Overview of online and offline reconstruction in ALICE for LHC Run 3

David Rohr

drohr@cern.ch

Connecting the Dots, 2020

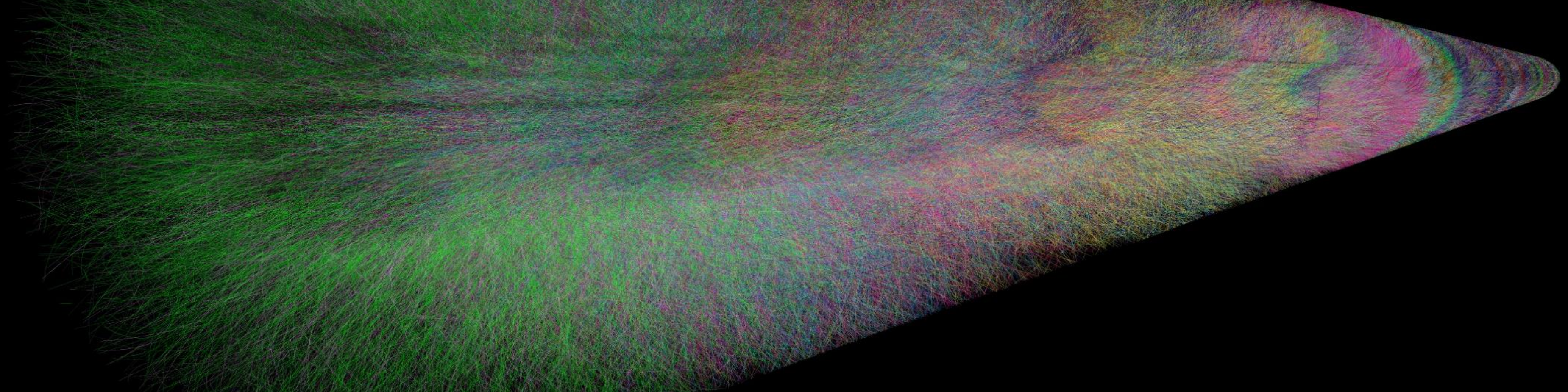
22.4.2020

ALICE in Run 3: 50 kHz Pb-Pb

Record large minimum bias sample.

- All collisions stored for main detectors → no trigger.
- Continuous readout → data in drift detectors overlap.
- 50x more events stored, 50x more data.
- Cannot store all raw data → online compression.

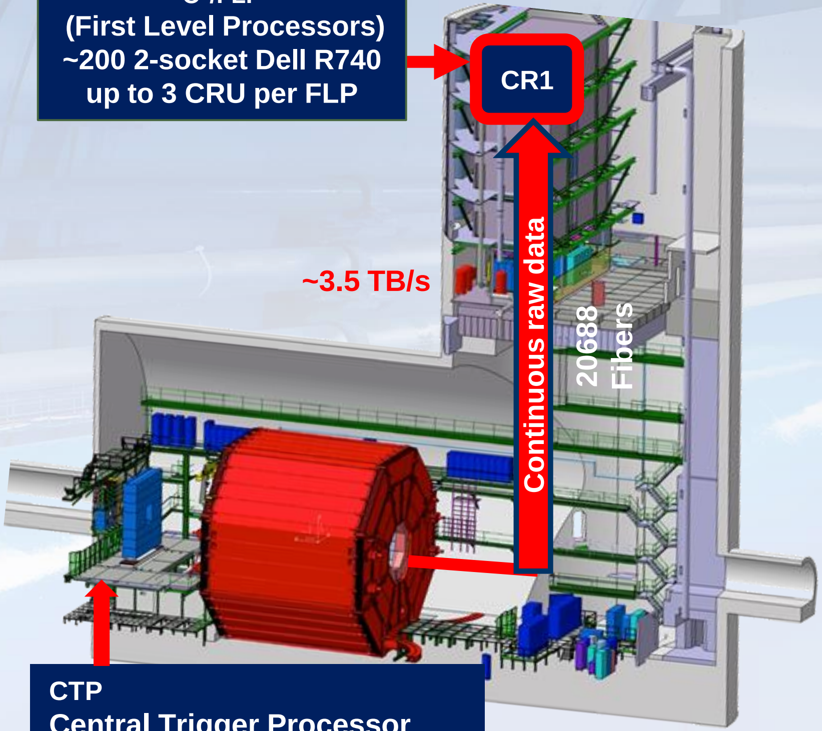
→ Use GPUs to speed up online processing.



- Overlapping events in TPC with realistic bunch structure @ 50 kHz Pb-Pb.
- Timeframe of 2 ms shown (will be 10 – 20 ms during production).
- Tracks of different collisions shown in different colors.

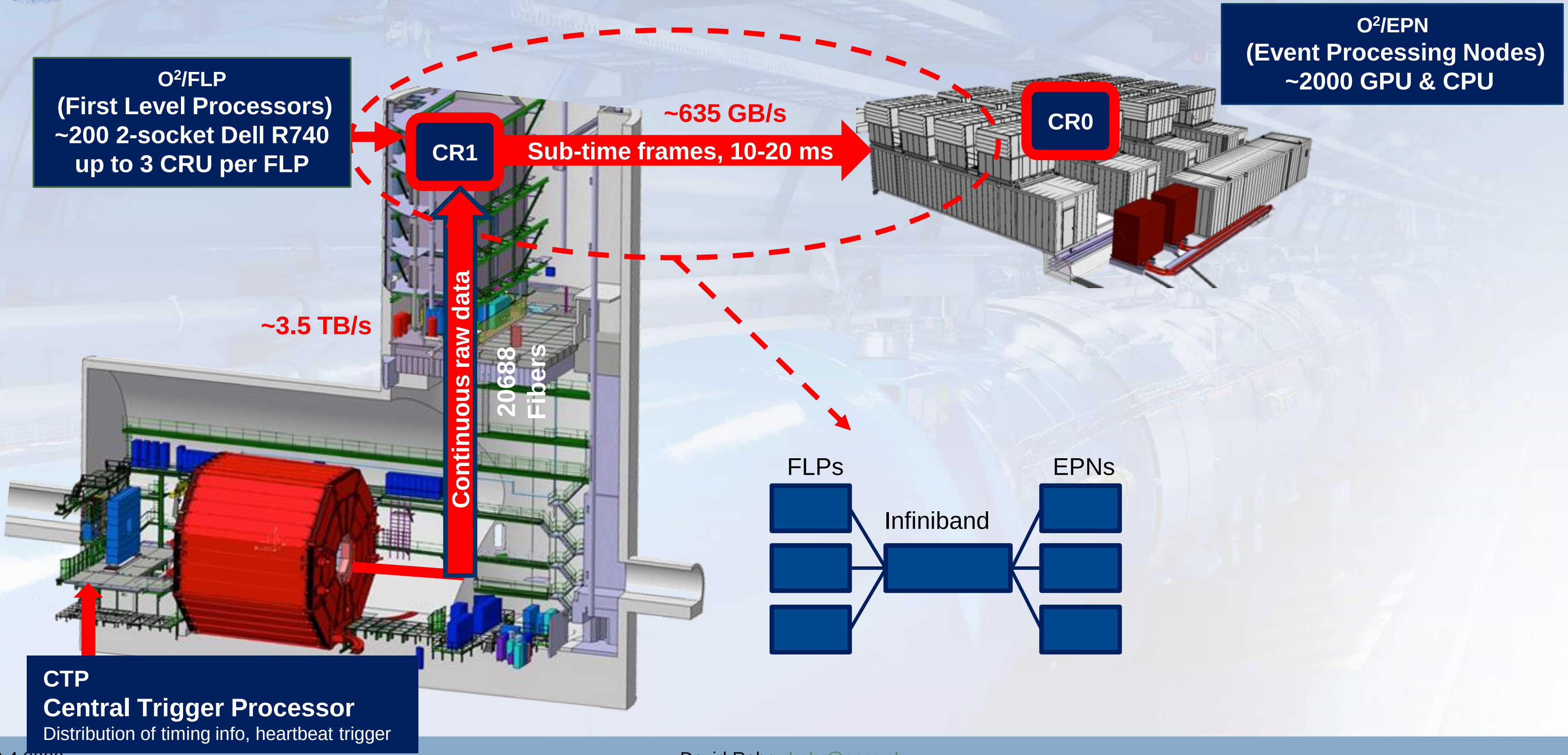
ALICE Raw Data Flow in Run 3

O^2/FLP
(First Level Processors)
~200 2-socket Dell R740
up to 3 CRU per FLP

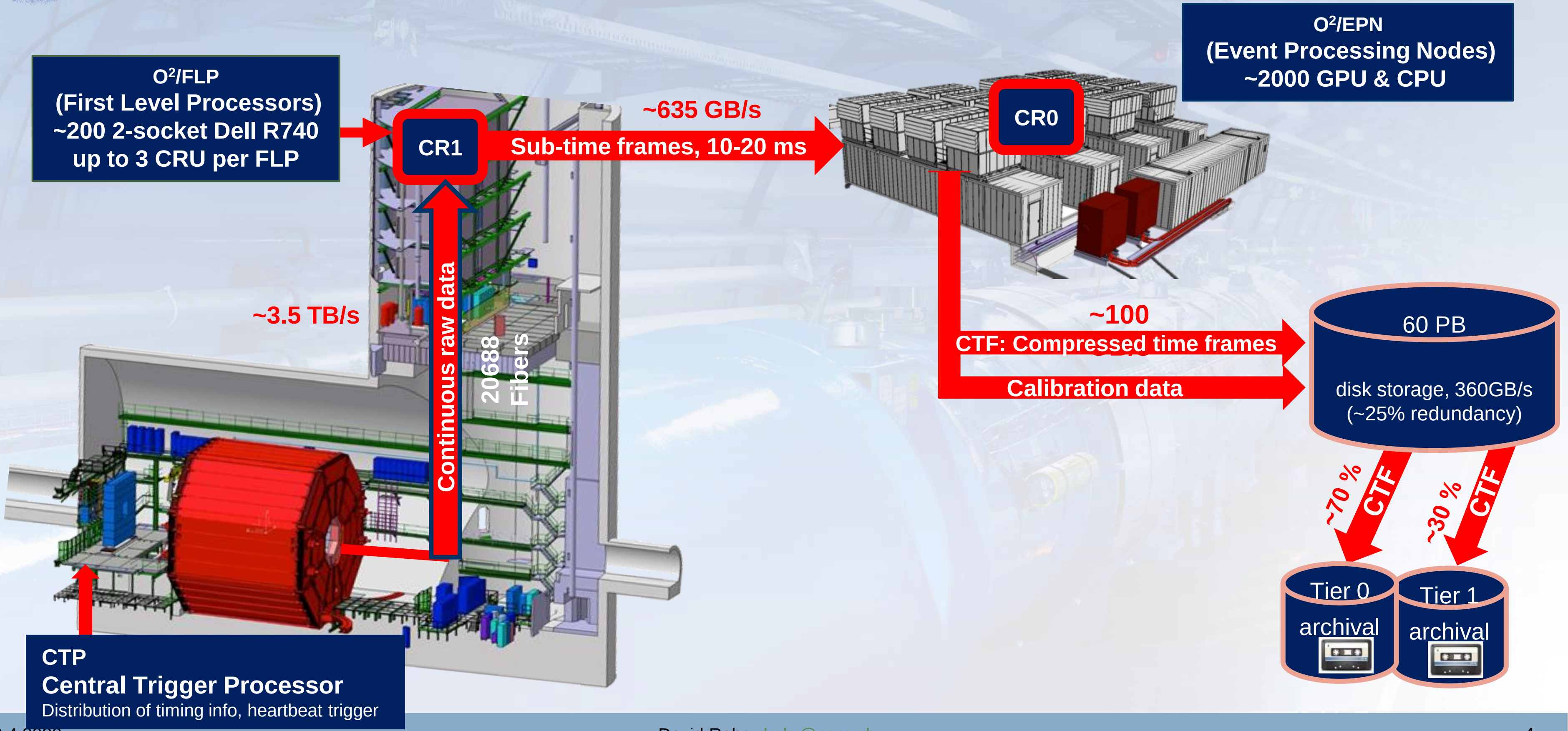


CTP
Central Trigger Processor
Distribution of timing info, heartbeat trigger

ALICE Raw Data Flow in Run 3

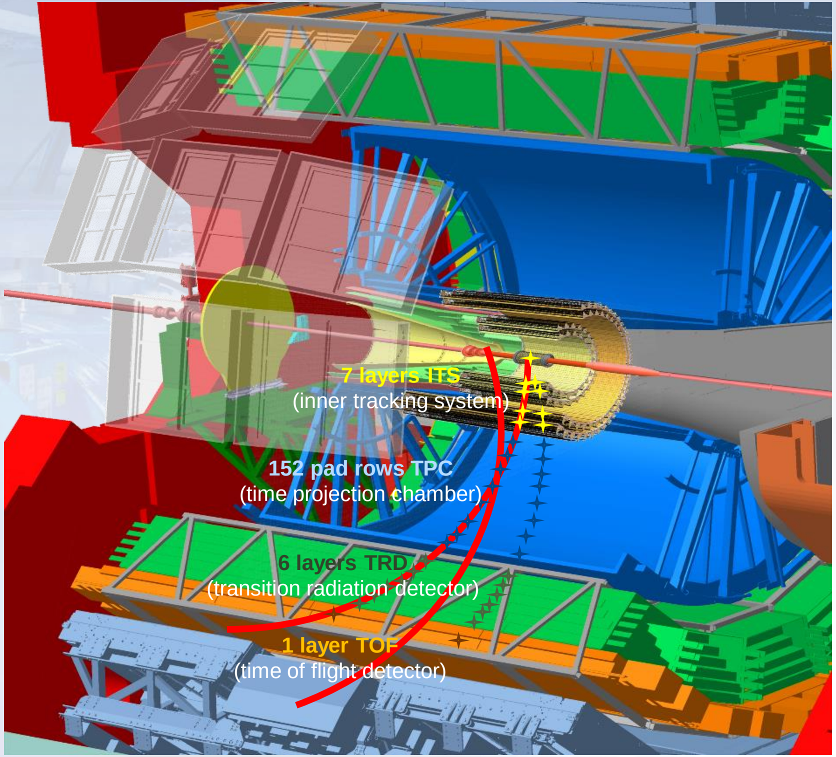


ALICE Raw Data Flow in Run 3



Tracking in ALICE in Run 3

- **ALICE uses mainly 3 detectors for tracking: ITS, TPC, TRD + (TOF)**
  - **7 layers ITS** (Inner Tracking System – silicon tracker)
  - **152 pad rows** TPC (Time Projection Chamber)
  - **6 layers TRD** (Transition Radiation Detector)
  - **1 layer TOF** (Time Of Flight Detector)



Processing in ALICE in Run 3

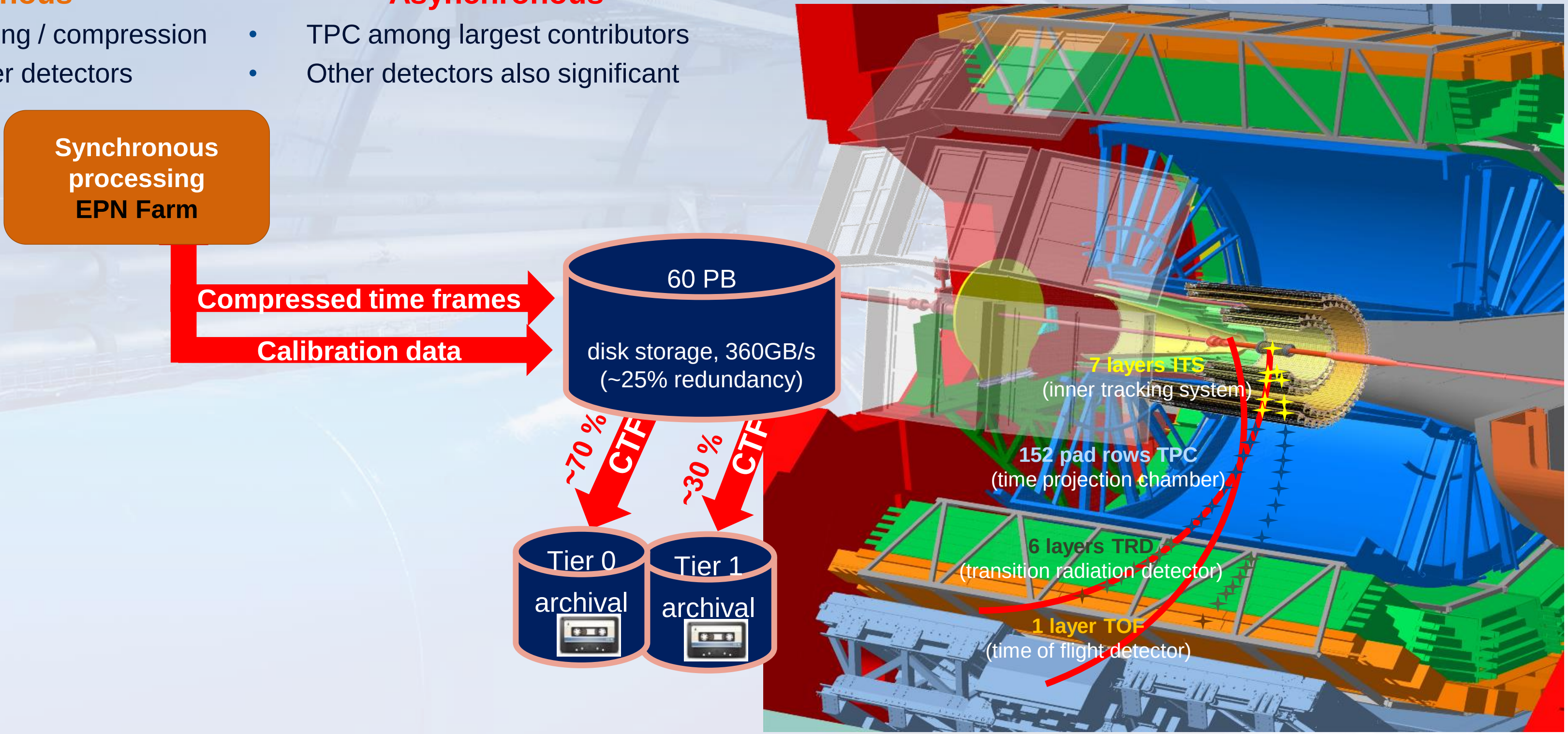
- **Bulk of computing workload:**

Synchronous

- >90% TPC tracking / compression
- Low load for other detectors

Asynchronous

- TPC among largest contributors
- Other detectors also significant

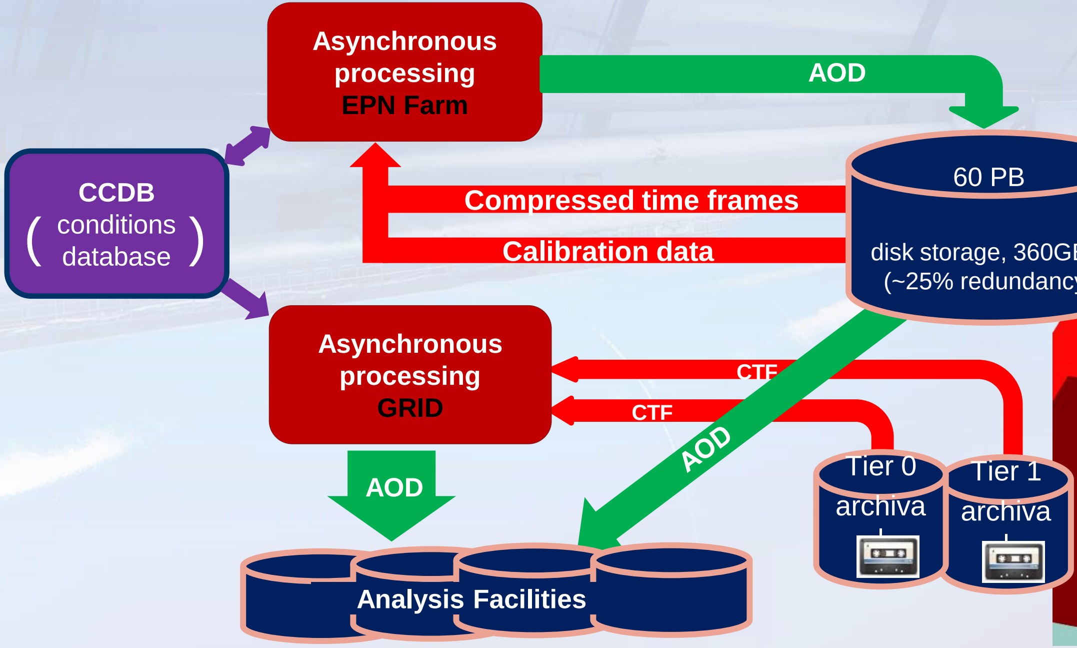


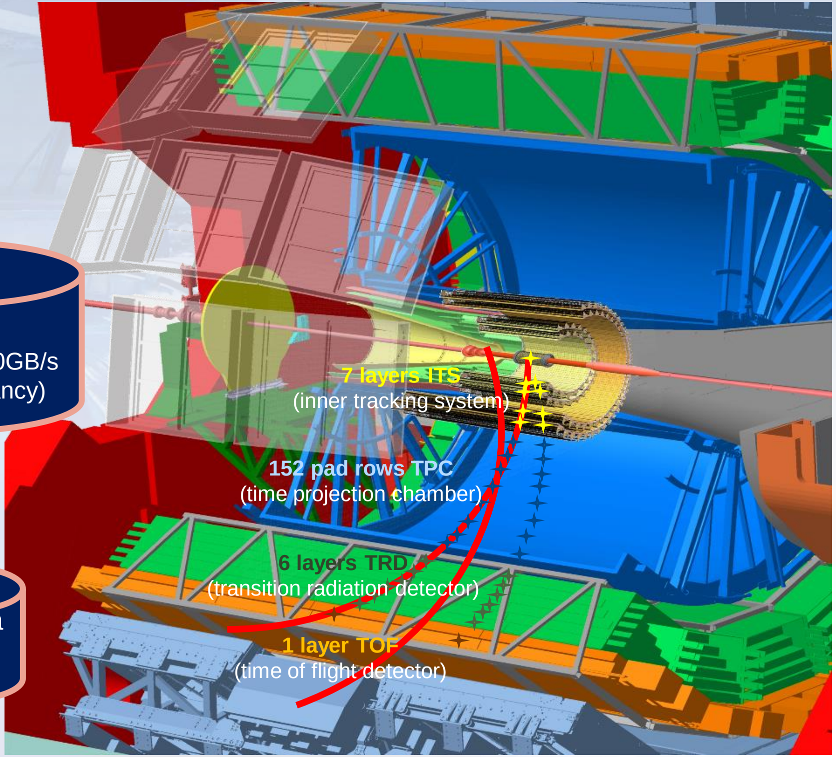
Processing in ALICE in Run 3

- **Bulk of computing workload:**

Synchronous
- >90% TPC tracking / compression
- Low load for other detectors

Asynchronous
- TPC among largest contributors
- Other detectors also significant





Tracking in ALICE in Run 3

- **Bulk of computing workload:**

Synchronous
- >90% TPC tracking / compression
- Low load for other detectors

Asynchronous
- TPC among largest contributors
- Other detectors also significant

- **ALICE GPU processing strategy**

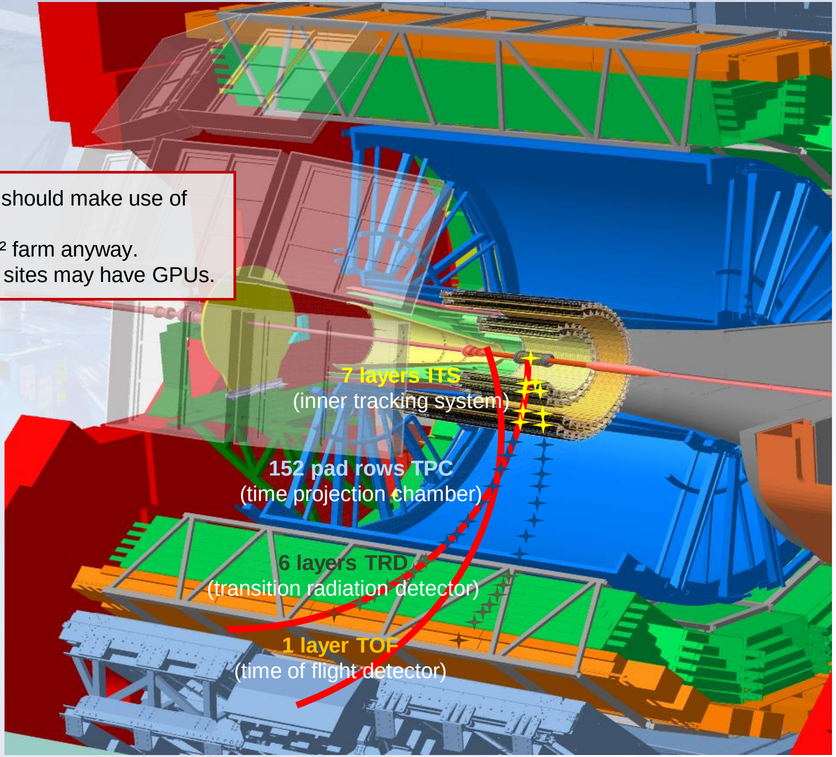
Baseline solution (almost available today): TPC + part of ITS tracking on GPU

– **Mandatory** solution to keep up with the data rate online.
– **Defines** number of servers / **GPUs**.

Optimistic solution (what could we do in the ideal case): Run most of tracking + X on GPU.

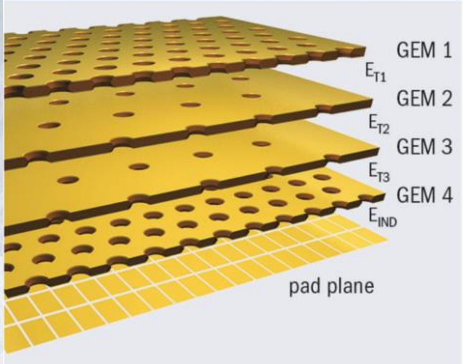
– Extension of baseline solution to make best use of GPUs.
  – Ideally, **full barrel tracking** without ever leaving the GPU.
  – In the end, we will probably be somewhere in between.

Asynchronous phase should make use of the available GPUs.
- Available in the O^2 farm anyway.
- Future HPC / grid sites may have GPUs.



ALICE TPC upgrades and implications

- **Need continuous TPC readout to store full minimum bias data sample.**
  - The TPC of **Run 1 and 2** uses **MWPC** (Multi Wire Proportional Chambers) readout and a **gating grid** to **suppress the ion back flow**.
  - Gating grid **limits readout to ~3 kHz**, prevents continuous readout.
  - → **Replace MWPCs with GEMs** (Gas Electron Multiplier), **Intrinsic ion back flow blocking** (**99%**), no gating grid.



- **Still, significantly more space charge in the TPC compared to Run 1 and 2.**
  - GEM amplification creates ~20 ions (gain 2000) per electron.
    - Dominant over ions from primary ionization.
    - Scales linearly with interaction rate up to **50 kHz**.
  - Ions drift from end plate to central electrode in ~200 ms.
  - Ion charge in the TPC produces an electric field, distorting the electrons during drift.
  - Up to **20 cm distortions** in radial direction.
  - Must be corrected to **O(0.1) mm** to maintain the **intrinsic TPC resolution**.

- ***z*-coordinate of TPC hits depends on time of the vertex:** $z \approx \pm(z_0 - v_{drift} * (t - t_{vertex}))$
  - Need to assign a hit to a vertex to obtain its *z* coordinate, but the vertex is usually only known after tracking.

- **Processing based on time frames (10 – 20 ms of continuous data) instead of events.**

Important challenges for Run 3 tracking / processing

- **Performance**
  - Have to process **50 times more events** than before.
- **Data compression**
  - Full minimum bias data taking, no event rejection, need to store **50 times more data** than before.
- **Calibration**
  - Need **calibration** procedure for **space charge distortions**.
- **Tracking**
  - Need to be able to **track TPC data with continuous readout**, and with **space charge distortions**.

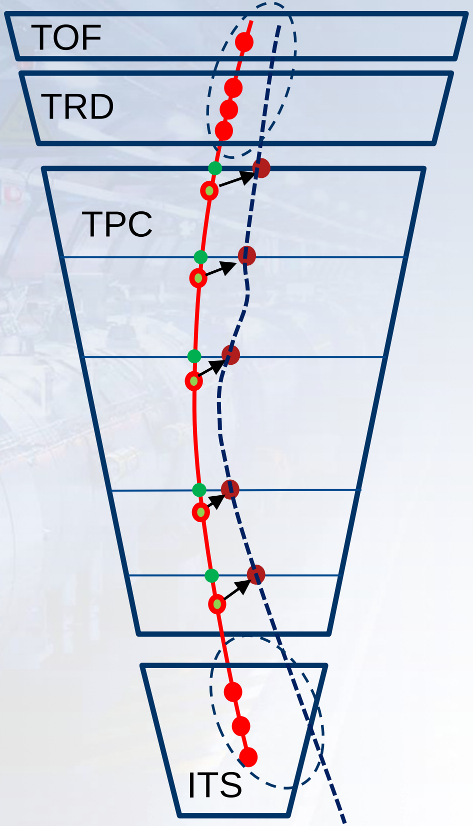
TPC Calibration

- **Most complicated TPC calibration is for space charge distortions (SCD).**
  - Other calibrations (like **d*E*/dx** run in parallel).
  - We foresee 2 SCD calibrations in Run 3:

1. **Track based**:
   - TPC Tracks reconstructed with relaxed cuts and matched to inner / outer detectors.
   - Track refit with only ITS / TRD / TOF information.
   - Residuals of TPC hits wrt. refitted tracks are collected.
   - TPC volume is voxelized, and a correction per voxel is calculated.
   - Corrections are smoothed to compensate for bad TRD chambers, holes in TRD / TOF acceptance.

- The track based TPC calibration corrects also several other effects:
  - Misalignment, drift velocity, E x B, …
- Needs a certain number of tracks in each voxel to extract the correction.
  - In Run 2, the calibration interval was 40 minutes. This will be reduced to O(1) minute in Run 3.
- Distortions fluctuate over time:
  - Scales with instantaneous luminosity, i.e. TPC occupancy, e.g. by beam burn-off.
    - Accounted for by scaling the correction map with the luminosity.
    - *To be precise, the difference to a static correction map at luminosity ~0 is scaled.*
  - Short-term fluctuations by LHC bunch structure, collision centrality, etc.
    - Not accounted for during Run 2, but effect below intrinsic TPC resolution.

→ Need a new method for short-term fluctuations in Run 3, correction stable over ~5 ms.



TPC Calibration

- **Most complicated TPC calibration is for space charge distortions (SCD).**
  - Other calibrations (like ***dE/dx*** run in parallel).
  - We foresee 2 SCD calibrations in Run 3:

1. **Track based**.
2. **Integration of digital currents**:
   - Aggregating the currents arriving at the TPC end plates.
   - Allows for the computation of:
     - The number of amplification ions produced.
     - Space charge produced by the ions.
     - Electric field in the TPC by the space charge.
     - Distortions of the electrons during the drift.
   - Since this is computationally very heavy, we aim to employ a neural network.

   - This requires the full ion history and thus continuous readout, impossible with triggered readout in Run 2.
   - Cannot correct other effects but SCD by design.
   - Probably less total precision than track-based distortion.
   - Fast enough to correct short-term fluctuations.
   - Could correct for the delta to average distortion by using the delta to average digital currents.

3. Alternative: Self-contained TPC calibration using cluster to track residuals.

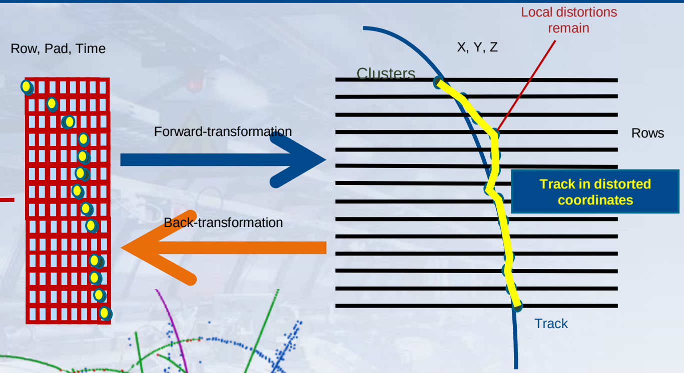
TPC Calibration

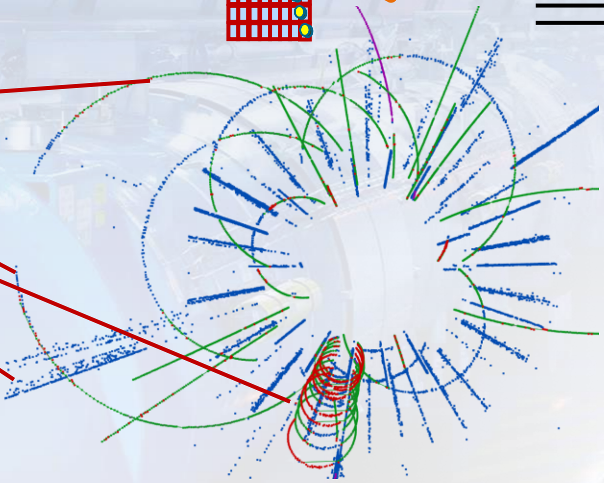
- **Most complicated TPC calibration is for space charge distortions (SCD).**
  - Other calibrations (like ***dE/dx*** run in parallel).
  - We foresee 2 SCD calibrations in Run 3:
    1. **Track based**.
    2. **Integration of digital currents**.

- **ALICE calibration procedure:**
  - In Run 1 and 2, ALICE was processing the data 4 times: Online (trigger / QA) / 2 calibration passes / physics pass.
  - This will be reduced to 2 passes: synchronous and asynchronous.
    - An intermediate postprocessing step will process the calibration input extracted by the synchronous pass and create the final calibration.
  - For the TPC this means:
    - Collection of the integrated digital currents:
      - Happens during synchronous processing on the readout card level (must happen in the FPGA, since servers / network might drop data rendering the next 180 ms of raw data unreconstructible).
    - Extraction of residuals for track-based calibration:
      - Happens during synchronous processing on the EPN. Only O(1%) of the tracks are needed. Peripheral collisions are selected.
    - Correction maps are created between synchronous and asynchronous (or at the beginning of asynchronous to avoid storing them).
  - The correction is applied as follows:
    - Average distortions are corrected by the track-based correction.
    - The track-based corrections are scaled with the luminosity.
    - Short term fluctuations are corrected by the integrated digital current method.

TPC Data Compression

- **The TPC data compression is composed of several steps:**
  - Raw data are clusterized.
  - Custom data format as integer / floating point with number of bits matching the TPC resolution.
  - Entropy is reduced as much as possible:
    - Clusters attached to tracks are stored as residual to the track.
    - Unattached clusters are sorted and stored as difference wrt. the previous cluster.
    - Correlated values treated together.
  - Clusters of tracks not used for physics are removed:
    - Strategy A: Identification and removal of unneeded clusters:
      - Secondary leg of track with p_T < 200 MeV/c.
      - Track segment with high inclination angle.
      - Low-p_T track below 50 MeV/c.
      - Noisy pads, charge clouds from low-p_T protons.
    - Strategy B:
      - Remove everything except for identified good tracks.
  - Remaining clusters are entropy-compressed with ANS encoding.





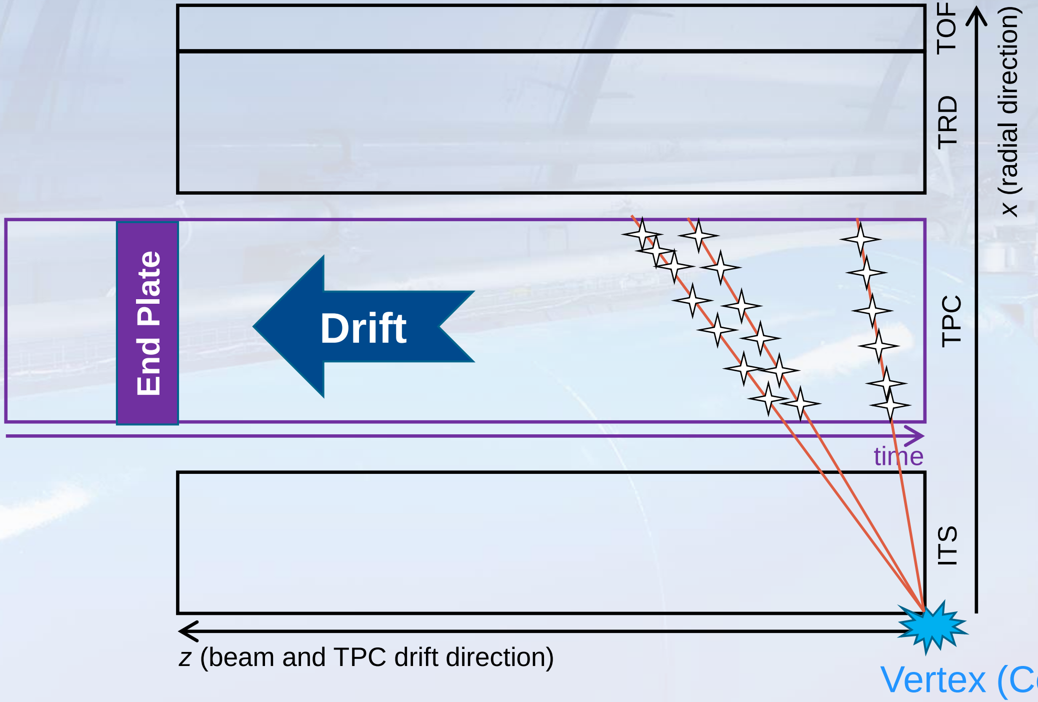
- **TPC is the largest data-contributor**
  - Must be reduced from 3.4 TB/s raw data to ~70 GB/s to storage.
  - At the storage level, other detectors contribute as well.
    - → Entropy-compressed, but not as sophisticated as TPC.

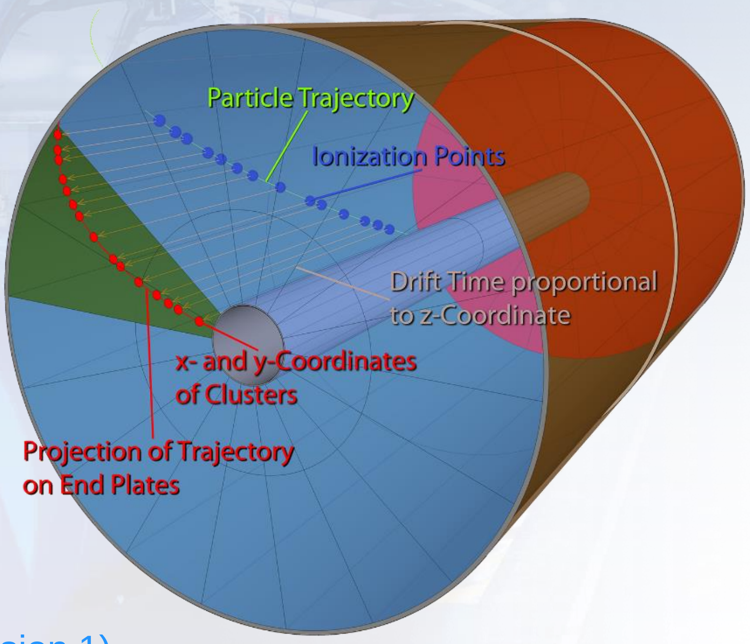
TPC Tracking

- **There are 2 (related) main challenges caused by continuous readout / space charge distortions**
  - How to assign a z-position to a cluster?
  - How to apply SCD corrections (inhomogeneous magnetic field, cluster error parameterization) if z is now known.
- **Tracking strategy:**
  - The naïve brute force approach:
    - We know all possible vertex times from the fast interaction triggers.
    - We can correct all clusters multiple times, for each possible vertex time.
    - Run the tracking multiple times, and select only the tracks belonging to the current vertex.
    - → Working but infeasible, increases processing time by factors.
  - → Need a better approach.

TPC Tracking

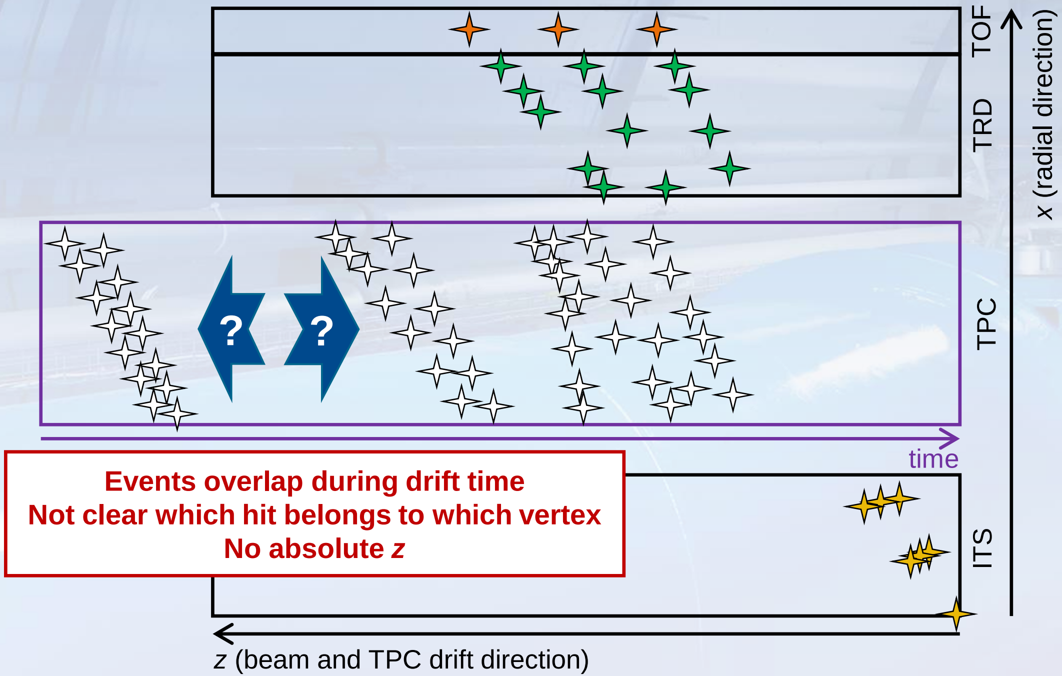
- **There are 2 (related) main challenges caused by continuous readout / space charge distortions**
  - How to assign a z-position to a cluster?
  - How to apply SCD corrections (inhomogeneous magnetic field, cluster error parameterization) if z is now known.

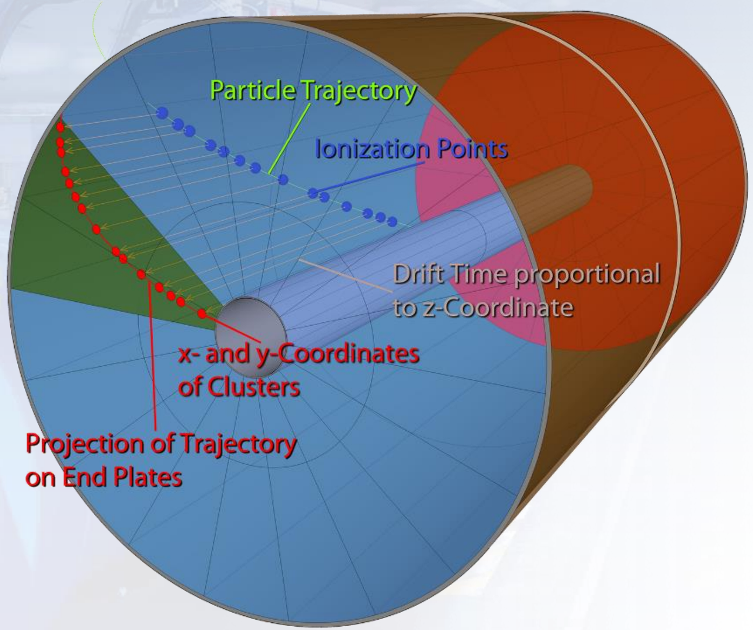




TPC Tracking

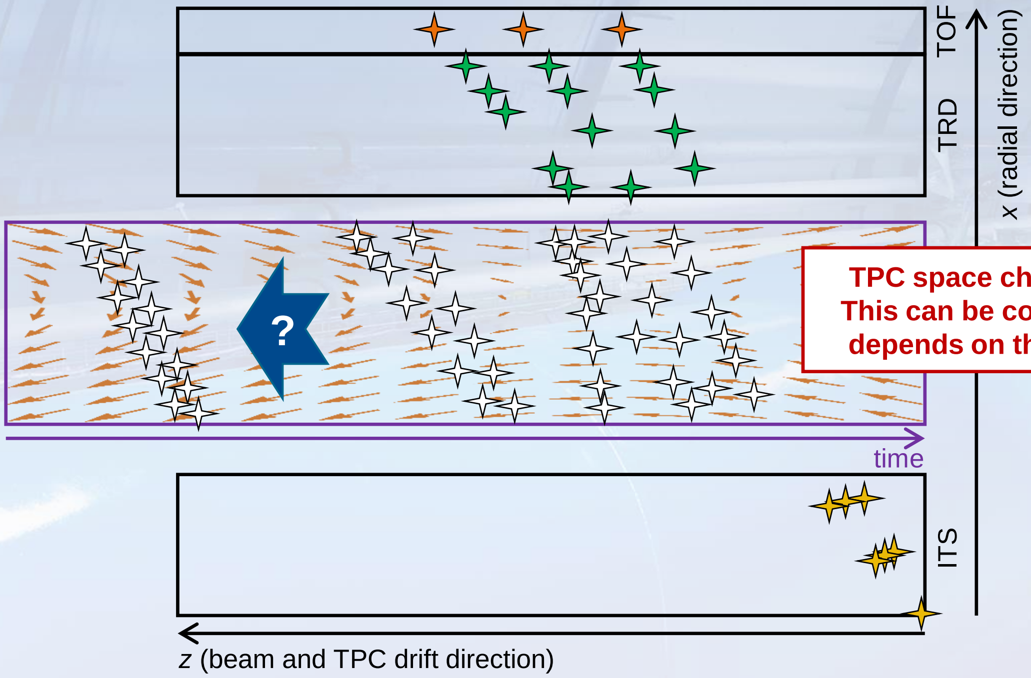
- **There are 2 (related) main challenges caused by continuous readout / space charge distortions**
  - How to assign a z-position to a cluster?
  - How to apply SCD corrections (inhomogeneous magnetic field, cluster error parameterization) if z is now known.





TPC Tracking

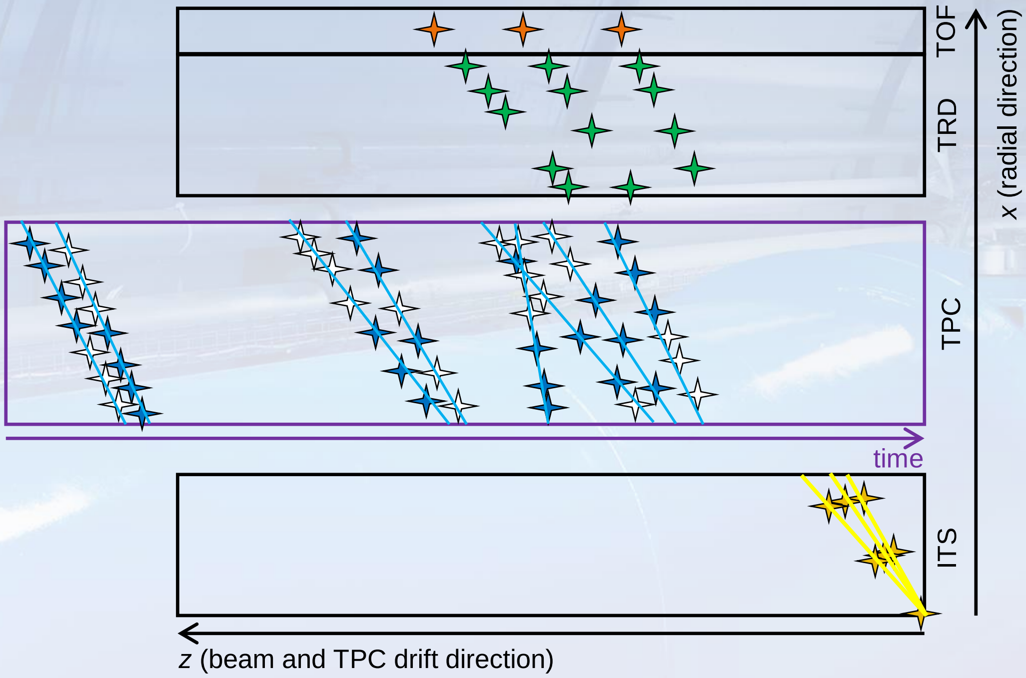
- **There are 2 (related) main challenges caused by continuous readout / space charge distortions**
  - How to assign a z-position to a cluster?
  - How to apply SCD corrections (inhomogeneous magnetic field, cluster error parameterization) if z is now known.



TPC space charge distorts the clusters. This can be corrected, but the correction depends on the *z* position of the cluster

TPC Tracking

- **There are 2 (related) main challenges caused by continuous readout / space charge distortions**
  - How to assign a z-position to a cluster?
  - How to apply SCD corrections (inhomogeneous magnetic field, cluster error parameterization) if z is now known.



- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.

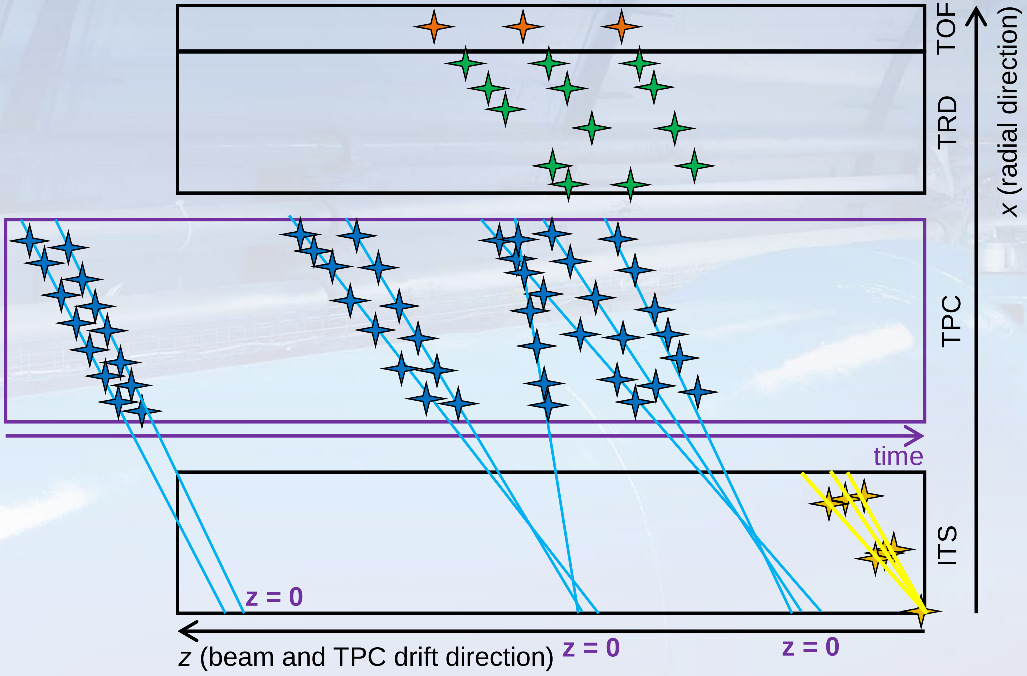
Precise tracking needs *z* for:

- Cluster error parameterization
- Inhomogeneous B-field
- Distortion correction

Effects smooth → irrelevant for initial trackletting

TPC Tracking

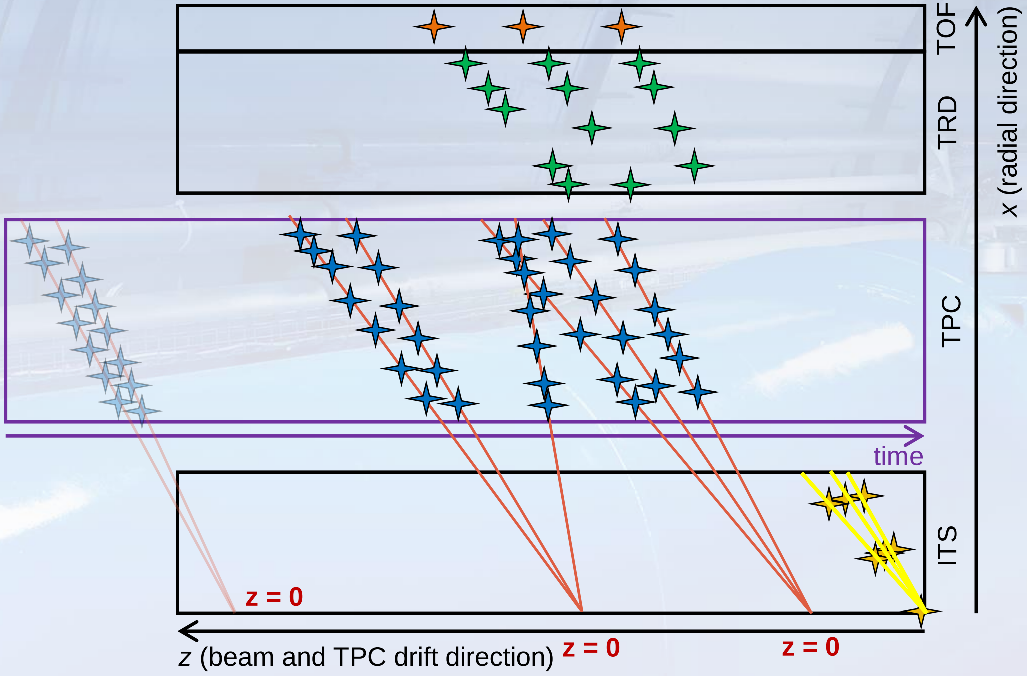
- **There are 2 (related) main challenges caused by continuous readout / space charge distortions**
  - How to assign a z-position to a cluster?
  - How to apply SCD corrections (inhomogeneous magnetic field, cluster error parameterization) if z is now known.



- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.
- Extrapolate to *x* = 0, define *z* = 0 as if the track was primary.
- Track following to find missing clusters. For cluster error parameterization, distortions, and B-field, shift the track such that *z* = 0 at *x* = 0.

TPC Tracking

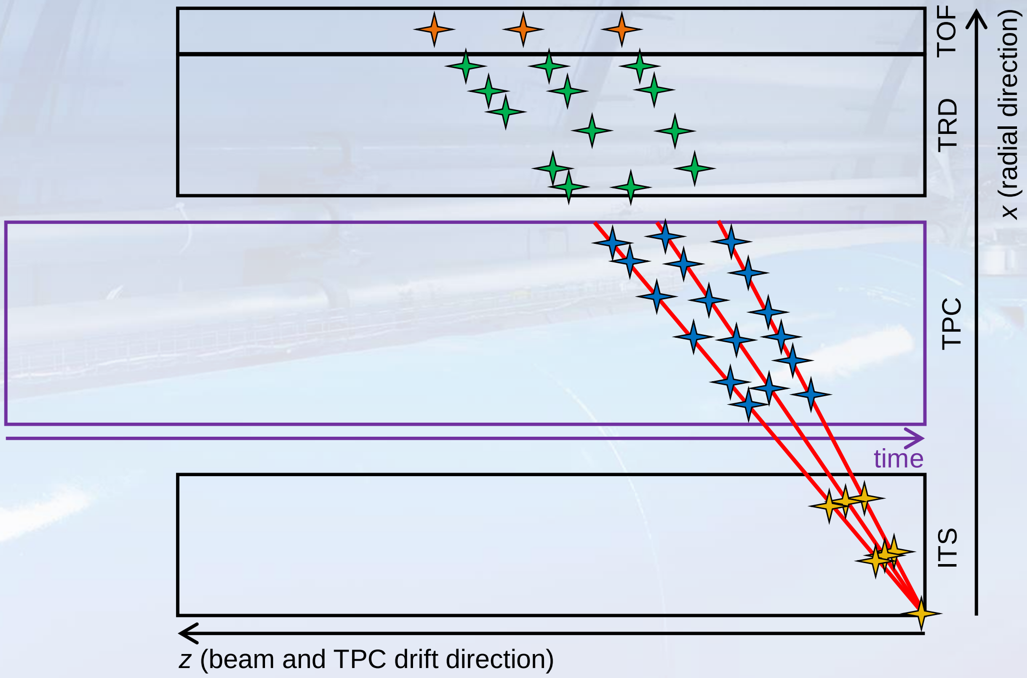
- **There are 2 (related) main challenges caused by continuous readout / space charge distortions**
  - How to assign a z-position to a cluster?
  - How to apply SCD corrections (inhomogeneous magnetic field, cluster error parameterization) if z is now known.



- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.
- Extrapolate to $x = 0$, define $z = 0$ as if the track was primary.
- Track following to find missing clusters. For cluster error parameterization, distortions, and B-field, shift the track such that $z = 0$ at $x = 0$.
- Refine $z = 0$ estimate, refit track with best precision
- For the tracks in one ITS readout frame, select all TPC tracks with a compatible time (from $z = 0$ estimate).

TPC Tracking

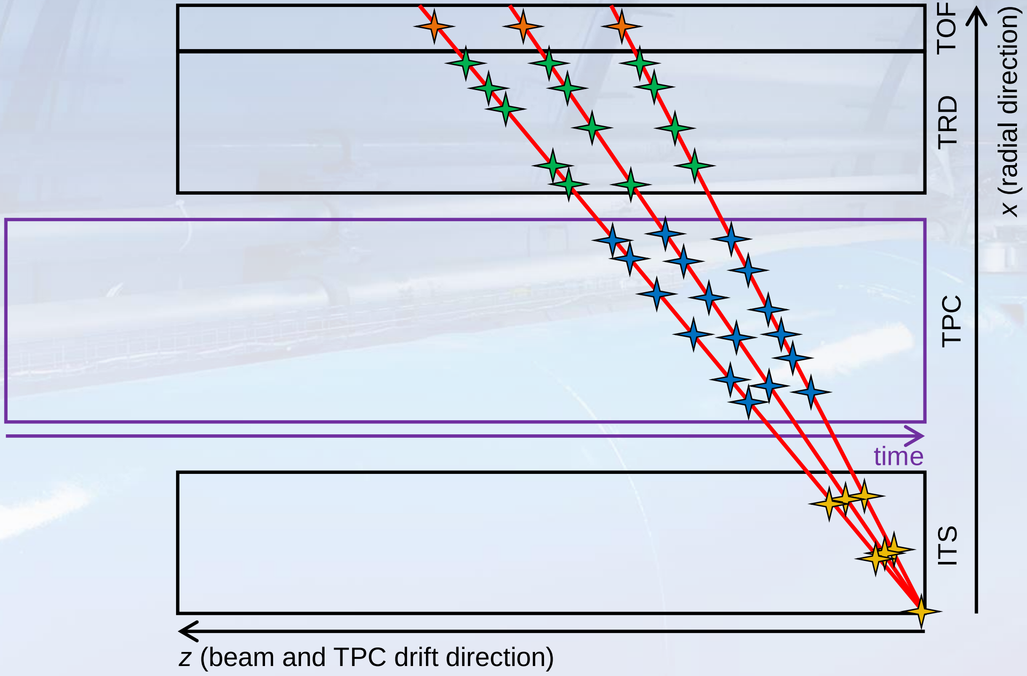
- **There are 2 (related) main challenges caused by continuous readout / space charge distortions**
  - How to assign a z-position to a cluster?
  - How to apply SCD corrections (inhomogeneous magnetic field, cluster error parameterization) if z is now known.



- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.
- Extrapolate to $x = 0$, define $z = 0$ as if the track was primary.
- Track following to find missing clusters. For cluster error parameterization, distortions, and B-field, shift the track such that $z = 0$ at $x = 0$.
- Refine $z = 0$ estimate, refit track with best precision
- For the tracks in one ITS readout frame, select all TPC tracks with a compatible time (from $z = 0$ estimate).
- Match TPC track to ITS track, fixing z-position and time of the TPC track.
- Refit ITS + TPC track outwards.

TPC Tracking

- **There are 2 (related) main challenges caused by continuous readout / space charge distortions**
  - How to assign a z-position to a cluster?
  - How to apply SCD corrections (inhomogeneous magnetic field, cluster error parameterization) if z is now known.



- Standalone ITS tracking.
- Standalone TPC tracking, scaling *t* linearly to an arbitrary *z*.
- Extrapolate to $x = 0$, define $z = 0$ as if the track was primary.
- Track following to find missing clusters. For cluster error parameterization, distortions, and B-field, shift the track such that $z = 0$ at $x = 0$.
- Refine $z = 0$ estimate, refit track with best precision
- For the tracks in one ITS readout frame, select all TPC tracks with a compatible time (from $z = 0$ estimate).
- Match TPC track to ITS track, fixing z-position and time of the TPC track.
- Refit ITS + TPC track outwards.
- Prolong into TRD / TOF.

TPC Tracking

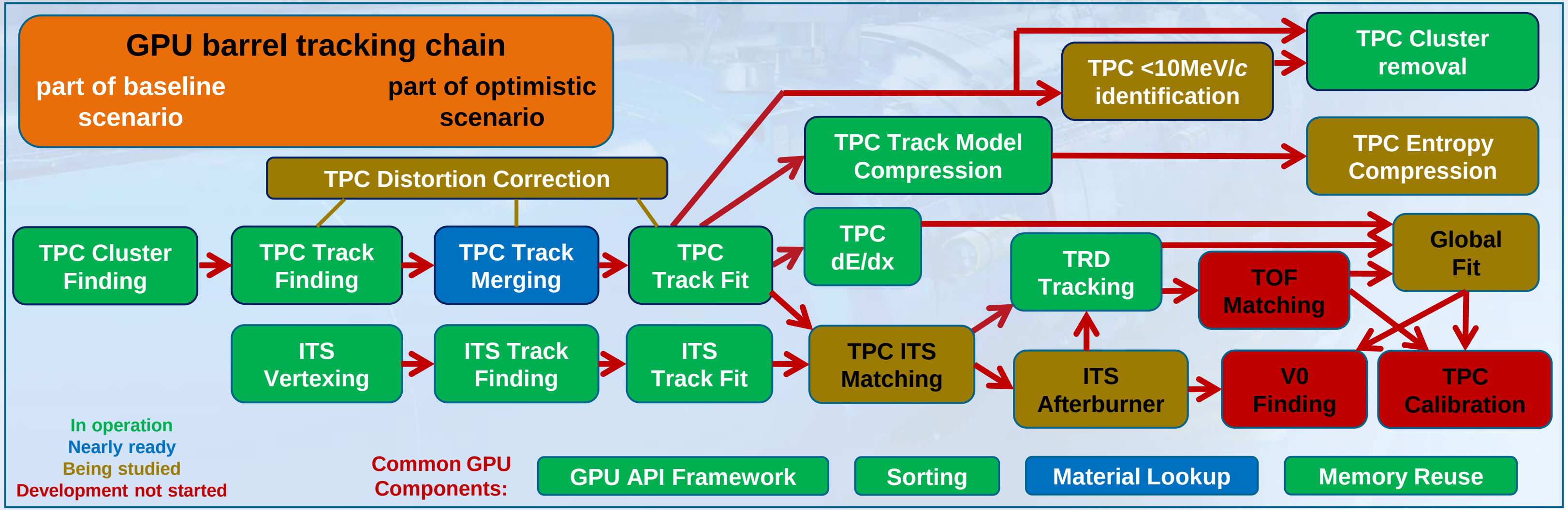
- **There are 2 (related) main challenges caused by continuous readout / space charge distortions**
  - How to assign a z-position to a cluster?
  - How to apply SCD corrections (inhomogeneous magnetic field, cluster error parameterization) if z is now known.
- **This works, but yields another subtle problem:**
  - Clusters are stored in a grid for fast cluster search during seeding / track following.
  - There is one common grid, cluster positions in the grid cannot be corrected.
  - Track position is in the “corrected coordinate system”, cannot be used to find clusters during track following, etc.
  - → Apply “inverse” correction to track position to identify grid cells for cluster search.
    - Select clusters in grid cell, and apply “forward” correction for candidates on the fly.
    - Select best cluster, and fit in corrected track coordinate system.
  - Requires repeated cluster transformation on the fly, but:
    - No need to store the clusters multiple times (TPC clusters are the largest data contribution).
    - Tracking runs only once.

Processing requirements

- **Synchronous processing:**
  - TPC data compression: Needs full TPC tracking for track model / cluster removal.
  - Calibration: Needs partial ITS / TPC / TRD / TOF tracking for a small subset of events.
  - → Full TPC tracking is largest compute contribution.
- **Asynchronous processing:**
  - Full TPC reconstruction.
    - Some additional steps like dE/dx and more sophisticated fit.
    - Overall, TPC faster than during synchronous processing: fewer clusters after removal, no clusterization, no compression.
  - Full ITS / TRD tracking.
    - More complex combinatorics than in TPC
  - Vertexing, etc.
  - → TPC tracking not the single dominant compute task.
- **TPC tracking defines synchronous workload and size of farm.**
  - Use GPUs, which are efficient at TPC tracking.
  - Processing partial time frames on the GPU would imply a special treatment at the borders.
  - Simpler to process full time frame on GPU at once, if possible (mostly a memory concern).
- **Optimistic scenario for asynchronous workload:**
  - Use GPUs for as many steps as possible, e.g. full barrel tracking.

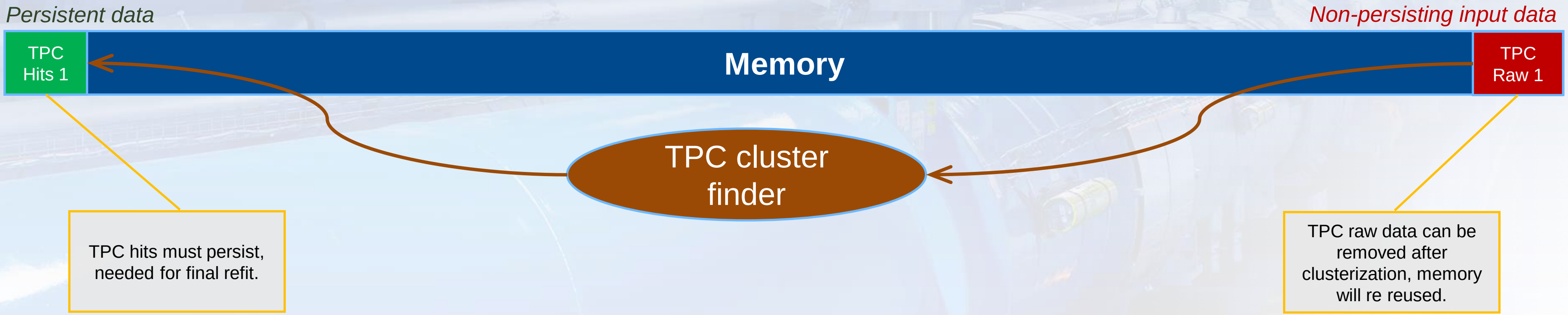
Overview of Barrel Tracking Chain on GPU

- **GPU components** for **baseline scenario** almost finished (**baseline = mandatory parts of synchronous reconstruction**):
  - **TPC distortion corrections** (most critical point now)
  - **Material lookup** during tracking not finished (not strictly needed for TPC).
  - **TPC Track Merger** still runs certain steps on the CPU, not critical.
  - **Junk identification below 10 MeV/*c*** missing (still searching for a good algorithm, affects compression ratio by ~15% in strategy A).
  - **TPC entropy compression** on GPU missing (not strictly needed, can run on CPU).
- Optimistic scenario for better GPU utilization in asynchronous reconstruction, work in progress.



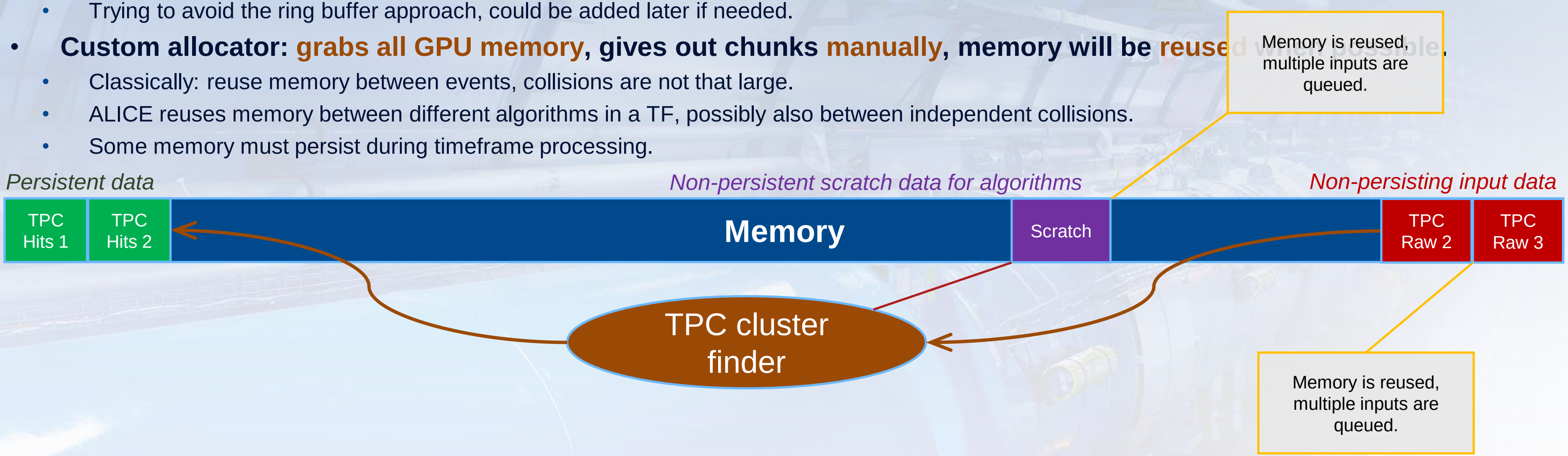
Memory requirements

- **ALICE reconstructs timeframes (TF) independently (128 – 256 orbits → ~10 - ~20 ms → ~500 - ~1000 collisions).**
  - One TPC drift time of data not reconstructible at TF border (~ 90 us) → < 0.5 – 1 % of statistics lost (baseline is 0.5 %).
  - Timeframe should fit in GPU memory. If not, could use kind of ring buffer, or reduce TF length to 128 orbits.
  - Trying to avoid the ring buffer approach, could be added later if needed.
- **Custom allocator: grabs all GPU memory, gives out chunks manually, memory will be reused when possible.**
  - Classically: reuse memory between events, collisions are not that large.
  - ALICE reuses memory between different algorithms in a TF, possibly also between independent collisions.
  - Some memory must persist during timeframe processing.



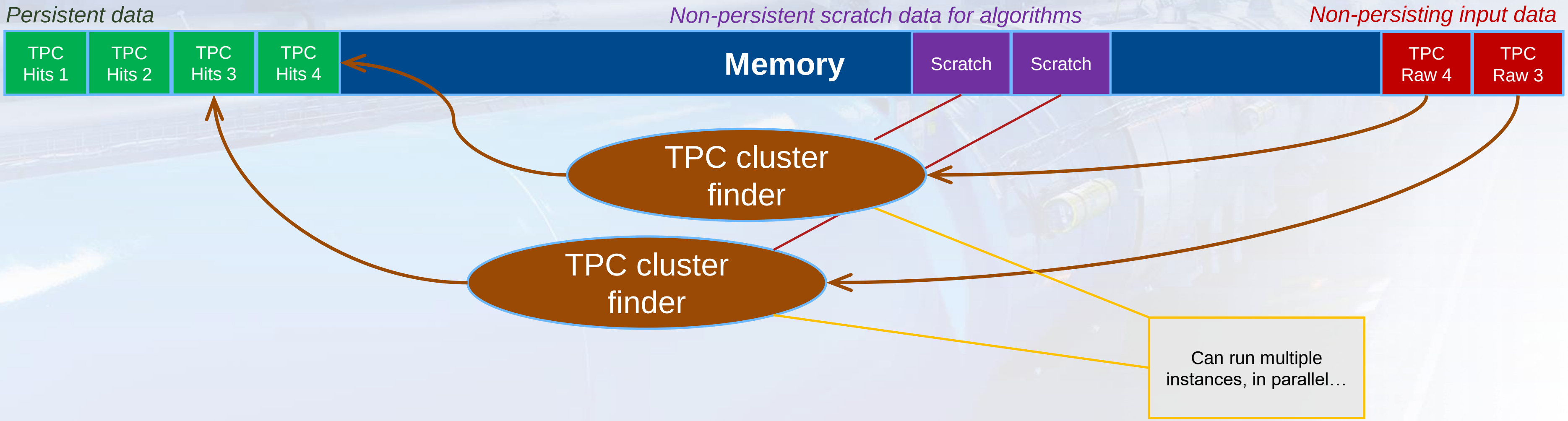
Memory requirements

- **ALICE reconstructs timeframes (TF) independently (128 – 256 orbits → ~10 - ~20 ms → ~500 - ~1000 collisions).**
  - One TPC drift time of data not reconstructible at TF border (~ 90 us) → < 0.5 – 1 % of statistics lost (baseline is 0.5 %).
  - Timeframe should fit in GPU memory. If not, could use kind of ring buffer, or reduce TF length to 128 orbits.
  - Trying to avoid the ring buffer approach, could be added later if needed.
- **Custom allocator: grabs all GPU memory, gives out chunks manually, memory will be reused when possible.**
  - Classically: reuse memory between events, collisions are not that large.
  - ALICE reuses memory between different algorithms in a TF, possibly also between independent collisions.
  - Some memory must persist during timeframe processing.



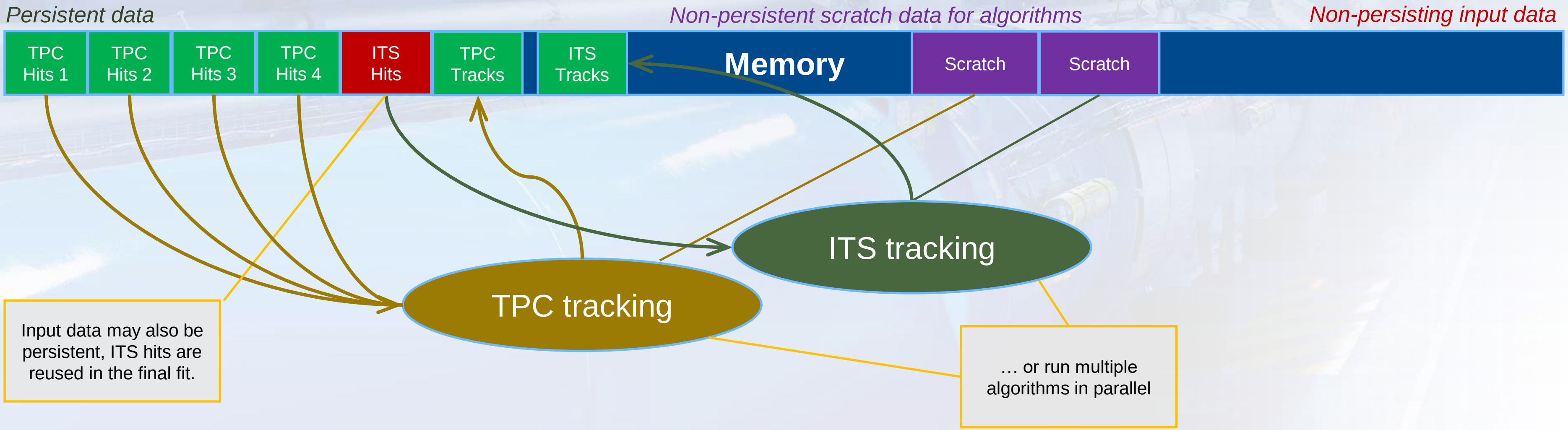
Memory requirements

- **ALICE reconstructs timeframes (TF) independently (128 – 256 orbits → ~10 - ~20 ms → ~500 - ~1000 collisions).**
  - One TPC drift time of data not reconstructible at TF border (~ 90 us) → < 0.5 – 1 % of statistics lost (baseline is 0.5 %).
  - Timeframe should fit in GPU memory. If not, could use kind of ring buffer, or reduce TF length to 128 orbits.
  - Trying to avoid the ring buffer approach, could be added later if needed.
- **Custom allocator: grabs all GPU memory, gives out chunks manually, memory will be reused when possible.**
  - Classically: reuse memory between events, collisions are not that large.
  - ALICE reuses memory between different algorithms in a TF, possibly also between independent collisions.
  - Some memory must persist during timeframe processing.



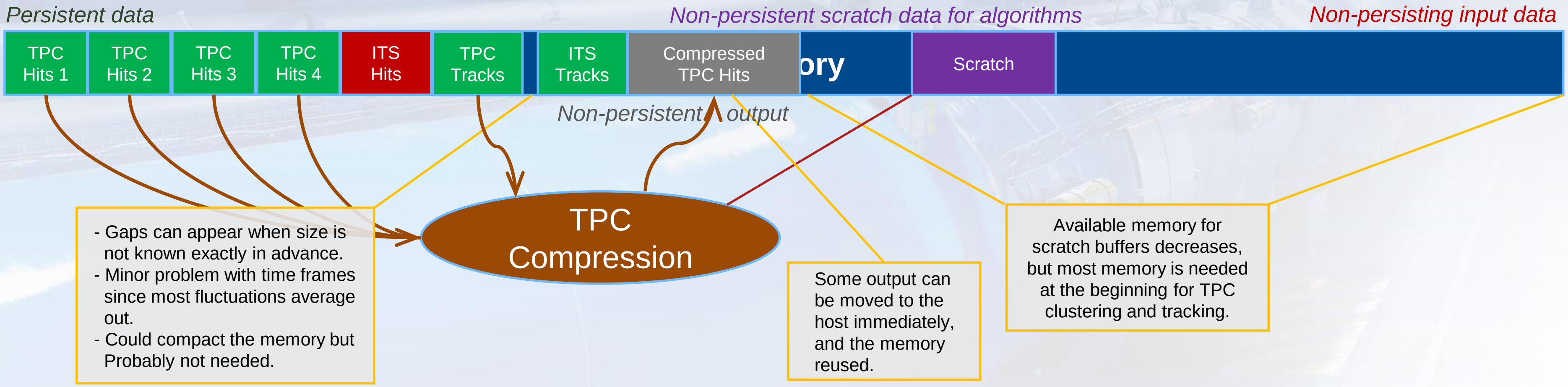
Memory requirements

- **ALICE reconstructs timeframes (TF) independently (128 – 256 orbits → ~10 - ~20 ms → ~500 - ~1000 collisions).**
  - One TPC drift time of data not reconstructible at TF border (~ 90 us) → < 0.5 – 1 % of statistics lost (baseline is 0.5 %).
  - Timeframe should fit in GPU memory. If not, could use kind of ring buffer, or reduce TF length to 128 orbits.
  - Trying to avoid the ring buffer approach, could be added later if needed.
- **Custom allocator: grabs all GPU memory, gives out chunks manually, memory will be reused when possible.**
  - Classically: reuse memory between events, collisions are not that large.
  - ALICE reuses memory between different algorithms in a TF, possibly also between independent collisions.
  - Some memory must persist during timeframe processing.



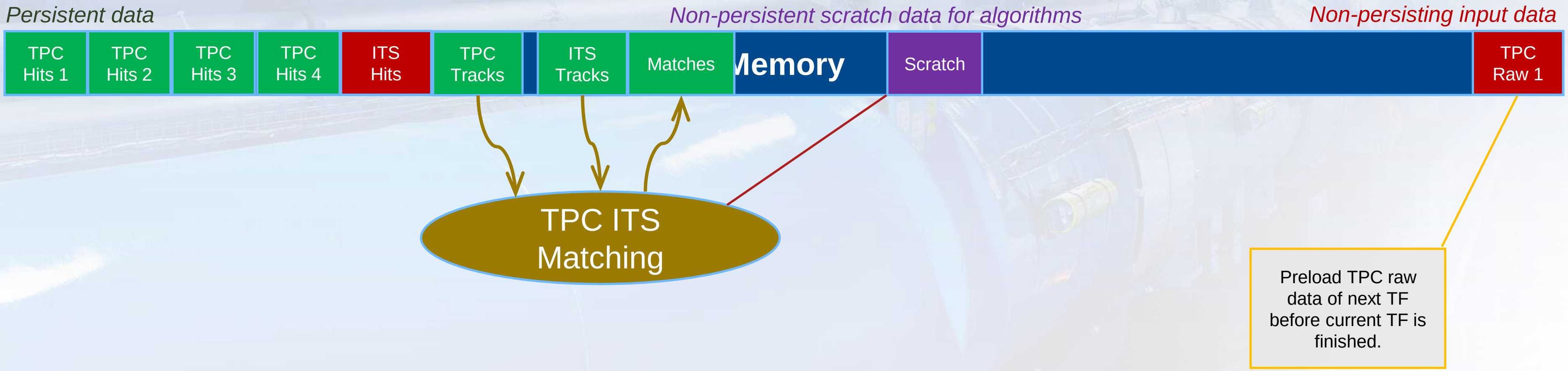
Memory requirements

- **ALICE reconstructs timeframes (TF) independently (128 – 256 orbits → ~10 - ~20 ms → ~500 - ~1000 collisions).**
  - One TPC drift time of data not reconstructible at TF border (~ 90 us) → < 0.5 – 1 % of statistics lost (baseline is 0.5 %).
  - Timeframe should fit in GPU memory. If not, could use kind of ring buffer, or reduce TF length to 128 orbits.
  - Trying to avoid the ring buffer approach, could be added later if needed.
- **Custom allocator: grabs all GPU memory, gives out chunks manually, memory will be reused when possible.**
  - Classically: reuse memory between events, collisions are not that large.
  - ALICE reuses memory between different algorithms in a TF, possibly also between independent collisions.
  - Some memory must persist during timeframe processing.



Memory requirements

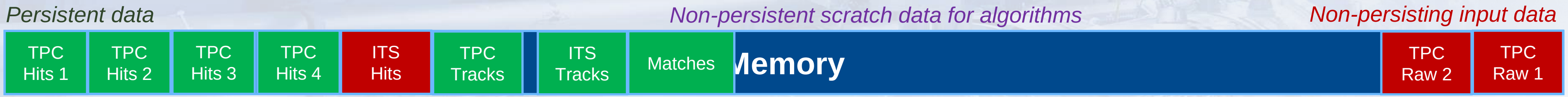
- **ALICE reconstructs timeframes (TF) independently (128 – 256 orbits → ~10 - ~20 ms → ~500 - ~1000 collisions).**
  - One TPC drift time of data not reconstructible at TF border (~ 90 us) → < 0.5 – 1 % of statistics lost (baseline is 0.5 %).
  - Timeframe should fit in GPU memory. If not, could use kind of ring buffer, or reduce TF length to 128 orbits.
  - Trying to avoid the ring buffer approach, could be added later if needed.
- **Custom allocator: grabs all GPU memory, gives out chunks manually, memory will be reused when possible.**
  - Classically: reuse memory between events, collisions are not that large.
  - ALICE reuses memory between different algorithms in a TF, possibly also between independent collisions.
  - Some memory must persist during timeframe processing.



Memory requirements

Work in Progress

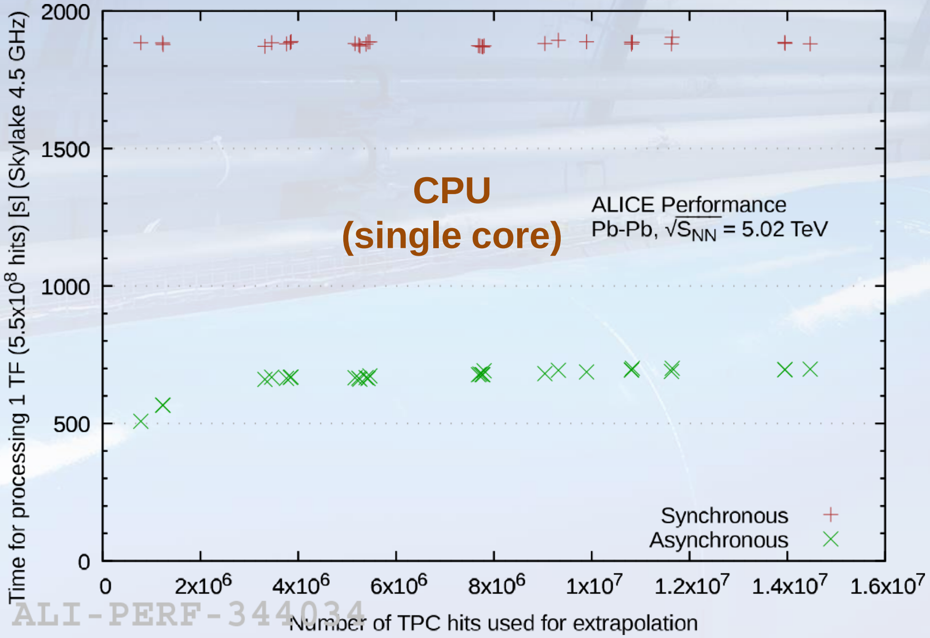
- **ALICE reconstructs timeframes (TF) independently (**128 – 256 orbits → ~10 - ~20 ms → ~500 - ~1000 collisions**).**
  - One TPC drift time of data not reconstructible at TF border (~ 90 us) → < 0.5 – 1 % of statistics lost (baseline is 0.5 %).
  - Timeframe should fit in GPU memory. If not, could use kind of ring buffer, or reduce TF length to 128 orbits.
  - Trying to avoid the ring buffer approach, could be added later if needed.
- **Custom allocator: grabs all GPU memory, gives out chunks manually, memory will be reused when possible.**
  - Classically: reuse memory between events, collisions are not that large.
  - ALICE reuses memory between different algorithms in a TF, possibly also between independent collisions.
  - Some memory must persist during timeframe processing.

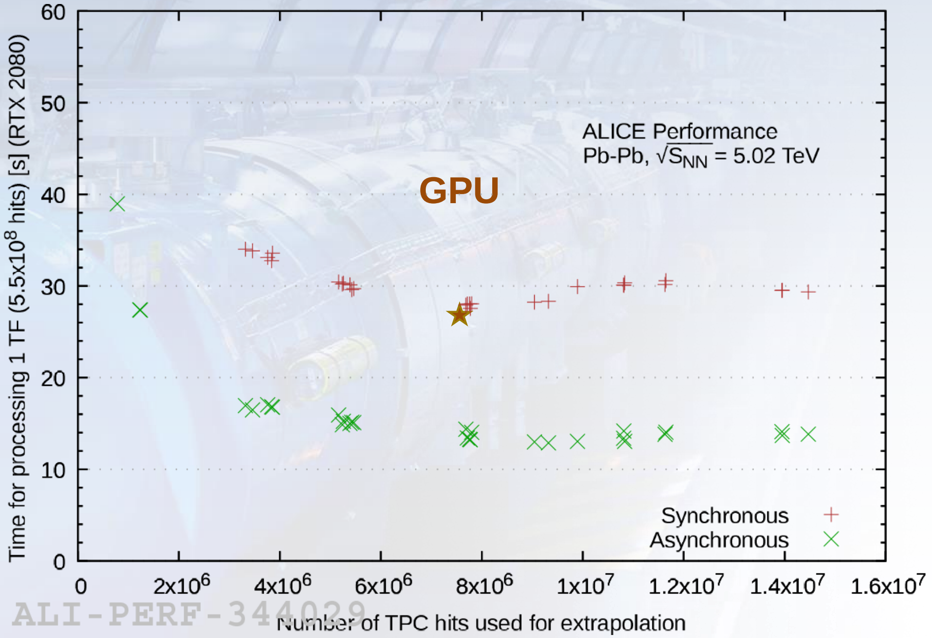


- **Estimated maximum memory needed during important steps (**with TF length = 128 orbits**):**
  - TPC Cluster finder: ~ 3 GB (+ input / scratch data, which is pipelined)
  - TPC Transformation: 12.1 GB
  - TPC Sector tracker: ~ 14.6 GB (including persistent memory from previous steps)
  - TPC Merger / track fit: 14.1 GB
  - TPC Compression: 12.9 GB
    - Later steps do not scale their scratch memory with TPC input → less memory intensive.
- **We assume a 16 GB GPU will suffice for TF of 128 orbits, unclear if 12 GB will suffice after optimizations.**

Performance (TPC processing only)

- **Critical assumption: The processing time must scale linearly (or less) with the input data size!**
  - Otherwise we must not process full time frames at once.
- **Cannot yet process a full time frame on GPU.**
  - Extrapolating to full time frame by extrapolating linearly in the number of clusters: The curves are flat → **Above assumption true**.
  - GPU slower for small input data due to insufficient parallelism, CPU slightly faster with small input due to better cache utilization.





Performance (TPC processing only)

- **Performance of Processing steps**
  (14.5 million TPC hits)

★ **Cluster finding has inefficient algorithm on CPU, dominates synchronous processing**
→ Asynchronous processing gives better GPU v.s. CPU estimate.

Step	AMD Radeon 7	NVIDIA RTX 2080 Ti	Intel Skylake 4.5 GHz 1 core	2080 Ti / CPU
Zero Suppression Decoding	38 ms	19 ms	986 ms	52x
Cluster Finding	87 ms	79 ms	★ 21343 ms	270x
Track Finding	109 ms	65 ms	8759 ms	135x
Track Fit	284 ms	243 ms	7204 ms	30x
Cluster Compression	137 ms	105 ms	1452 ms	14x
Synchronous Processing Total	657 ms	511 ms	★ 39816 ms	78x
dE/dx calculation	61 ms	22 ms	906 ms	41x
Asynchronous Processing total	304 ms	237 ms	★ 13381 ns	56x

- **Relative performance of GPU Models (Sync + Async):**

★ **Best options from AMD / NVIDIA**

GPU Model	Performance		GPU Model	Performance
NVIDIA RTX 2080 Ti ★	100.0%		NVIDIA V100	88.5%
NVIDIA Quadro RTX 6000 (active)	105.8%		NVIDIA T4	59.3%
NVIDIA Quadro RTX 6000 (passive) ★	94.5%		AMD Radeon 7 ★	77.8%
NVIDIA RTX 2080	83.5%		AMD MI50 ★	74.1%
NVIDIA GTX 1080 *(sorting excluded)*	60.1%		Intel Skylake 1 core 4.5 GHz *(async)*	1.77%

Performance (TPC processing only)

- **Performance of Processing steps**
  (14.5 million TPC hits)

★ **Cluster finding has inefficient algorithm on CPU, dominates synchronous processing**
→ Asynchronous processing gives better GPU v.s. CPU estimate.

Step	AMD Radeon 7	NVIDIA RTX 2080 Ti	Intel Skylake 4.5 GHz 1 core	2080 Ti / CPU
Zero Suppression Decoding	38 ms	19 ms	986 ms	52x
Cluster Finding	87 ms	79 ms	★ 21343 ms	270x
Track Finding	109 ms	65 ms	8759 ms	135x
Track Fit	284 ms	243 ms	7204 ms	30x
Cluster Compression	137 ms	105 ms	1452 ms	14x
Synchronous Processing Total	657 ms	511 ms	★ 39816 ms	78x
dE/dx calculation	61 ms	22 ms	906 ms	41x
Asynchronous Processing total	304 ms	237 ms	★ 13381 ms	56x

Most promising candidates from AMD / NVIDIA:
AMD Vega20 or NVIDIA Turing architectures

- **No big performance difference between professional / consumer cards, as expected since all code is single precision.**
- **2000 GPUs sufficient in all cases, with respectively more margin for NVIDIA Turing.**

- **Relative performance of GPU Models (Sync + Async):**

★ **Best options from AMD / NVIDIA**

GPU Model	Performance	GPU Model	Performance
NVIDIA RTX 2080 Ti ★	100.0%	NVIDIA V100	88.5%
NVIDIA Quadro RTX 6000 (active)	105.8%	NVIDIA T4	59.3%
NVIDIA Quadro RTX 6000 (passive) ★	94.5%	AMD Radeon 7 ★	77.8%
NVIDIA RTX 2080	83.5%	AMD MI50 ★	74.1%
NVIDIA GTX 1080 *(sorting excluded)*	60.1%	Intel Skylake 1 core 4.5 GHz *(async)*	1.77%

Summary

- ALICE will record 50 kHz Pb-Pb minimum bias data in Run 3 without trigger.
  - Continuous TPC readout, time frames of 10 – 20 ms instead of events.
- Storage of all data needs sophisticated data compression and online processing.
- Processing farm used for synchronous (online) and asynchronous processing (periods without beam).
  - Usage of GPUs mandatory in synchronous processing (baseline scenario).
  - Aiming to use GPUs as much as possible also in asynchronous processing (optimistic scenario).
  - Baseline scenario almost ready, promising candidate for optimistic scenario is the full barrel tracking chain.
- Most demanding calibration: TPC Space Charge distortions:
  - 2 Approaches combined: Track based calibration + Integrated digital currents.
- Tracking algorithm adapted to work with TPC distortions and continuous readout.
- TPC data needs to be compressed for 3.4 TB/s (uncompressed raw data) to O(70) GB/s (to disk buffer / permanent storage).
  - Improved version of Run 2 compression with online clusterization and entropy encoding, track model compression added.
  - In addition, clusters attached to tracks not used for physics are removed.
- Plan to process a full TF at once on the GPU.
  - Processing times scales linearly with input data size after a certain minimum size needed to fully exploit the GPU parallelism.
  - GPU memory size a concern, need ~16 GB for 10 ms time frames, which comes with < 1% loss of statistics.
- Many GPU models evaluated, RTX 2080 Ti can replace ~56 Skylake CPU cores at 4.5 GHz.
  - Most promising candidates are RTX 2080 Ti, Quadro RTX 6000, Radeon 7, MI 50, will need < 2000 GPUs.
  - Little performance difference between consumer / professional cards.
  - Stability is a concern, no guarantee for gaming cards but in the past also no problems with gaming cards at smaller scale.
  - Important features of professional cards are support, passive cooling, and to some extent memory size.