Accelerated tracking using GPUs at CMS High Level Trigger

Andrea Bocci, Elena Corni, Adriano Di Florio, Sushil Dubey, Shashi Dugad,
Matti Kortelainen, Vincenzo Innocente, Dario Menasce, Marco Rovere, Mia Tosi,
Felice Pantaleo

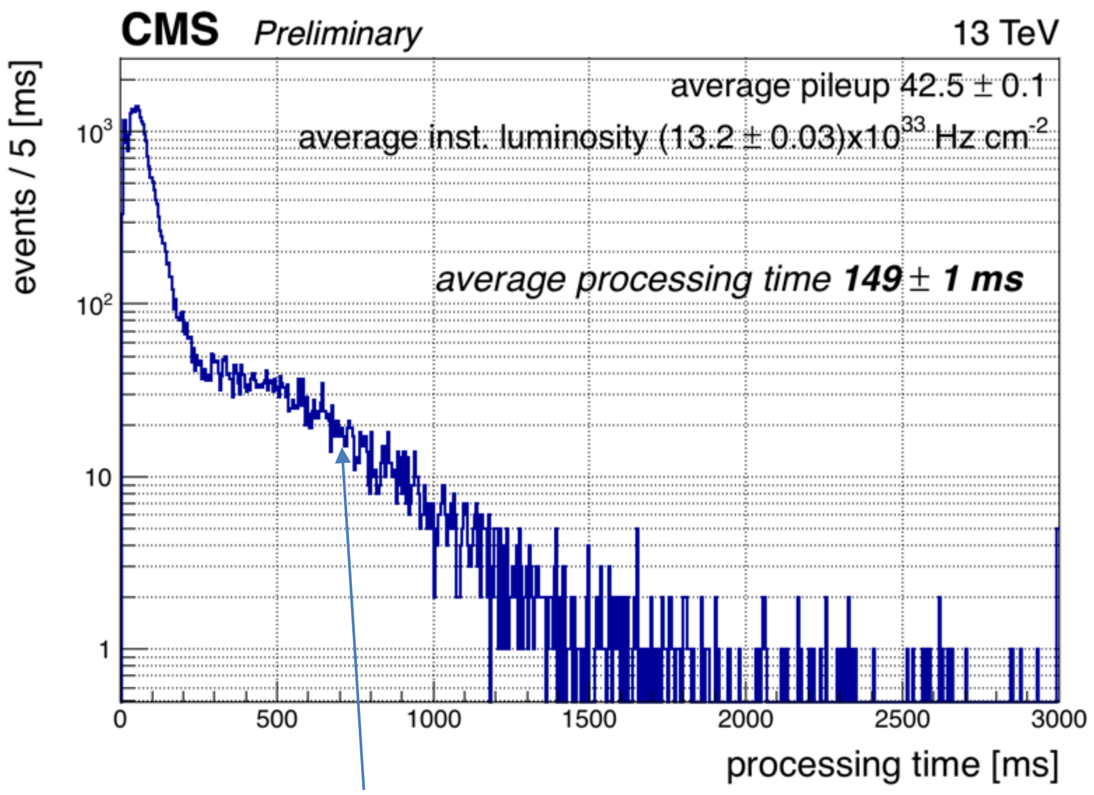
On behalf of the CMS Collaboration

felice@cern.ch

Bundesministerium
für Bildung
und Forschung

GPUs at HLT during run 3

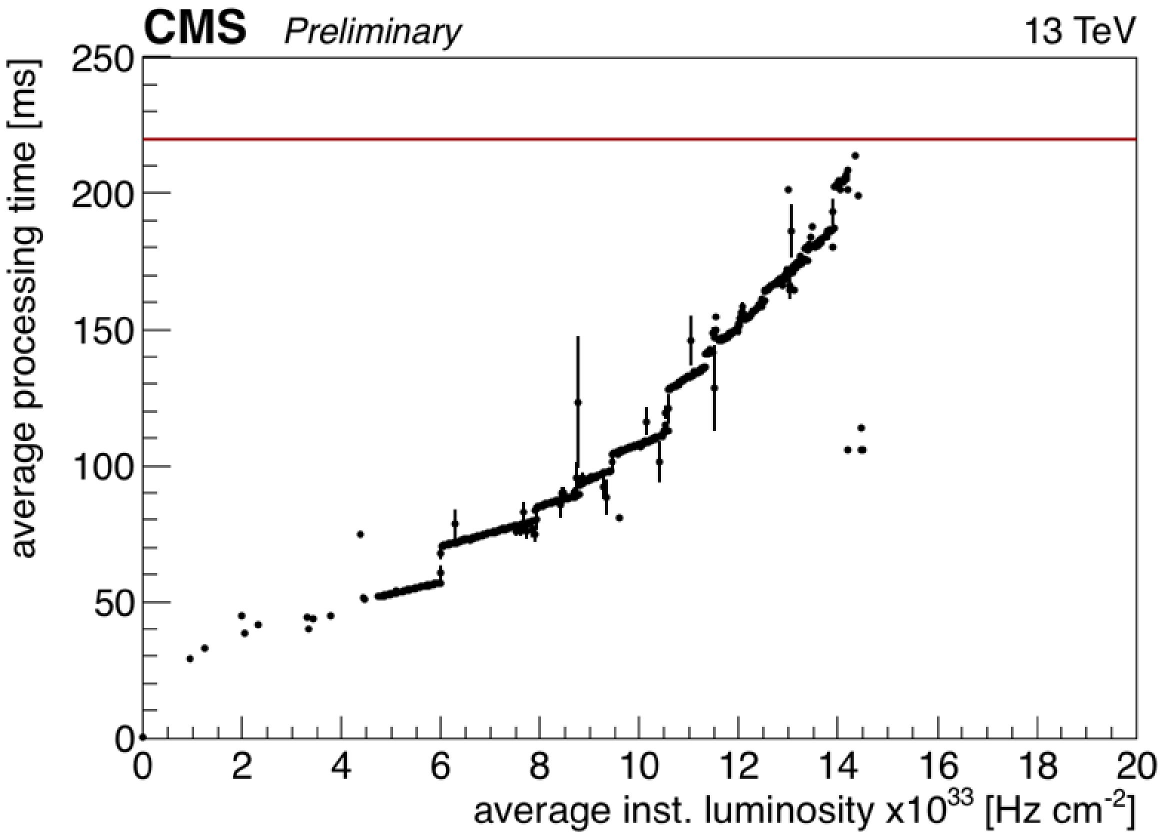
- Today the online farm consists of ~20k Intel Xeon cores
  - One event per logical core
- At the moment tracks are not reconstructed for all the events at the HLT
  - In 2016: 64% and 44% of events run pixel clustering and pixel tracking respectively
- This will be even more difficult at higher pile-up
  - Combinatorics time in seeding O(μ!)
  - More memory/event
- Profit from the end-of-year upgrade of the Pixel to redesign the seeding code
  - Exploiting the information coming from the 4^{th} layer would improve efficiency, b-tag, IP resolution
- GPUs are becoming wider
  - Thousands of threads on the fly
- Future-proof solution: scaling parallel algorithms inside the event



full track reconstruction and particle flow e.g. jets, tau

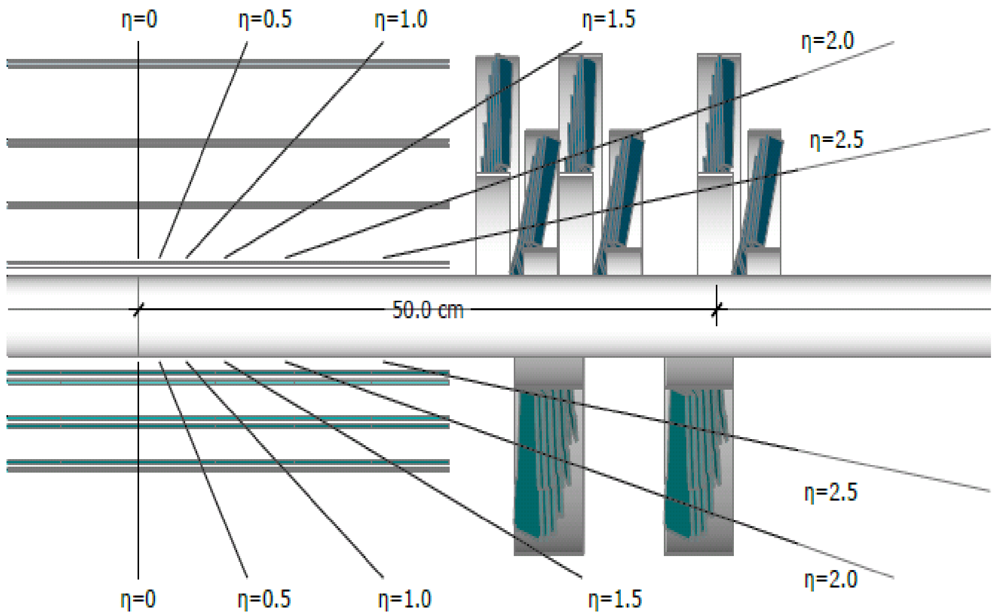
GPUs at HLT during run 3

- Today the online farm consists of ~20k Intel Xeon cores
  - One event per logical core
- At the moment tracks are not reconstructed for all the events at the HLT
  - In 2016: 64% and 44% of events run pixel clustering and pixel tracking respectively
- This will be even more difficult at higher pile-up
  - Combinatorics time in seeding O(μ!)
  - More memory/event
- Profit from the end-of-year upgrade of the Pixel to redesign the seeding code
  - Exploiting the information coming from the 4th layer would improve efficiency, b-tag, IP resolution
- GPUs are becoming wider
  - Thousands of threads on the fly
- Future-proof solution: scaling parallel algorithms inside the event



From RAW to Tracks during run 3

- Trigger avg latency should stay within 220ms
- Reproducibility of the results (bit-by-bit equivalence CPU-GPU)
- Integration in the CMS software framework



- Ingredients:
  - Massive parallelism within the event
  - Independence from thread ordering in algorithms
  - Avoid useless data transfers and transformations
  - Simple data formats optimized for parallel memory access
- Result:
  - A GPU based application that takes RAW data and gives Tracks as result

Algorithm Stack

Raw to Digi

↓

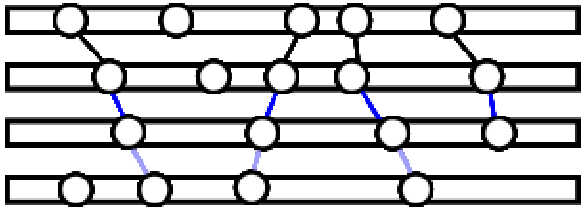
Hits - Pixel Clusterizer

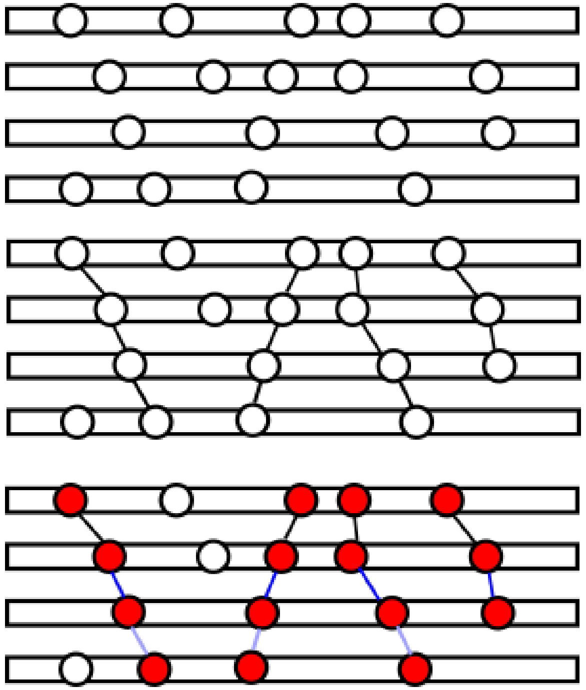
↓

Hit Pairs – FKDTree*

↓

Ntuplets - Cellular Automaton



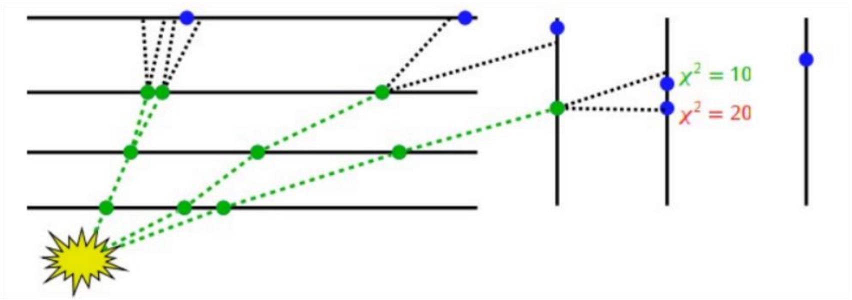


*See talk “Fast GPU Nearest Neighbors search algorithms for the CMS experiment at LHC”

Algorithms Compared

Triplet propagation

Propagate 1-2-3 triplet to 4th layer and search for compatible hits



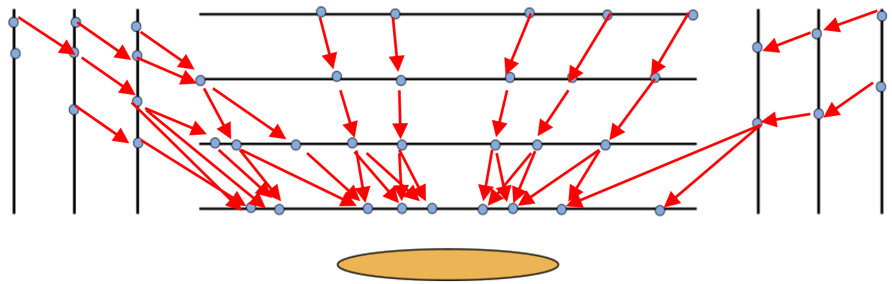
Natural continuation of the current approach from pairs to triplets

Cellular Automaton

Create hit pairs from pairs of adjacent layers

Join compatible pairs that share hits

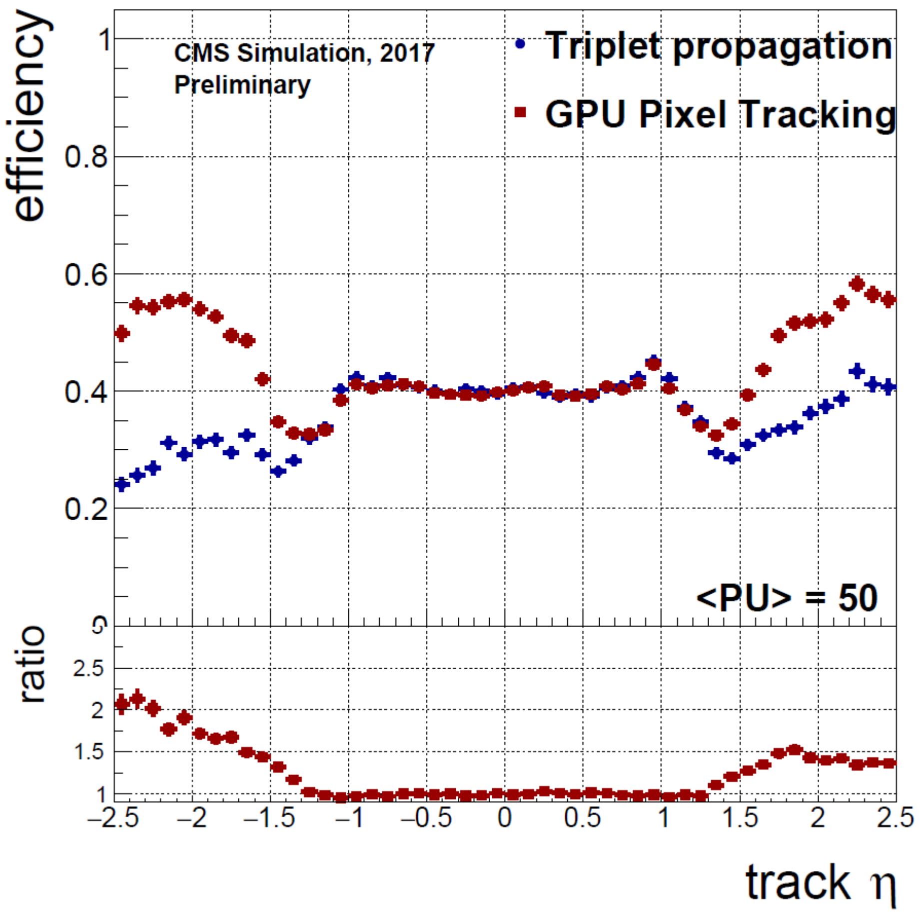
Compatibility checked

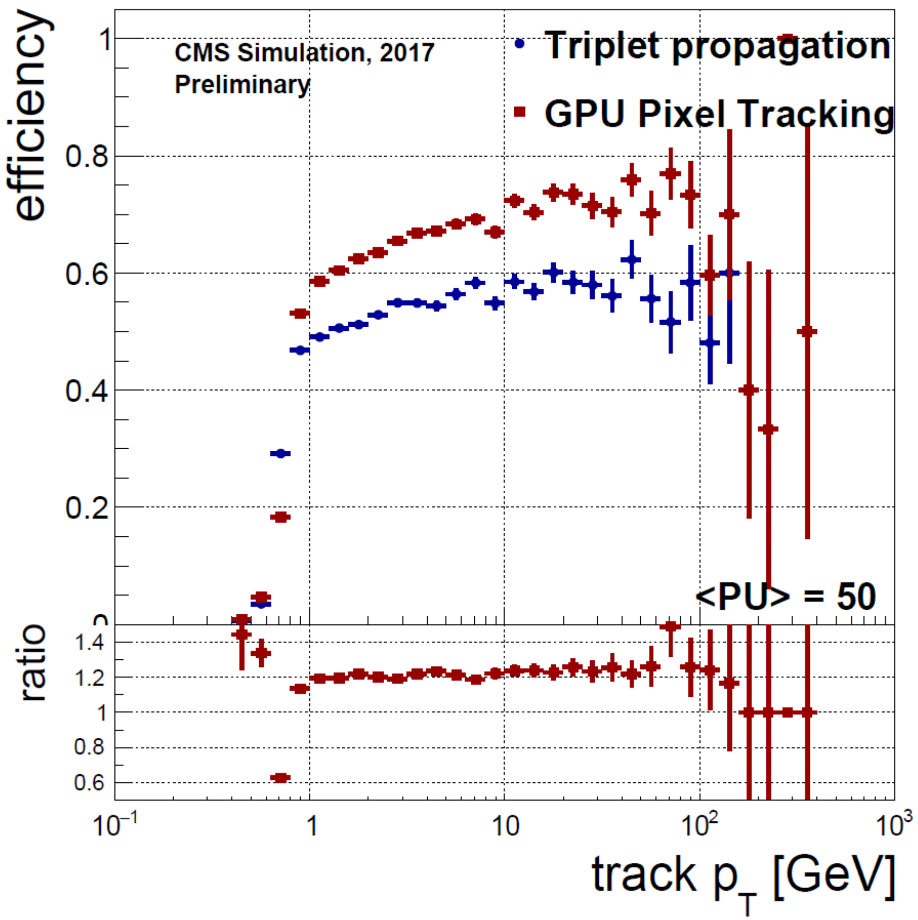


Evolution step, analogous to Game of Life, creates quadruplets

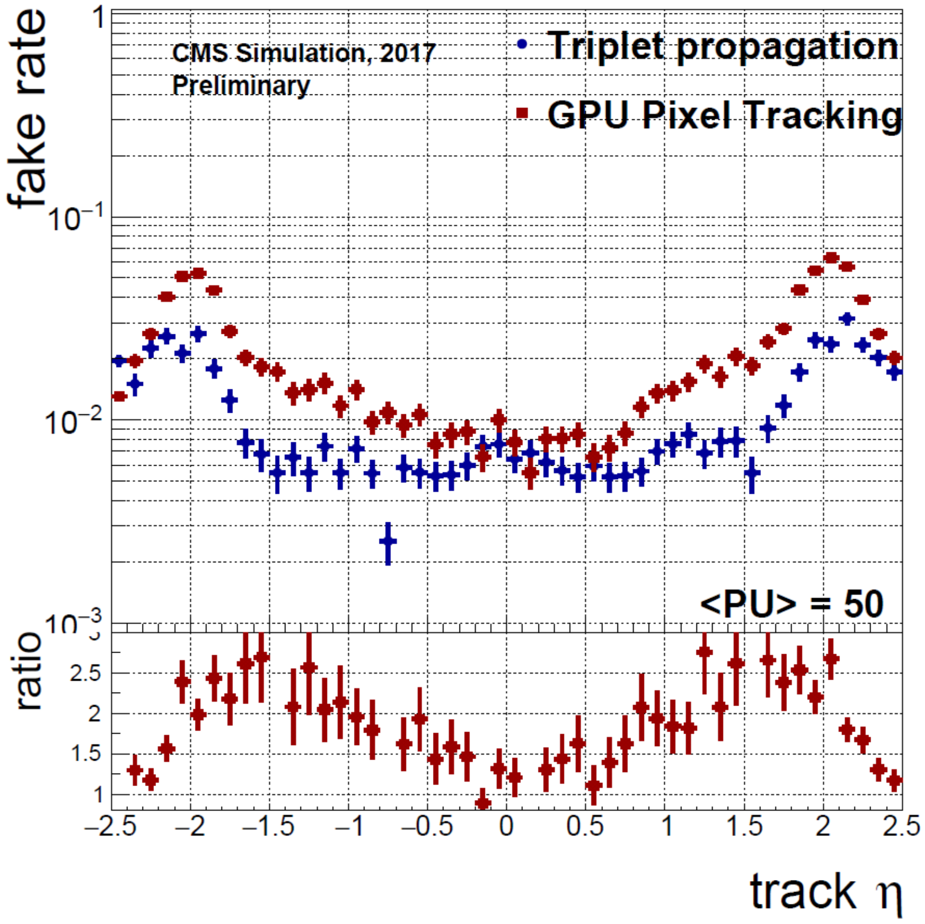
Calculations are simple, and localized in memory, straightforward to parallelize efficiently

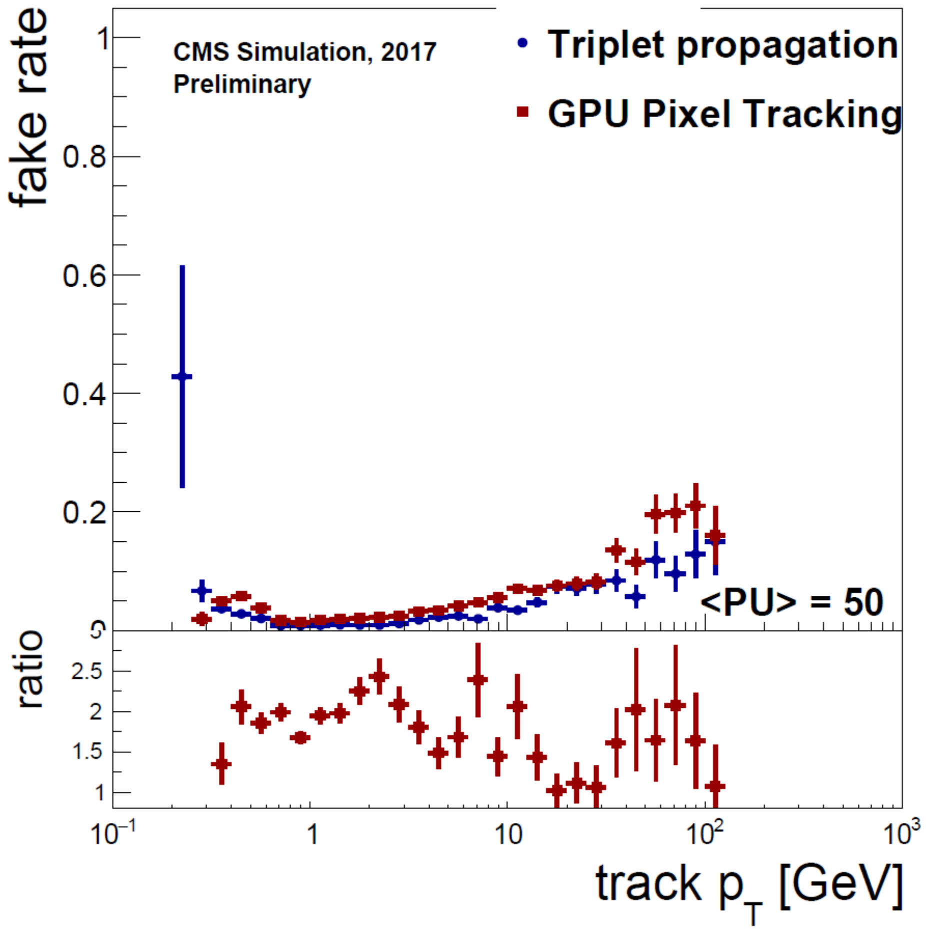
Physics performance ttbar 50 pileup





Physics performance ttbar 50 pileup



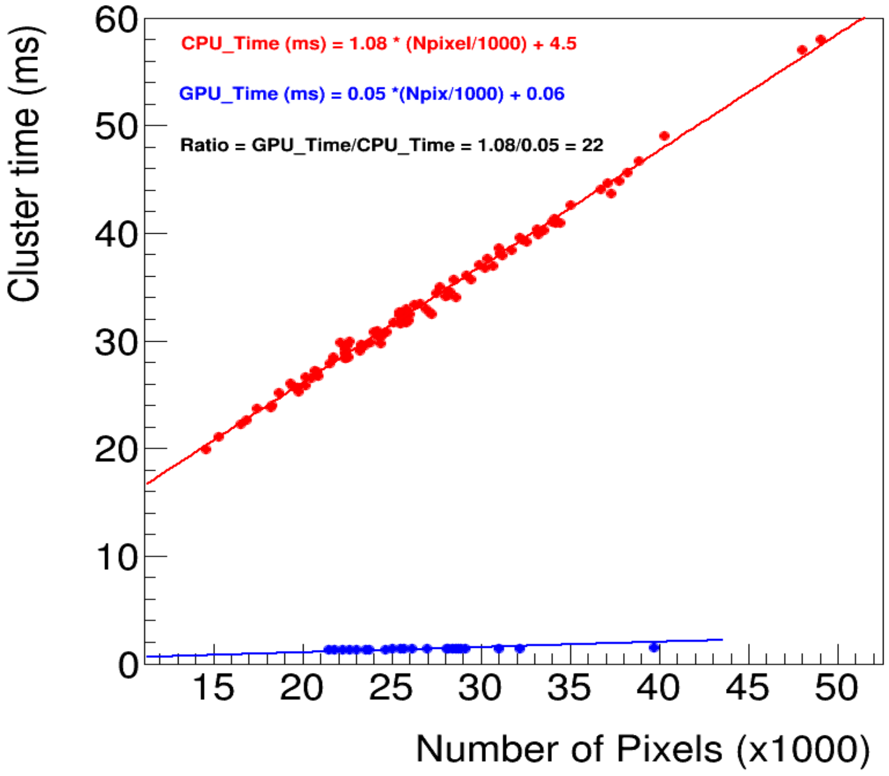


Timing performance ttbar 50 pileup

Events with PU50 are not getting even close to saturate the GPU

- Only 2-5% of the GPU busy
- ~100MB GPU DRAM used per event
- This allows us to offload many events on the same GPU by many threads

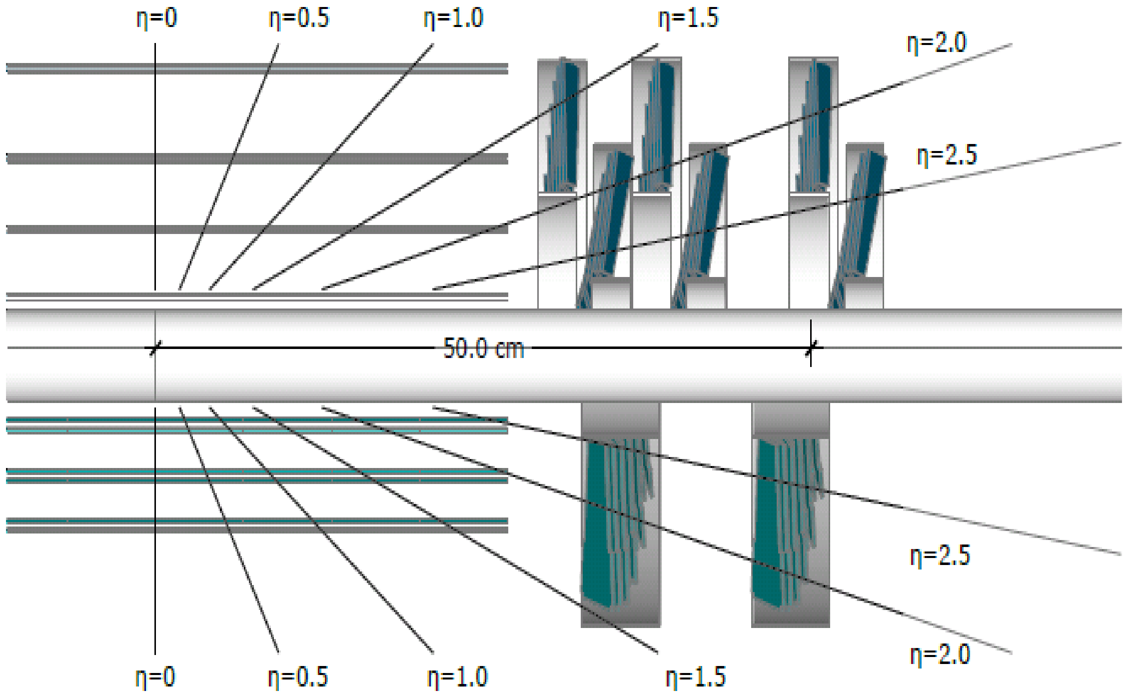
	time per event CPU (ms)	time per event GPU (ms)
Triplet propagation	66.3	N/A
CA	22	1.6 (15.2)



- Hardware used:
  - CPU Intel 4771K
  - GPU NVIDIA K40

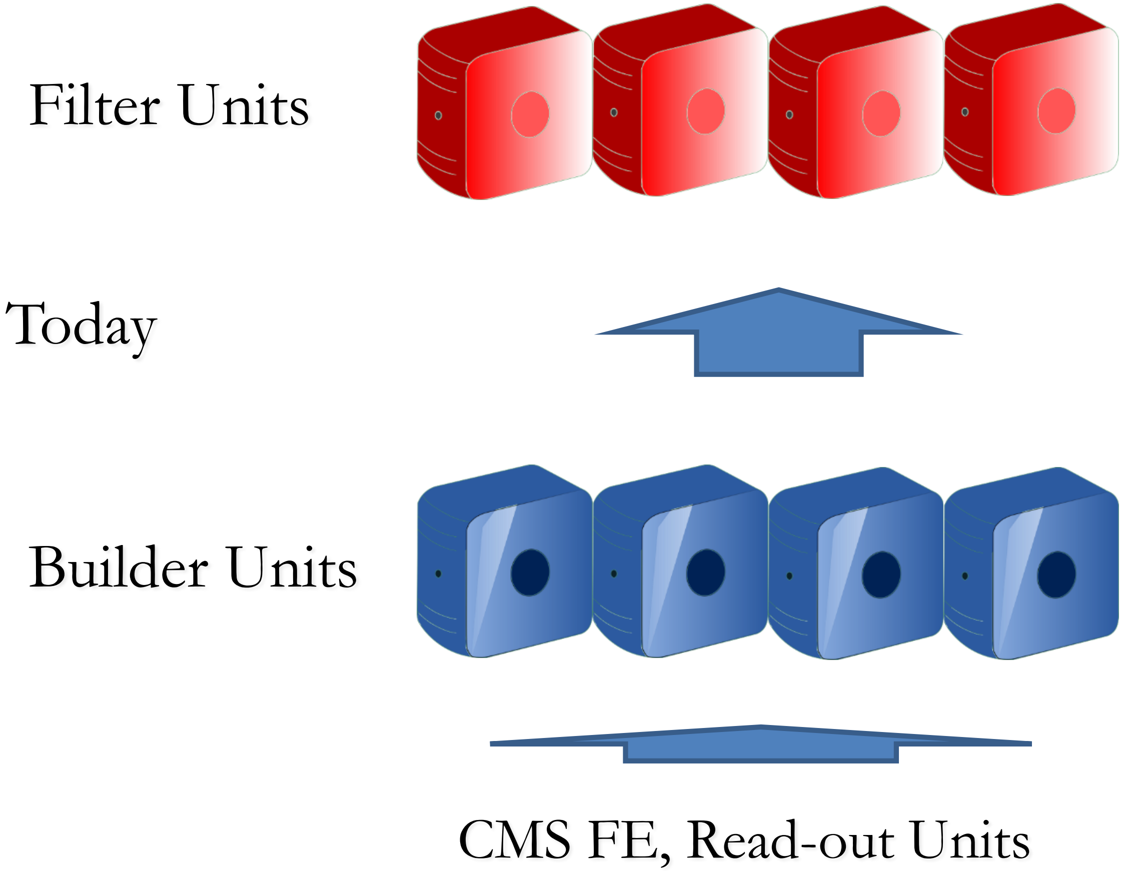
Motivation for the CA before run3

- A gradual restructure of the code at algorithmic level
  - Make use of parallel friendly algorithms
  - At the beginning of run 3 we won't need to compare apples to oranges
  - Sequential CA produces exactly the same results as the parallel CA
  - Expose parallelism
- Porting from CUDA to sequential C++
  - 2x speedup wrt 2016 pixel tracking
  - 5x less fake rate wrt 2016



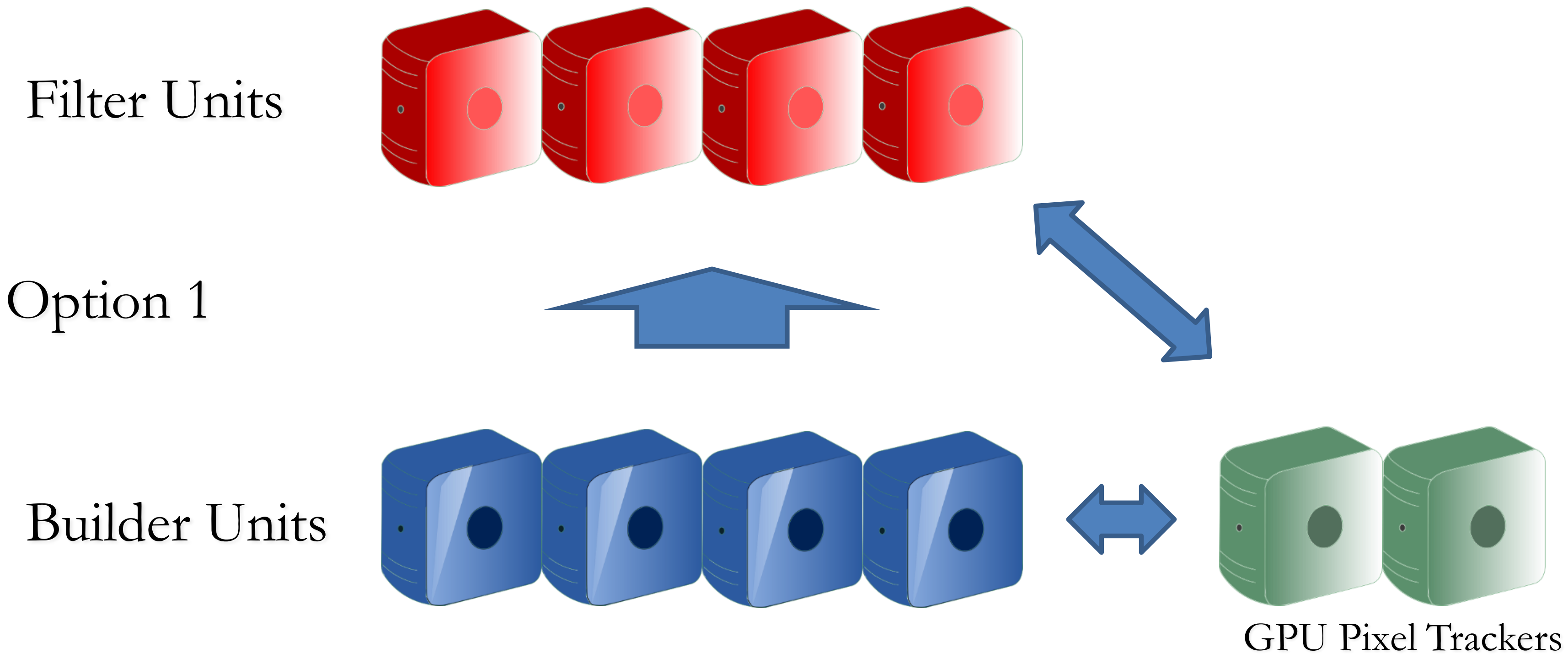
Integration in the HLT Farm

- Different possibleideas depending on :
  - the fraction of the events running tracking
  - other parts of the HLT reconstruction requiring a GPU



Integration in the HLT Farm

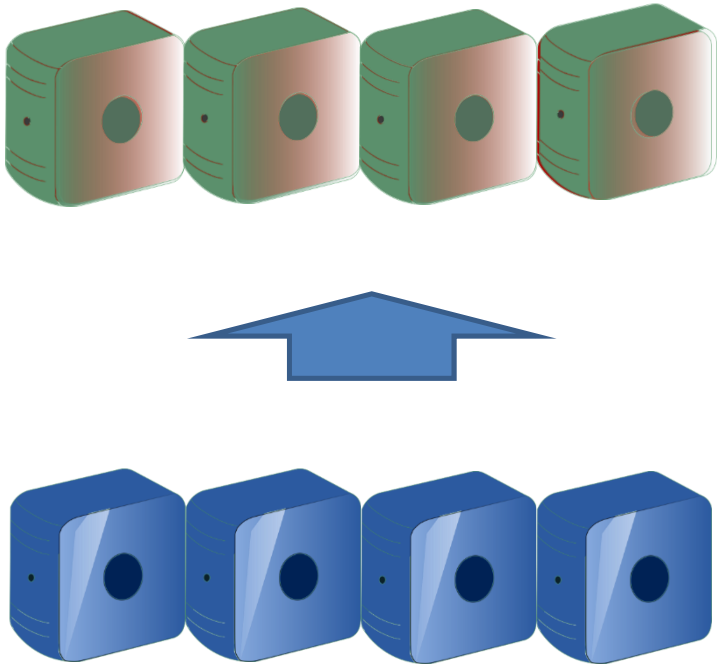
- A part of the farm is dedicated to a high density GPU cluster
- Tracks (or other physics objects like jets) are reconstructed on demand



Integration in the HLT Farm

- Every FU is equipped with GPUs
  - tracking for every event

GPU Filter Units

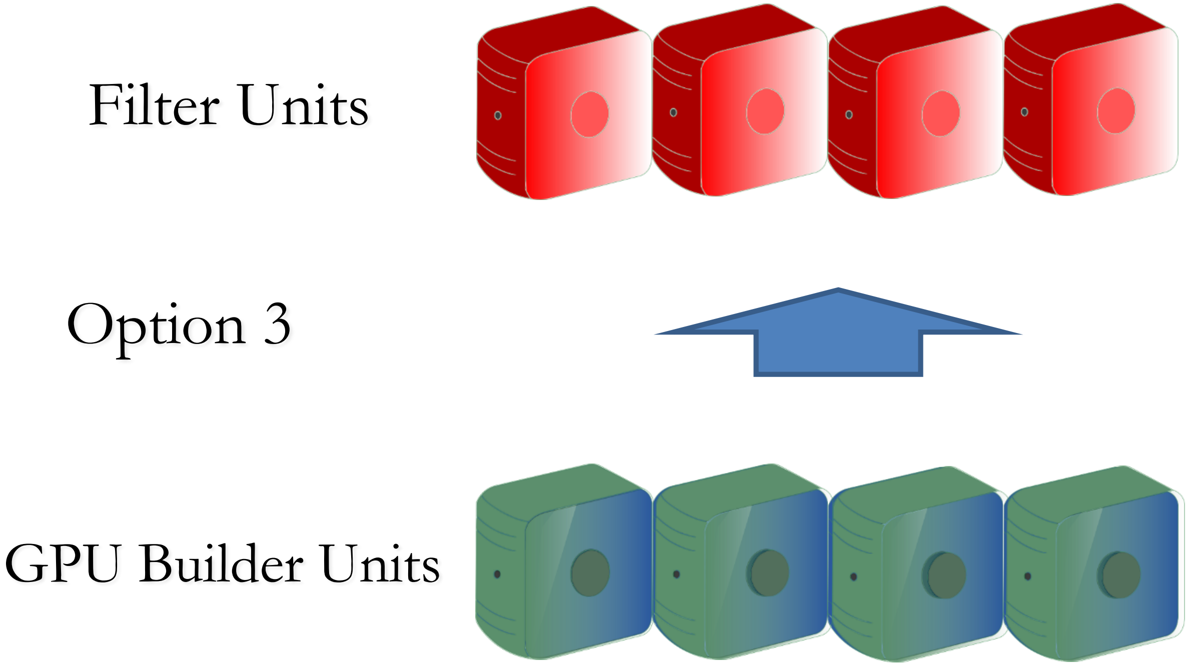


Option 2

Builder Units

Integration in the HLT Farm

- Builder units are equipped with GPUs:
  - events with already reconstructed tracks are fed to FUs with GPUDirect
  - Use the GPU DRAM in place of ramdisks for building events.



Conclusion

- Redesign of algorithms for parallel architectures will allow us to deal with the astonishing and always increasing performance of the LHC
  - Improvements in performance may come even when running sequentially
- The GPU and CPU algorithms run in CMSSW and produce the same bit-by-bit result
- Running Pixel Tracking at the CMS HLT will become cheap even with PU ~ 50 – 70
- What's next:
  - Merge all the standalone demonstrators in a single one from RAW data to Tracks
  - Measure performance for HL-LHC pileup conditions (i.e. PU ~ 140-200)

Questions?

felice@cern.ch