SimTauCPLink

Towards performant $\tau$ reconstruction in HGCAL

TauCQM Meeting 20.11.2023

Andreas Gruber, Erica Brondolin, Marco Rovere
Motivation

• The τ is one of the most difficult particles in reconstruction – will be a good proxy for the overall reconstruction performance of HGCAL

• τ reconstruction can benefit from unprecedented fine granularity available in HGCAL

• It would be interesting to study if HGCAL allows for dedicated π⁰ reconstruction

Performance depending on variables shown, but in general we see a performance drop in HGCAL for now
The High Granularity Calorimeter (HGCAL)

• Imaging calorimeter with both lateral and longitudinal fine granularity

• Will replace the existing endcap detectors of CMS for the HL-LHC

• Designed both to withstand the high radiation and deal with a pileup of up to 200
HGCAL – two sections

• Electromagnetic (CEE):
  Active material: Silicon
  In three sensitive thicknesses: 300, 200, and 120 μm
  with an area of 1.18 cm$^2$ or 0.52 cm$^2$
  Passive: Copper, Tungsten and Lead absorbers
  26 layers: 26.3 $X_0$ and 1.73 $\lambda$

• Hadronic (CEH):
  Active material: Silicon in inner region, Scintillators
  and silicon-photomultiplier in outer region
  Passive: Steel absorbers
  7 all-Si layers, 21 scintillator layers
  $\sim$ 9 $\lambda$
HGCAL reconstruction: The Iterative CLustering (TICL)

- Modular framework
  - Independent developments
  - Algorithms can be easily swapped for comparisons
- Well-defined interface
- Validation-driven: → Today’s focus
TICL: CaloParticles

- **CaloParticles** are simulated objects created from a SimTrack helping to understand the decay chain – the CaloParticle will store the hits left in HGCal by a SimTrack (or the SimTrack’s children)
  - so far only present in HGCal (recently PR to include them also in barrel)

- **SimClusters** are the calorimetric equivalent of a SimTrack in HGCal: created for each SimTrack entering HGCal, storing information about the hits left in HGCal

- The **CaloParticle** basically is the first stable particle in the decay chain that itself (or via its children in Geant4) left hits in HGCal and can consist of multiple SimClusters
CaloParticle (CP) from SimTrack (ST) 703: $\pi^-$

CP from ST 701: $\pi^+$

CP from ST 702: $\pi^+$

Event displays
Event display: All CPs

CP from ST 701: $\pi^+$
CP from ST 702: $\pi^+$
CP from ST 703: $\pi^-$
CaloParticle distribution for a $Z \to \tau\tau$ sample
Goals for the SimTauCPLink

With this new object, we would like to:

• Identify which CaloParticles are coming from a $\tau$ decay
• Establish a connection to their corresponding GenParticle in simulation
• Save the resonances in the $\tau$ decay and flag the decay products (e.g. $\rho$ or $\pi^0$)
• Include the object as default in Phase-2 workflows in CMSSW
SimTauCPLinkProducer

```
class SimTauProducer : public edm::stream::EDProducer<> {

  void buildSimTau(SimTauCPLink &t, 
                  uint8_t generation, 
                  int resonance_idx, 
                  const reco::GenParticle & gen_particle, 
                  int gen_particle_key, 
                  edm::Handle<std::vector<CaloParticle>> calo_particle_h, 
                  const std::vector<int> & gen_particle_barcodes); 
  void produce(edm::Event&, const edm::EventSetup&) override;

  (...)
};
```
SimTauCPLinkProducer

class SimTauProducer : public edm::stream::EDProducer<> {

    void buildSimTau(...);
    void produce(edm::Event&, const edm::EventSetup&) override;

    const edm::EDGetTokenT<std::vector<CaloParticle>> caloParticle_token_;
    const edm::EDGetTokenT<std::vector<reco::GenParticle>> genParticles_token_;
    const edm::EDGetTokenT<std::vector<int>> genBarcodes_token_;
};
void BuildSimTauCPLink (...) {
    daughters = gen_particle.DaughterRefVector();
    bool is_leaf = (daughters.size() == 0);
    if (is_leaf) {
        auto &found_in_caloparticles = std::find_if(...);
        calo_particle_leaves.push_back(found_in_caloparticles);
        return;
    } else if (generation != 0) {
        resonances.push_back(gen_particle.pdgId, resonance_idx);
    }
    for (auto daughter: daughters) {
        buildSimTauCPLink(...)
    }
}
void BuildSimTauCPLink(...) {
    daughters = gen_particle.DaughterRefVector();
    bool is_leaf = (daughters.size() == 0);
    if (is_leaf) {
        auto &found_in_caloparticles = std::find_if(...);
        calo_particle_leaves.push_back(found_in_caloparticles);
        return;
    } else if (generation != 0) {
        resonances.push_back(gen_particle.pdgId, resonance_idx);
    }
    for (auto daughter: daughters) {
        buildSimCPLink(...);
    }
}
void BuildSimTauCPLink (...) {
    daughters = gen_particle.DaughterRefVector();
    bool is_leaf = (daughters.size() == 0);
    if (is_leaf) {
        auto &found_in_caloparticles = std::find_if(...);
        calo_particle_leaves.push_back(found_in_caloparticles);
        return;
    } else if (generation != 0) {
        resonances.push_back(gen_particle.pdgId, resonance_idx);
    }
    for (auto daughter: daughters) {
        buildSimTauCPLink(...);
    }
}
void BuildSimTauCPLink (...)
{
    daughters = gen_particle.DaughterRefVector();
    bool is_leaf = (daughters.size() == 0);
    if (is_leaf) {
        auto &found_in_caloparticles = std::find_if(...);
        calo_particle_leaves.push_back(found_in_caloparticles);
        return;
    } else if (generation != 0) {
        resonances.push_back(gen_particle.pdgId, resonance_idx);
    }
    for (auto daughter: daughters) {
        buildSimTauCPLink(...);
    }
}
void BuildSimTauCPLink (...)
{
  daughters = gen_particle.DaughterRefVector();
  bool is_leaf = (daughters.size() == 0);
  if (is_leaf) {
    auto &found_in_caloparticles = std::find_if(...);
    calo_particle_leaves.push_back(found_in_caloparticles);
    return;
  } else if (generation != 0) {
    resonances.push_back(gen_particle.pdgId, resonance_idx);
  }
  for (auto daughter: daughters) {
    buildSimTauCPLink(...);
  }
}
Example of a reconstructed $\tau$ decay

<table>
<thead>
<tr>
<th>16</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>-211 0 coming from:</td>
<td>-213 -1</td>
</tr>
<tr>
<td>22 1 coming from: 111 0 coming from:</td>
<td>-213 -1</td>
</tr>
<tr>
<td>22 1 coming from: 111 0 coming from:</td>
<td>-213 -1</td>
</tr>
<tr>
<td>L 16</td>
<td>-1 CP</td>
</tr>
<tr>
<td>L -211 0 CP</td>
<td>0</td>
</tr>
<tr>
<td>L 22 1 CP</td>
<td>1</td>
</tr>
<tr>
<td>L 22 1 CP</td>
<td>2</td>
</tr>
<tr>
<td>R -213</td>
<td>-1</td>
</tr>
<tr>
<td>R 111 0</td>
<td></td>
</tr>
</tbody>
</table>
The SimTauCPLink class

class SimTauCPLink {
    std::vector<std::pair<int, int>> resonances;
    std::vector<DecayNav> leaves;
    CaloParticleRefVector calo_particle_leaves;
    struct DecayNav {
        int pdgId;
        int resonance_idx;
        int calo_part_idx;
        int gen_part_idx;
    }
}

20.11.2023

TauCQM
Example of a reconstructed $\tau$ decay

16 $-1$: $\nu_\tau$ with no associated CaloParticle
-211 0 coming from: -213 $-1$
22 1 coming from: 111 0 coming from: -213 $-1$
22 1 coming from: 111 0 coming from: -213 $-1$
L 16 $-1$ CP $-1$
L -211 0 CP 0
L 22 1 CP 1
L 22 1 CP 2
R -213 $-1$
R 111 0
Example of a reconstructed $\tau$ decay

16 -1
-211 0 coming from: -213 -1: $\pi^-$ from $\rho$ (resonance index -1)
22 1 coming from: 111 0 coming from: -213 -1: $\gamma$ from $\pi^0$ from $\rho$
22 1 coming from: 111 0 coming from: -213 -1: $\gamma$ from $\pi^0$ from $\rho$

L 16 -1 CP -1
L -211 0 CP 0
L 22 1 CP 1
L 22 1 CP 2
R -213 -1
R 111 0

$\rightarrow \rho$ decaying to $\pi^-$ and $\pi^0$
Example of a reconstructed $\tau$ decay

16 -1  
-211 0 coming from: -213 -1  
22 1 coming from: 111 0 coming from: -213 -1  
22 1 coming from: 111 0 coming from: -213 -1  
L 16 -1 CP -1  
L -211 0 CP 0  
L 22 1 CP 1  \rightarrow \text{print out final leaves vector (pdgId, res_idx, CP_idx)}  
L 22 1 CP 2  
R -213 -1  
R 111 0
Example of a reconstructed \( \tau \) decay

\[
\begin{align*}
16 &\rightarrow -211 \ 0 \text{ coming from: } -213 \ -1 \\
22 &\ 1 \text{ coming from: } 111 \ 0 \text{ coming from: } -213 \ -1 \\
22 &\ 1 \text{ coming from: } 111 \ 0 \text{ coming from: } -213 \ -1 \\
L &\ 16 \ -1 \ CP \ -1 \\
L &\ -211 \ 0 \ CP \ 0 \\
L &\ 22 \ 1 \ CP \ 1 \\
L &\ 22 \ 1 \ CP \ 2 \\
R &\ -213 \ -1 \\
R &\ 111 \ 0 \quad \rightarrow \text{ print out final } \text{resonances vector (pdgId, res_idx)}
\end{align*}
\]
Next steps

• We are ready to do a PR for this object to be merged in CMSSW

• How would you think this object could be useful in general? E.g., could this be integrated in the existing TauValidation (as it’s only working in HGCal so far)?

• What should we include in the class? Examples:
  – An isRhoResonance method
  – The \( \tau \) decay modes (as they are already available in CMSSW)
  – …
Thank you for your attention