Proceedings of the CTD 2022
PROC-CTD2022-02
April 6, 2023

Standalone track reconstruction and matching algorithms for GPU-based High level trigger at LHCb

Brij Kishor Jashal

On behalf of LHCb Collaboration,
Instituto de Física Corpuscular,
University of Valencia, Spain

ABSTRACT

The LHCb Upgrade in Run 3 has changed its trigger scheme to implement a full software selection in two steps. In the first step, HLT1, the entire implementation will be on GPUs, running a fast selection aimed at reducing the visible collision rate from 30 MHz to 1 MHz. This selection relies on partial event reconstruction. A version of this reconstruction starts with two monolithic tracking algorithms: VELO-pixel tracking and HybridSeeding on Scintillating-Fiber tracker, which reconstruct track segments in standalone sub-detectors. These segments are then joined through a matching algorithm to produce 'long' tracks, forming the basis of the HLT1 reconstruction. We discuss the principle of these algorithms, as well as the details of their implementation, which allows them to run in a high-throughput configuration. Emphasis is put on the optimization of the algorithms themselves to take advantage of the GPU architecture. Finally, we present results in the context of the LHCb performance requirements for Run 3.

PRESENTED AT

Connecting the Dots Workshop (CTD 2022)
May 31 - June 2, 2022

1 Introduction

The LHCb detector is a single-arm forward spectrometer, covering the pseudo-rapidity range from two to five, designed for the study of particles containing b or c quarks. LHCb detector has went through major transformation wherein more than 90% of the active detector channels have been replaced, as a result the LHCb detector was mostly dismantled and a new detector has been installed [2]. In comparison to Run-2, this new detector is capable of sustaining up to a five times higher instantaneous luminosity, $2 \times 10^{33}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$, and six times the number of primary interactions. To meet the upgrade requirements, an efficient trigger system is the key for Run-3 data taking. To handle such high luminosity and still record the data without any loss of interesting physics, the trigger system should have (a) the capability to reduce 5 TB per seconds to a manageable 10 Giga-bytes per second data rates. (b) should make a decision in the limited time budget (c) should handle granular type small size workloads of O(150) kB (d) should have built in parallelism to effectively use underlying hardware to handle high throughput and (e) should be a cost effective solution that can be integrated within the LHCb online system. LHCb has managed to meet these requirements with a revolutionary two stage full software-only High Level Trigger (HLT) system wherein partial detector reconstruction and selection is carried out in the first stage of HLT (HLT1) built using Graphical Processing Units (GPUs) [2]. The output of HLT1 stage is written to the buffer storage and alignment and calibration procedure are carried out before the full detector reconstruction and selection in the second stage of the HLT (HLT2) system using x86 based CPUs. The upgraded LHCb dataflow focusing on the real-time aspects is shown in Figure 1 and different parts have been detailed in [7],[9],[10], [11].

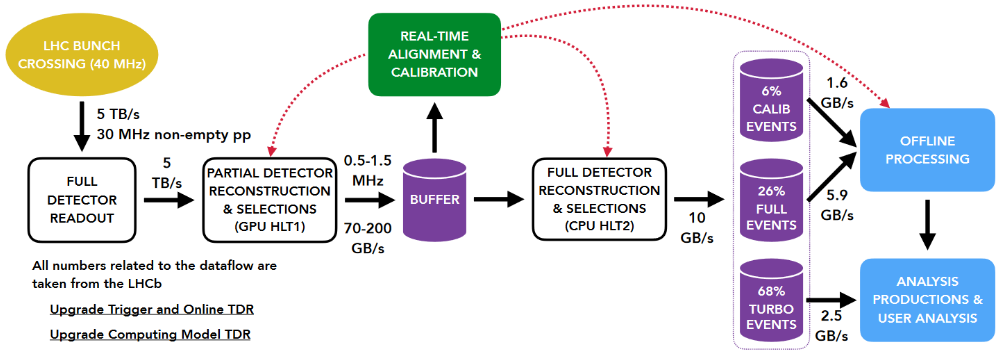


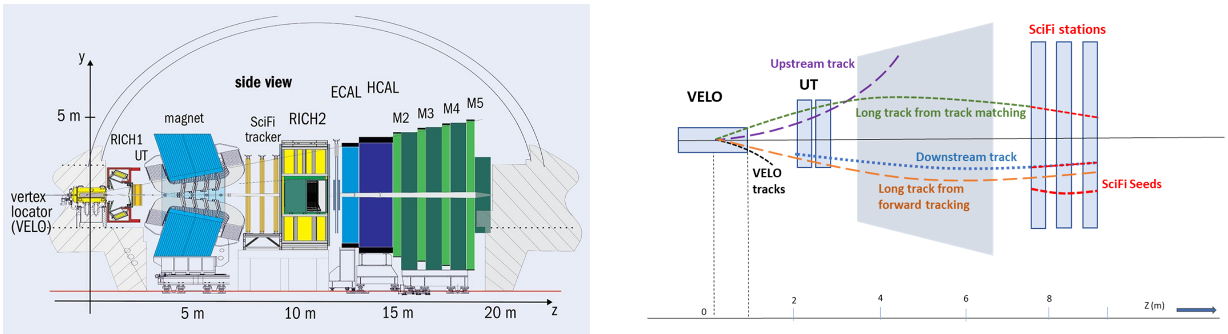
Figure 1: LHCb upgrade data-flow focusing on the real-time aspects.

2 LHCb detector and tracking system for Run-3

The three main tracking detectors in LHCb are the Vertex locator detector (VELO) which is a 52 plane silicon pixel detector, the Upstream tracker (UT) that is a silicon strip detector with 4 planes and the Scintillating Fiber (SciFi) detector placed after the dipole magnet. A Schematic of LHCb detector and different track types is shown in Figure 2. At the beginning of the Run-3 the VELO and SciFi are being commissioned while the UT is being prepared for installation at the end of 2022. Thus, the tracking system has to be adapted to run without UT.

Two new algorithms under commissioning at GPU based HLT1 stage of the data-flow chain are presented here. The first is a standalone track reconstruction algorithm for SciFi which produces SciFi track seeds and the second is a matching algorithm that matches these SciFi seeds with VELO tracks to create long tracks. This provides an alternate to the baseline HLT1 configuration [8] where long tracks are created by extending the VELO tracks to the SciFi stations and matched with SciFi hits. Figure 2.b shows different track types of LHCb.

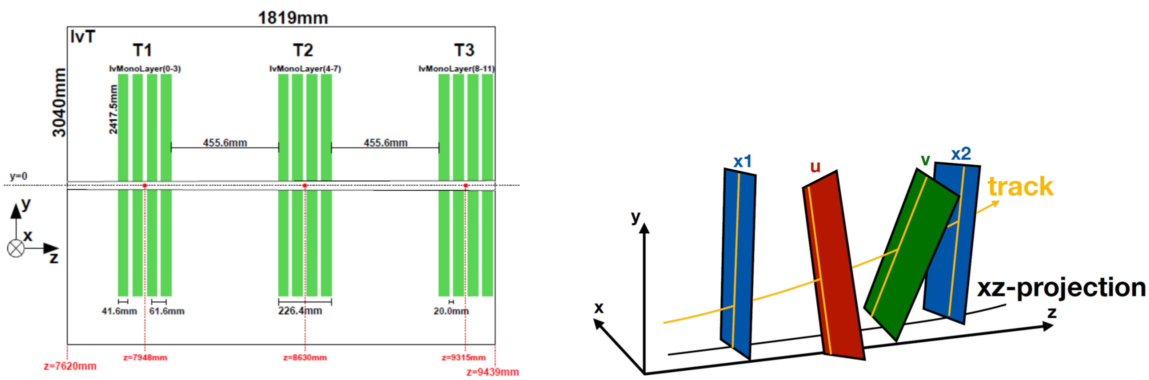
The SciFi detector is built using polystyrene scintillating fibres. A total of about 10,000 km of fibre in SciFi is organized in 1024 fibre mats with each mat containing 6 layers of fibres with 512 fibres across. These



(a) (b)

Figure 2: Schematic of (a) The upgraded LHCb detector (b) Sketch of LHCb Track types.

mats are then combined together in fibre modules with each module containing 8 mats. These modules are placed in the 3 stations (T1,T2 and T3) with each station having 4 layer (X, U, V, X) where the U and V are the stereo layers tilted by $\pm 5^o$ as shown in Figure 3.b. The first two stations (T1 and T2) contain 5 modules for each half and third station T3 contains 6 modules per half. Figure 3.a illustrates the SciFi detector and Figure 3.b shows the position of SciFi layers in each station with a track traversing a station in x,y,z coordinates.



(a) (b)

Figure 3: Schematic of (a) The SciFi detector layout, and (b) Position of SciFi layers in each station with a track traversing a station in x,y,z coordinates.

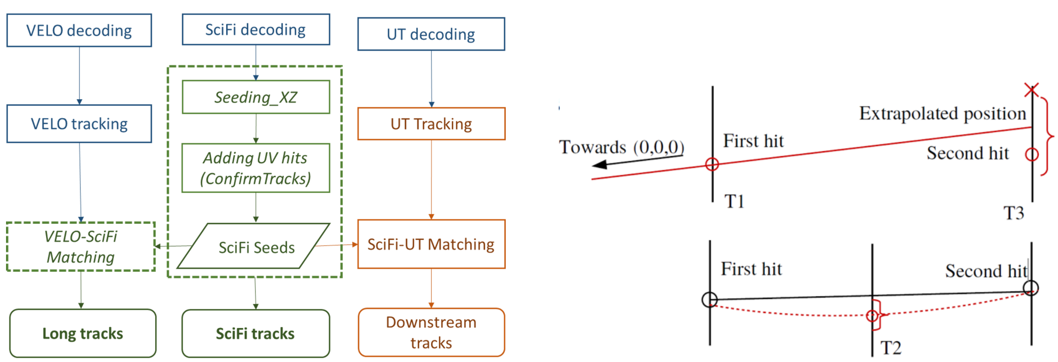
Measuring a precise location of hits in 3-dimensional space is crucial for tracking algorithms. In the SciFi detector the x and z information is obtained by identification of the unique SiPM channel where the ends of the fibres are connected and their respective position in the detector. By Combining the hit information from the UV layers with respect to x and z, we can calculate the y information.

3 A standalone SciFi seeding algorithm at HLT1

The SciFi Seeding algorithm for HLT1 on GPUs is an adaptation of the HybridSeeding algorithm implementation at HLT2 level based on CPUs [3]. The algorithm has been optimized for the performance, efficiency and throughput needs of HLT1 taking advantage of the heterogeneous computing architectures support under the Allen project [8]. It supports multiple architectures of GPU and CPU wthat can be chosen at the compilation time and is organized in two cases optimized for varying initial layer. Each case consists of two main stages of the algorithm, the first being *Seeding_XZ* which performs a pattern recognition in the *xz-plane*, that is the bending plane for tracks, followed by the second stage called *Seeding_confirmTracks* wherein the xz candidates are confirmed after adding the y information. These stages are explained below.

Seeding_XZ: In the first case, the first X layers of the three SciFi stations are used while in the second case, the second X layers are used. For every hit in the first X-layer of the T1, a small search window is opened in the first X-layer of T3 and all the hits in that window are matched to create two-hit combinations. The position and width of this search window are determined based on the assumption that the particle's origin vertex is (0,0,0) and that the minimum track momentum is $p_{min} > 3$ GeV/c. At this stage the tracks are assumed to be straight lines in the bending plane and the size of the search window in the T3 is adjusted using parameters which are tuned using simulation samples and depend on the position of layer of a respective station and the momentum range.

For every two-hit candidate, a charge-momentum estimation and slope calculations are performed using the extrapolation of the doublets to $z = 0$. This allows to define a tight search window for hits in the first X-layer in T2. Taking bending into account, all the hits from the T2 first X-layer within the tolerance window are added to create three-hit combinations as depicted in the Figure 4.b. The three-hit combinations are fitted with a parabola with cubic correction estimation on simulated samples to predict positions in the remaining layers. This is then extended to remaining layers. The candidates with at-least two additional hits from remaining layers, i.e at least 5 out of 6 possible hits, are selected and a track fit is performed in the x-z plane. Candidates with a high track quality are kept. This same procedure is performed in the second case where the second X layers of all stations are used as the initial set of layers. This helps in minimising the tracking performance losses due to hit detection inefficiencies from any specific layer. Each track candidate is assigned a score based on the number of hits and χ^2 value. A GPU optimized shared memory $O(N)$ voting algorithm is used for clone removal. The *Seeding_XZ* candidates still contains about 50% ghost or fake tracks as these candidates are merely the superposition of the real track candidates in the x-z plane since there is no y information yet.



(a) (b)

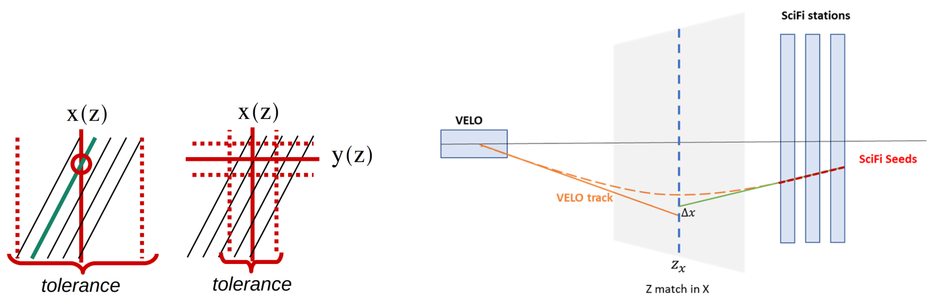
Figure 4: Schematic of (a) SciFi Seeding and Matching algorithm in HLT1 chain, (b) *Seeding_XZ* two-hit and three-hit combinations

Seeding_confirmTracks: For each XZ candidate, a large tolerance window for initial set of UV layer is opened. This is shown in Figure 5.a. In a tolerance window, many different fibres cross a given hypothesis and each fibre crosses that hypothesis at a different position of $x(z)$, giving y information. For all the hits collected in that window, the corresponding y and tolerance in $y(z)$ is determined. The tolerance window is updated at each hit found for subsequent layers. Tracks with a combined total of 10 hits or higher from X and UV layers are accepted and a linear model in y is used to calculate track parameters. The candidates with best χ^2 are confirmed as SciFi seeds. All these operations are carried out in parallel for all the XZ candidates.

4 VELO-SciFi track matching algorithm

VELO tracks are matched to the SciFi track seeds reconstructed by the standalone seeding algorithm to create *long-tracks*. Before beginning with the matching step, some of the VELO tracks are filtered out if they do not meet some basic criteria. The tracks generated due to back-propagation of the particles are removed and only the tracks that fall within the geometrical acceptance of the SciFi are considered for matching. This algorithm is adapted from a CPU based HLT2 matching algorithm called PrMatchNN [4]. A track state in LHCb at position z_i is defined as a vector of form $\vec{S}_i = (x, y, t_x, t_y, q/p)^T$ where x, y are coordinates, t_x and t_y are the slopes in x_z and y_z plane respectively and q/p is inverse track momentum times its charge q at $z = z_i$. A fast q/p estimation of tracks called the "p_T *kink*" approximation method is used to predict the tolerances at matching plane in z for a given momentum requirement, which is different for x_z and y_z planes as depicted in Figure 5.b. The actual momentum kink depends on the integrated magnetic field along the path followed by the track and is calculated using MC simulations taking the magnetic field map of LHCb detector into account Candidates with small differences in their slope with Δt_x ¡ 1.5 and Δt_y ¡ 0.02 are matched. In addition, for matching planes in the in x_z and y_z the differences in the extrapolated positions are required to be Δx ¡ 20 mm and for Δy ¡ 150 mm respectively.

Best candidates are selected using χ^2 minimization in Δx, Δy and Δt_y. Further, a clone killing procedure is applied wherein the tracks which share a VELO track are compared and only the ones with best χ^2 are kept. This produces *long-tracks* as the output.



(a) (b)

Figure 5: (a) Tolerance windows in UV search. (b) Kink approximation in the X_Z plane used in the matching algorithm.

5 GPU implementation features

To achieve the HLT1 level throughput and performance, it was necessary to design these algorithms as per the underlying GPU hardware architectures and efficiently utilize the built functionality of parallel programming from framework. The Allen framework provides built-in event level parallelization as every event in LHCb is an independent physics event. In the following, some of the key algorithm level design considerations are described.

- GPU Memory access: The choice of GPU memory type for a given operation can make significant difference in performance. In the SciFi seeding algorithm, SciFi hits are read multiple times in a random (non-coalesced) order and therefore the shared memory storage is preferred as compared to global memory. Each hit can be stored in a single float of 4 bytes and in the SciFi seeding algorithm we only load hits from 6 (X or UV) layers at a time. Thus if we take 300 hits as nominal for each layer then we have a total of 1800 hits in total which in turn require $7.2kB$ of shared memory. The binary search implementation for making pair candidates from hits is about 1.5 times faster using shared memory over global memory.

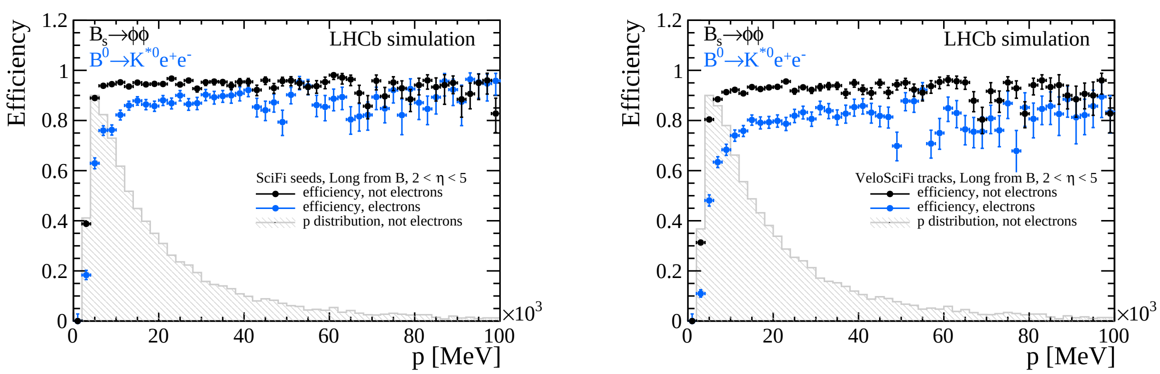
- Parallelisation: In the *Seeding_XZ* part, all the operations of creating two-hit, three-hit and multi-hit combinations are parallelised over the hits of the starting layer. In the *Seeding_confirmTracks* part, all of the operations of adding UV hits are done in parallel for each starting XZ seed from previous step. The VELO-SciFi matching algorithm is parallelised over the VELO tracks while matching with the SciFi track seeds.

- Thread block size optimization: From GPU hardware architecture perspective, A set of 32 threads that execute the same instructions are grouped into warps. A thread block is made up of several warps (maximum 64 for RTX A5000). A group of thread blocks is assigned to a Streaming Multi-processor(SM). An SM is a general purpose processor with limited amount of shared memory, caches, registers and execution cores. Optimal thread block size configuration is important to maximise the occupancy of the GPU and depends on the type of the GPU and its architecture. The *Seeding_XZ* and *Seeding_confirmTracks* parts require about 8 KB of shared memory per block to store the hits. This can be best met with 4 warps per block which gives a total of 128 threads. For Velo-SciFi Track Matching the block size is kept at 32.

6 Performance of algorithms.

The performance of SciFi Seeding and VELO-SciFi Matching algorithms presented above fully meets the HLT1 requirements. The sequence of HLT1 formed by these two algorithms produces *long-tracks* which is an alternative track reconstruction approach to the baseline approach called *Forward-tracking*. One should note that these performance figures are without the UT detector and are fully compatible with the baseline requirements without any cuts on transverse momentum. Few important performance parameters are presented below. Some additional performance plots are available in [5].

- Efficiency: Figure 6 shows the track reconstruction efficiency for *long-tracks* from B decays is 83% while for tracks with $p > 5$ GeV, the efficiency is 92%. For electrons, overall efficiency is 70% and for $p > 5$ Gev, efficiency is 75%.



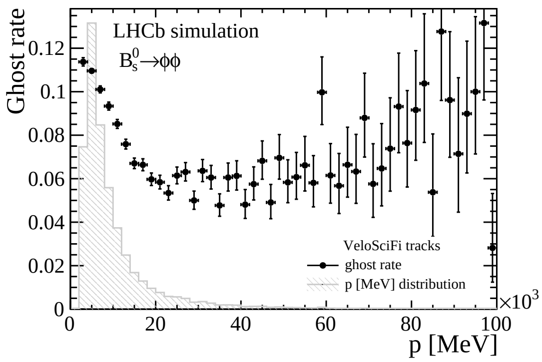
(a) SciFi seeding

(b) VELO-SciFi track matching

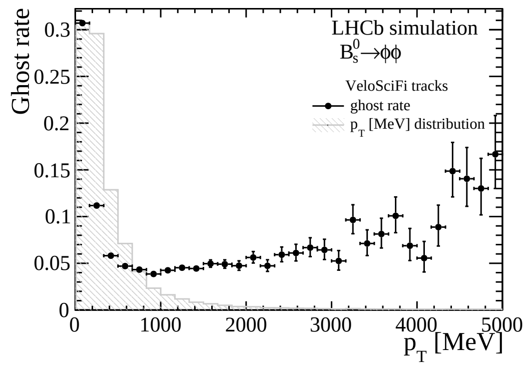
Figure 6: Tracking efficiencies for HLT1 from B decays as function of momentum p using 5000 simulated events for $B_s \to \phi\phi$ and $B^0 \to K^{*0}e^+e^-$

- Ghost rates: The Ghost rates for all reconstructed tracks stands at about 9% while for the momentum range of $p > 5$ GeV and $p_T > 0.5$ GeV, it comes down to 5%, as shown in Figure 7.

- Throughput: The baseline requirement for HLT1 throughput for Run-3 is 30 MHz. That means that the HLT1 needs to be able to process the input rate of about 30 million events per second and



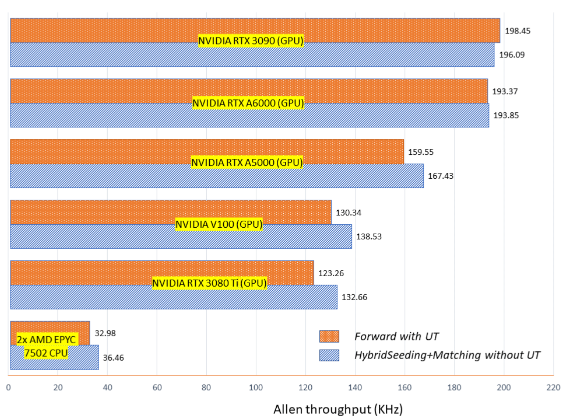
(a)



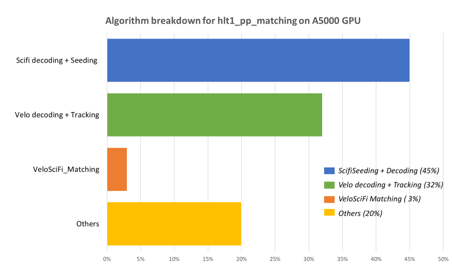
(b)

Figure 7: Ghost rates for HLT1 *long-tracks* from B decays as a function of momentum p and transverse momentum p_T. The plots show reconstructed tracks from 5000 simulated $B_s^0 \to \phi\phi$ events using VELO-SciFi track matching

produce output rate of about 1 million events per second. Figure 8.a shows the throughput of *VELO-SciFi Matching* sequence for different GPUs and CPUs without UT in comparison with the baseline (*Forward-tracking*) sequence with UT. In addition, Figure 8.b shows the breakdown of *VELO-SciFi Matching* sequence algorithms on RTX A5000 GPU.



(a)



(b)

Figure 8: (a) Throughput of *VELO-SciFi Matching* sequence (red) and default sequence (forward with UT) (blue) sequence on single GPU cards (NVIDIA GeForce RTX 3090, RTX A6000, RTX A5000, GeForce RTX 2080 Ti and NVIDIA V100). In addition the throughput of Allen compiled for CPU is shown for a CPU server (AMD EPYC 7502). The CPU throughput was measured on a single NUMA domain and multiplied by two. (b) Breakdown of the *VELO-SciFi Matching* sequence.

7 Conclusions and prospects

In the context of LHCb's Real Time Analysis data-flow for Run-3, two new algorithms for the GPU-based first stage of the High Level Trigger at LHCb have been presented along with their key design aspects on GPU architecture. A standalone seeding algorithm for the SciFi detector and a matching algorithm for SciFi

Seeds and VELO tracks. The performance of these algorithm meets the baseline requirements for Run-3 and integrates well with the HLT1 sequence. In addition, the availability of SciFi seeds at HLT1 level provides opportunities for adding downstream tracking to HLT1 by adding UT hits after the commissioning of the UT detector.

ACKNOWLEDGEMENTS

The author would like to acknowledge the support of the LHCb collaboration by thanking the LHCb computing, Online, RTA and simulation teams for their support and for producing the simulated LHCb samples used to develop and benchmark our algorithms. This work has been developed in the framework of the RTA project with contributions from members of RTA team and LHCb collaboration. we thank our colleagues of RTA for their support and dedication. We also acknowledge the funding support from MICINN national agency in Spain.

References

[1] LHCb's momentous metamorphosis, "CERN courier," [https://cerncourier.com/a/lhcbs-momentous-metamorphosis/].

[2] LHCb collaboration, "LHCb Upgrade Technical Design Reports," [LHCB-TDR-015, LHCB-TDR-016, LHCB-TDR-017 and LHCB-TDR-018].

[3] Aiola, S., Amhis, Y., Billoir, P., Jashal, B. K., Henry, L., Oyanguren, A., et al. (2021), "Hybrid seeding:A standalone track reconstruction algorithm for scintillating fibre tracker at lhcb," Computer Physics Communications **107713**, 260 (2021) [doi:https: //doi.org/10.1016/j.cpc.2020.107713].

[4] Esen, Sevda and De Cian, Michel, "A Track Matching Algorithm for the LHCb upgrade," LHCb Upgrade Technical Design Reports **CERN-LHCb-PUB-2016-027**, Dec (2016) [https://cds.cern.ch/record/2238266].

[5] LHCb collaboration, "Standalone track reconstruction and matching algorithms for the GPU-based High Level Trigger at LHCb," LHCB-FIGURE-2022-010 **LHCB-FIGURE-2022-010**, Mar (2022) [https://cds.cern.ch/record/2811214].

[6] Quagliani Renato, "Study of double charm B decays with the LHCb experiment at CERN and track reconstruction for the LHCb upgrade," CERN-THESIS-2017-254 **CERN-THESIS-2017-254**, Oct (2017) [https://cds.cern.ch/record/2296404].

[7] R. Aaij *et al.* [LHCb], "A Comparison of CPU and GPU Implementations for the LHCb Experiment Run 3 Trigger," Comput. Softw. Big Sci. **6** (2022) no.1, 1 doi:10.1007/s41781-021-00070-2 [arXiv:2105.04031 [physics.ins-det]].

[8] R. Aaij *et al.* "Allen: A high level trigger on GPUs for LHCb," Comput. Softw. Big Sci. **4**, no.1, 7 (2020) doi:10.1007/s41781-020-00039-7 [arXiv:1912.09161 [physics.ins-det]].

[9] T. Boettcher, "Allen in the first days of Run 3," PROC-CTD2022-33.

[10] A. Scarabotto, "Tracking on GPU at LHCb's fully software trigger," PROC-CTD2022-28.

[11] P. A. Günther, "LHCb's Forward Tracking algorithm for the Run 3 CPU-based online track reconstruction sequence," PROC-CTD2022-17.