Line Segment Tracking: Improving the Phase 2 CMS High Level Trigger Tracking with a Novel, Hardware-Agnostic Pattern Recognition Algorithm

E Vourliotis[1a] and P Chang[2], P Elmer[3], Y Gu[1], J Guiang[1], V Krutelyov[1], B V Sathia Narayanan[1], G Niendorf[4], M Reid[4], M Silva[2], A Rios Tascon[3], M Tadel[1], P Wittich[4], A Yagil[1]
on behalf of the CMS Collaboration

[1]University of California San Diego, CA, US
[2]University of Florida, FL, US
[3]Princeton University, NJ, US
[4]Cornell University, NY, US

E-mail: [a]emmanouil.vourliotis@cern.ch

Abstract. Charged particle reconstruction is one the most computationally heavy components of the full event reconstruction of Large Hadron Collider (LHC) experiments. Looking to the future, projections for the High Luminosity LHC (HL-LHC) indicate a superlinear growth for required computing resources for single-threaded CPU algorithms that surpass the computing resources that are expected to be available. The combination of these facts creates the need for efficient and computationally performant pattern recognition algorithms that will be able to run in parallel and possibly on other hardware, such as GPUs, given that these become more and more available in LHC experiments and high-performance computing centres. Line Segment Tracking (LST) is a novel such algorithm which has been developed to be fully parallelizable and hardware agnostic. The latter is achieved through the usage of the Alpaka library. The LST algorithm has been tested with the CMS central software as an external package and has been used in the context of the CMS HL-LHC High Level Trigger (HLT). When employing LST for pattern recognition in the HLT tracking, the physics and timing performances are shown to improve with respect to the ones utilizing the current pattern recognition algorithms. The latest results on the usage of the LST algorithm within the CMS HL-LHC HLT are presented, along with prospects for further improvements of the algorithm and its CMS central software integration.

1 Motivation and the Line Segment Tracking Algorithm

The High Luminosity Large Hadron Collider (HL-LHC) is the planned upgrade of the Large Hadron Collider (LHC) of CERN, with the target to collect data of proton-proton collisions corresponding to an integrated luminosity of more than $3000\,\mathrm{fb}^{-1}$ [1]. This can only be achieved by considerably enhancing the instantaneous luminosity, which, in turn, implies a drastic increase in the number of simultaneous collisions (pileup, PU). Because of this, the computational complexity of event reconstruction is projected to exceed the available computing resources, especially for the highly combinatorial task of trajectory pattern recognition of charged particles. This leads to both an increased timing, jeopardizing the ability

to reconstruct the data at the desired rate, and an increased cost due to the higher demands for processing power.

To accommodate the HL-LHC conditions, the LHC experiments are planning a major upgrade of their software and hardware infrastructure (Phase 2). The Line Segment Tracking (LST) algorithm aims at improving and parallelizing on Graphics Processing Units (GPUs) the charged hadron trajectory pattern recognition of the Phase 2 CMS experiment [2]. It uses as input the hits of the CMS Phase 2 outer tracker (OT) [3] and associates them to inner tracker (IT) tracks, ultimately producing a collection of OT+IT and OT-only track candidates. The early stages of the algorithm rely to a significant degree on the characteristics of CMS Phase 2 OT, qualitatively shown in Fig. 1: one of the key aspects of its design is that each layer comprises of 2 closely-spaced silicon sensors. In this way, two hits are recorded on each layer and are linked by the LST algorithm in a single pair of hits, called Mini Doublet (MD). The MDs are useful in reducing the combinatorics for the trajectory patterns and have the advantage of being locally reconstructed, which is utilized by the LST algorithm to parallelize their creation. Another handle for the reduction of the combinatorics is the definition of a lower p_{T} threshold for track reconstruction by tuning the search window for hit pairs. The current lower p_{T} threshold for LST is set at 0.8 GeV.

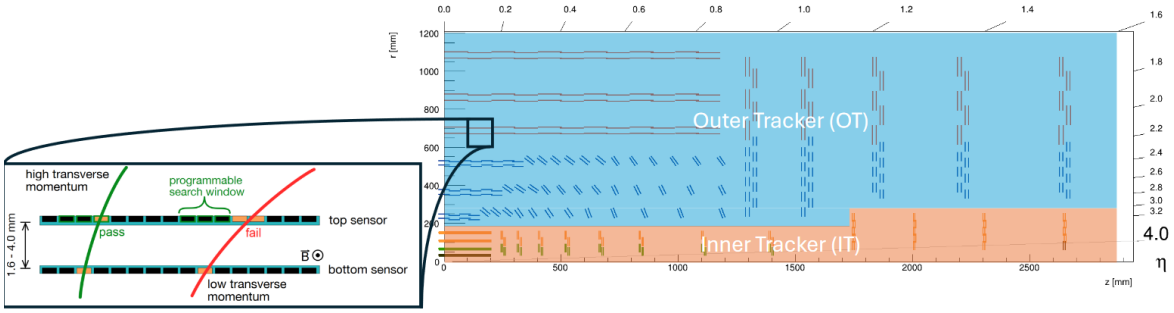


Figure 1: A qualitative representation of the expected Phase-2 CMS tracker geometry [3].

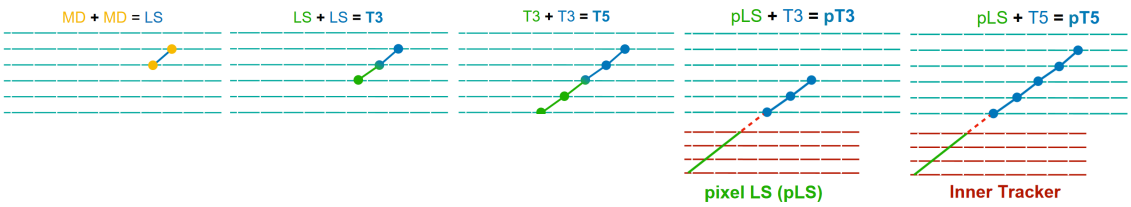
MDs serve as the elementary building blocks for the LST algorithm to create tracks. Based on precomputed connection maps for modules in the IT and OT, fulfilling geometric criteria that physical charged particle patterns obey, two MDs are linked to create a Line Segment (LS). The selections used to create LST objects are described in Ref. [4], and incorporate also machine learning methods, as detailed in Ref. [5]. Two LSs with a common MD are subsequently linked to form a T3, and two T3s with a common MD are linked to form a T5. T3s and T5s are combined with IT tracks (pLSs) to create pT3s and pT5s. Since the reconstruction of the above objects only requires local information, it can be massively parallelized, as all objects of the same kind can be created concurrently. Out of those objects, only a subset is propagated to downstream algorithms to be made into full tracks:

- pT5s, providing the majority of the efficiency.
- pT3s, complementing the pT5 efficiency.
- T5s, enabling the reconstruction of displaced tracks.
- Unlinked pLSs, covering the track reconstruction at high $|\eta|$, where there is no OT for the LST algorithm to create other objects.

The objects created by the LST algorithm are summarized in Fig. 2. Previously, the LST algorithm performance had been demonstrated in the offline reconstruction setup in Ref. [6]. It is worth noting that the LST algorithm is written using the Alpaka abstraction framework [7, 8, 9], therefore it seamlessly runs on multiple hardware devices.

2 Tracking in the Phase 2 CMS High Level Trigger

The High Level Trigger (HLT) is one of the two tiers of the system that collect the events of interest for CMS. It consists of a farm of processors running a version of the full event reconstruction software optimized for fast processing, and reduces the event rate before data storage. As it is responsible for the acquisition of the data as they are produced by the (HL-)LHC, timing plays an important role for the algorithms it runs. Apart from that, it needs to be general enough to cover for the majority of the potential physics signals.



(a) LS (b) T3 (c) T5 (d) pT3 (e) pT5

Figure 2: A qualitative representation of the different objects created by the LST algorithm [10].

The pattern recognition of tracks is of major importance for the HLT, as it is the basis of the reconstruction of most of the physics particles, while it needs to be run at a disproportionately short time for the combinatorial complexity of the task, especially at the harsh PU of HL-LHC. The CMS Phase 2 HLT uses the Combinatorial Kalman Filter (CKF) to reconstruct tracks with $p_{\mathrm{T}} > 0.9\,\mathrm{GeV}$ [11]. The track reconstruction is performed in two stages (iterations) based on different sets of initial track estimations (track seeding): the initial step produces tracks from pixel seeds with at least 4 hits (quads) created by the Patatrack algorithm [12, 13], while the highPtTriplet step produces tracks from pixel seeds with 3 hits (triplets), created by the legacy pixel seeding algorithm [14]. It is worth noting that the CKF algorithm used for the trajectory pattern recognition (track building) in this configuration is inherently sequential, and is implemented on Central Processing Unit (CPU). Once the built tracks, i.e. the set of hits originating from the same track, have been identified, these undergo a fitting procedure to extract the final track parameters, and are selected with requirements based on those parameters (tracking ID). The "highPurity" selection is applied, which provides a good balance between high efficiency and low fake+duplicate rate for prompt tracks [14]. This baseline configuration will be mentioned below as "Base CKF".

The LST algorithm can be an ideal candidate to run at the CMS Phase 2 HLT. LST allows for the parallel processing of track reconstruction on GPUs, hence keeping the timing under control, while extending the physics acceptance of the HLT to displaced tracks. In the following, a few potential configurations for integrating the LST algorithm in the CMS Phase 2 HLT are documented. The LST algorithm is utilized as a replacement for track building for the initial step, using pixel seeds with at least 3 hits as pLSs. Since the highPurity tracking ID has been optimized for prompt tracks, it is not applied to the LST objects targeting displaced tracks (T5s). This leads to a high efficiency for displaced tracks, without any significant increase in the fake and duplicate rate. Finally, as mentioned above, LST cannot build tracks for $|\eta| \gtrsim 2.5$, as that region is outside of the OT acceptance (Fig. 1). As a result, the CKF algorithm still needs to run in the highPtTriplet step to recover efficiency in those high $|\eta|$ regions. Different configurations are being used for the seeding of this "recovery iteration": in the "LST with CKF on Legacy Triplets" configuration, legacy triplets are used, in the "LST with CKF on LST Quads" configuration only the quad LST pLSs are used, while in the "LST with CKF on LST Quads+Triplets" configuration both the quad and triplet LST pLSs are used. The last two configuration imply that the LST algorithm can be used not only as a track building but also as a track seeding algorithm. All of the configurations described above are summarized in a more condensed format in Table 1.

3 Physics Performance and Throughput

This section is dedicated to the measurement of the physics and computational performance of the CMS Phase 2 HLT configurations using LST for track reconstruction. Three metrics are used for the physics performance:

- Efficiency: The fraction of the matched simulated tracks from the hard-scattering vertex.
- Fake rate: The fraction of reconstructed tracks not matched to any simulated track.
- Duplicate rate: The fraction of reconstructed tracks matched to any simulated track that is matched to multiple reconstructed tracks.

For the measurement of the efficiency, the simulated tracks from the hard-scattering vertex are matched to the reconstructed tracks. A given simulated track is considered a match to a reconstructed one if more than 75% of the hits of the reconstructed track originate from the simulated track. For the measurement of the fake and the duplicate rate, all simulated tracks are used for the matching. In the following, any selections applied to the simulated or reconstructed tracks (depending on the metric, as

Table 1: Summary of the HLT tracking sequence setup for each configuration described in this note [10].

Iteration	Procedure	Base CKF	LST with CKF on Legacy Triplets	LST with CKF on LST Quads	LST with CKF on LST Quads+Triplets
Initial	Seeding	Patatrack quads	Patatrack quads + Legacy Triplets	Patatrack quads + Legacy Triplets	Patatrack quads + Legacy Triplets
	Building	CKF	LST	LST	LST
	Tracking ID	highPurity	highPurity (pT3, pT5, pLS) None (T5)	highPurity (pT3, pT5) None (T5)	highPurity (pT3, pT5) None (T5)
HighPt Triplet	Seeding	Legacy triplets	Legacy triplets	LST pLS quads	LST pLS quads+triplets
	Building	CKF	CKF	CKF	CKF
	Tracking ID	highPurity	highPurity	highPurity	highPurity

described above) are shown on the plots. The radial distance and the z position of the production vertex of the tracks are denoted as r_{vertex} and z_{vertex} respectively. A simulated $t\bar{t}$ sample produced with 200 PU for the upgraded Phase 2 detector geometry is used for the measurements below.

Figure 3 shows the efficiency of the different configurations tested as a function of the simulated track p_T (left) and r_{vertex} (right). It is obvious that the efficiency is lower when using only quads in the recovery iteration, highlighting the importance of triplets for track seeding in the current setup. When triplets are used, the LST configurations reach an efficiency that is comparable to the one by Base CKF, or even higher for $p_T \lesssim 5\,\text{GeV}$. The efficiency as a function of r_{vertex} demonstrates the fact that any configuration using LST for track building allows for acceptance of displaced tracks ($r_{vertex} \gtrsim 5$ cm). This constitutes a completely new feature for the CMS HLT. Notably, the efficiency drops at the radial distances roughly corresponding to tracker layers, with an endpoint at $\sim$35 cm, where less than 4 OT layers are available, so no T5s can be built.

The fake rate is lower for any configuration using LST for track building. Importantly, the right plot of Fig. 4 shows that most of the fake rate reduction comes for tracks with $p_T < 10\,\text{GeV}$. Given that the bulk of tracks have low p_T, this implies a significant reduction of computational resources downstream, as less tracks need to be processed. The left plot of Fig. 4 indicates a higher duplicate rate when the recovery CKF iterations runs on legacy triplets, as both legacy triplets and LST pLSs have the potential to reconstruct the same track. When only the LST pLSs are used for track seeding, the overall duplicate rate is lower throughout the whole p_T range.

Based on the computational performance of the current offline tracking reconstruction, displaced tracking takes as much time as the prompt track reconstruction. As displaced tracking is completely missing from the current CMS Phase 2 HLT configuration, its addition would imply a 50% reduction of the throughput. Table 2 shows the throughput of the tracking sequence for CMS Phase 2 HLT configurations using LST, normalized to that of the Base CKF configuration. The measurements were performed with 2 threads (for CPU), pinned to 2 specific CPU cores, and 2 streams (for GPU) with local access to the input. An AMD EPYC "Milan" 7763 CPU and an NVIDIA "Ampere" A30 PCIe GPU were used. The results imply that a throughput reduction of at most 30% is expected when running LST, hence including displaced tracking and improving various performance metrics, as outlined above. The throughput reduction comes only for the LST configuration that run single-threaded on CPU. When the LST configurations are executed on GPU, the throughput is comparable with the Base CKF one or even increases. It was observed that most of the slowdown for the LST configurations is actually coming from the recovery iteration, which is run with CKF.

4 Summary and Outlook

LST is a novel, hardware-agnostic pattern recognition algorithm, targeting application to the CMS Phase 2 tracking. The algorithm can bring improvements both to the physics performance, as it extends the acceptance of the current tracking implementation to displaced tracks, and to the computational performance, as it efficiently runs on GPUs, potentially increasing the event reconstruction throughput. As such, it is a suitable candidate for performing the charged particle trajectory pattern recognition at the CMS Phase 2 HLT, where both the physics acceptance and timing considerations are of utmost importance. This work presented the first exploratory integration of the LST algorithm in the CMS

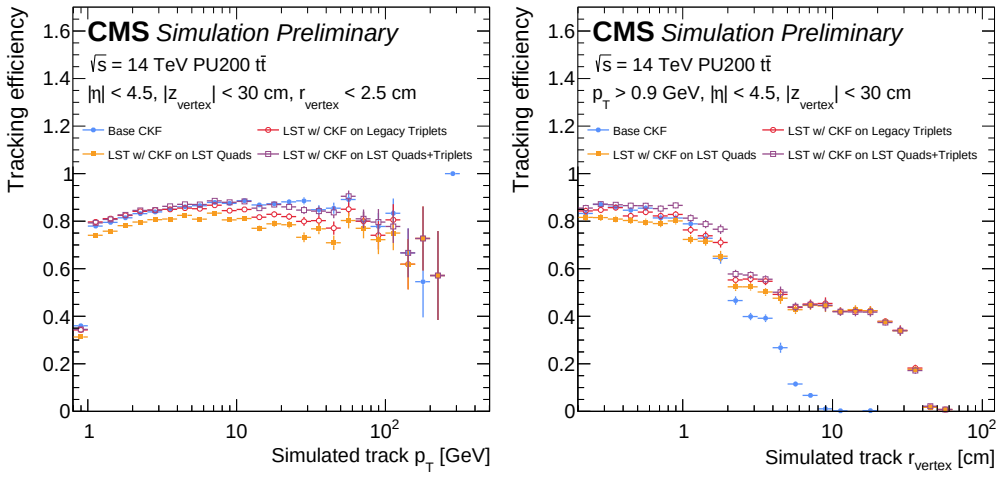


Figure 3: The tracking efficiency is shown for Base CKF (blue), LST with CKF on Legacy Triplet (red), LST with CKF on LST Quads (orange) and LST with CKF on LST Quads+Triplets (purple) as a function of the simulated track p_{T} (left) and $\mathrm{r}_{\mathrm{vertex}}$ (right) [10].

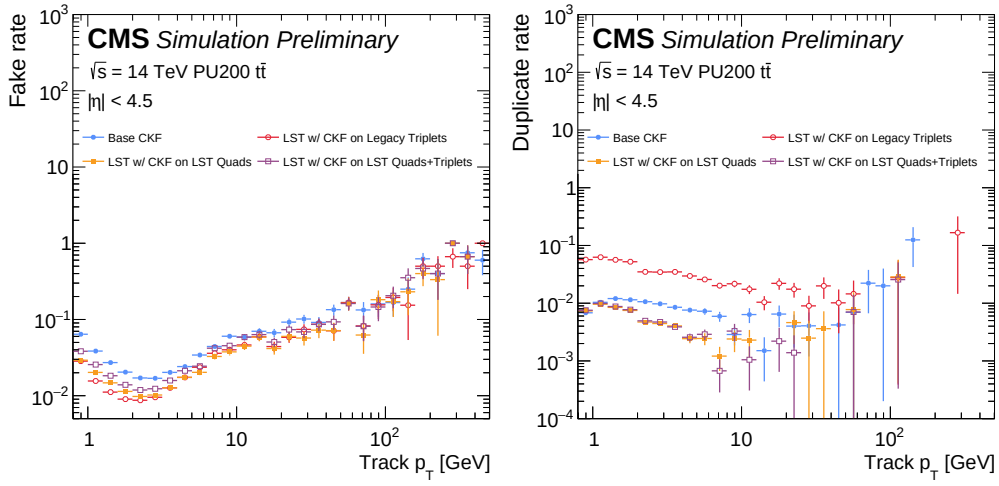


Figure 4: The tracking fake rate (left) and duplicate rate (right) are shown for Base CKF (blue), LST with CKF on Legacy Triplet (red), LST with CKF on LST Quads (orange) and LST with CKF on LST Quads+Triplets (purple) as a function of the reconstructed track p_{T} [10].

Table 2: Throughput of the HLT tracking sequence for different configurations, normalized to that of the Base CKF one [10].

	LST with CKF on Legacy Triplets	LST with CKF on LST Quads	LST with CKF on LST Quads+Triplets
LST on CPU Throughput / Base CKF	0.72 ± 0.07	0.86 ± 0.07	0.70 ± 0.09
LST on GPU Throughput / Base CKF	1.03 ± 0.09	1.35 ± 0.12	0.92 ± 0.09

Phase 2 HLT, demonstrating a lot of potential for the future.

On top of the improvements showcased above, more developments are planned both for the LST algorithm and for the CMS Phase 2 HLT configuration. The former involve the creation of more objects, the integration of more machine learning methods and the optimization of its implementation on CPU for LST, while the latter revolve around the optimization of the usage of the Patatrack algorithm for track seeding and the usage of the mkFit algorithm [15, 16] for the track building of the recovery iteration.

Acknowledgements
This work was supported by the U.S. National Science Foundation under Cooperative Agreements OAC-1836650, PHY-2323298, and PHY-2121686 and grant PHY-2209443.

References

[1] Aberle O et al 2020 High-Luminosity Large Hadron Collider (HL-LHC): Technical design report CERN-2020-010 URL https://cds.cern.ch/record/2749422

[2] CMS Collaboration 2008 *Journal of Instrumentation* **3** S08004

[3] CMS Collaboration 2017 The Phase-2 Upgrade of the CMS Tracker CMS-TDR-014 URL https://cds.cern.ch/record/2272264

[4] Chang P, Elmer P, Gu Y, Krutelyov V, Niendorf G, Reid M, Narayanan B V S, Tadel M, Vourliotis E, Wang B, Wittich P and Yagil A 2022 *Journal of Physics: Conference Series* **2375** 012005

[5] CMS Collaboration 2023 Improved Performance of Line Segment Tracking Using Machine Learning CMS-DP-2023-075 URL https://cds.cern.ch/record/2872904

[6] CMS Collaboration 2023 Performance of Line Segment Tracking algorithm at HL-LHC CMS-DP-2023-019 URL https://cds.cern.ch/record/2857438

[7] Worpitz B 2015 Investigating performance portability of a highly scalable particle-in-cell simulation code on various multi-core architectures URL http://dx.doi.org/10.5281/zenodo.49768

[8] Zenker E, Worpitz B, Widera R, Huebl A, Juckeland G, Knüpfer A, Nagel W E and Bussmann M 2016 Alpaka - an abstraction library for parallel kernel acceleration (IEEE Computer Society) (*Preprint* 1602.08477) URL http://arxiv.org/abs/1602.08477

[9] Matthes A, Widera R, Zenker E, Worpitz B, Huebl A and Bussmann M 2017 Tuning and optimization for a variety of many-core architectures without changing a single line of implementation code using the alpaka library (*Preprint* 1706.10086) URL https://arxiv.org/abs/1706.10086

[10] CMS Collaboration 2024 Performance of the Line Segment Tracking Algorithm in the CMS Phase-2 High Level Trigger Tracking CMS-DP-2024-014 URL https://cds.cern.ch/record/2890677

[11] CMS Collaboration 2021 The Phase-2 Upgrade of the CMS Data Acquisition and High Level Trigger CMS-TDR-022 URL https://cds.cern.ch/record/2759072

[12] Bocci A, Innocente V, Kortelainen M, Pantaleo F and Rovere M 2020 *Frontiers in Big Data* **3**

[13] CMS Collaboration 2022 Performance of Run-3 HLT Track Reconstruction CMS-DP-2022-014 URL https://cds.cern.ch/record/2814111

[14] CMS Collaboration 2014 *Journal of Instrumentation* **9** P10009

[15] Lantz S, McDermott K, Reid M, Riley D, Wittich P, Berkman S, Cerati G, Kortelainen M, Reinsvold A H, Elmer P, Wang B, Giannini L, Krutelyov V, Masciovecchio M, Tadel M, Würthwein F, Yagil A, Gravelle B, Norris B 2020 *Journal of Instrumentation* **15** P09030

[16] CMS Collaboration 2022 Performance of Run 3 track reconstruction with the mkFit algorithm CMS-DP-2022-018 URL https://cds.cern.ch/record/2814000