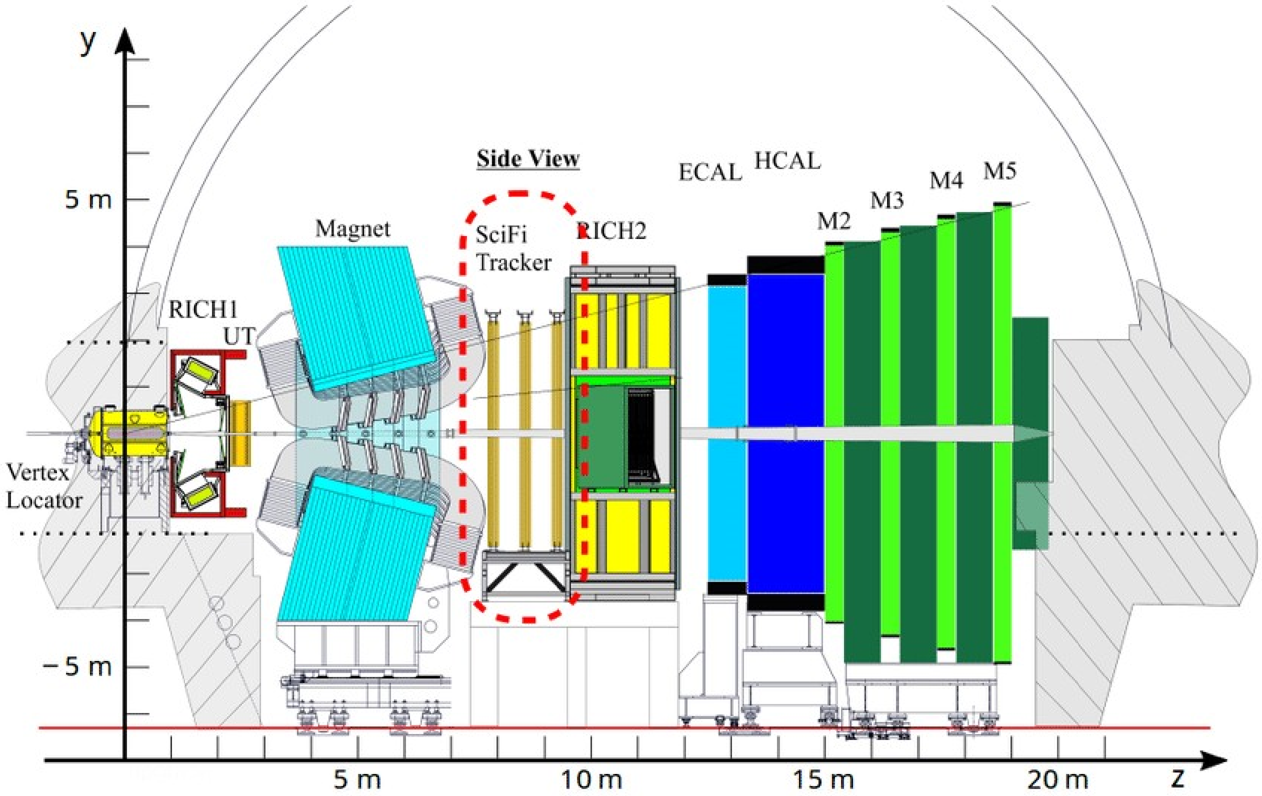
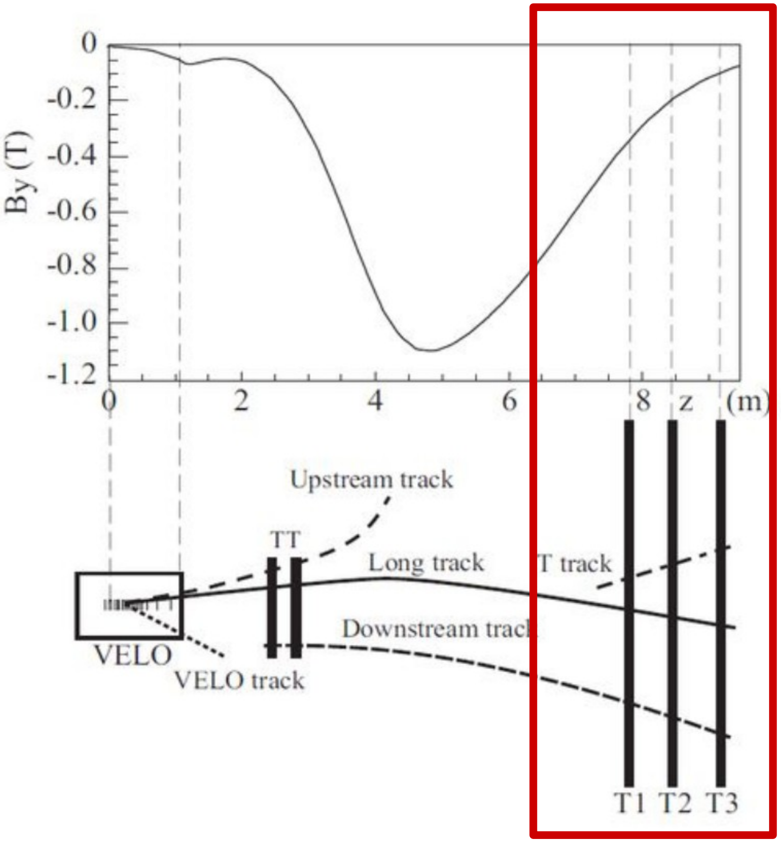
Standalone track reconstruction and matching algorithms for the GPU-based High Level Trigger at LHCb

Louis Henry, on behalf of the LHCb collaboration
CERN, 13/10/2023

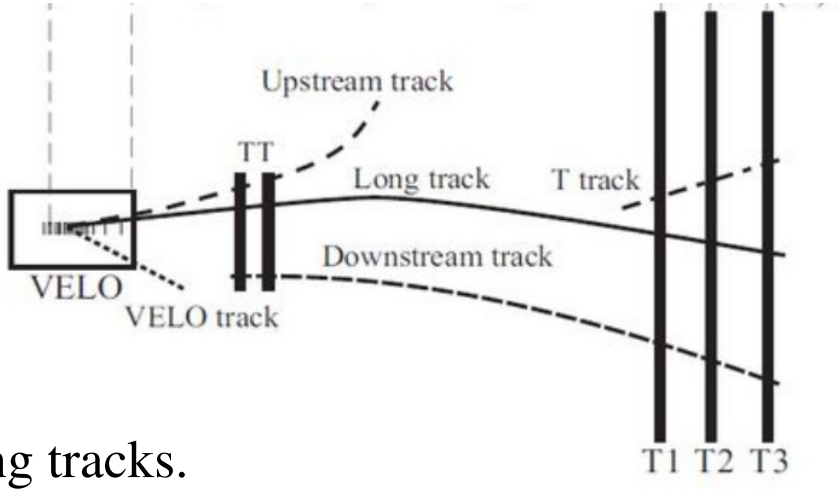
The LHCb detector

- LHCb is a detector along the LHC, specialised in the study of beauty and charm hadrons [JINST 3 (2008) S08005]
- Three tracking subdetectors used to reconstruct tracks: VELO, UT and SciFi



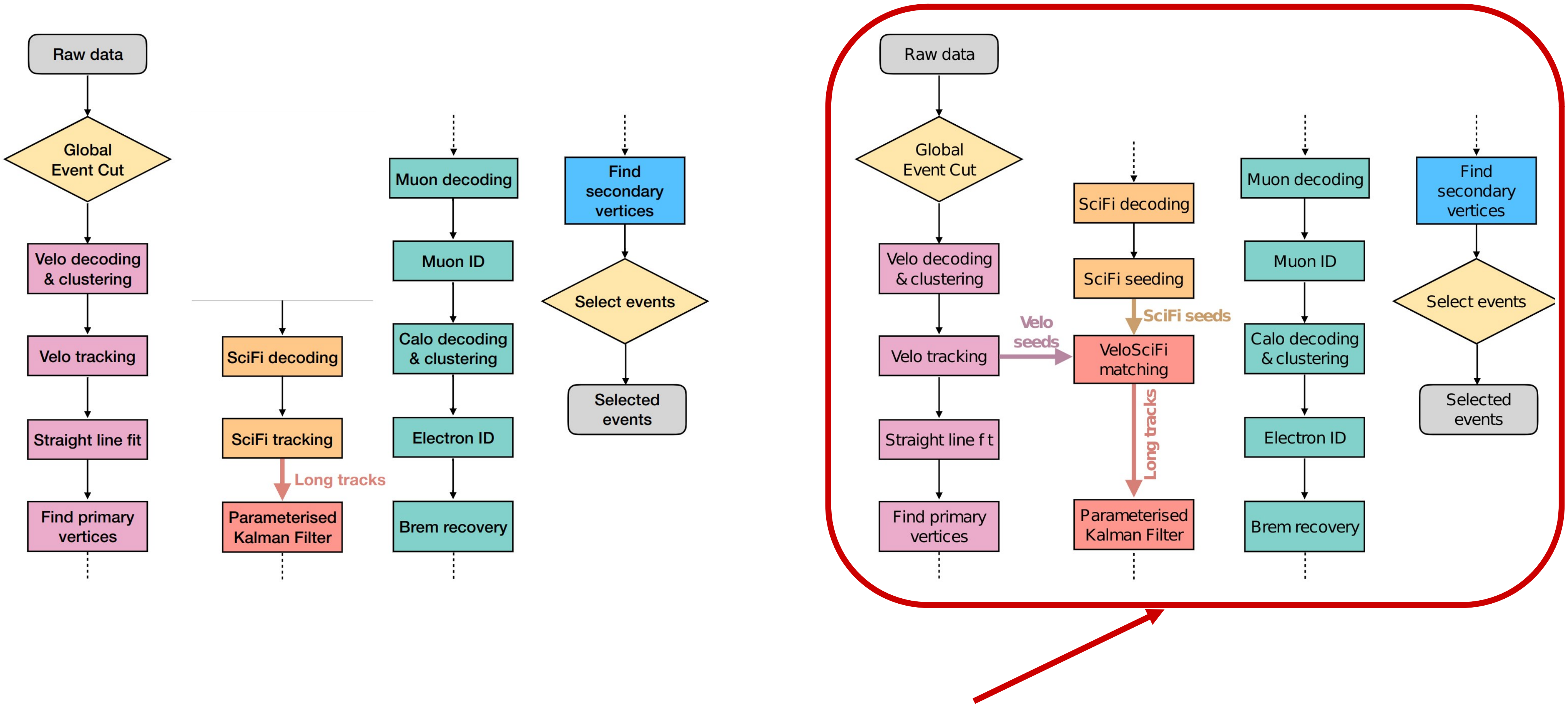


Track types, trigger levels, and how they relate



- LHCb trigger in two levels:
  - HLT1 (GPUs) reduces the rate from 30 MHz to 1 MHz using partial signatures of interesting physics.
  - HLT2 (CPUs) reduces the rate further from 1 MHz to few kHz, reconstructing the entire event.
- By design, HLT1 focuses on ‘easy’ signatures → Long tracks.
  - Consequently, penalty on displaced modes such as LLPs.
- Two ways of reconstructing Long tracks:
  - Reconstruct VELO segments, then propagate them to SciFi: “Forward” strategy
  - Reconstruct VELO segments, SciFi segments (“Seeds”) then Match them: “Matching” strategy.
- In the HLT1 menu, only Forward strategy implemented as baseline due to Seed reconstruction cost.

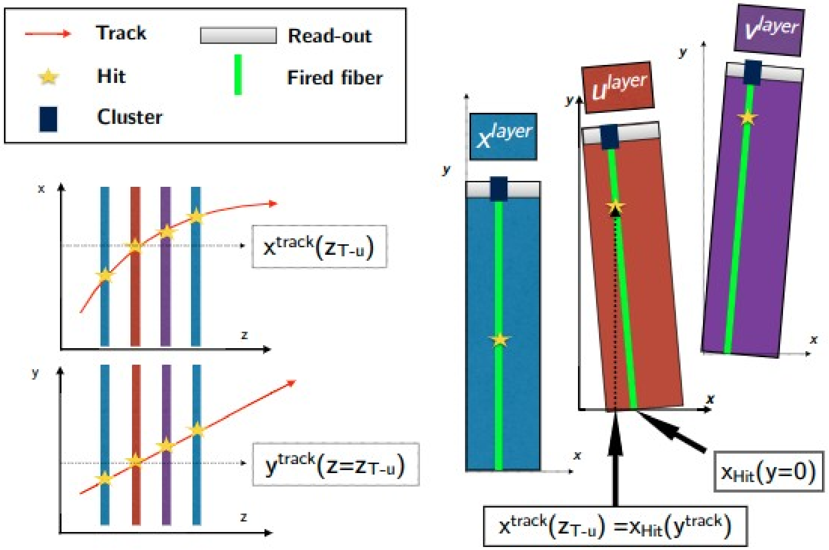
The two possible HLT1s



This talk

Hybrid seeding: overall strategy

- SciFi: **three** stations arranged in a x-u-v-x geometry, u and v being layers titled by a +/- 5° stereo angle.
  - Easier to get x coordinate than y coordinate.
  - But ~only residual B_y field → simpler y trajectory (line).



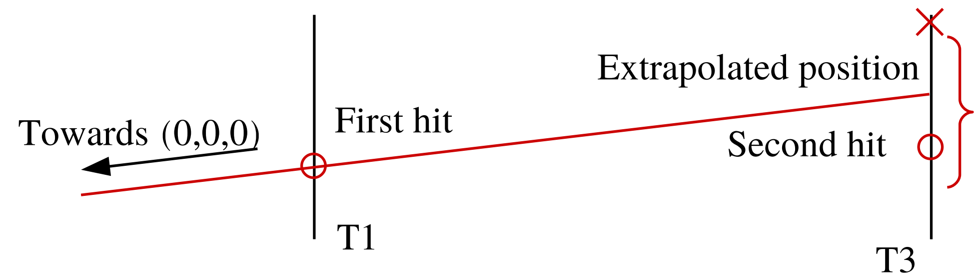
- Combinatorics too large to tackle all at once → **iterative strategy**: go for high-momentum first (~straight lines), cleanup the environment progressively.

- Each iteration starts with different pair of layers in T1 & T3.
  - Covers for hit inefficiency → modest theoretical cap on efficiencies.

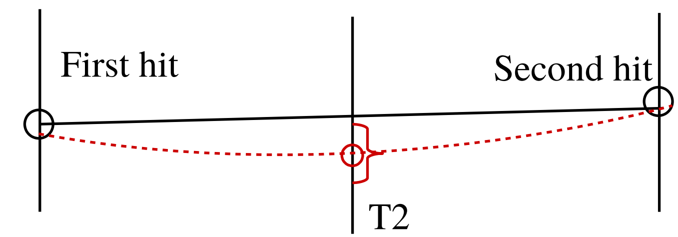
Hybrid seeding: the gist of the algorithm

seed_xz

- Starts with doublet search in T1 & T3, windows depending on minimum p, taking charge asymmetry in consideration.



- For each doublet, already a charge-momentum estimation → narrower windows to look for 3rd hit in T2 station, taking bending into account.



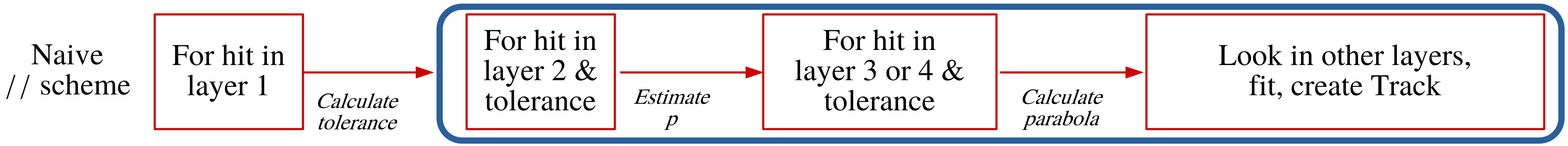
- Triplet → track model. We look for remaining hits (2 minimum) → **XZ segment**.

seed_confirmtracks

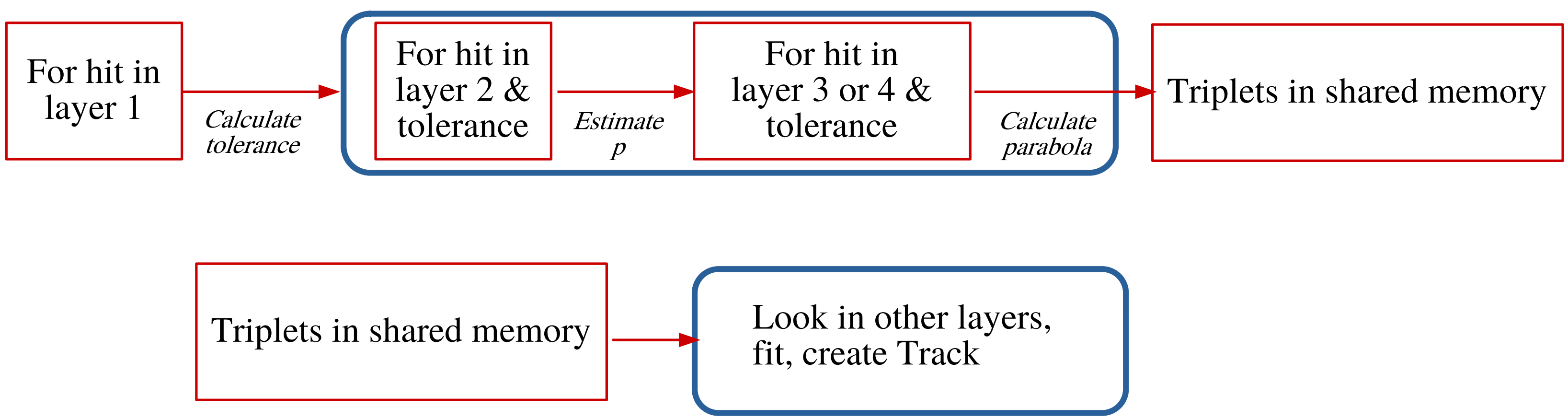
- XZ segments are dominated by fake tracks → need for U/V combination.
  - XZ segment provides us with x(z) equation, so hits in U/V layers can be translated into y coordinate, and thus to t_y.
- Real tracks have **~ constant t_y** if no scattering and come from close to the origin.

Seeding in XZ: seed_xz

- XZ search: naive sequence would use one level of parallelisation (over first hits)



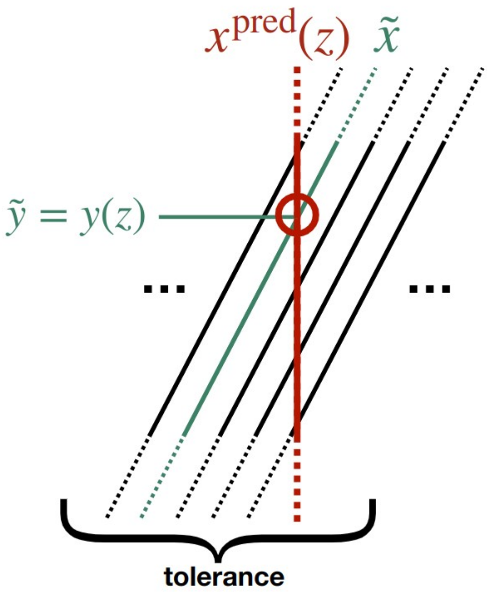
- Studies on MC show that 80% of triplets get promoted to full track → costly for many threads without a triplet to wait for the other ones to finish.
  - New scheme is in two parallel sequences:



- First sequence is fast, and hit-or-miss (many doublets do not have a triplet); second sequence is slow but high occupancy of threads.

Adding U/V hits: seed_confirmtracks

- XZ segment provides us with x(z) equation, so hits in U/V layers can be translated into y coordinate, and thus to t_y.



- Real tracks have **~ constant t_y** if no scattering and come from close to the origin.

- Solution adopted on CPUs: Hough clustering.
  - Costs too much memory in GPUs, although solutions are being developed.

- Instead, adapt seed_xz approach and use all hits in a first layer as seeds to start the t_y evaluation, and look for hits along the defined line.
  - Update t_y value with subsequent hits
  - Ignoring the tilt in dz/dy of layers, transformation function $x_{observed} \rightarrow y$ is a simple multiplication.

- Parallelised over first hits, two passes with different first layers to cover inefficiencies.

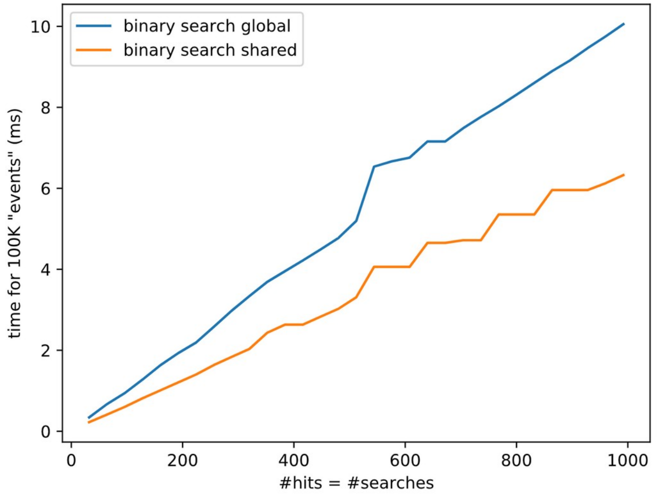
Cloning, flagging, storing hits

- Seeding relies on several “passes” on data, flagging hits to focus on more difficult tracks.
- We found that this reconstitution of data was more costly than anything: better do two passes with different seed layers.

GPU	P Min	Layer 1	Layer 2	Layer 3
Case 0	3000	T1X1	T3X1	T2x1
Case 1	3000	T1X2	T3X2	T2x2

- This hinges on a $O(n_{\text{Tracks}})$ clone removal → voting algorithm.
  - Each track “votes” on its hits using its $(n_{\text{hits}}, \chi^2)$ score, tracks that get outvoted are removed.

- Seeding reads hits very often → need to store them in coalesced container.
  - Store the x coordinates of the hits only (4 bytes);
  - Only consider 6 layers at a time, 300 hits per layer → 7.2 KB in shared emory.
  - Fallback to global in case of overflow, very rare.



A few tricks down memory lane

- Storing variables in registers allows for fastest treatment.
  - Sometimes, precalculating quantities does not help.

- Rewrote the algorithm to make sure that, whenever possible:
  - Loops are unrolled (example 1);
  - Array indices are known at compile time;
  - Conditional index increments are delayed to the end of the kernel (example 2).

Example 1

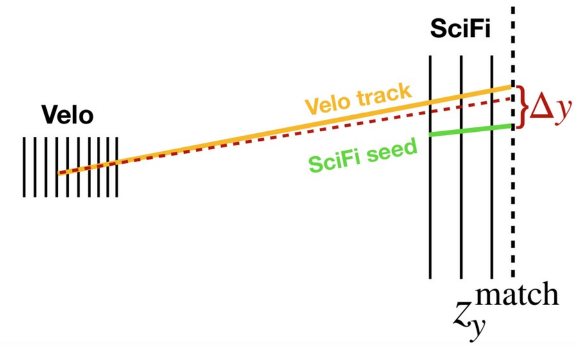
```
float x, tx; // registers
float dz[6]; // registers ?
float x_pred[6]; // registers ?
for (int i=3 ; i<6 ; i++) {
    x_pred[i-3] = x + tx * dz[i];
}

// Compiler will unroll everything:
register x_pred_0 = x + tx * dz_3;
register x_pred_1 = x + tx * dz_4;
register x_pred_2 = x + tx * dz_5;
```

Example 2

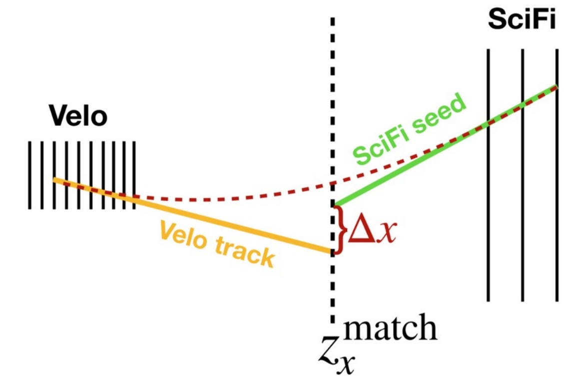
```
int hits[6]; // local variable
int nHits = 0;
for (int i=0 ; i<6 ; i++) {
  hits[i] = idx; // always store (idx may be invalid)
}
// ...
for (int i=0 ; i<6 ; i++) {
  if (hits[i] != -1) tracks[threadIdx.x].idx[nHits++] = hits[i];
}
```

Matching seeds to VELO tracks

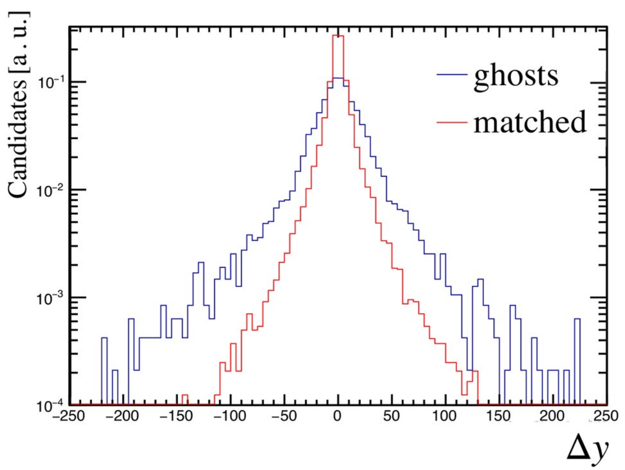
- Hybrid seeding gives us O(80) tracks per event, to match with O(100) Velo tracks without UT hits.



- Parallelise over SciFi seeds, extrapolate all tracks in a straight line to a "kink" plane in xz.

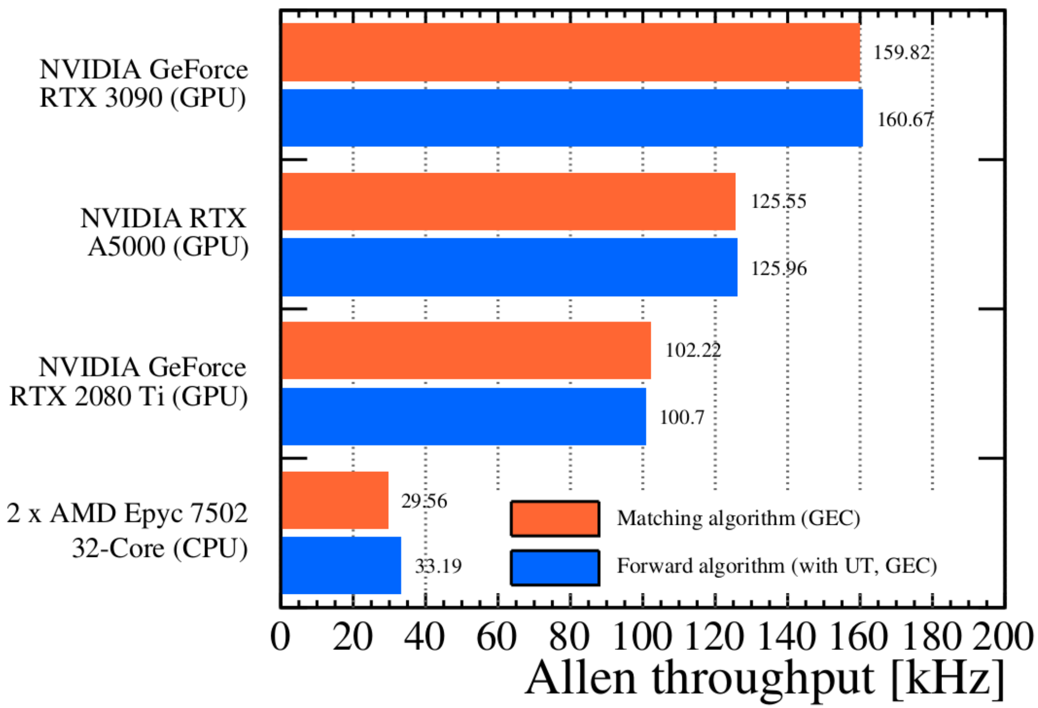


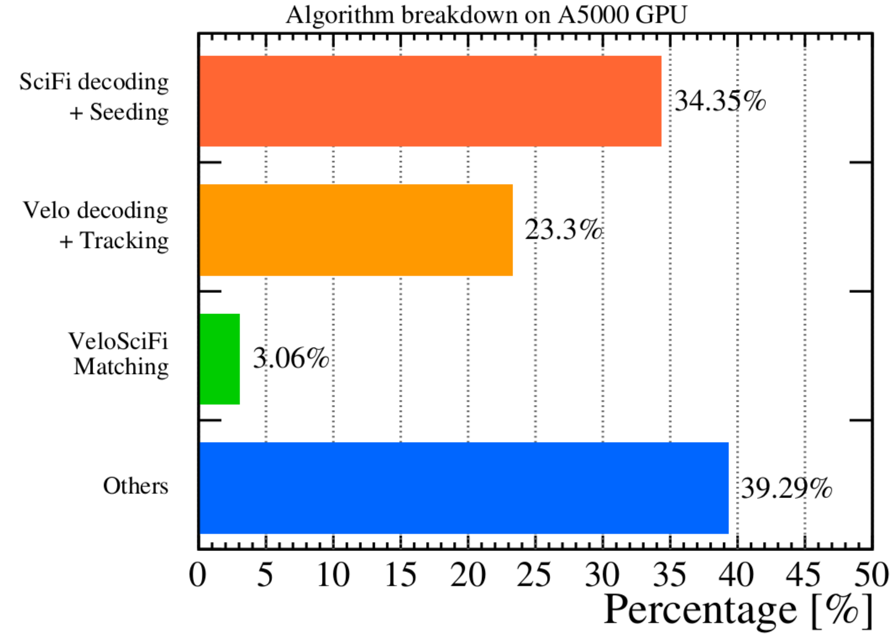
- Magnetics field, tolerances, parameterised using simulation.



- Clone killing is simpler than in the seeding: criterion of shared VELO segment → only the pair with the smallest χ^2 is kept.

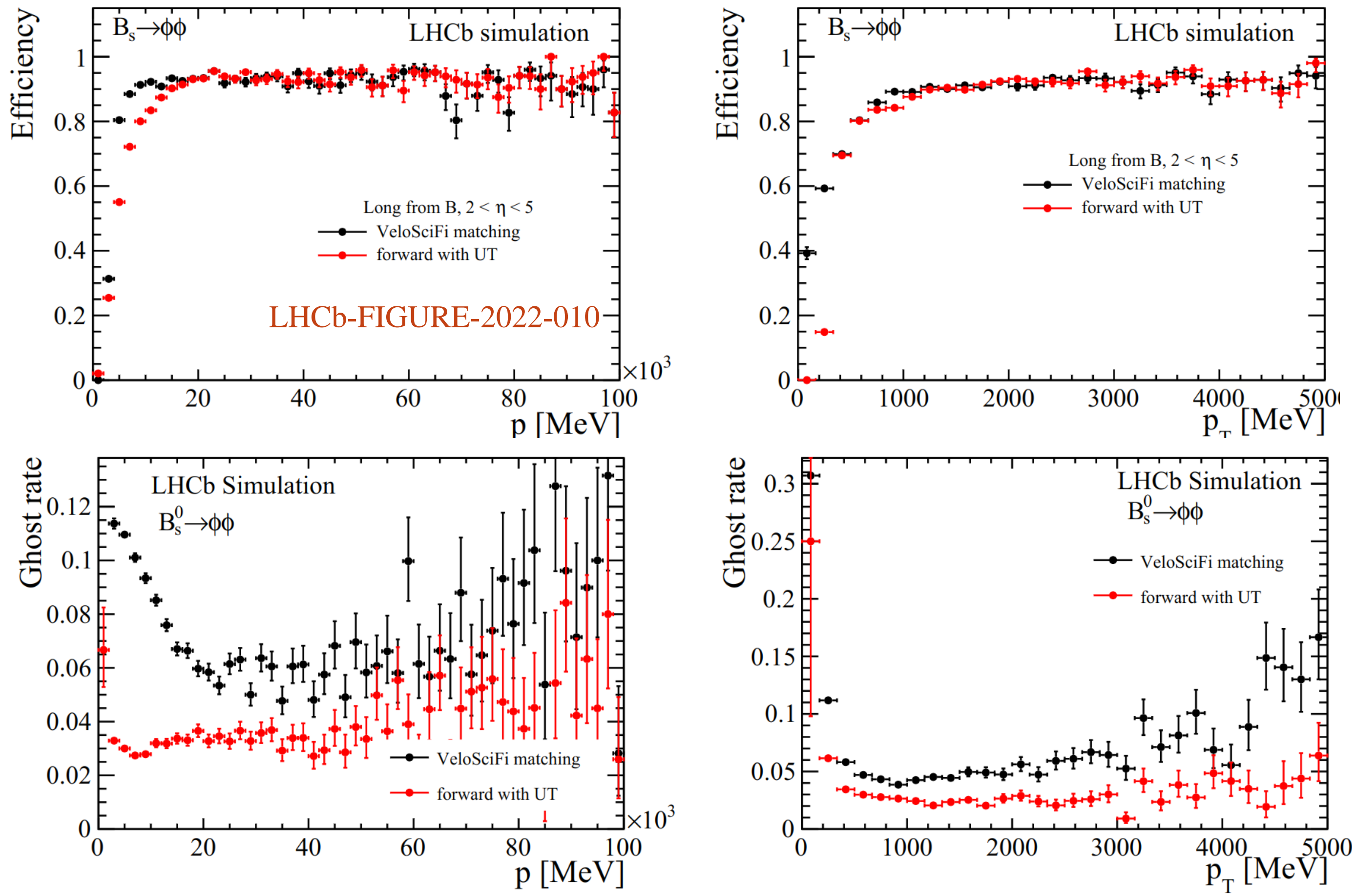
Impact on throughput





- Seeding + matching does not significantly change the throughput of HLT1.
- Most of the cost is on the seeding itself, as could be expected.

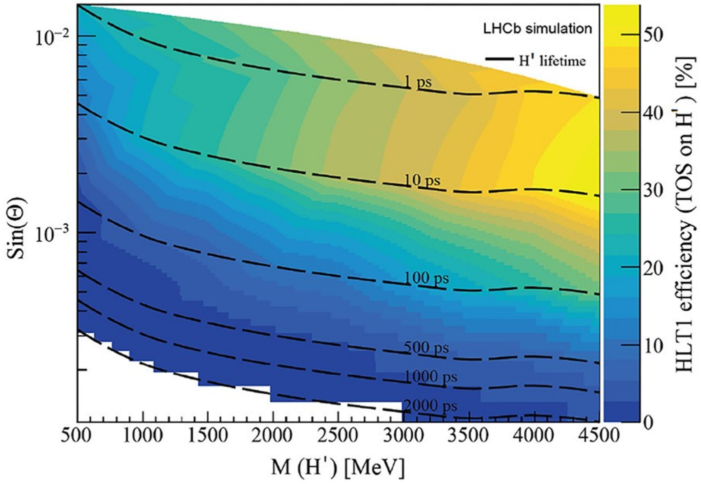
Physics results



- No UT hits in the hybrid seeding – matching → larger ghost rate.
- Efficiency on Long tracks a bit smaller than Forward at high momentum, but much larger at smaller momenta.
- Much more flexible strategy: soft tracks, low p_T, opens possibility for downstream tracking.

Conclusion

- Successfully implemented a standalone reconstruction algorithm in high-throughput constraints.
  - Allows to reconstruct Long tracks in alternative way;
  - Competitive efficiency/fast rate compared to Forward baseline strategy;
  - Much better efficiency at low momenta;
  - Opens door for downstream reconstruction in HLT1.
- This algorithm is already being used in commissioning.
- Considering the possibility to run the seeding+matching and the forward concurrently for maximal efficiency.
- HLT1 downstream reconstruction and T-tracks only trigger lines are being prepared.
  - Huge potential impact on long-lived particle searches.



10.3389/fdata.2022.1008737