Dubrovnik, Apr 30th – May 3rd, 2019
-- Game of Flavours - CMS Heavy Flavour Tagging Workshop 2019 --

tracking

at CMS

mia tosi
on behalf of the CMS collaboration
**introduction**

**the tracking challenge at the LHC**
- typically 30 charged particles within the tracker acceptance per proton-proton collision and 40 collisions per event: $\mathcal{O}(1200)$ charged particles per event
- these need to be reconstructed with:
  - very high efficiency (>90%)
  - precise track parameters
  - very low fake rate: $\mathcal{O}(\sim\%)$
  - quickly (stringent CPU limits)
- very strong requirements on track reconstruction algorithms
- track reconstruction is not just about reconstructing charged particles: used in almost every element of reconstruction

tracking is a key ingredient of reconstructing the full event

see Petra and Chris’ slides for details on Flavour Tagging approaches in ATLAS and CMS
CMS tracker: Silicon Strips

- $O(10\ M)$ strips
- $O(200)\ m^2$ of sensors
- hit resolution: $(10,40) \times (230,530)\ \mu m$
- occupancy: 1-3%
- coverage up to $|\eta| < 2.5$

Sub-detectors
- Inner Barrel (TIB): 4
- Inner Disks (TID): 3 (x 2)
- Outer Barrel (TOB): 6
- Endcap (TEC): 9 (x 2)

many layers: redundancy
12 hits per track on average
CMS tracker: Silicon Pixel

- 127 M pixels,
- 100 x 150 $\mu m^2$ in size
- hit resolution: 10 x (20,40) $\mu m$
- occupancy: $10^{-3}$
- coverage up to $|\eta| < 2.5$ (even 3.0)
- layer position [cm]:
  - BPix: 2.9, 6.8, 10.9, 16.0
  - FPix: 29.1, 39.6, 51.6
- high segmentation: high quality seeds for offline tracking

since 2017, one additional tracking point, in both barrel and forward regions:
- 4-hit seeds
- lower fake rate!
smaller radius of the innermost pixel layer
- closer to the interaction region
- improve tracking and vertexing performance
- reduced material budget
- reduce multiple scattering
track reconstruction in CMS

- few, but precise measurements
- non negligible amount of dead material inside the tracker volume

| $R_{\text{inner}}$ [cm] | $R_{\text{outer}}$ [m] | $|\eta|$ coverage | B field [T] | $X_0$ @ $|\eta| = 0$ | $p_T$ resolution @1 (100) GeV, $|\eta| = 0$ | $d_0$ resolution @1 (100) GeV, $|\eta| = 0$ [$\mu$m] |
|-------------------------|-----------------------|-------------------|------------|-----------------|------------------|-----------------|
| 3                       | 1.1                   | 3.0               | 3.8        | 0.4             | 0.7 (1.5)%       | 90 (20)         |

main tracking algorithm: Combinatorial Track Finder
used in iterative steps

- limits the number of combinatorics in pattern recognition
- tracking reach guarantee, w/o degrading computing performance

high-quality tracks are reconstructed first, their hits are removed, other (more difficult) tracks are reconstructed from the remaining hits

i.e. w/ more multiple scattering, loops, displaced tracks..
in each iteration, tracks are reconstructed in four steps:

1. **seeding**: use combination of pixel, strip or mixed hits
   provides track candidates,
   with an initial estimate of the trajectory parameters and their uncertainties

2. **pattern recognition**: alignment uncertainty taken into account
   hit compatible w/ predicted track position are added (Kalman update)
   to the trajectory track parameters are updated

3. **final fit**: taking into account the Field non uniformity and a detailed description of the material budget
   provides the best estimate of the parameters of each smooth trajectory
   after combining all associated hits [outlier hits are rejected]

4. **selection**:
   sets quality flags based on the fit $\chi^2$ and the track compatibility w/ interaction region
   aims to reject fake tracks *tracks sharing too many hits are also cleaned as duplicates*
in each iteration, tracks are reconstructed in four steps:

1. **seeding**: use combination of pixel, strip or mixed hits
   provides track candidates,
   with an initial estimate of the trajectory parameters and their uncertainties

2. **pattern recognition**: alignment uncertainty taken into account
   hit compatible w/ predicted track position are added (Kalman update)
   to the trajectory track parameters are updated

3. **final fit**: taking into account the Field non uniformity and a detailed description of the material budget
   provides the best estimate of the parameters of each smooth trajectory
   after combining all associated hits [outlier hits are rejected]

4. **selection**: 
   sets quality flags based on the fit $\chi^2$ and the track compatibility w/ interaction region
   aims to reiect fake tracks **tracks sharing too many hits are also cleaned as duplicates**

**iterative procedure**
Iterative Tracking in CMS

tracks reconstruction is an **iterative procedure**:

- the **InitialStep** makes use of high-$p_T$ quadruplets coming from the beam spot region
- subsequent steps use triplets, or improve the acceptance either in $p_T$ or in displacement
- the later steps use seeds w/ hits from the strip detector to find detached tracks,
- final steps are dedicated to special phase-space
  - highly dense environment (i.e. w/in jets)
  - clean environment (i.e. muons)

---

**CMS Simulation preliminary** 13 TeV

- **Initial**
- **HighPtQuad**
- **LowPtQuad**
- **LowPtTriplet**
- **DetachedQuad**
- **DetachedTriplet**
- **MixedTriplet**
- **PixelLess**
- **TobTec**
- **JetCore**
- **Muon inside-out**
- **Muon outside-in**

**CMS Simulation preliminary** 13 TeV

- **Initial**
- **HighPtTriplet**
- **LowPtQuad**
- **LowPtTriplet**
- **DetachedQuad**
- **DetachedTriplet**
- **MixedTriplet**
- **PixelLess**
- **TobTec**
- **JetCore**
- **Muon inside-out**
- **Muon outside-in**

---

**Iteration** | **Seeding** | **Target track**
---|---|---
Initial | pixel quadruplets | prompt, high $p_T$
LowPtQuad | pixel quadruplets | prompt, low $p_T$
HighPtTriplet | pixel triplets | prompt, high $p_T$ recovery
LowPtTriplet | pixel triplets | prompt, low $p_T$ recovery
DetachedQuad | pixel quadruplets | displaced--
DetachedTriplet | pixel triplets | displaced-- recovery
MixedTriplet | pixel+strip triplets | displaced--
PixelLess | inner strip triplets | displaced+
TobTec | outer strip triplets | displaced++
JetCore | pixel pairs in jets | high-$p_T$ jets
Muon inside-out | muon-tagged tracks | muon
Muon outside-in | standalone muon | muon

---

**h\(t\) event tracks (\(\text{PU}=35\))**

- $p_T > 0.9$ GeV, $|\eta| < 2.5$

---

**Sim. track prod. vertex radius (cm)**

- $\text{Simulated track } p_T$ (GeV)
- $\Delta R$ (jet,track)
Iterative Tracking in CMS

Tracks reconstruction is an iterative procedure:
• the InitialStep makes use of high-$p_T$ quadruplets coming from the beam spot region
• subsequent steps use triplets, or improve the acceptance either in $p_T$ or in displacement
• the later steps use seeds w/ hits from the strip detector to find detached tracks,
• final steps are dedicated to special phase-space
  • highly dense environment (i.e. w/in jets)
  • clean environment (i.e. muons)

Tracks w/ high $p_T$ can be reconstructed fast!
Run2 performance

few challenges:

- $50\text{ns} \rightarrow 25\text{ns}$ out-of-time pile up introduce few mitigations like the (strip) cluster charge cut

- much higher PU and luminosity (w.r.t. Run1 and design)
few challenges:

- 50ns → 25ns out-of-time pile up
- much higher PU and luminosity (w.r.t. Run1 and design)
- commissioning the new pixel detector during the run

- increase efficiency
  (above all at high pseudo-rapidity)
- decrease fakerate
- improve pT resolution
  (mainly in the transition region)
Run2 performance

few challenges:

- 50ns → 25ns out-of-time pile up
- much higher PU and luminosity (w.r.t. Run1 and design)
- commissioning the new pixel detector during the run
- some detector issues/features

extrapolate track to tracker module and check for the presence of a reconstructed hit

[inactive/problematic modules crossed out]

single hit reconstruction efficiency: \( \geq 99\% \)

the inefficiency mainly depends on the particle flux, inner layers are more sensitive than the outer ones
in general, **high tracking** and **vertexing performance** (despite difficult circumstances), thanks to significant improvements, which were made during both the LS1 and Run2:

- new iterations, new tuning, PU mitigation,
- code re-engineering, new seeding framework,
- **Cellular Automaton seeding**, mitigation strategy, etc..

<table>
<thead>
<tr>
<th></th>
<th>2017</th>
<th>2018</th>
</tr>
</thead>
<tbody>
<tr>
<td>tracker + muon seeded</td>
<td>99.0%</td>
<td>99.5%</td>
</tr>
<tr>
<td>tracker-only seeded</td>
<td>97.0%</td>
<td>98.0%</td>
</tr>
</tbody>
</table>

*Tracking efficiency using events from $Z \rightarrow \mu \mu$ and $J/\psi \rightarrow \mu \mu$*
new track seeding algorithm based on **Cellular Automaton (CA)** technique

- it starts from a list of layers and their pairings
  - a graph of all possible connections between layers is created
  - doublets (cells) are created for each pair of layers [compatible with a region hypothesis]

- fast computation of the compatibility between 2 connected cells

<table>
<thead>
<tr>
<th>Quadruplet Algorithm</th>
<th>Time per event</th>
<th>speedup wrt. 2016</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPU Cellular Automaton</td>
<td>(1.2 ± 0.9) ms</td>
<td>24.4×</td>
</tr>
<tr>
<td>CPU Cellular Automaton</td>
<td>(14.0 ± 6.2) ms</td>
<td>2.1×</td>
</tr>
<tr>
<td>Triplet Propagation</td>
<td>(72.1 ± 25.7) ms</td>
<td>0.4×</td>
</tr>
<tr>
<td>2016 Pixel Tracks</td>
<td>(29.3 ± 13.1) ms</td>
<td>1×</td>
</tr>
</tbody>
</table>

- **x5 faster** than old algorithm
- **x2 faster** than 2016 configuration

- **performance** (with new pixel detector)
  - almost same efficiency in the barrel
  - 50% efficiency gain in the endcap
  - **x4 reduction in fakes**!
cellular automaton seeding

- more robust
- smaller complexity vs PU than 2016 track seeding despite the increased number of layer combinations involved
- in pattern recognition, no additional gain

despite the increase in the number of pixel layers

~20% faster track reconstruction wrt to 2016 tracking @ <PU> = 70
Run2 performance

- precise knowledge of the detector alignment crucial for track reconstruction
- detector components have been observed to move
e.g. tension of support structure due to thermal transients or magnetic field changes
- CMS employs an automatic procedure
to monitor movements of top level hierarchical mechanical structures
(half-barrels and half-disks)
when appropriate, detector geometry is updated based on these online results

- $\phi$-dependent mass bias in $m_{\mu\mu}$ @high muon rapidity
  - greatly reduced w/ the update of the alignment
- track impact parameters ($d_{xy}$ and $d_z$) are sensitive to
  - Lorentz Angle mis-calibrations
  - misalignment in the pixel detector
  - residual bias is nicely recovered by the time granularity alignment
  - still sub-optimal performance in certain period could be improved further
in Run2, we decide to keep the Tracking@HLT as close as possible to the offline track reconstruction on both the algorithms & configurations but, time is a constraint

⇒ reduce #iterations wrt offline tracking
⇒ constrain tracking regions (i.e., regional tracking)

during Phase1 pixel commissioning, failures observed mostly geometrically contained
⇒ in 2017, adopted a Static mitigation via dedicated iterations in specific $\eta$-$\phi$ regions

however, recovery is insufficient for additional (dynamic) pixel issues [like the DC/DC converter issue]
⇒ in 2018, adopted the Dynamic mitigation of pixel issues [trade off among efficiency/fake and timing]

CA seeding since 2017

<table>
<thead>
<tr>
<th>N</th>
<th>Step Name</th>
<th>Seeding</th>
<th>Target Track</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Iter0</td>
<td>pixel tracks (from quadruplets)</td>
<td>prompt, high $p_T$</td>
</tr>
<tr>
<td>1</td>
<td>Iter1</td>
<td>pixel quadruplets</td>
<td>prompt, low $p_T$</td>
</tr>
<tr>
<td>2</td>
<td>Iter2</td>
<td>pixel triplets</td>
<td>recovery (not only high $p_T$)</td>
</tr>
<tr>
<td>3</td>
<td>Triplet recovery</td>
<td>pixel triplets in $\eta$-$\phi$ region</td>
<td>static triplet recovery</td>
</tr>
<tr>
<td>4</td>
<td>Doublet recovery</td>
<td>Pixel doublets in $\eta$-$\phi$ region</td>
<td>static doublet recovery</td>
</tr>
</tbody>
</table>

nearly ideal efficiency is achieved

efficiency is almost flat as a function of #PU
during Run3, there will be a not negligible degradation of both Pixel and Strip detectors wrt nominal performance due to the accumulated radiation

- large portions of TIB1/2 inactive due to leakage currents

⇒ mitigation strategies need to be developed
plans: run3  
DeepCore

- algorithms like the Particle Flow, the b-tagging and the jet sub-structure rely on the good performance of the track reconstruction w/in jets
- tracking inside jets starts to become inefficient for $p_T > 500$ GeV
- in Run2, dedicated step has been added: **Jet Core**
  - the merged clusters in the pixel detector affect already the seeding step
- in Run3, the **DeepCore** will be deployed
  - basic idea is to skip the pixel clustering, exploit directly the RAW pixel data and reconstruct the seed of tracks w/in the jets
  - develop a convolutional Neural Network (cNN) to reproduce the «function»

- almost cancelled seeding inefficiencies
- fake rate reduction up to 60%
- seeding timing reduced by 85%

hits from different tracks can result in a merged pixel cluster in the core of high-$p_T$ jets (the effect is even more pronounced for b-jets, because the B decay happens closer to the pixel detector)

see Jean-Roch’ slides for more details on ML approaches in CMS (and Tobias Golling’s in ATLAS)
plans: run3

- to ensure smooth operation of the heterogeneous HLT farm during Run4
- accelerated RAW data to Pixel Track and Vertices reconstruction
- currently, at PU = 50, pixel tracks are reconstructed only for ~10% of the events @HLT
- profit from the upgrade of the Pixel to redesign the pixel track algorithm
  - developed 2 Multiple Scattering-aware fits:
    - Riemann Fit: improve parameter resolutions and fake rejection
    - Broken Line Fit: mitigate duplicates in the pixel endcap
  - developed a seeds cleaner “fishbone” mit negate duplicates in the pixel endcap
  - implemented the z-clustering by DBSCAN: density-based spatial clustering of applications w/ noise
plans: run3

- to ensure smooth operation of the heterogeneous HLT farm during Run4
- accelerated RAW data to Pixel Track and Vertices reconstruction
- currently, at PU = 50, pixel tracks are reconstructed only for ~10% of the events @HLT
- profit from the upgrade of the Pixel to redesign the pixel track algorithm
- developed 2 Multiple Scattering-aware fits: Riemann Fit, Broken Line Fit
- developed a seeds cleaner "fishbone"
- implemented the $z$-clustering by DBSCAN to improve parameter resolutions and fake rejection
- mitigate duplicates in the pixel endcap density-based spatial clustering of applications

6th Patatrack Hackathon

Save the date!

CERN Idea Square, 1–5 July 2019

https://indico.cern.ch/event/799486
plans: phase2

present tracker designed for an integrated lumi of 500/fb and <PU> ~ 30-50

Requirements for Phase2

- High radiation tolerance to operate efficiently up to 3000/fb
- increased granularity to maintain channel occupancy around or below the % level
- reduced material in the tracking volume
- contribution to the Level-1 trigger
- extended tracking acceptance
- robust pattern recognition

see Daniel’ slides for more details on Flavour Tagging at HL-LHC
**conclusion**

- despite challenging circumstances at the LHC in Run2
  high tracking and vertexing performance
  → large performance improvements
due to the upgrades to the pixel
- the simulation is accurate on predicting tracking behaviour
- detailed studies of detector material
- track efficiency measurements based on data-driven techniques
- alignment has become an even more challenging task
  as the experiments cope w/
  the impact of detector movements and increased precision

**new developments** targeting Run3
  mitigation strategy, tracking in dense environment,
  pixelTracks, and ML oriented:
  DNN track selection, DeepCore, see Jean-Roch Vlimant’s talk
  CNN mergeCluster, CNN cluster shape filter

**synergy** w/ the tracker experts group
  and the physics object reconstruction experts
  is a key ingredient for the developments
  and establishment of the expertise
  try to not loose the expertise built in the last years,
  and build a new team of experts

...paper on performance of tracking in Run2 on going
A helical trajectory can be expressed by 5 parameters, but the parameterization is not unique.

Given one parameterization, we can always re-express the same trajectory in another parameterization.

In general terms, the five parameters are:

- **signed radius of curvature** (units of cm), which is proportional to particle charge divided by the transverse momentum, $p_T$, (units of GeV);
- **angle of the trajectory** at a given point on the helix, in the plane transverse to the beamline (usually called $\phi$);
- **angle of the trajectory** at a given point on the helix with respect to the beamline ($\theta$, or equivalently $\lambda = \pi/2 - \theta$), which is usually expressed in terms of pseudorapidity $\eta = -\ln(\tan(\theta/2))$;
- **offset or "impact parameter"** relative to some reference point (usually the beamspot or a selected primary vertex), in the plane transverse to the beamline (usually called $d_{xy} = d_0$);
- **impact parameter** relative to a reference point (beamspot or a selected primary vertex), along the beamline (usually called $d_z$).
Fishbone

After using the CA for producing N-tuplets, “fishbone” seeds can be produced to account for module/layer overlaps

Only the best chi² or highest grade n-tuplet is fitted (pentuplets in mixed barrel-endcap region)

Presented for the first time as a possible mitigation of duplicates in the pixel EC in January 2017, today available in CUDA soon in C++
new developments: tracking on GPU

- to ensure smooth operation of the heterogeneous HLT farm during Run4
- accelerated RAW data to Pixel Track and Vertices reconstruction
  with some improvements in efficiency and performance

- profit from the upgrade of the Pixel to redesign the pixel track algorithm
  from scratch

- developed 2 Multiple Scattering - aware fits:
  - Riemann Fit
  - Broken Line Fit

- developed a seeds cleaner "fishbone"

- exploit the $z$-clustering by DBSCAN
  to improve parameter resolutions and fake rejection
  to mitigate duplicates in the pixel endcap
in general, high tracking and vertexing performance (despite difficult circumstances), thanks to significant improvements, which were made during both the LS1 and Run2: new iterations, new tuning, PU mitigation, code re-engineering, new seeding framework, Cellular Automaton seeding, mitigation strategy, etc.

Tracking efficiency using events from $Z \rightarrow \mu\mu$ and $J/\Psi \rightarrow \mu\mu$

<table>
<thead>
<tr>
<th>Year</th>
<th>Tracker + Muon Seeded</th>
<th>Tracker - Only Seeded</th>
</tr>
</thead>
<tbody>
<tr>
<td>2017</td>
<td>97.0%</td>
<td>97.9%</td>
</tr>
<tr>
<td>2018</td>
<td>99.0%</td>
<td>98.8%</td>
</tr>
</tbody>
</table>
the vertex reconstruction: **Deterministic Annealing Clustering**

- Minimize $\chi^2$-like energy
- Inspired by physical annealing
  - A system reaches the state of minimal energy by going from high T to low T
  - Treat it as an ensemble of systems

$c_{ik} \rightarrow p_{ik}$

**Assignment probabilities**

$$p_{ik} = \frac{\rho_k e^{\frac{(z_i - z_k)^2}{T \sigma_i^2}}}{\sum_k \rho_k e^{\frac{(z_i - z_k)^2}{T \sigma_i^2}}}$$

1. **maximize Entropy for fixed E ($\leftrightarrow$ T)**
   - @ fixed T minimize free energy $F$ wrt cluster positions $z_k$
   - Reaches constellation w/ no net “forces” on clusters (iteratively)

2. when minimum reached, reduce T and repeat until $T = T_{\text{min}}$

**NB:**

Algorithm not robust wrt outliers that pass track selection

- **Outlier rejection** competes w/ splitting, only below $T_{\text{min}}$
- Allows tracks to be associated to no vertex at all $\rightarrow$ outlier suppression, like in AVF
- Clusters with too few tracks are discarded
bibliography

details of the tracking and vertexing performance results