC (endcap) and RPC systems;

MuonInternalReco : For muon track reconstruction with standalone detector;

Introduction: BuildFile (II)

MuonReconstruction : For global reconstruction (with tracker);

MuonIsolation : For isolation (HLT specific);

⋆ **Warning** ⋆ : if you want to use `GlobalMuonReconstruction` (see after), also the MuonInternalReco group must be included, since Global reco uses StandAlone (A new group will appear in forthcoming ORCA releases)

How to use the groups:

```
<environment>
  <group name=MuonInternalReco>
  <use Muon>

  ...
</environment>
```

Ex 1.1: Muon tracks

- What is a **Muon** track? There is not (yet) a specific ``MuonTrack'' class, so a Muon track is just a `RecTrack`, just as the tracker ones.
- Can use what you already learned from Tracker tutorial!!
- Which reconstructor are available?
  - ⋆ **StandAloneMuonReconstructor** uses only Muon detectors, no tracker, no IP constraint: BuildFile group MuonInternalReco (Muon)
  - ⋆ **GlobalMuonReconstruction** uses Muon **and** Tracker. MuonInternalReco (Muon) and MuonReconstruction (MuonReco). Uses StandAloneμReco as "seed".
  - ⋆ **L2MuonReconstruction** - **L3MuonReconstruction** HLT reconstruction: seed L2 from L1, L3 from L2;
  - ► **Warning** you are highly discouraged to use directly L2/3MuonReconstruction: they are meant for HLT only, not for off-line. To use HLT, please use the new HLT package.

Ex 1.1: StandAloneμ

- login into `lxplus`, setup an ORCA_7_6_1 working area and check out MuonAnalisys/MuonTutorial/

```
scram project ORCA ORCA_7_6_1
cd ORCA_7_6_1/src
cmscvsroot ORCA
cvs co -r Tutorial0104
    MuonAnalysis/MuonTutorial
cd MuonAnalysis/MuonTutorial
```

- in `test`: dumpMuonTrack.cpp (register the Observer to CARF), BuildFile, setup script;
- in `src`: concrete code `DumpMuonTrack` (Observer) (to be written by you!)
- in test: source `setup.csh|sh` : fetch the Pool catalog and write a .orcarc ;

Ex 1.1: StandAloneμ (2)

- access to TTrack via
```
AutoRecCollection<TTrack, G3EventProxy*>
    muons(``StandAloneMuonReconstruction'');
```
- WARNING for propagation through iron, we still use `GEANE`, which need to be initialized!
  `setenv GEANEUSED TRUE` or
  `export GEANEUSED=TRUE`
  if not, you'll get an exception (and no muon)
- Q: Wait:a track is made from hits and segments (more after). Who is responsible to reconstruct them?
  A: NOT YOU! they are reconstucted *on demand*, don't worry!
- in the BuildFile you need only the `MuonInternalReco` group
- template in src/MuonTrackDump.cc_ex1.1, solution in MuonTrackDump.cc_sol1.1

Ex 1.1: Globalμ

- This reconstructor uses both Muon and Tracker hits, starting from StandAlone reconstructed muons;
- The best approximation of an *off-line* reconstructed muon today;
- If you want to use `GlobalMuonReconstuction`, just add a second call of `DumpMuonTrack::printMuonTracks` with different argument;
- Note the the innermost state is now very close to the IP, while was far before: we are using tracker.
- You need to add MuonReconstruction in the BuildFile (also CaloRecHitReader (Calorimetry))

Ex 1.2: access μ hits

- A RecTrack has a collection of `TrajectoryMeasurement`
- Each TM has:

  predicted state: as predicted by the extrapolation from the previous state;

  updated state: when the info about the actual hit (if found) is used;

  RecHit: the hit (of any kind) found and used to update: if not found, an invalid one is set, and updated==predicted
- Ex: access and dump TM's, predicted, updated state and recHit.
- template in src/MuonTrackDump.cc_ex1.2, solution in MuonTrackDump.cc_sol1.2

Ex 1.3: apply IP constraint

- None of this reconstruction uses the beam spot constraint
- Very useful for StandAlone reco, more delicate for Global one, where one should reconstruct a vertex, not use the IP

Two different solution:

▶ MuonUpdatorAtVertex or

▶ MuonFitWithVertexConstraint

- MuonUpdatorAtVertex is in Muon/MuonTrackFinder
- It performs the constraint to IP, a given Vertex or a vertex at a given position
- have a method which returns a `MuonVertexMeasurement`, with the measurement at vertex
- Does not update the input RecTrack, nor create a new one

Ex 1.3: apply IP constraint (2)

- MuonFitWithVertexConstraint is in MuonReco/MuonReconstruction
- As before, constraint with IP or generic Vertex;
- have a method which returns a vector<RecTrack>, which uses all hit of input RecTrack plus the vertex one;
- Does not update the input RecTrack, but create a new one(s)
- template in src/MuonTrackDump.cc_ex1.3, solution in MuonTrackDump.cc_sol1.3

Homework:

get the primary vertex using `HistogramPVFromHits` (pixel), then apply the vtx constraint to that vtx and not to the beam spot

Ex 2: Access detector info

- Goals:
  - access geometry information
  - access SimHits
  - access Digis
  - access RecHits
- 3 subsystems in Muon: DT, CSC, RPC
- geometry access and navigation only partially uniform
- access to SimHits, Digis and RecHits uniform thanks to common interface (CommonDet, shared also with Tracker)
- will give example for all the 3 subsystem

Ex 2: Geometry

- A reconstruction geometry is implemented in ORCA
- It does **not** describe all the details of the detector
- It is focused on reconstruction, and is designed to be as simple as possible
- It is different from the detector description used in OSCAR (simulation), which describe as many detail as possible
- Different application have different needs, thus "different" geometry
- Both geometries come from the same database!! Oscar one describe much more detail than Orca's one.
- eg. in ORCA only detector are described, not passive material

Ex 2.1: DT geometry

Goal Get number of wires for each DT chamber in central wheel

Quick description of DT geometry:

- CMSMuonBarrel: the whole system
- has: 4 `DetLayers`, cylinders, aka stations, used for navigation during track finding
- MuBarChamber: access to 4D segments (RecHits)
- each chamber has 3 (2) MuBarSL: access to 2D segments (RecHits)
- each SL has 4 MuBarLayer: access to SimHits, Digis, Hits (RecHit)
- each Layer has $\mathcal{O}(100)$ MuBarWire
- Possible to navigate from top to bottom in different ways

Ex 2.1: DT geometry (2)

template in src/MuonRecHitDump.cc_ex2.1,
solution in MuonRecHitDump.cc_sol2.1

- Access to Dt Geometry via `Singleton`:

```
MuBarrelSetup* mbSetup =
  Singleton<MuBarrelSetup>::instance();
const MuBarDetectorMap& map =
  mbSetup->map();
```

- get all MuBarChamber via MuBarDetectorMap::chambers()
- for each chamber, loop through all Sl, then all layer and count wires
- each chamber (as well as SL and Layer) have a "name", MuBarChamberId, via id() method. MuBarChamberId::wheel() gives the wheel of chamber ($[-2, +2]$)
- print nWires only if wheel()==0

Ex 2.2: Get RPC SimHits

template in src/MuonRecHitDump.cc_ex2.2,
solution in MuonRecHitDump.cc_sol2.2

- For all system, the SimHits are accessible via SimDet
- A SimDet is accessible from a DetUnit
- There are different ways to get all DetUnits
- A possibility, is to get them via the DetLayers, which have a methods ::detUnits() Not necessarily the more efficient way, eg for DT
- To get DetLayers:
  - Get Rpc setup
  - Get simplyfied geometry from setup
  - Get DetLayers

```
MRpcSetUp* mrpcSetup =
  Singleton<MRpcSetUp>::instance();
CMSMuonRpc* cmsrpc=mrpcSetup->CMSMRpc();
vector<DetLayer*> dls=cmsrpc->allLayers();
```

Ex 2.2: Get RPC SimHits (2)

- Other methods can be
  - Get directly all DetUnit via MRpcMap, in turn accessed as a singleton
  - Navigate through the Rpc geometry in an other way:
    - get all MRpcDetectors from setup
    - get all MRpcChamber from MRpcDetectors (MRpcDetectors is composed of MRpcChambers, so use allComponents() method)
    - get all DetUnit from MRpcChamber (again as components): there are 1 to 3 DetUnit for each chamber, as in reality “chambers”are made by 1 to 3 “detectors”
- Once we have the DetUnit, we should get the SimDet
- Check if the SimDet is really present (in real world there are not such a thing as a SimWhatever...)
- Finally, the SimDet can give us the SimHits

Ex 2.3: Get CSC Digis

template in src/MuonRecHitDump.cc_ex2.3,
solution in MuonRecHitDump.cc_sol2.3
CSC geometry

- CmsMuonEndcap: collection of DetLayers (disk)
- MuEndcapSystem (all), MuEndcap (2), MuEndStation (the disks 4+4), MuEndRing (2 or 4)
- MuEndChamber: provides 4d segments (RecHit)
- MuEndLayer (6 for each chamber), provides SimHits, Digis (separately for Wires and Strips) and hits (aka cluster) (RecHit)
- To get the digis we must get all the MuBarLayers
- Completely different approach (better?)
- Use MuEndLayerIterator

Ex 2.3: Get CSC Digis (2)

- As before, get the Endcap setup MuEndSetUp via singleton
- Get MuEndSystem from the setup

```
MuEndcapSetUp * setup =
  Singleton<MuEndcapSetUp>::instance();
MuEndcapSystem * endcapSystem =
  setup->MEndcap();
```

- Instantiate a MuEndLayerIterator with the system (the constructor get the system as argument)
- Iterate over all the layer
- Get the Digis
- That's it!

Ex 2.4: Print all RecHits

template in src/MuonRecHitDump.cc_ex2.4,
solution in MuonRecHitDump.cc_sol2.4

C'mon, it's the last exercise!

- The goal os to print all the RecHits of the 3 systems
- For RPC, the MRpcDetUnit provide the only RecHits
- For the CSC and DT, there is a hierarchy of RecHits, as there is a hierarchy of Dets
- CSC: Chamber provides segments, Layers provide Hits
- DT: Chamber provide 4D segments, SL 2D segments, layer hits
- Moreover, the segments are composed RecHits, made of RecHits
- Let's concentrate on highest level RecHits, ie segments for DT and CSC

Ex 2.4: Print all RecHits (2)

- We already know how to access DT chamber, CSC chamber and Rpc DetUnits
- all these object are `Det`, so have a common interface
- To get RecHits from a Det, just use Det::recHits()
- Q: wait! I know that these 2 types of RecHits are different (segment vs hit). How can it be that they are all the same object (RecHit)??
- A: good question! They have the same interface, but have different information. Eg. only the segments have direction. When used in the Kalman fit, every RecHit provide information according to what it really is.
- Print position and direction of DT and CSC, and check that direction is not defined for RPC

Homework

- Repeat ex 2.1,2,3 for all 3 sub-systems
- Print also the low level RecHits of CSC and DT
  - Access them via appropriate Det (MuEndLayer, and MuBarSL/Layer, respectively)
  - Do the same also via the composite RecHit interface (namely via RecHit::recHits()). Do you get the same number of RH in both cases?
- Compare SimHits and RecHits to get residuals for all sub-system
- Get a reconstructed muon, get all the Dets which contribute with a RecHit (valid or not), and look if there are other RH in that Det, and how far

Conclusion

- This was a short and quick overview of Muon reconstruction software

Conclusion

- This was a short and quick overview of Muon reconstruction software
- Lot of details have been left out

Conclusion

- This was a short and quick overview of Muon reconstruction software
- Lot of details have been left out
- Do you think you could do better?

Conclusion

- This was a short and quick overview of Muon reconstruction software
- Lot of details have been left out
- Do you think you could do better?
- You have idea about major improving?

Conclusion

- This was a short and quick overview of Muon reconstruction software
- Lot of details have been left out
- Do you think you could do better?
- You have idea about major improving?
- There is a lot of room for improvement and for people willing to contribute!

Conclusion

- This was a short and quick overview of Muon reconstruction software
- Lot of details have been left out
- Do you think you could do better?
- You have idea about major improving?
- There is a lot of room for improvement and for people willing to contribute!
- Don't stay just a *user*, become a developer!

Conclusion

- This was a short and quick overview of Muon reconstruction software
- Lot of details have been left out
- Do you think you could do better?
- You have idea about major improving?
- There is a lot of room for improvement and for people willing to contribute!
- Don’t stay just a *user*, become a developer!

- from *Picked up for you* this week:

Conclusion

- This was a short and quick overview of Muon reconstruction software
- Lot of details have been left out
- Do you think you could do better?
- You have idea about major improving?
- There is a lot of room for improvement and for people willing to contribute!
- Don't stay just a *user*, become a developer!

- from *Picked up for you* this week:

"You must be the change you want to see in the world", M.K.Ghandi

Conclusion

- This was a short and quick overview of Muon reconstruction software
- Lot of details have been left out
- Do you think you could do better?
- You have idea about major improving?
- There is a lot of room for improvement and for people willing to contribute!
- Don't stay just a *user*, become a developer!

- from *Picked up for you* this week:

"You must be the change you want to see in the CMS"

Ex 1.4: Muon isolation

template in src/MuonTrackDump.cc_ex1.4,

solution in MuonTrackDump.cc_sol1.4

Warning: use also MuonTrackDump.h_ex1.4

uncomment MuonIsolation etc. in the BuildFile

recompile only `dumpMuonTrack` (scram b dumpMuonTrack)

⋆⋆⋆⋆ Warning ⋆⋆⋆⋆

This isolation is meant for HLT selection, not tuned for off-line analysis! Work on off-line isolation is in progress

- Example how to use HLT isolation with tracker
- Please refer to `HLT` package to apply selection on HLT (isolation included)

Ex 1.4: Muon isolation (2):

- Initialization: we need a MuonIsolation, the an extractor and an isolator
- The extractor fetch the information (here tracks), the isolator apply the algo on fetched object
- Techincal trick: the initialization must be done after a G3Setup have been dispatched (need to build tracker geometry): not possible in the obs c'tor Two solution
  - Have an observer of both `G3Setup` and `G3EventProxy` When G3Setup is dispatched, initialize isolation (ie inside upDate(G3Setup*)) (more correct);
  - Initialize isolation at the first event (done in solution)
- Must set strategy (with extractor, isolator and name ="TRACKER")
- Then simply ask to MuonIsolation if a given RecTrack isIsolated