Muon Software Tutorial

Rick Wilkinson
Caltech
The Basics

Q: Is there a Muon class?
A: No. A muon is just a RecTrack, the same class as the Tracker uses.

Q: What information can I get from a RecTrack?
A: All this…

GlobalPoint vertexPosition() const;
GlobalVector momentumAtVertex() const;
TrackCharge charge() const;
Measurement1D transverseImpactParameter() const;
vector<RecHit> recHits() const;
float normalisedChiSquared() const;
TrajectoryStateOnSurface outermostState() const;
… and more.
Creating Standalone Muon Tracks

Q: How do I get muon tracks from the muon detectors?
A: $\text{AutoRecCollection<TTrack, G3EventProxy*> muons("MuonReconstructor");}$

Q: Wait! What’s this about TTracks? I thought they were RecTracks?
A: TTrack inherits from RecTrack, and interfaces it to the reconstruction framework COBRA. It’s nothing you have to worry about.

Q: Is there anything special I need to do?
A: You need to set up GEANE to propagate the tracks through the iron!
   $\text{setenv GEANEUSED TRUE}$

Q: What about the segments, clusters, digis, etc.?
A: They’ll be reconstructed $\text{on-demand.}$
Looping Over Muons

Use the AutoRecCollection, just like Stephan showed you:

```cpp
AutoRecCollection<TTrack, G3EventProxy*> muons("MuonReconstructor");
AutoRecCollection<TTrack, G3EventProxy*>::iterator muonItr;

for(muonItr = muons.begin(); muonItr != muons.end(); ++muonItr) {
    cout << "the charge of this muon is " << (*muonItr)->charge() << endl;
    float pt = (*muonItr)->momentumAtVertex().perp();
    cout << "the pT of this muon is " << pt << endl;
}
```

Q: What RecHits are used in the track?

A: In the muon barrel, the tracks fit to track segments. In the endcaps, the tracks fit to the individual 2D points, which are selected using track segments. More on all that later.
Other Options

Q: What muon Reconstructors are there?
A: MuonReconstructor: -- standalone, find its own seeds
    L2MuonReconstructor: -- standalone, starts from L1 seed
    L3MuonReconstructor: -- with inner tracker hits, starts from L2 track

Q: Are these tracks vertex constrained?
A: No. To add a vertex constraint to the track, you must do a Kalman Filter update of the track by hand, adding the vertex point.

```cpp
// define beam spot with error
// sigma(x) = sigma(z) = 0.1 mm
// sigma(z) = 5.3 cm
GlobalPoint p(0.0,0.0,0.0);
GlobalError e(0.0001, 0., 0.0001, 0., 0., 28.09);
TrajectoryStateOnSurface traj = muonItr->innermostState();
MuonUpdatorAtVertex updator(p,e);
MuonVertexMeasurement vm = updator.update(traj);
TrajectoryStateOnSurface traj_trak = vm.stateAtTracker();
```
First Exercise

The first exercise in the hands-on session will be to create a job which loops over muons and prints them out. Check out the MuonAnalysis package, go to MuonTutorial/test, and modify, build, and run MuonTracks.

If you get brave, feel free to play around by:
• extracting information from the RecTrack
• extracting the RecHits and looking at them
• looking at the TrajectoryStateOnSurface at various points

If you want to try the L2MuonReconstructor or the L3MuonReconstructor, you’ll need to uncomment-out some packages in the BuildFile I provided. Try it!

A solution is provided in MuonTracksSolution.cpp
Muon Subdetectors

For precise position measurements:
- Drift Tubes (DT)
  - barrel
- Cathode Strip Chambers (CSC)
  - endcaps

For precise time measurement:
- Resistive Plate Chambers (RPC)
  - barrel and endcap
  - Primarily for triggering
Drift Tubes (DT) - Barrel
Cathode Strip Chamber (CSC) - Endcaps

- Wires
- Strips
- Muon
- Cathode wires
- Cathode
- Induced charge
- Cathode with strips
- Avalanche
- Wires
- Plane cathode
Resistive Plate Chambers (RPC)
CSC Geometry

- CmsMuonEndcap
  - Groups chambers into tracking DetLayers
- MuEndcapSystem
- MuEndcap (2)
- MuEndStation (4)
- MuEndRing (4 in ME1, 2 in other stations)
- MuEndChamber (18 or 36)
  - Stores MuEndSegments
- MuEndLayer (6 per chamber)
  - Stores SimHits
  - Stores MuEndWireDigis
  - Stores MuEndStripDigis
  - Stores RecHits (clusters)

MuEndChamberSpecs:
- Information common to a type of chamber (gas gains, shape, wire grouping...)

MuEndLayerGeometry:
- Handles the math of tilted gangs of wires crossing trapezoidal strips
DT and RPC Geometry
(note the different design philosophies)

• CMSMuonBarrel
• MuBarChamber
  – Stores SimHits,
  – Stores Digis
  – Stores RecHits (segments)
• MuBarSL
  – 2 superlayers measure $\phi$
  – one measures $r$
• MuBarLayer
  – 4 layers in a superlayer
  – RecHits to be added here
• MuBarWire
• CMSMuonRPC
  – Groups into tracking layers
• MRpcDetector
  – MRpcGlobalReader stores all digis
• MRpcChamber
• MRpcDetUnit
  – 1, or sometimes 3 per chamber
  – Stores SimHits,
  – Can fetch Digis
  – Stores RecHits (clusters)
DetUnit is the detector class where the data is stored

LocalPoint toLocal(const GlobalPoint &);
GlobalPoint toGlobal(const LocalPoint &);
SimDet * simDet(); // can be used to get SimHits
DetUnit::RecHitContainer recHits();
Accessing Data: CSC

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

MuonEndLayerIterator layerItr(endcapSystem);
MuonEndLayer * layer;
while (layer = layerItr.next()) {
    vector<RecHit> recHits = layer->recHits();
    vector<MuEndWireDigi> wireDigis = layer->getWireDigis();
}
Accessing Data: DT

MuBarrelSetup * setup = Singleton<MuBarrelSetup>::instance();
CMSMuonBarrel * mb = setup->MBarrel();

for( int ilayer = 0; ilayer < 4; ++ilayer) {
    DetLayer * layer = mb->barrelLayers()[ilayer];
    for(int ichamber = 0; ichamber < layer->detUnits().size(); ++ichamber) {
        DetUnit * chamber = layer->detUnits()[ichamber];
        SimDet * simDet = chamber->simDet();
        if(simDet != 0) {
            SimDet::SimHitContainer simHits = simDet->simHits();
        }
    }
    DetUnit::RecHitContainer recHits = station->recHits();
}
MRpcDetector * mrpc = Singleton<MRpcSetUp>::instance()->getDetector();
vector<MRpcDigi> mrdigis = mrpc->giveDigis();
vector<DetUnit*> dets = Singleton<MRpcMap>::instance()->getDetUnits();
vector<DetUnit*>::iterator it;
for (it = dets.begin(); it != dets.end(); it++) {
    SimDet * simDet = (*it)->simDet();
    if(simDet != 0) {
        SimDet::SimHitContainer simHits = (*it)->simDet()->simHits();
    }
}
Second Exercise

The goal of the second exercise will be to write routines which print out:

• The positions of all DT segments, in global coordinates
• The first CSC strip digi in each layer
• The positions of all RPC SimHits, in global coordinates

Modify and build the HitsDigisRecHits program in MuonAnalysis/MuonTutorial/test.
Technical Overview of the L1 Muon Trigger Software

- MuEndWireDigi
- MuEndStripDigi
- MuBarDigi
- MRpcDigi

L1MuCSCTrigger
- L1MuCSCTrackStub
- L1CSCTrackFinder
- L1CSCTrack

L1MuDTTrig
- L1MuDTTrackSegPhi
- L1MuDTTrackSegEta
- L1MuDTTrackFinder
- L1MuDTTrack

L1GlobalMuonTrigger
- L1MuGMTCand

L1GlobalTrigger
Hand-waving Overview of the L1 Muon Trigger Software

- **CSC Digis (Strip + Wire)**
- **DT Digis**
- **RPC Digis**

**Local Trigger (LCT)**

- **CSC Regional Trigger**
- **DT Regional Trigger**
- **RPC Trigger**

**L1 Global Muon Trigger**
Muon Isolation

- Has algorithms for Calo, Pixel, and Tracker
- Has cuts hardcoded for given “nominal” efficiencies.

```cpp
MuonIsolation * isolation = new MuonIsolation();
MuIsoCaloExtractor * ce = new MuIsoCaloExtractor();
MuIsoCaloIsolator * ci = new MuIsoCaloIsolator();
    ci->setNominalEfficiency(0.97);
    isolation->setStrategy("CALO", ce, ci);

// … Later, in event loop over muons
    if(isolation->isIsolated(muon, "CALO")) {
```
Analysis

• For the old-fashioned standard ntuple-maker, look in MuonReco/MuonReconstruction/test/MuonReconstructionNtuple.cc
  – Great source of example code!

• TAGs can be created in the database, using COBRA
• TAGs can be used for:
  – Fast event selection
  – Plotting directly in Lizard
  – Making PAW ntuples
  – Making ROOT trees
Future Software Plans

• Speed things up
  – Most of the time spent propagating through iron

• Refactor code
  – Separate clustering & segment-building from detector & digitization

• Code review?
  – Unify interfaces between the three subdetectors

• Thermal neutron background
  – biggest problem is in the endcaps

• GEANT 4, DDD

• Continue to develop and refine geometry, digitization, and reconstruction.