Present and future of online tracking in CMS

Adriano Di Florio
(INFN & Politecnico Bari)
On Behalf of CMS Collaboration

CTD 2022
1st June 2022
Princeton University

CMS - Triggering and tracking

L1 Trigger

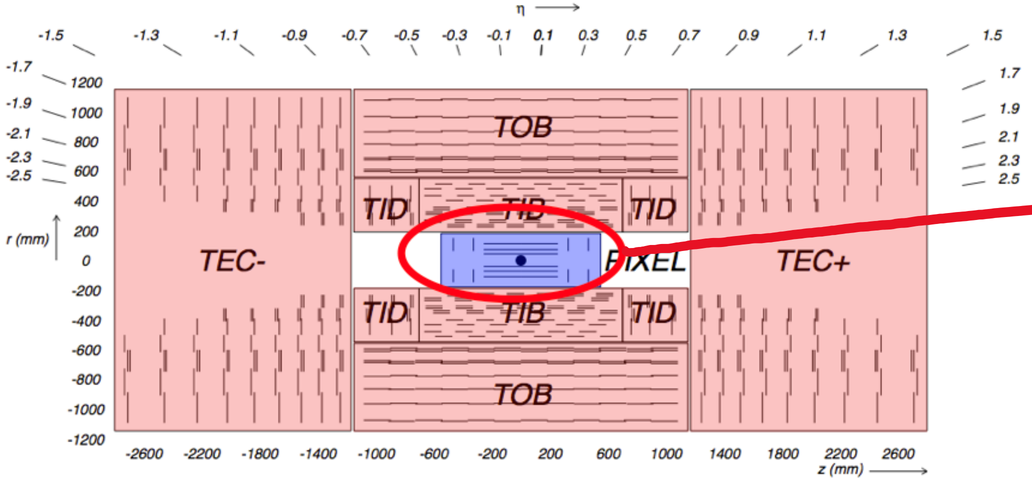
- 40 MHz input / 100 KHz output.
- Processing time: O(μs).
- Coarse local reconstruction.
- FPGAs / Hardware implemented.

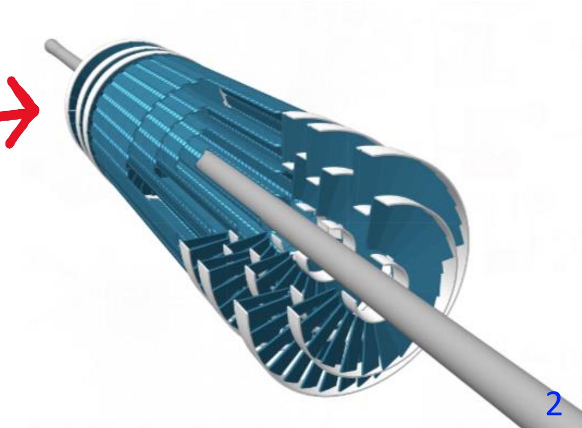
High Level Trigger (HLT)

- ~100 KHz in / ~1 KHz out.
- 500 KB / event.
- Processing time: O(100s) ms.
- Simplified global reconstruction.

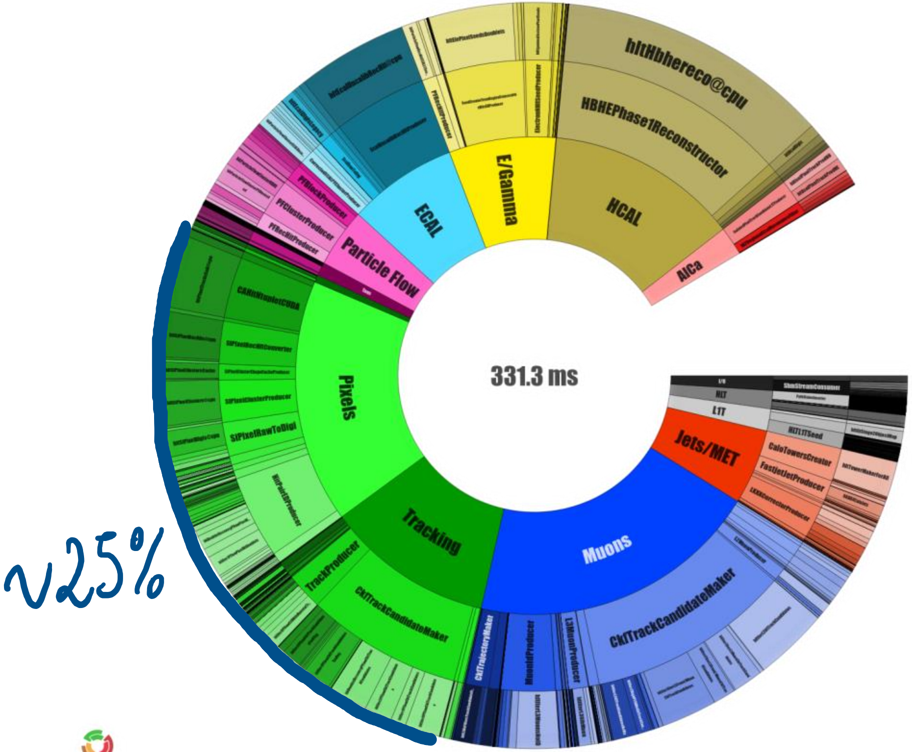
ONLINE TRACK RECONSTRUCTION (HLT)

Practically the same iterative reconstruction procedure as the one run offline. It has to undergo stringent time limits : O(100) ms.





Where we stood? Run2



CMS and LHC scenario at the end of Run2

- peak average instantaneous luminosity of 2×10^{34} cm^2s^{-1}
- about 50 proton-proton collisions per bunch crossing
- 100 kHz input rate (from the Level 1 Trigger rate)

A traditional CPU farm

- Over 1000 machines for 716 kHS06
- 30k physical CPU cores / 60k logical cores
- HLT running with multithreading
- 15k jobs with 4 threads

CMS track reconstruction algorithm at the HLT was based on an iterative approach, consisting of three main iterations:

- **iter0:** seeded by 4-hit global pixel tracks (p_T > 0.8 GeV)
- **iter1**: seeded by 4-hit global pixel tracks (p_T > 0.4 GeV)
- **iter2**: regional (jets) and seeded by 3 pixel hits (p_T > 0.4 GeV)

Where we stood? … CTD19

Patatrack:

Accelerated Pixel Track reconstruction in CMS

Andrea Bocci[1], Vincenzo Innocente[1], Matti Kortelainen[2],
Felice Pantaleo[1], Roberto Ribatti[3],
Marco Rovere[1], Gimmy Tomaselli[3]

[1]CERN – Experimental Physics Department, [2]FNAL,
[3]Scuola Normale Superiore di Pisa

felice@cern.ch

1

Where we stood? … CTD19

Conclusion

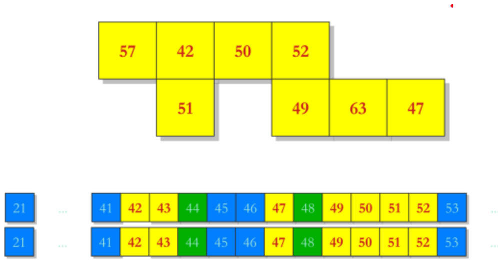
- A GPU-based full reconstruction of the Pixel detector from RAW data decoding to Pixel Tracks and Vertices determination has been implemented
- This reconstruction is fully integrated in the CMS Software
  - Conversion to the legacy data formats and the standard validation can be run on demand
- Can achieve better physics performance, faster computational performance at a lower cost with respect to the baseline solution
- The focus during LS2 will be to maximize code sharing to have the very same workflow running on GPUs and CPUs
  - Already achieved for many critical algorithms

32

Parallelism Exposed

Local pixel tracker reconstruction:

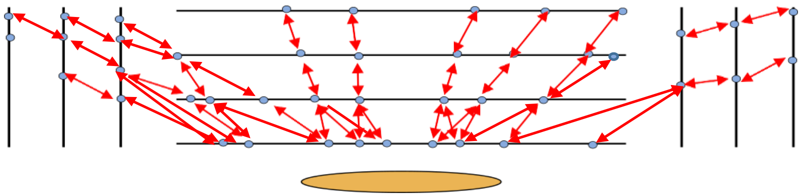
- *raw data unpacking and decoding*: parallelised across all input pixel hits
- clustering of the pixel hits parallelised across the pixel detectors and across the input pixel hits
- conversion to global coordinates parallelised across each cluster



Seeds Building

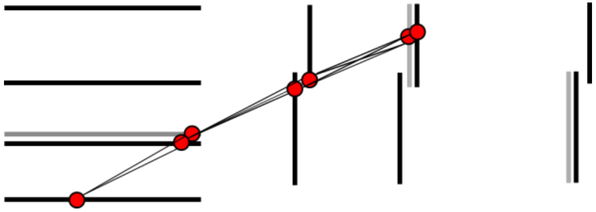
- *doublets:* parallelised on the hits of each layer
- *n-tuplets:*

    1. 2D parallelisation on the inner and outer layers
    2. Cellular Automaton (CA) algorithm with depth-first search



n-tuplets cleaning

- Fishbone algorithm merges overlapping ntuplets
- 2D parallelisation over ntuplets and possible duplicates



Track Fitting: (Eigen-based) parallelised over the ntuplets

Pixel Vertexing

- along z cluster tracks: parallelised across all input tracks
- split low quality vertices: parallelised across the vertices

Run3 HLT: offloading to GPU

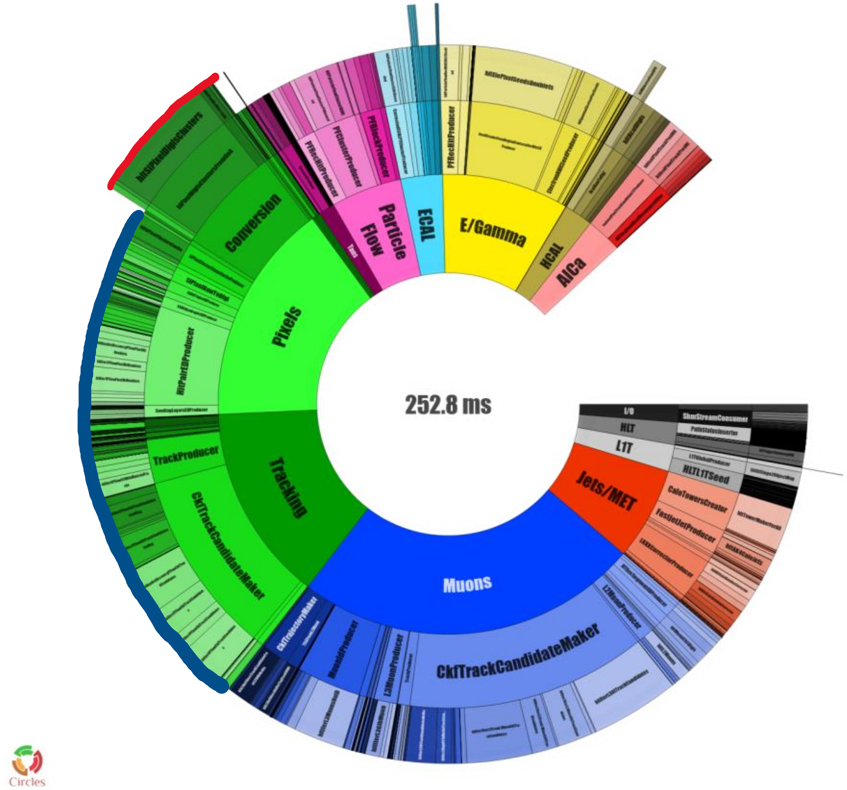
Final integration in the experiment's software in 2020-2021 (after 5 years of effort).

Even if intially targetting Phase II, things evolve rapidly and Run3 became an ideal benchmark:

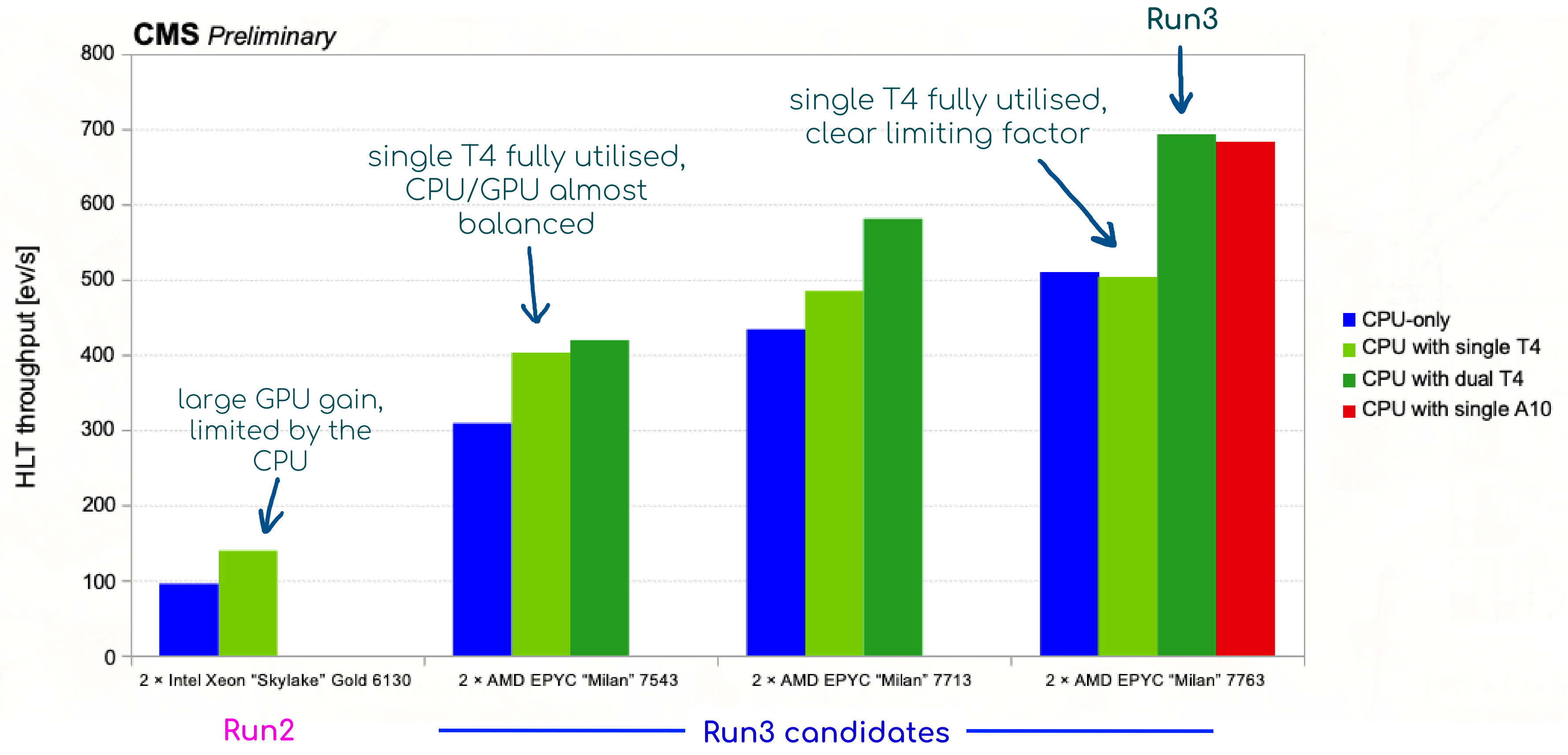
- no external pressure from LHC conditions.
- gain experience.
- take advantage of the extra computing capacity (e.g. scouting).

CMS HLT will offload four main components to GPUs:

- **pixel tracker local reconstruction.**
- **pixel-only track and vertex reconstruction.**
- electromagnetic and hadronic calorimeter local reconstruction.



HLT Throughput



Single Iteration approach

Run3 HLT Tracking:

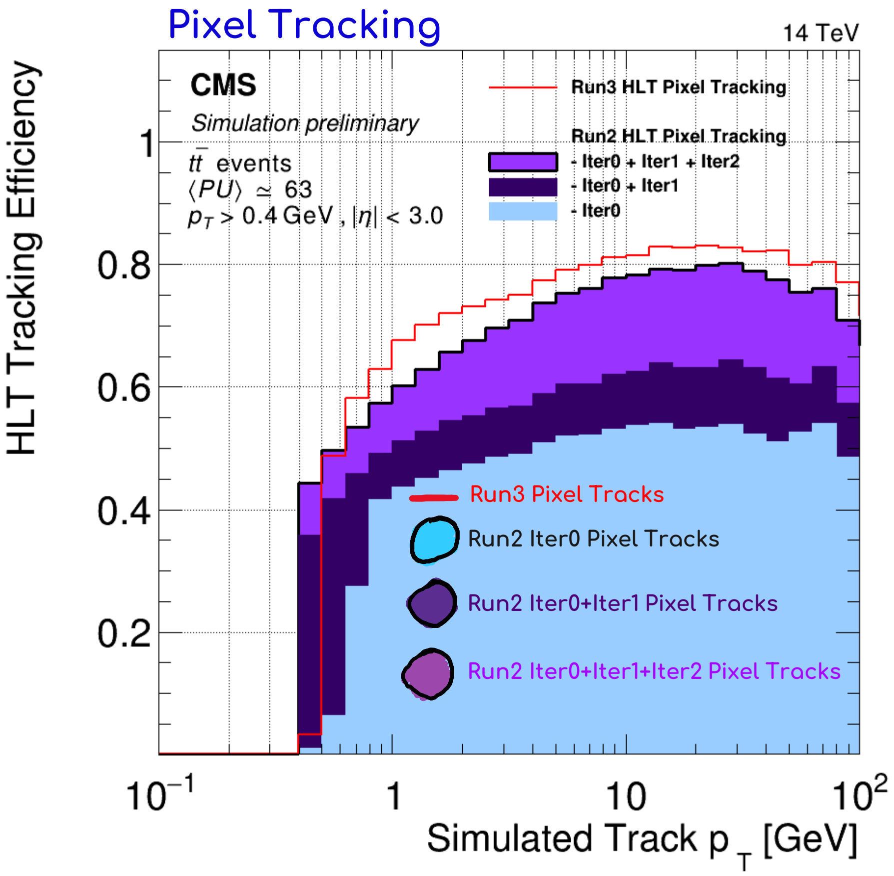
- Two pillars:
  1. profit from pixel tracks GPU offload.
  2. Retain (or improve) Run2 performance.

- Given the better performance of pixel tracks, Run 3 HLT tracking is based on a single iteration approach seeded by Patatrack pixel tracks (with n_{hits}>=3).

- Pixel vertices (on GPU) are reconstrcuted from pixel tracks (n_{hits}>=4 & p_T>0.5 GeV).

- A subset of (few) *trimmed* vertices ($\Sigma p_T^2 > 0.3 \cdot max(\Sigma p_T^2)$) is used to select seeds (as it was in Run2).

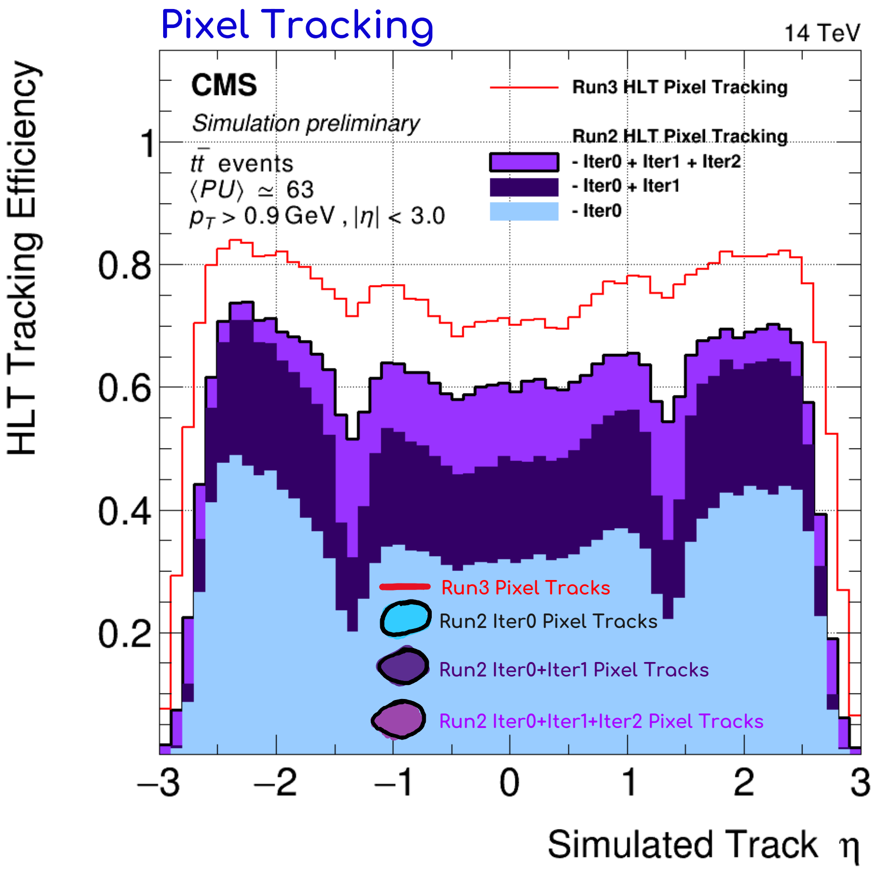
- **Physics performance are retained (or improved) & timing reduced by ~25% (on CPU).**



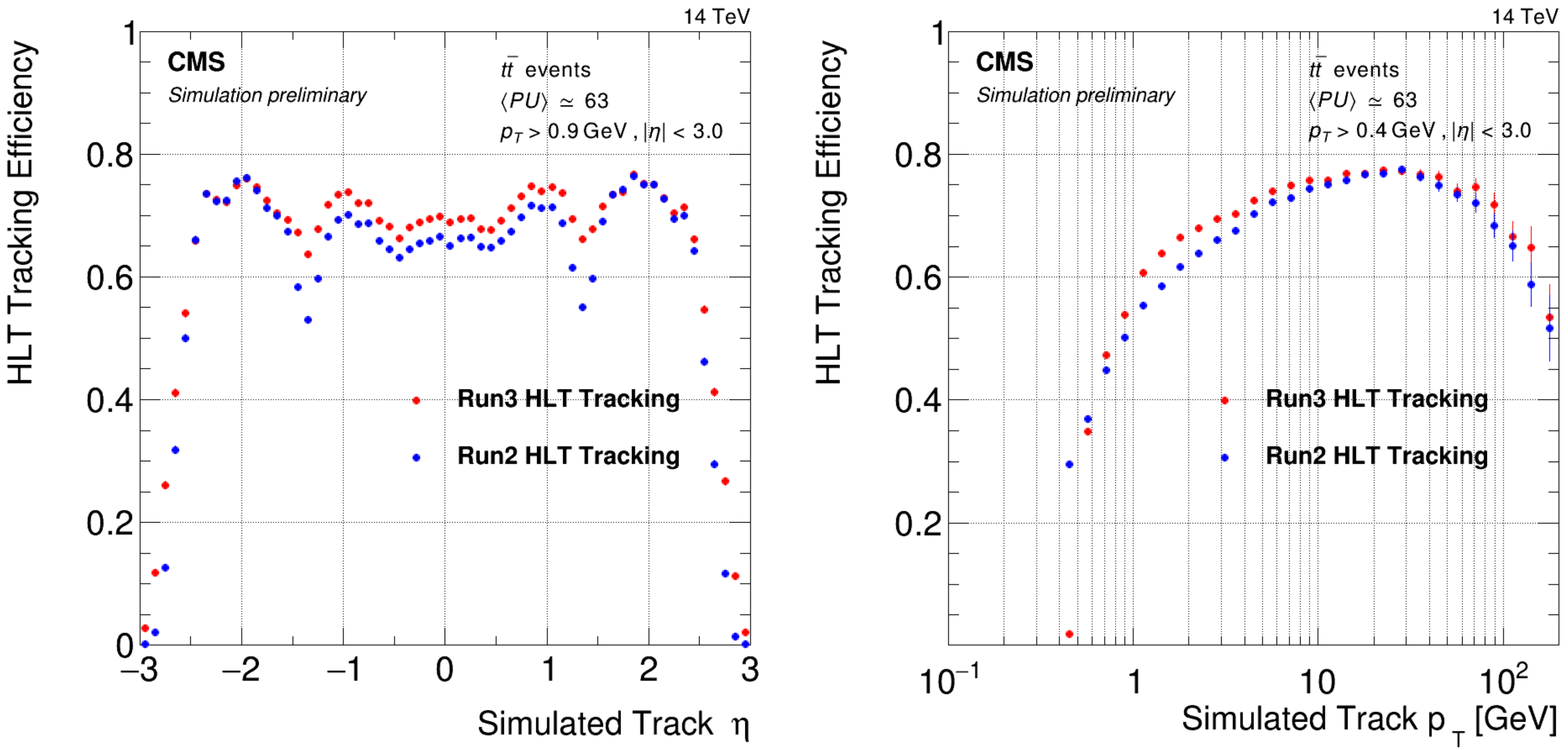
Single Iteration Approach

Run3 HLT Tracking:

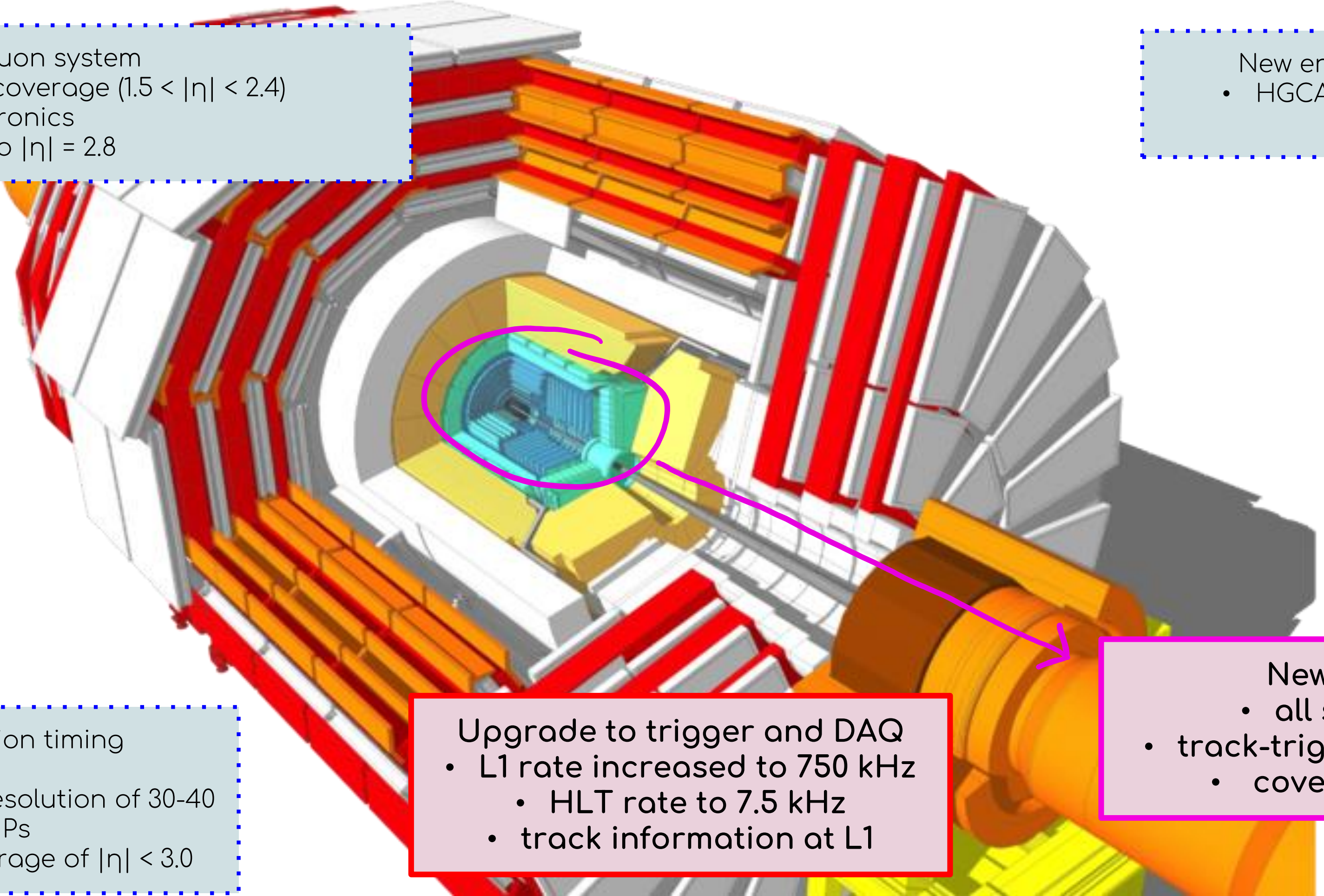
- Two pillars:
  1. profit from pixel tracks GPU offload.
  2. Retain (or improve) Run2 performance.
- Given the better performance of pixel tracks, Run 3 HLT tracking is based on a single iteration approach seeded by Patatrack pixel tracks (with $n_{hits}>=3$).
- Pixel vertices (on GPU) are reconstrcuted from pixel tracks ($n_{hits}>=4$ & $p_T>0.5$ GeV).
- A subset of (few) *trimmed* vertices ($\Sigma p_T^2>0.3\cdot max(\Sigma p_T^2)$) is used to select seeds (as it was in Run2).
- Physics performance are retained (or improved) & timing reduced by ~25% (on CPU).



Full Track - Efficiency



CMS@Phase II



Improved muon system
- new RPC coverage (1.5 < |η| < 2.4)
- new electronics
- GEM up to |η| = 2.8

New endcap calorimeters
- HGCAL: high granularity
- 4D showers

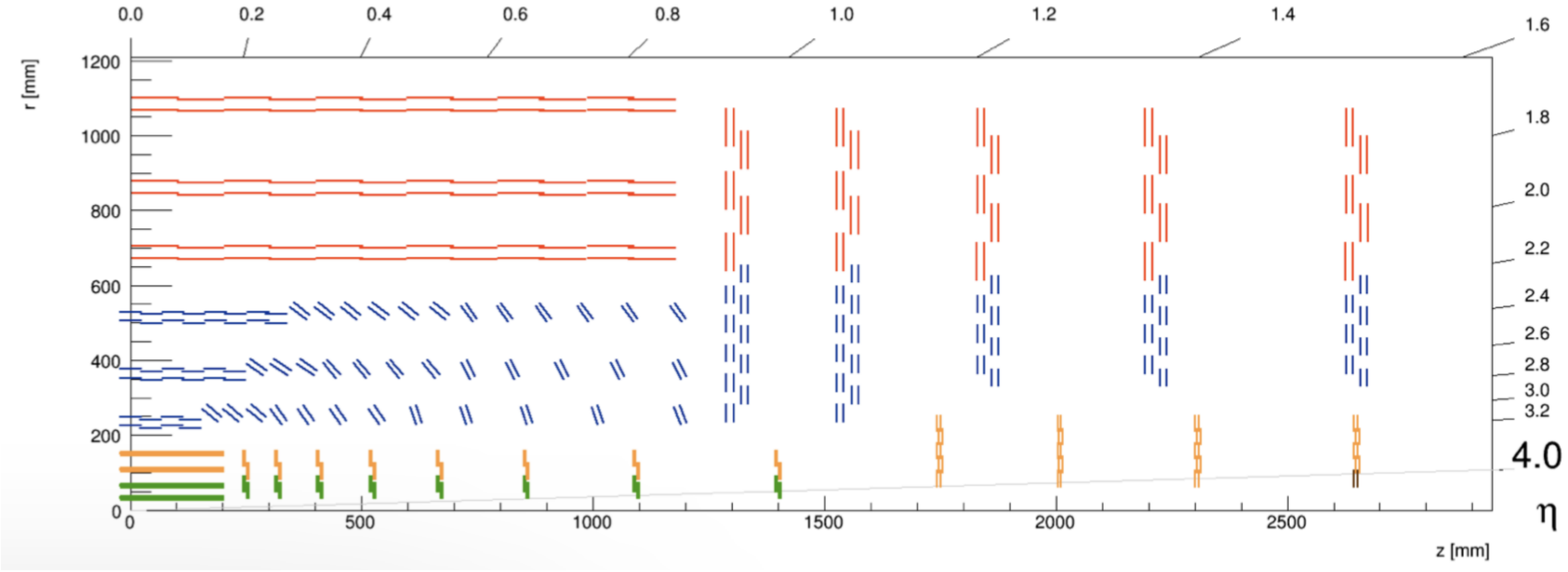
New precision timing detector
- timing resolution of 30-40 ps for MIPs
- full coverage of |η| < 3.0

Upgrade to trigger and DAQ
- L1 rate increased to 750 kHz
- HLT rate to 7.5 kHz
- track information at L1

New inner tracker
- all silicon tracker
- track-trigger @ 40 MHz
- coverage to |η| < 4

CMS Tracker @Phase II

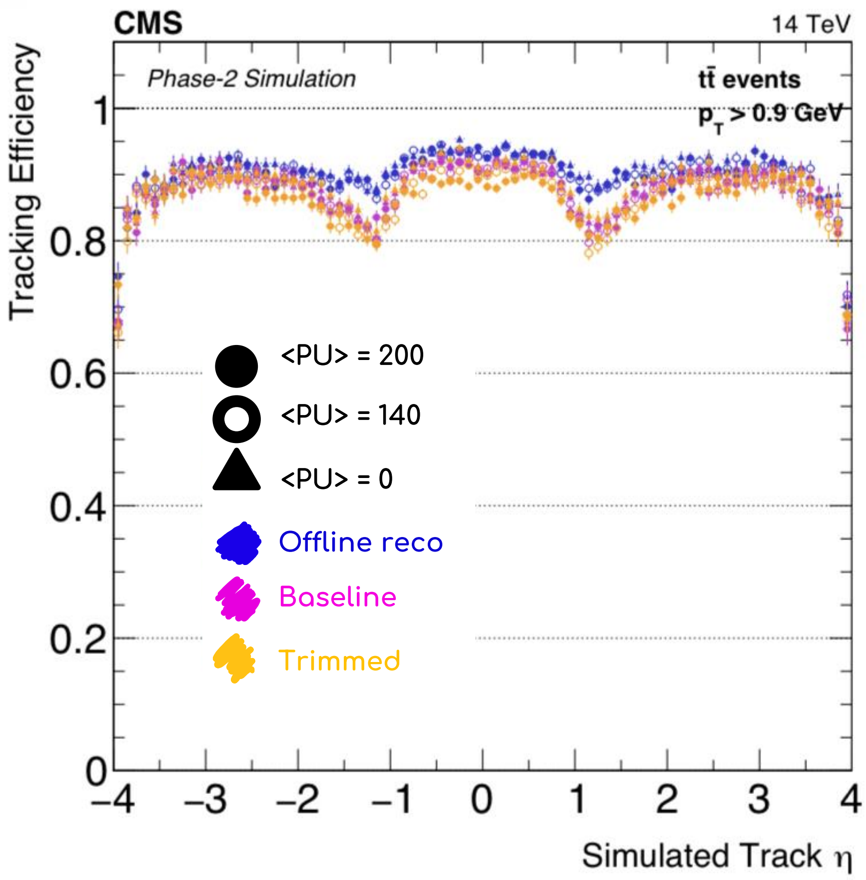
New CMS tracker with extended coverage (|η|<4) and increased number of layers.



Iterative Tracking for Phase II

In the Phase-2 Upgrade of the CMS Data Acquisition and High Level Trigger TDR:

- Starting point: offline Phase II track reconstruction.
- Redefining and adapting the iterations to reduce timing. HLT *baseline* tracking configuration with two iterations:
    1. First iteration: seeded by pixel tracks (n_{hits}=4).
    2. Second iteration: seeded by pixel triplets.
- In addition: a *trimmed* configuration (mimicking what is done Run 3) for which the seeds are selected to be compatible with a set of (~10) trimmed vertices.
- **Performance of *baseline* competitive with *offline reco* and timing reduced of a factor 6. The *trimmed* configuration brings a furter 20-30% timing reduction.**

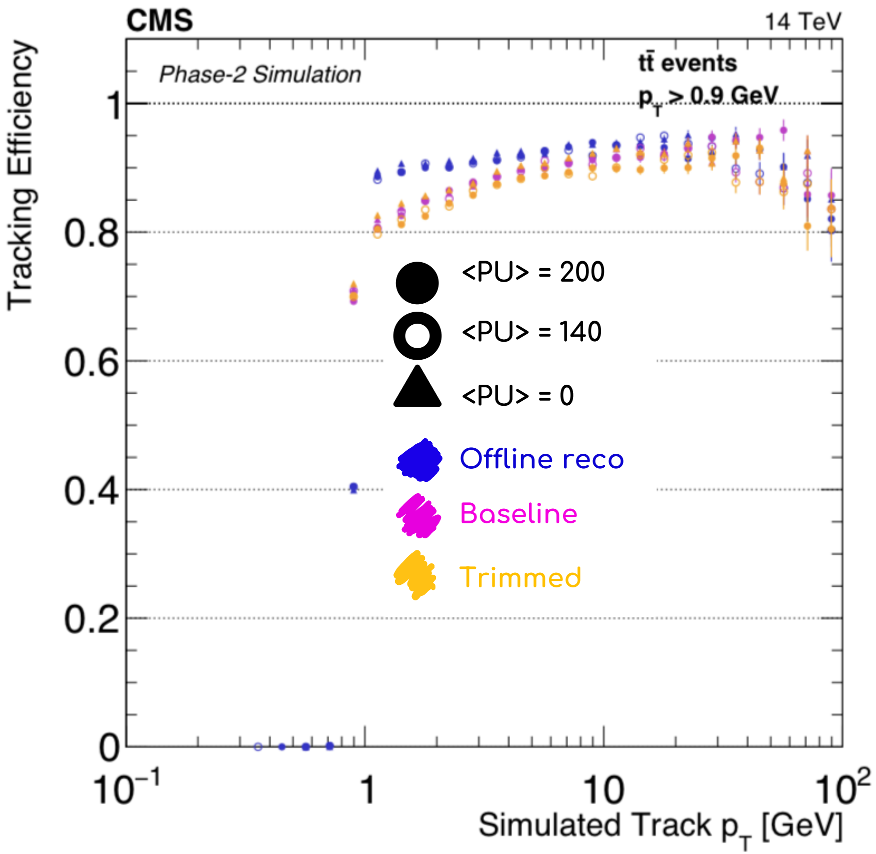


Including PU tracks

Iterative Tracking for Phase II

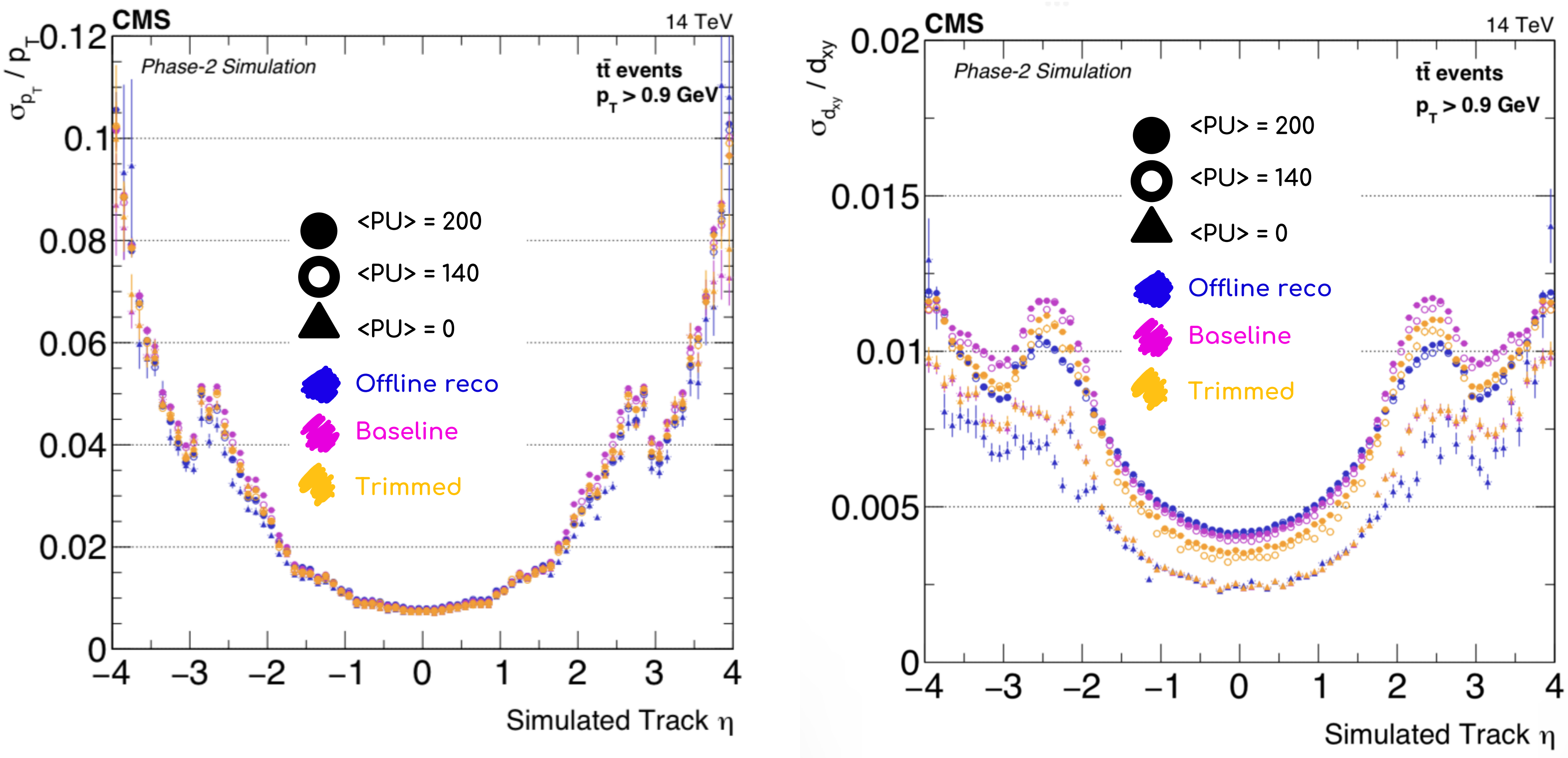
In the Phase-2 Upgrade of the CMS Data Acquisition and High Level Trigger TDR:

- Starting point: offline Phase II track reconstruction.
- Redefining and adapting the iterations to reduce timing. HLT *baseline* tracking configuration with two iterations:
    1. First iteration: seeded by pixel tracks (n_{hits}=4).
    2. Second iteration: seeded by pixel triplets.
- In addition: a *trimmed* configuration (mimicking what is done Run 3) for which the seeds are selected to be compatible with a set of (~10) trimmed vertices.
- Performance of *baseline* competitive with *offline reco* and timing reduced of a factor 6. The *trimmed* configuration brings a furter 20-30% timing reduction.

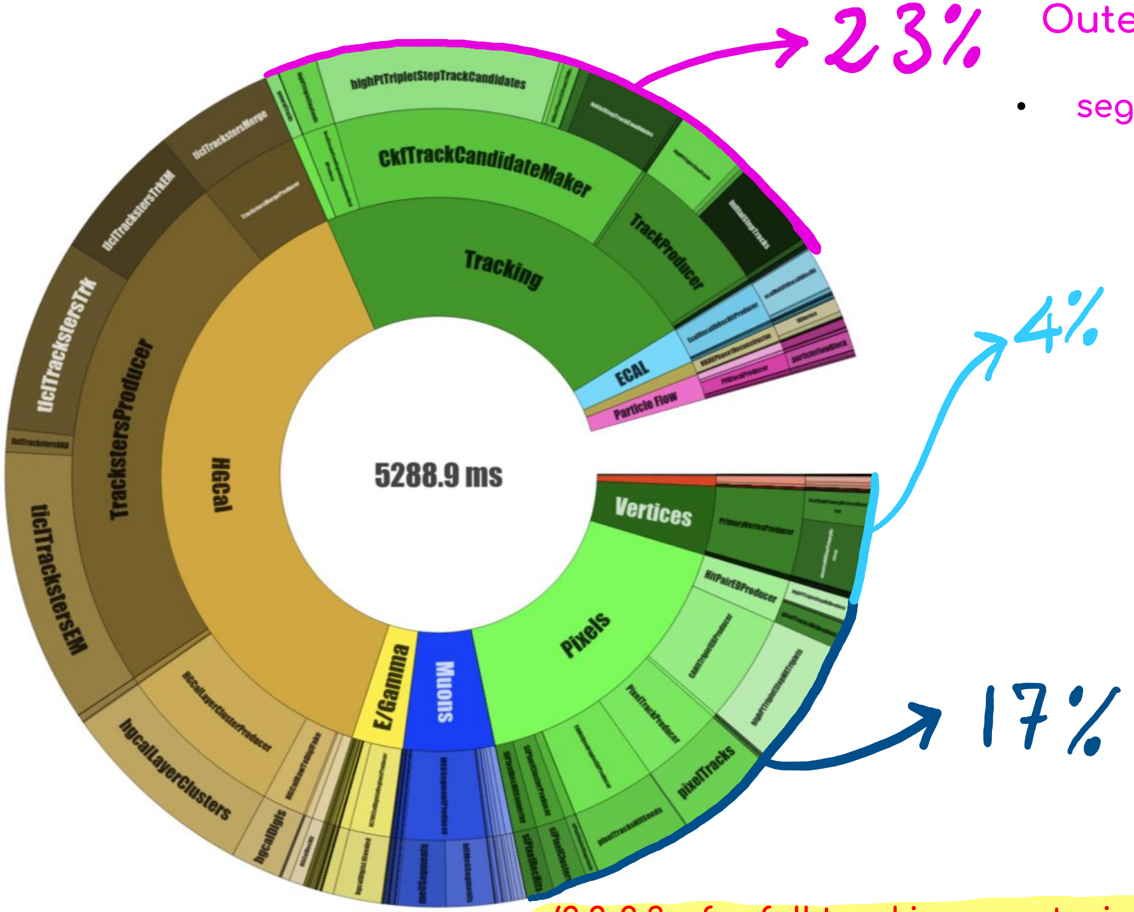


Including PU tracks

Resolutions



Phase II HLT Timings



Outer Tracker Reconstruction

- segment linking (or LST) algorithm

Vertex Reconstruction

- no more "negligible".
- annealing algorithm (clustering).
- adaptive fitting.
- offloadable on GPU?

Pixel detector reconstruction

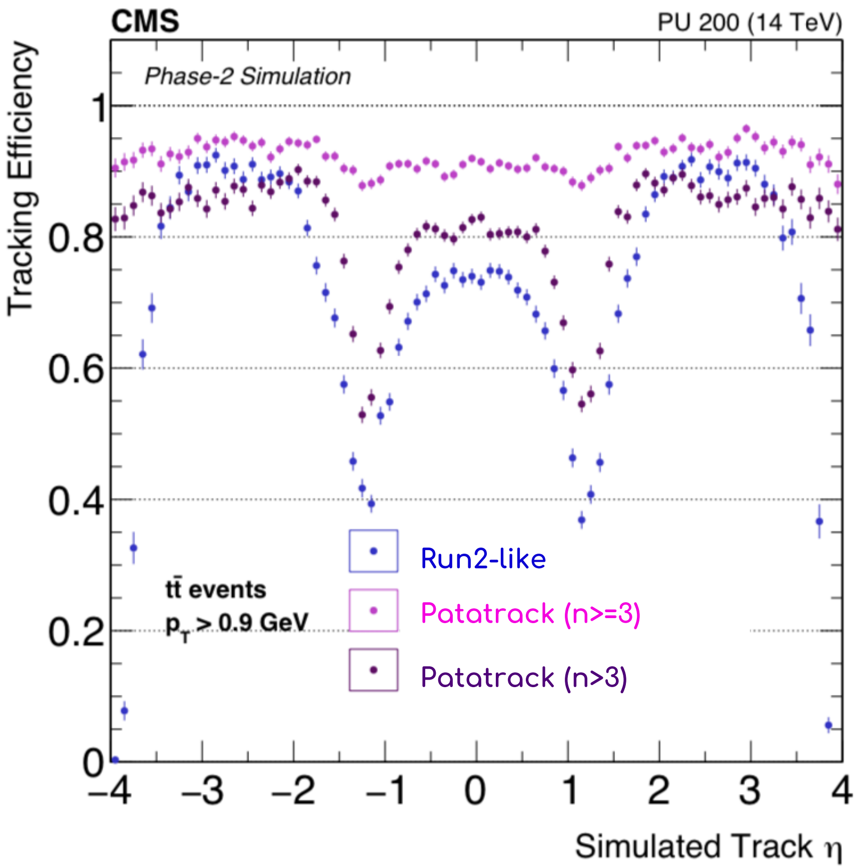
- may profit from patatrack pixel tracking!
- how to go regional?

(2.2-2.3 s for full tracking + vertexing 8 threads 8 streams on an HLT node)

Patatrack Pixel Tracking for PhaseII

Patatatrack Pixel Tracks for PhaseII:

- Profit from developments done for Run3.
- Adapting to the new geometry and PU conditions.
- Tested in the TDR running on CPU.
- Defining a new set of iterations replacing pixel ntuple seeds with pixel tracks.
- Targetting full offload to GPU within the year.

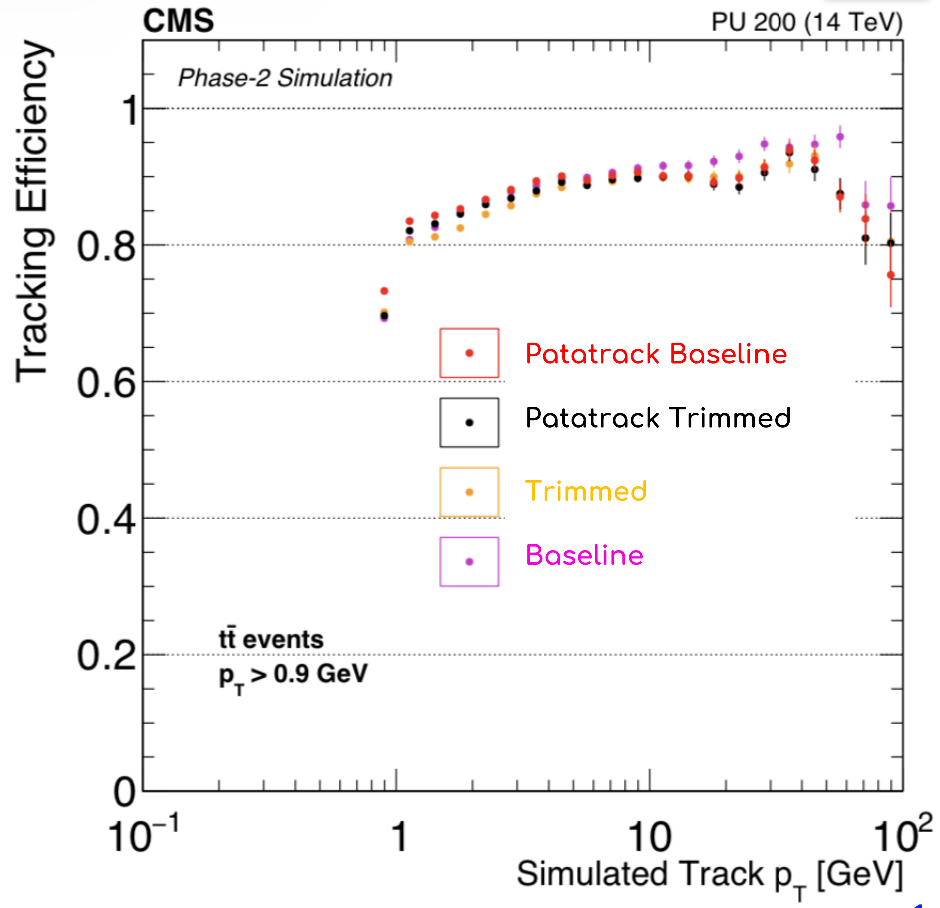


Including PU tracks

Patatrack Pixel Tracking for PhaseII

Patatatrack Pixel Tracks for PhaseII:

- Profit from developments done for Run3.
- Adapting to the new geometry and PU conditions.
- Tested in the TDR running on CPU.
- Defining a new set of iterations replacing pixel ntuple seeds with pixel tracks.
- Targetting full offload to GPU within the year.
- Performance competitive with *baselines* and up to 25% timing reduction (on CPU!) and 43% of tracking is made offloadable on GPU (as a bonus).

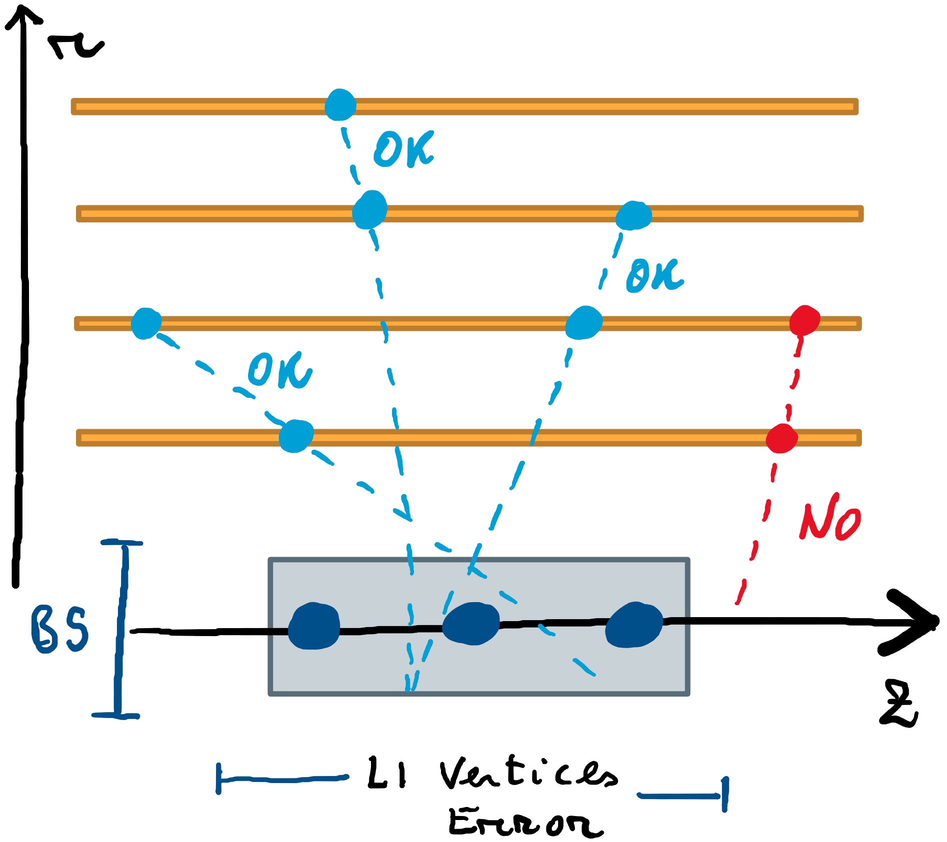


Including PU tracks

+L1 Vertexing Trimming

Patatatrack Pixel Tracks for PhaseII (+L1 Vertexing):

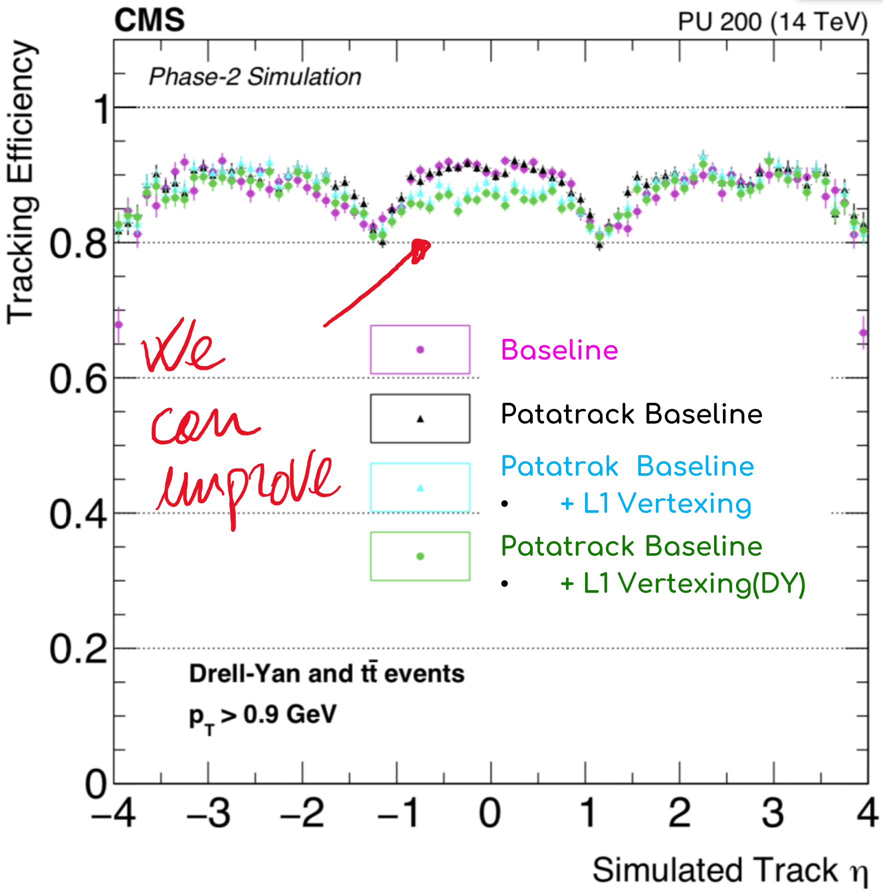
- Patatrack pixel tracks may be reconstructed only globally.
- Through an Level-1 is the histogram based algorithm *FastHisto* which coarsely clusters the tracks during the histogram forming step within fixed bins.
- The three vertices reconstructed with the largest Σp_T^2 are stored.
- These vertices are used to define a region of interest for pixel tracks reconstruction (at the seeding stage).
- Performance competitive with *baselines* and up to 20% in timing reduction. Room for improvement in the barrel.



+L1 Vertexing Trimming

Patatatrack Pixel Tracks for PhaseII (+L1 Vertexing):

- Patatrack pixel tracks may be reconstructed only globally.
- Through an Level-1 is the histogram based algorithm *FastHisto* which coarsely clusters the tracks during the histogram forming step within fixed bins.
- The three vertices reconstructed with the largest Σp_T^2 are stored.
- These vertices are used to define a region of interest for pixel tracks reconstruction (at the seeding stage).
- Performance competitive with *baselines* and up to 20% in timing reduction. Room for improvement in the barrel.



Including PU tracks

Segment Linking (LST)

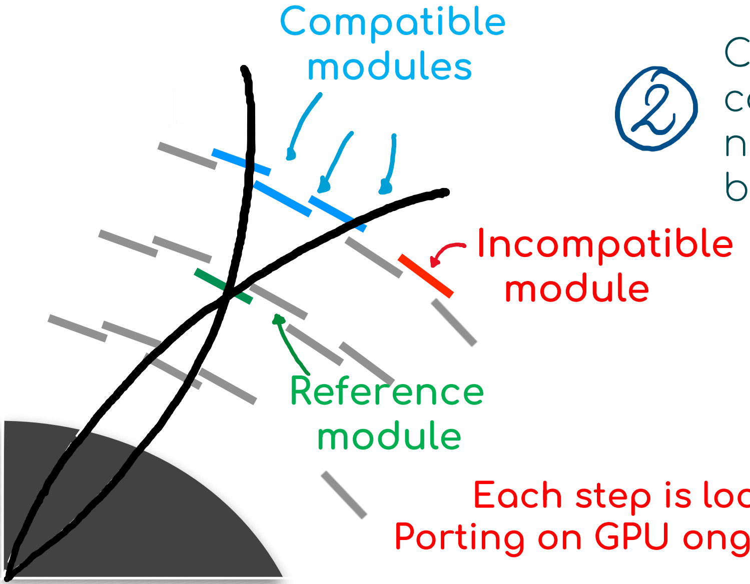
Segment Linking (or Line Segment Tracking) in a nutshell

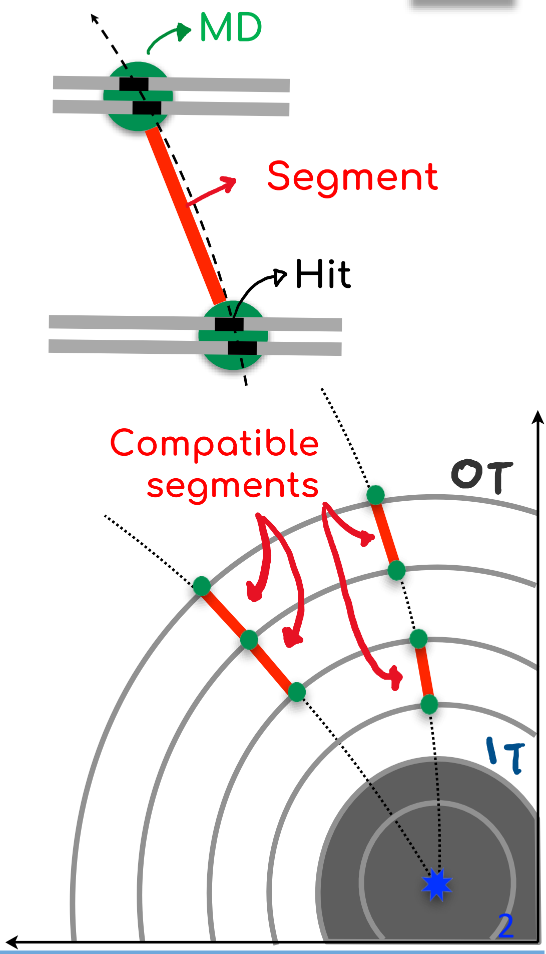
- Algorithm to build tracks in the OT ($|\eta|<2.5$).

(1) Take advantage of new double layered "p_T modules": build mini-doublets (MD) in each layer ($pT > 1$ GeV, complementary to L1 Tracking).

(2) Create line segments connecting compatible MDs in neighboring layers using a pre-built module map.

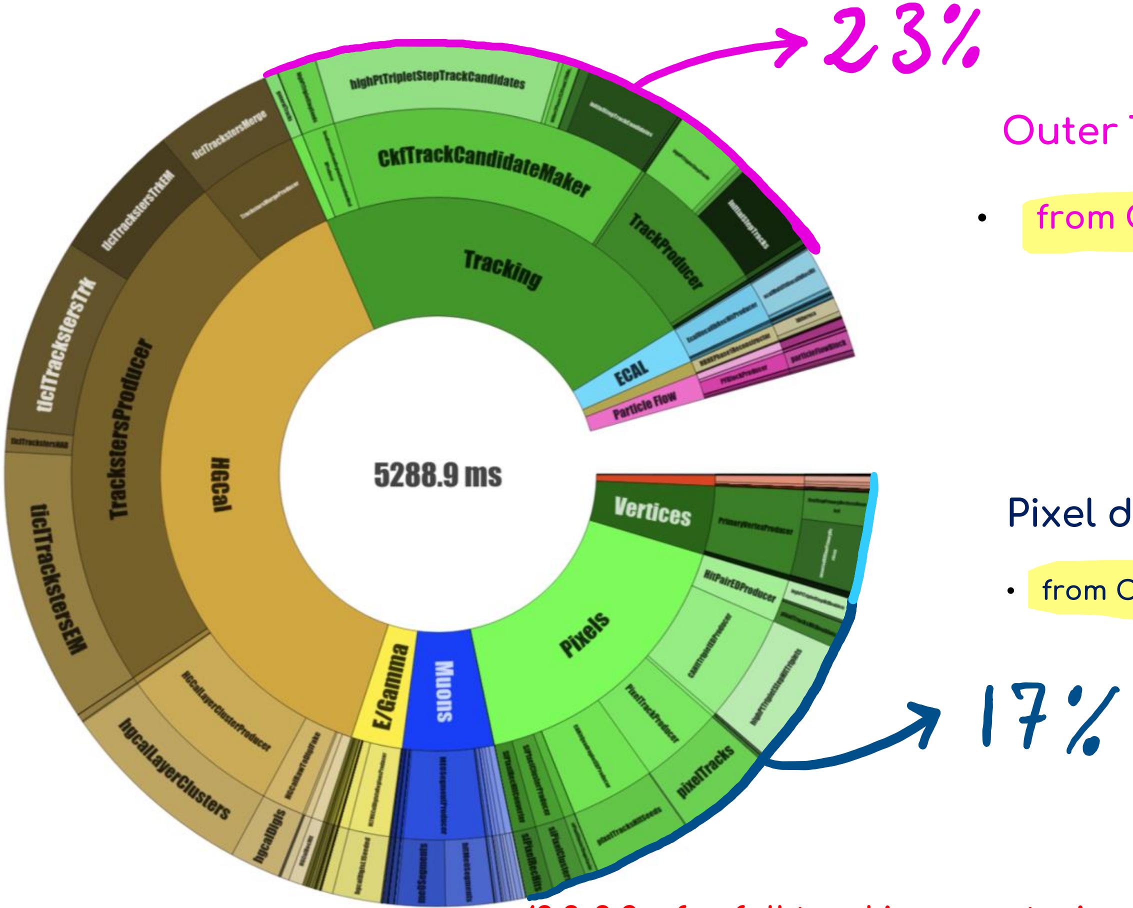
(3) Connect line segments to build tracks.





Each step is localized == Easily parallelizable
Porting on GPU ongoing, promising results yesterday.

Phase II HLT Timings (on GPU-ish)



Outer Tracker Reconstruction

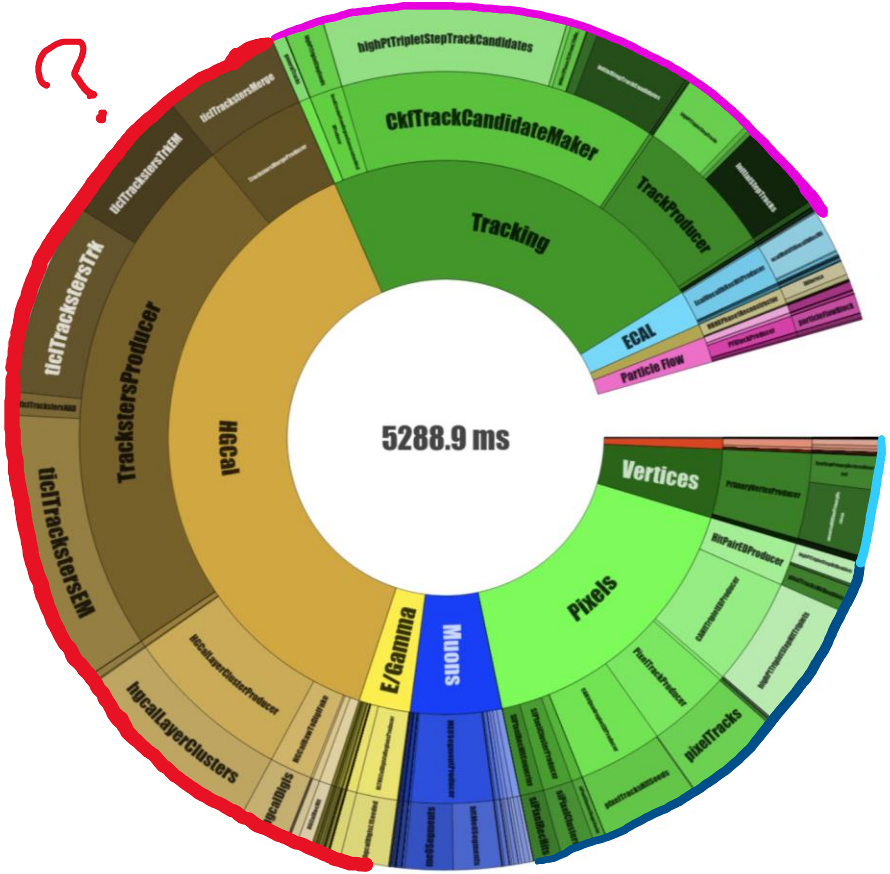
- from O(1s) to O(10s ms)

Pixel detector reconstruction

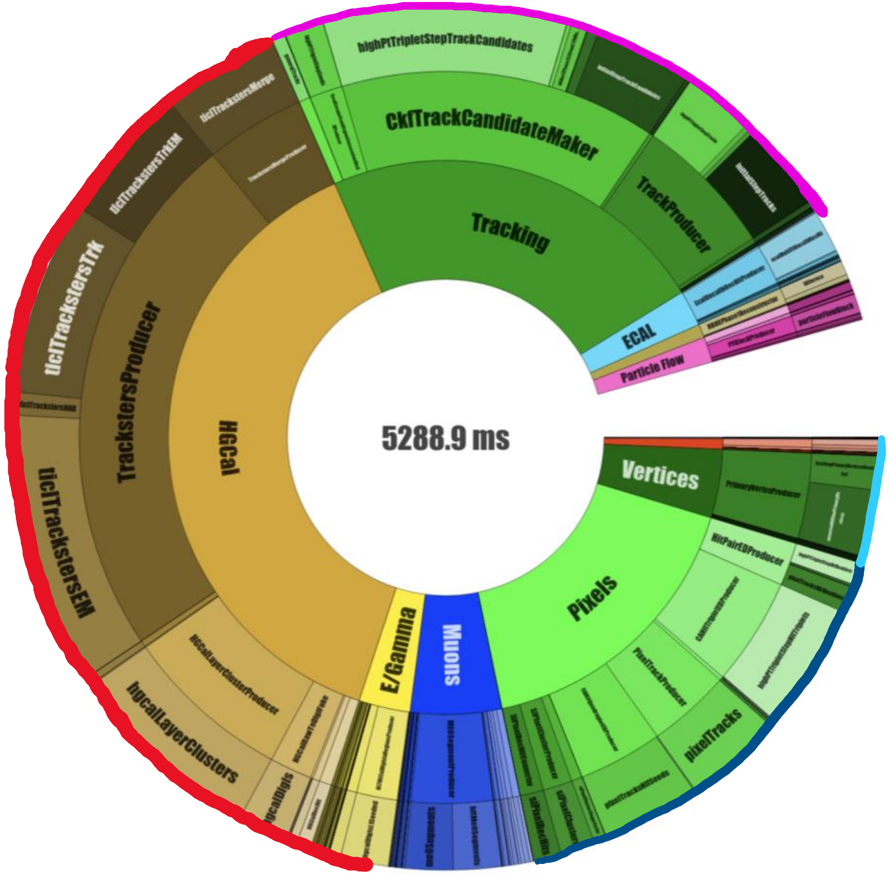
- from O(1s) to O(10s ms)

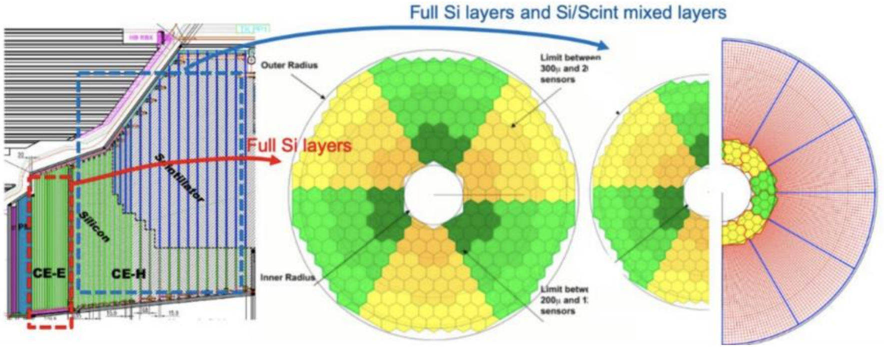
(2.2-2.3 s for full tracking + vertexing 8 threads 8 streams on an HLT node)

Phase II Timings



CMS HGCal





Major CMS Phase2 upgrade.

Silicon sensors (EM + HAD)

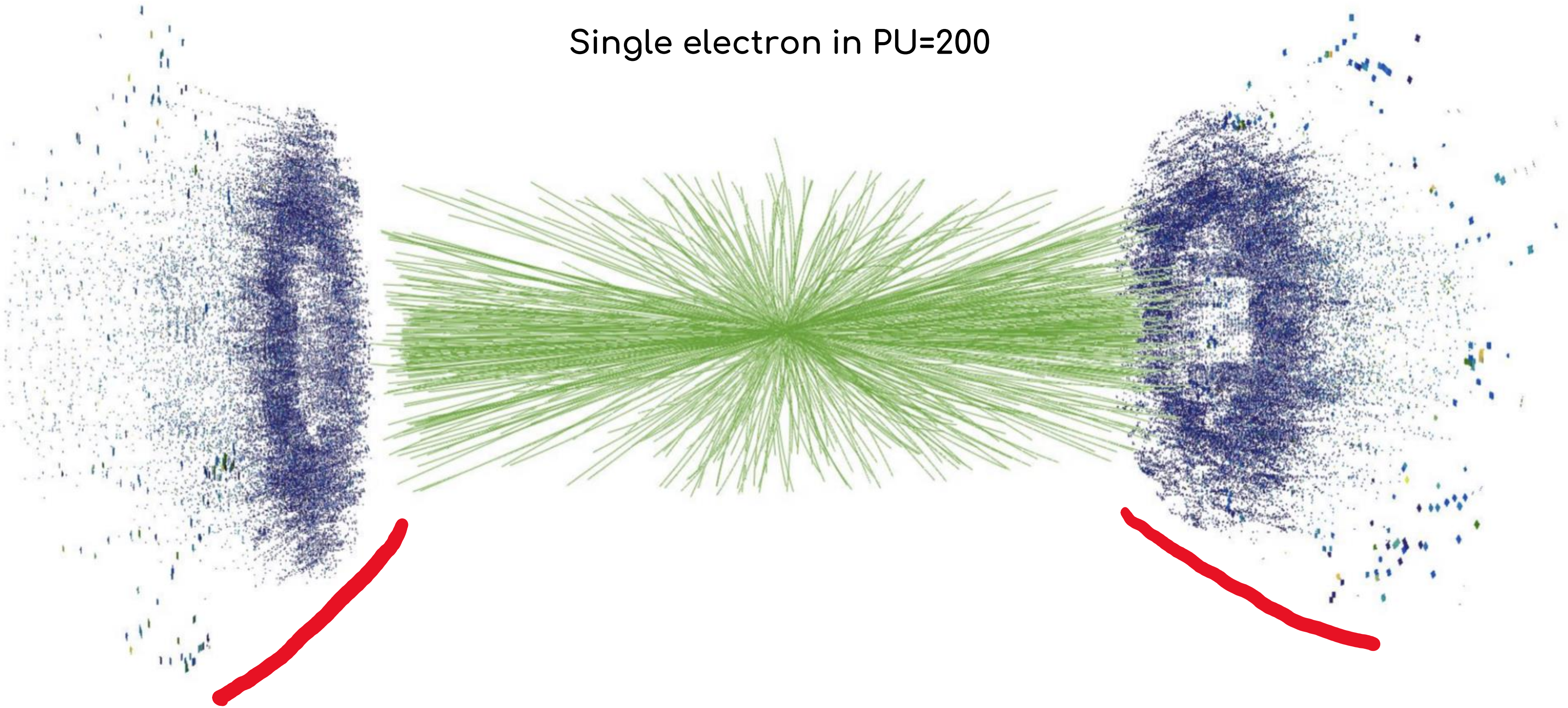
- 28 (EM) + 22 (HAD) layers
- about ~6M channels, cell sizes (about 0.5 cm2 and 1.2 cm2)

Plastic Scintillator + SiPM (HAD)

- 14 layers
- ~4K tiles (~240K channels)

A tiny e^-

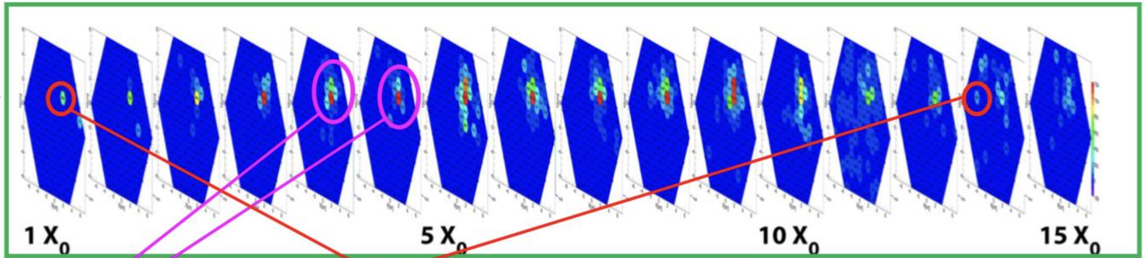
Single electron in PU=200

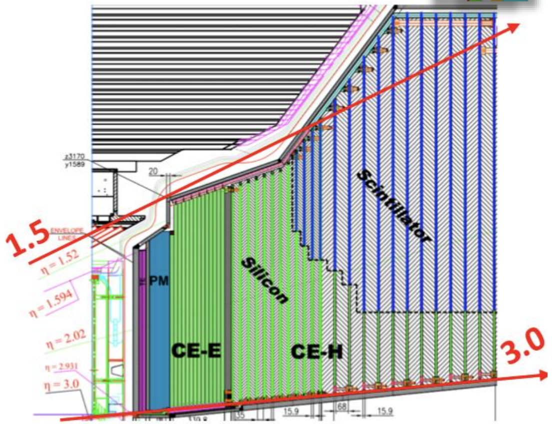


HGCAL Reco in a nutshell

HGCAL: **a new imaging calorimeter (both hadronic & elettromagnetic)** with very fine lateral and longitudinal segmentation, and precision timing capabilities. **Completely new reconstruction needed.**

32 GeV electron



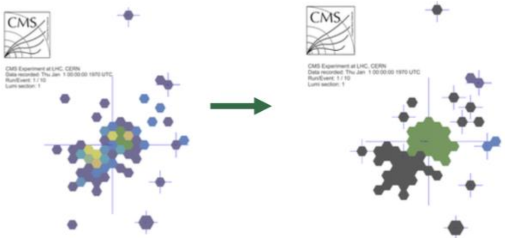


1.RecHits

Reconstructed hits: energy deposit in a sensor.

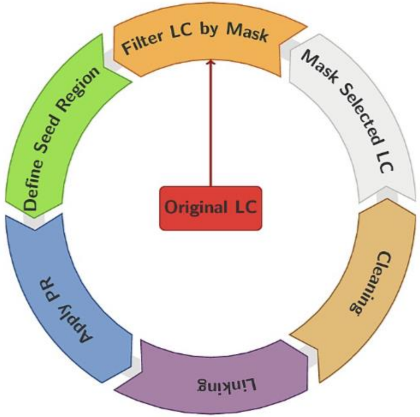
2. 2D Clusters

- 2D clusters (Layer Clusters): RecHits clustered on the same layer.
- **CLU**stering by **E**nergy (CLUE) algorithm. Fast. Using a concept of local energy density. Designed for high PUs. **GPU-friendly.**



3.3D Clusters

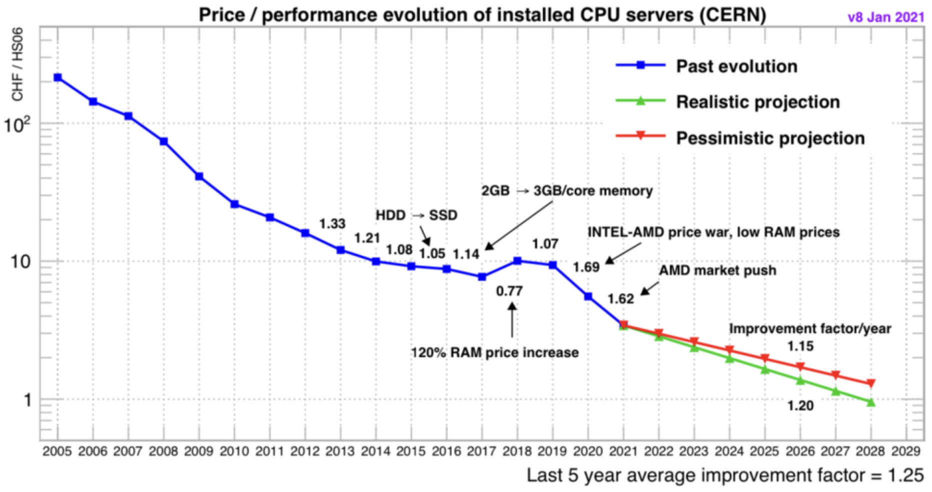
- 3D clusters (tracksters): Layer Clusters from different layers linked.
- The **IT**erative **CL**ustering **framework (TICL)**
  - **Modular;**
  - **Flexible/efficient/versatile;**
  - **Iterative:**
    1. Track seeded EM
    2. Unseeded EM
    3. Track seeded HAD
    4. Unseeded HAD



But why invest so much (*) effort?

(*) O(10) FTE

Let's crunch some numbers



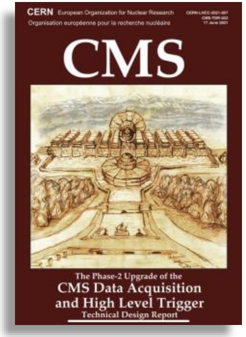
Extrapolation by CERN IT of server price/performance

- based on the servers installed in 2013-2021
- servers for CMS HLT follow the same trend
- **pessimistic: +15% /y**
- realistic: +20% /y

Assuming the same trend for GPU price/performance

- same technology and fabrication process
- compete for the same market
- **observed: +30% /y**
- for A10 (2021) vs T4 (2018)

Porting to accelerators helps? From The Phase-2 Upgrade of the CMS Data Acquisition and High Level Trigger TDR:



CPU-only

- 1.55 CHF/HS06 in 2028

50% code ported

- 0.70 CHF/HS06 in 2028

80% code ported

- 0.22 CHF/HS06 in 2031

	Run-2	Run-3	Run-4	Run-5
peak luminosity	2×10^{34} cm^{-2}s^{-1}	2×10^{34} cm^{-2}s^{-1}	5×10^{34} cm^{-2}s^{-1}	7.5×10^{34} cm^{-2}s^{-1}
pileup	50	50	140	200
HLT input rate	100 kHz	100 kHz	500 kHz	750 kHz
HLT output rate	1 kHz	< 2 kHz	5 kHz	7.5 kHz
HLT farm size	0.7 MHS06	0.8 MHS06	16 MHS06	37 MHS06

Lesson 1 : SoA

- SoAs improve access to global memory and exploit CPU vectorization.
- Device data uses the SoA format (easy kernel mapping).
- Takes advantage of **memory coalescing** and **warp alignment.**
- **Fixed size**: template geometry, conditions.
- **CMS** is currently investigating a good SoA-abstraction implementation.

```c
//Structure of Arrays
struct pointlist3D {
  float x[N];
  float y[N];
  float z[N];
};
struct pointlist3D points;
float get_point_x(int i) {
     return points.x[i]; }
```

```c
//Array of Structures
struct point3D {
  float x;
  float y;
  float z;
};
struct point3D points[N];
float get_point_x(int i) {
     return points[i].x; }
```

Lesson II : CPU Fallback

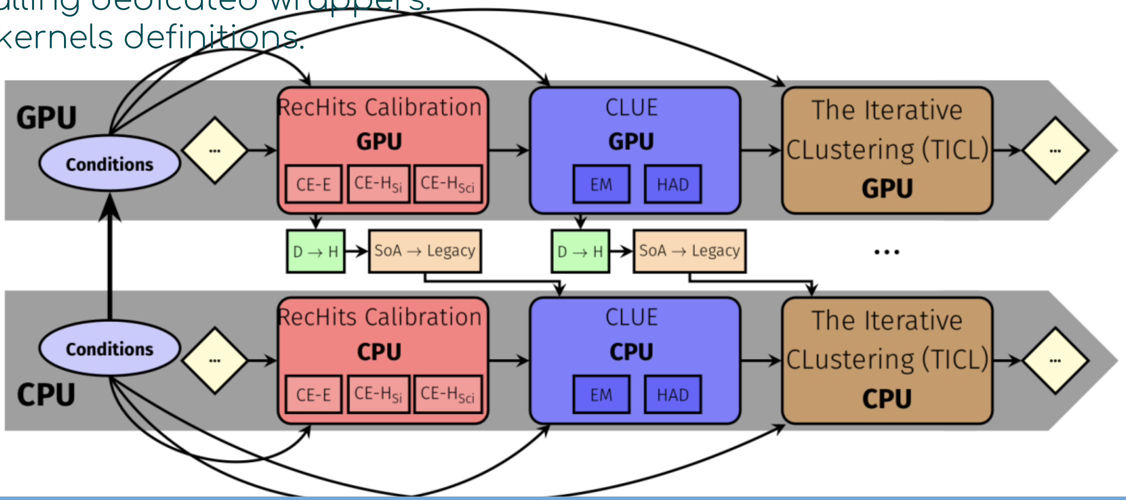
Full reconstruction chain designed to be runnable on **both CPU and GPU** depending on the accelerator availability.

Configuration-wise:

- Different modules run on CPUs and GPUs, where conditions are deployed.
- GPU → CPU data conversion modules bring **flexibility** and ease **validation**.
- User transparent.

Development-wise:

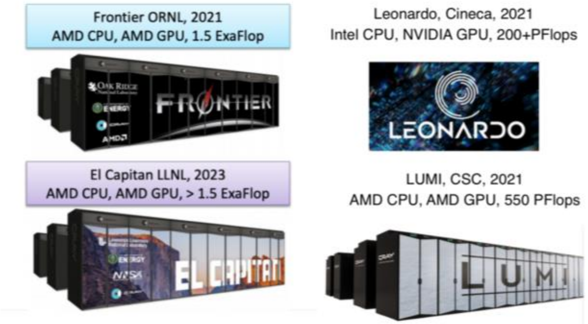
- Producer modules calling dedicated wrappers.
- CPU and GPU share kernels definitions.



Lesson III : Portability

Portability: support multiple accelerator platforms with minimal changes to code base.

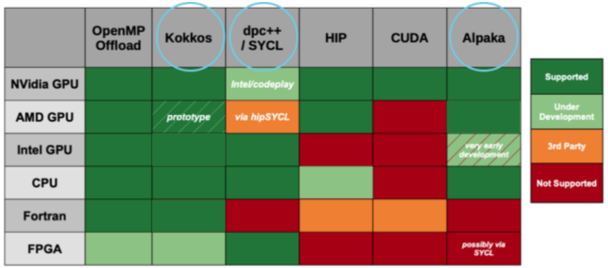
- **Rewriting the same code for each architecture is not feasible**
- **Easier maintenance**
- **Avoid vendor lock-in!**
- Going to offline distributed reconstruction means «heterogeneity», also: HPCs (5% for CMS in 2019-2020)!



A complete C++ standard for heterogeneous computing is **way in the future**. Need to rely on portability layers:

- Kokkos, Alpaka

In Run 3 timescale:
- Given the use cases, we require the portability layer to have good CPU and CUDA backend
- Migrate CUDA GPU codes to use portability layer

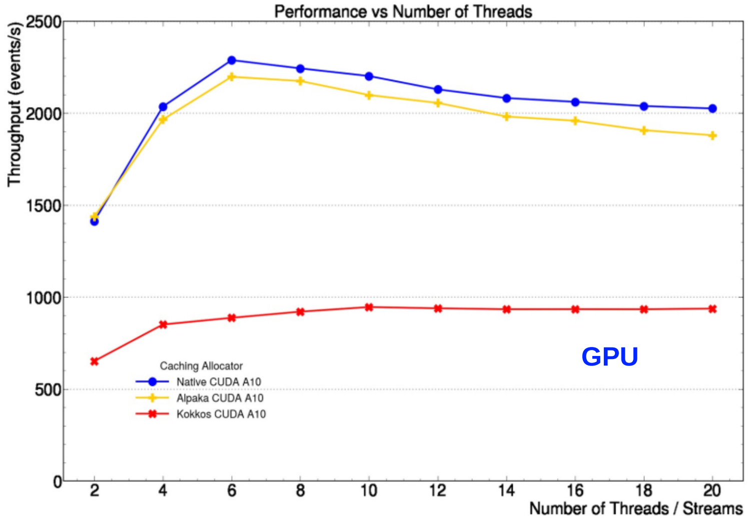


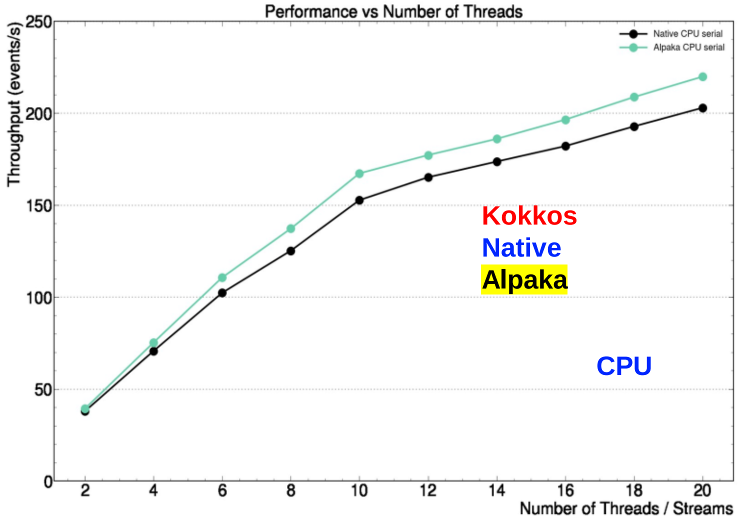
In Run 4 timescale:
- Support as much architectures as we can
- Landscape (software & hardware) maybe very different by then: no decision casted in stone.
- May need to think beyond GPUs (FPGAs?)

alpaka kokkos

Lesson III : Portability

- Patatrack and HEP-CCE's pixeltrack-standalone project (git)
    - prototype different data structures user friendly SoA abstractions
    - port to **different backends**
    - CMSSW independent
    - test **different performance portability solutions:** Kokkos, Alpaka
- Throughput results for the patatrack-standalone prototype:





alpaka achieves ~ 95% of the native performance for the CUDA and CPU backends
It has been chosen as Run-3 perfromance portability layer.

Summary? Further Readings

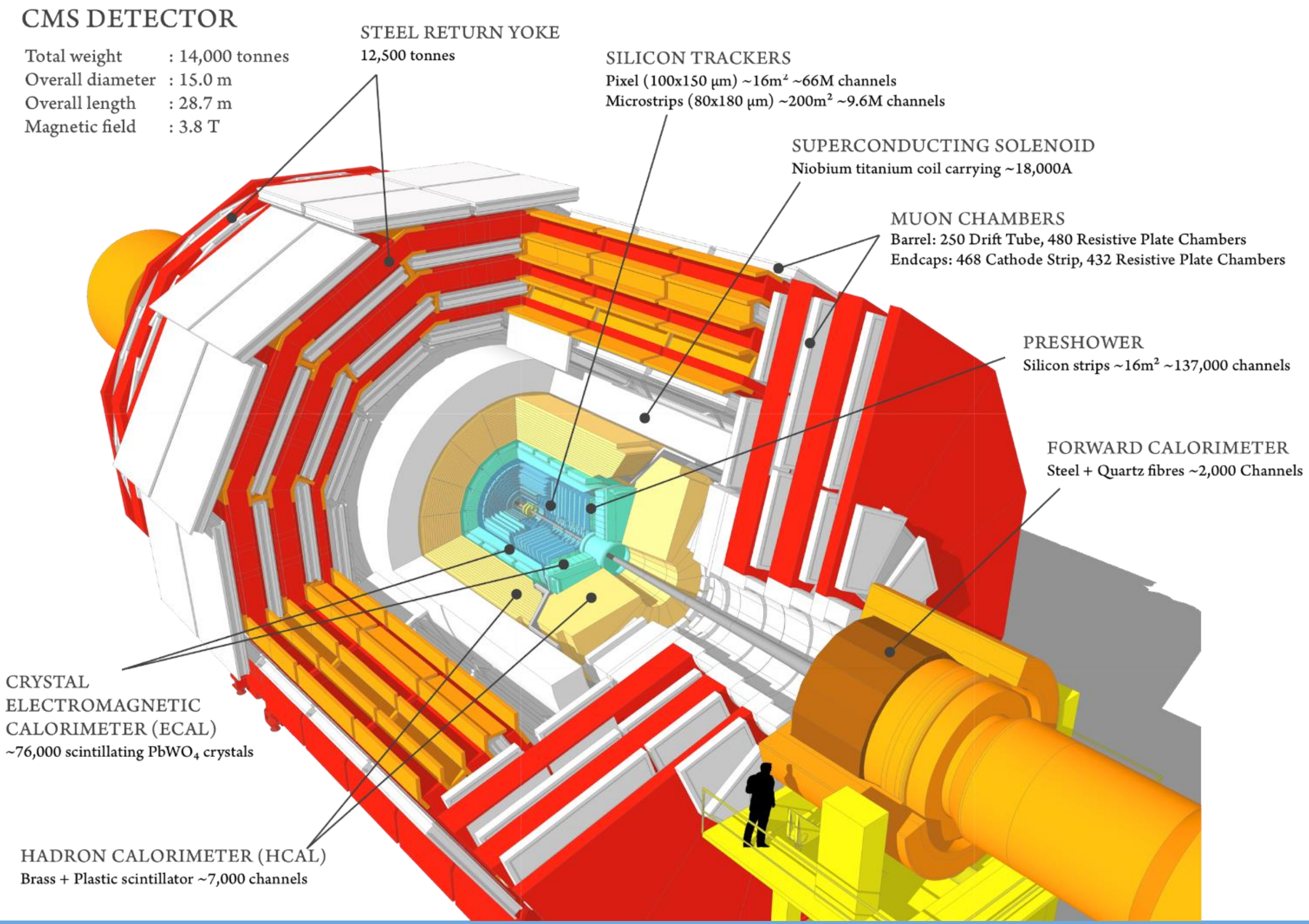
- Performance portability for the CMS Reconstruction with Alpaka
- Clustering in the Heterogeneous Reconstruction Chain of the CMS HGCAL Detector
- Developing GPU-compliant algorithms for CMS ECAL local reconstruction during LHC Run 3 and Phase 2
- CLUE: a clustering algorithm for current and future experiments
- The Iterative Clustering framework for the CMS HGCAL Reconstruction
- Patatrack standalone
- Compute Accelerator Forum / HSF Reconstruction and Software Triggers - Patatrack and ACTS
- CMS Phase2 CMS TDR
- Reproducibility
- Validating GPU and CPU workflows
- Run3 HLT Plots

Thanks!

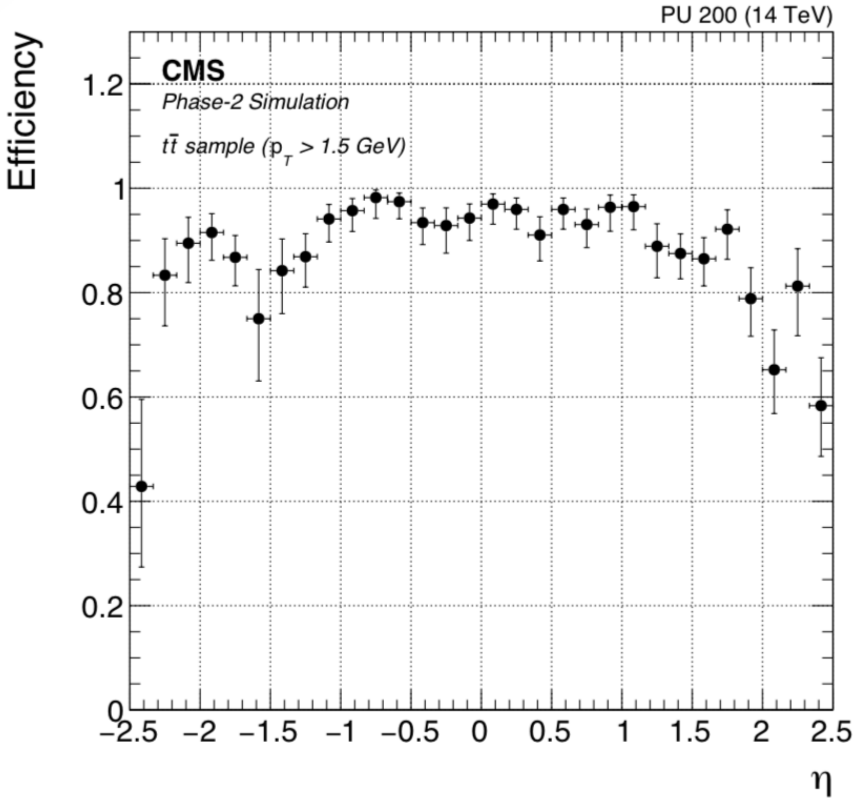
Backup

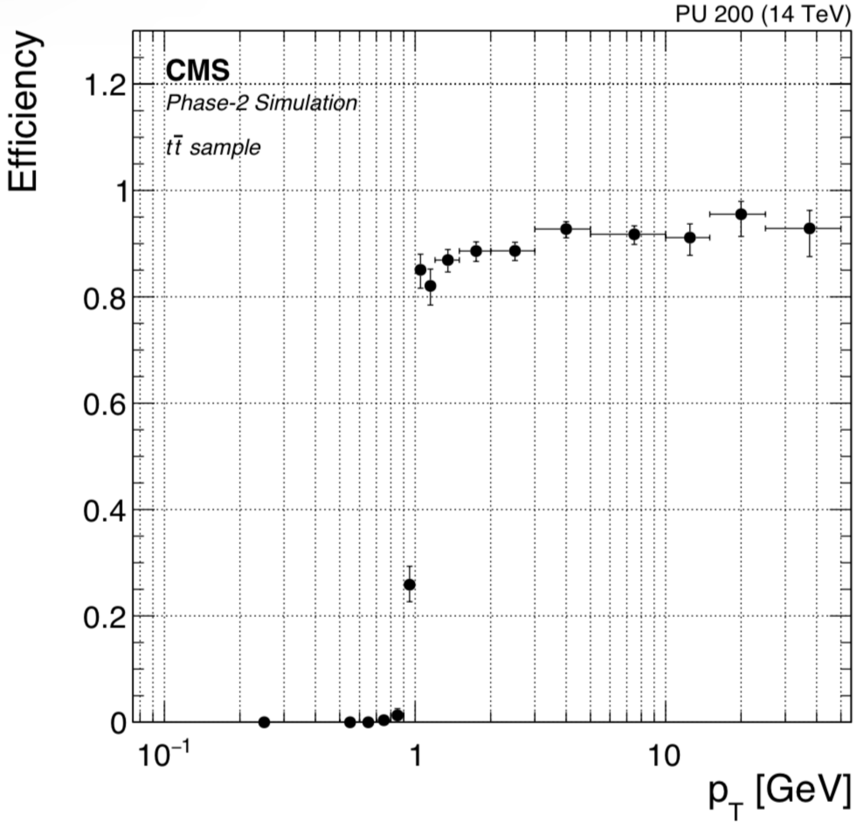
CMS Detector

•



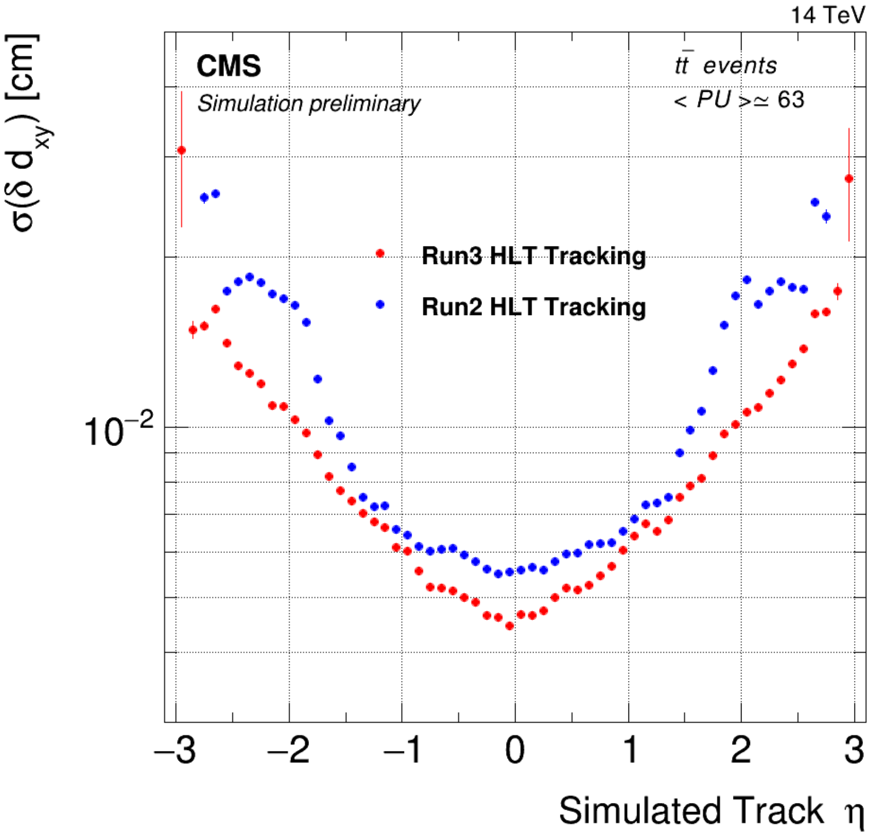
Segment Linking - Efficiencies

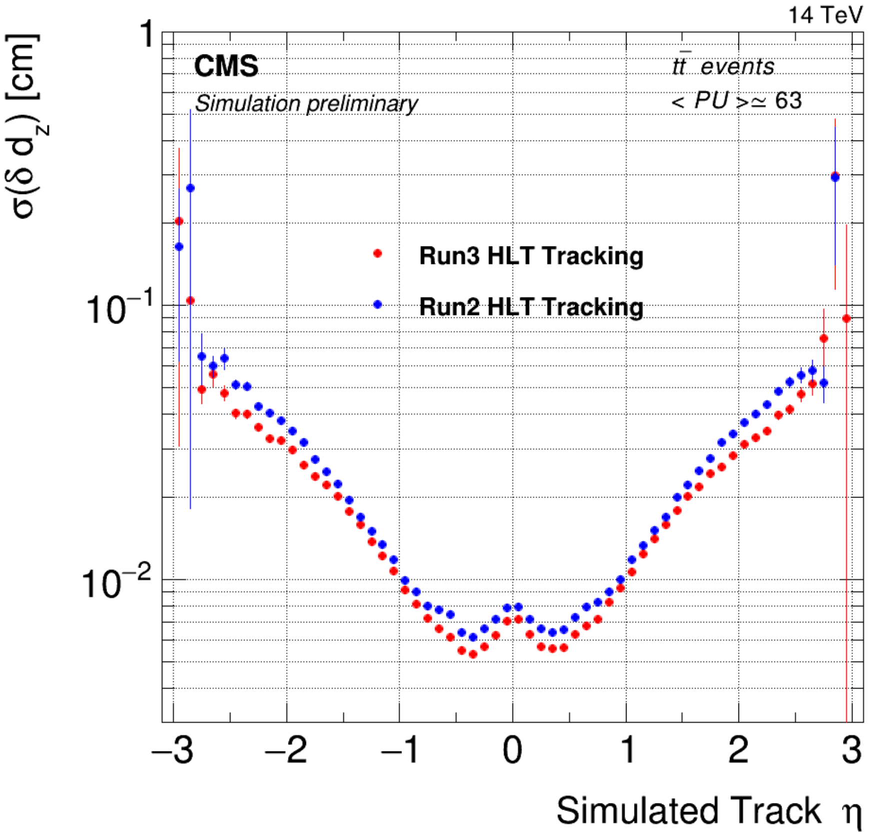




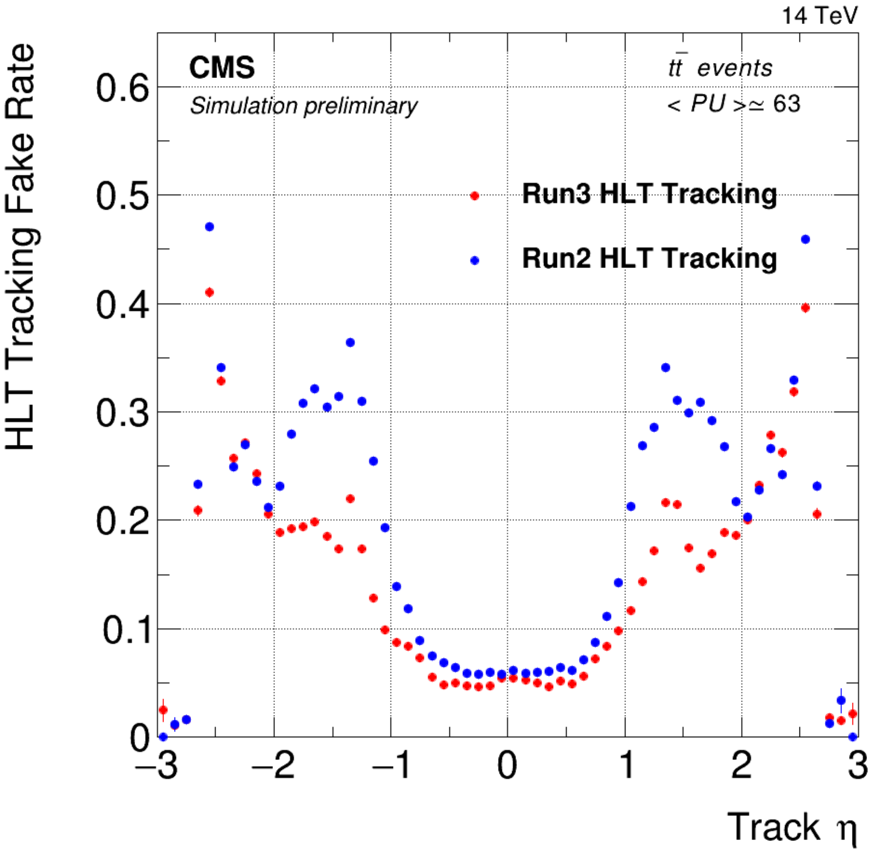
caveat: these are the latest public plots from TDR. Much improvement in the last year

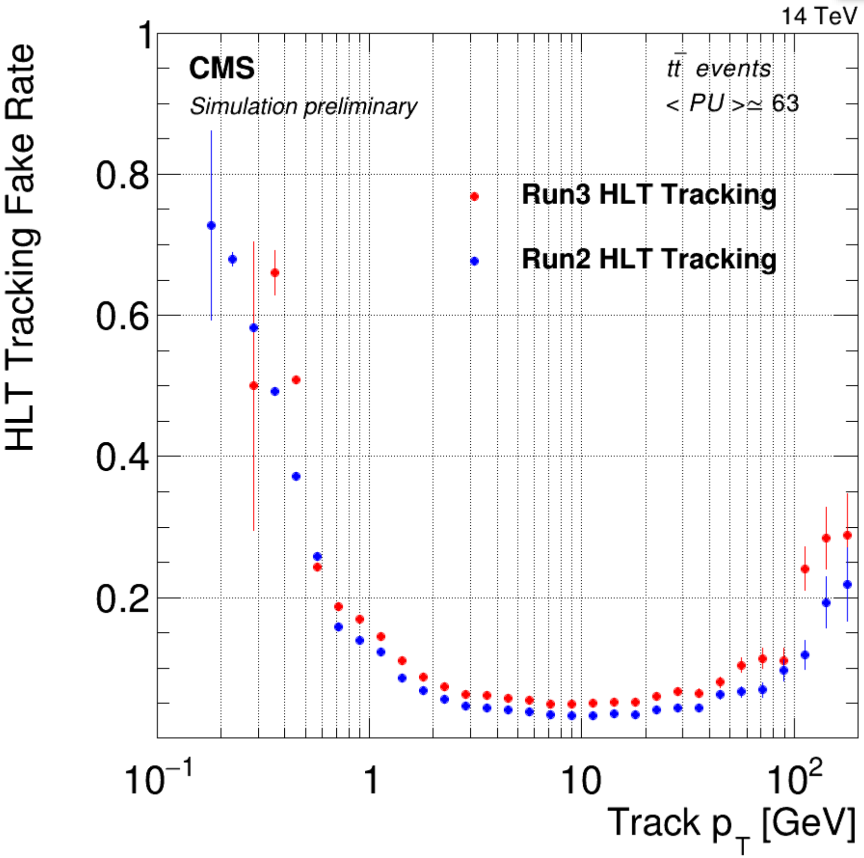
Full Track - Resolution



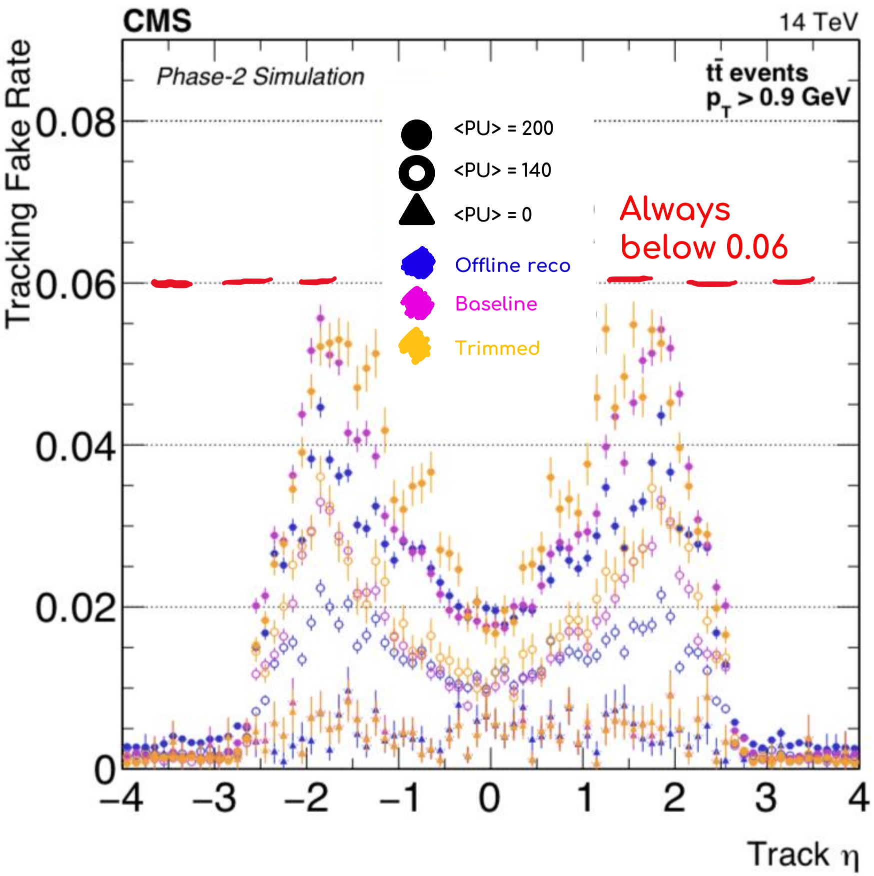


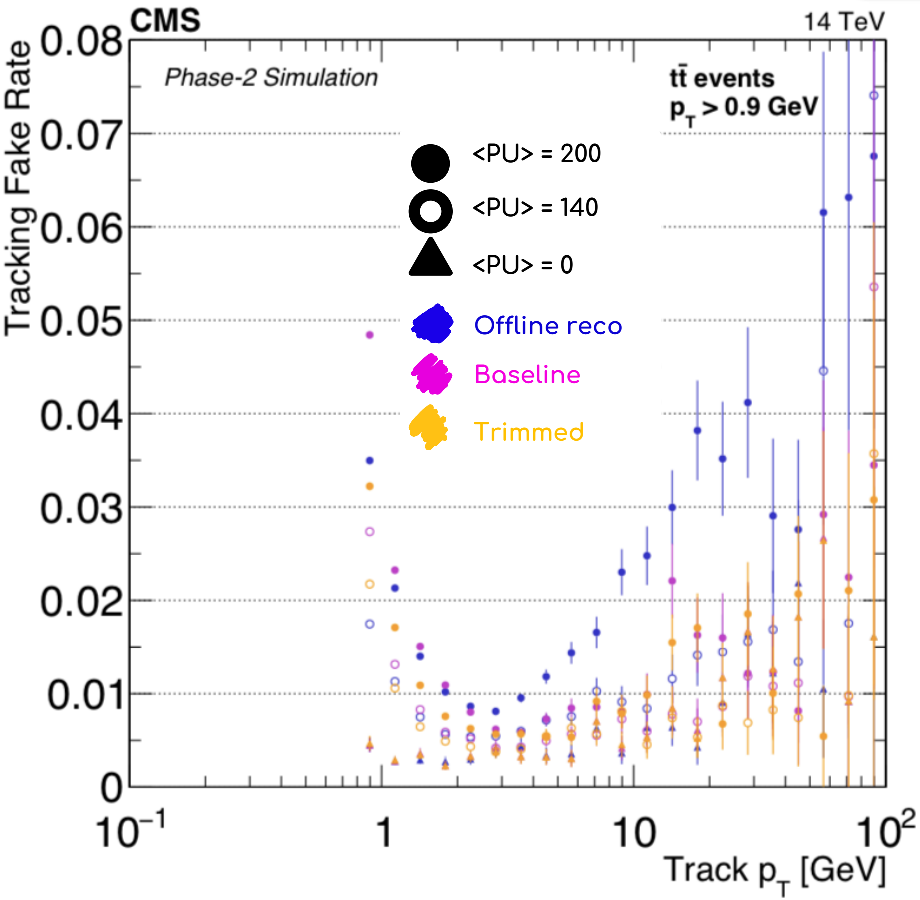
Full Track - Fake Rate





Fake rate



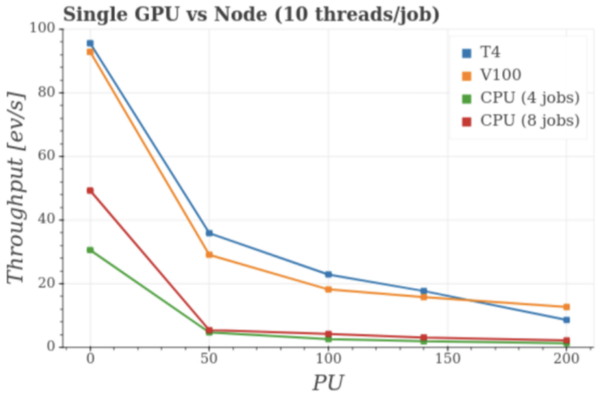


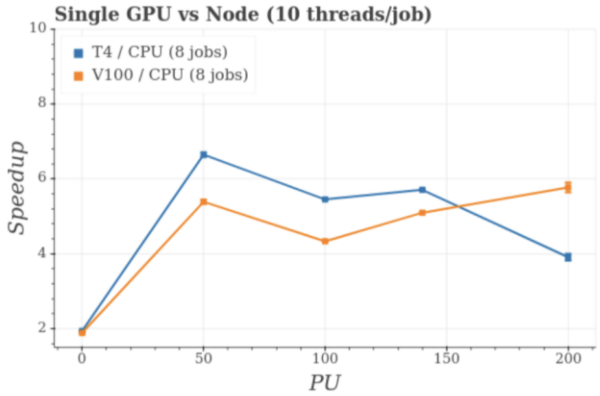
Including PU tracks

CLUE on GPU

Performance of RecHit Calibration [4] + CLUE: Throughput and Speedup

⇒ ***Full*** Intel(R) Xeon(R) Silver 4114 with 40 logical cores vs. ***Single GPU*** (T4-16GB or V100-32GB), 10 CPU threads per job, 512 GPU threads per block





⇒ The speedup peaks between ***5*** and ***6*** for PU140 (Run4) and PU200 (Run5)

⇒ Additional measurements allows to conclude that ***data conversion modules and recursion functions do not affect the throughput***: CLUE is the bottleneck

17 / 22