
UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

# Enhancing the capabilities of the Event Filter tracking in the ATLAS experiment at the HL-LHC

Diploma Thesis

**Eleni Xochelli**

**Supervisor:** Christos Antonopoulos

February 2025

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250


UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

# Enhancing the capabilities of the Event Filter tracking in the ATLAS experiment at the HL-LHC

Diploma Thesis

**Eleni Xochelli**

**Supervisor:** Christos Antonopoulos

February 2025



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250


ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

# Επέκταση των δυνατοτήτων του συστήματος ανίχνευσης τροχιών στο στάδιο Event Filter του πειράματος ATLAS στο HL-LHC

Διπλωματική Εργασία

**Ελένη Ξωχέλλη**

**Επιβλέπων:** Χρήστος Αντωνόπουλος

Φεβρουάριος 2025



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Approved by the Examination Committee:

Supervisor **Christos Antonopoulos**
Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Tamara Vazquez Schroeder**
Tenure-track researcher, Institut de Física d'Altes Energies, Barcelona

Member **Nikolaos Bellas**
Professor, Department of Electrical and Computer Engineering, University of Thessaly



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

# Acknowledgements

I am sincerely thankful to both the Trigger and ACTS teams at CERN for welcoming me as a student, sharing their knowledge, and tirelessly answering all my questions. These people and this environment have been instrumental in shaping this work. I am especially grateful to my CERN supervisor, Tamara, for her guidance and support throughout this thesis.

For all the inspiration and encouragement over the years, including during this thesis, I am truly grateful to my university and professors, especially Prof Antonopoulos. I could not have completed this thesis without his help.

Finally, to my friends—all my Greek favorites and all my CERN loves—who, despite being scattered across Europe and beyond, always find ways to stay close and supportive—I am endlessly thankful.

I hope this work properly reflects the dedication and support of everyone who believed in it.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

# DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

«Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism».

The declarant

Eleni Xochelli



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Diploma Thesis

**Enhancing the capabilities of the Event Filter tracking in the ATLAS experiment at the HL-LHC**

**Eleni Xochelli**

# Abstract

The computational needs in High-Energy Physics experiments like ATLAS, are ever-growing in an effort to adapt to new technologies and explore new physics. For the upcoming HL-LHC upgrade, the Event Filter (EF) system of ATLAS will be used to recover the properties of charged particles and make real-time decisions at a higher rate than ever before. In preparation for this upgrade, multiple development branches are working towards a testing baseline for hardware comparisons and a final decision.

This Thesis attempts to address the EF tracking challenges by facilitating the integration of open-source solutions from A Common Tracking System (ACTS) to existing trigger systems. Under the ACTS umbrella, the Traccc and detray projects provide GPU-focused alternatives for tracking.

The first half of this work details the implementation of a tracking geometry conversion from ACTS' CPU version to a detray format, that offers a GPU-friendly geometry design. This constitutes a first step towards a broader integration plan, making GPU tracking solutions more accessible to trigger. In its second part, this Thesis combines the regional tracking reconstruction of Athena with Traccc tracking algorithms. Athena software defines regions of the detector possibly containing muon particles, whereas Traccc is responsible for the tracking reconstruction. This proof-of-concept execution pipeline provides promising early results, encouraging further developments.

In total, this work contributes to GPU-EF tracking efforts to test and optimize new execution chains. Both implementations explore new capabilities for EF tracking, as part of broader efforts of the Trigger and ACTS R&D teams in preparation for HL-LHC.

**Keywords:** Event Filter, Tracking reconstruction, ACTS, Traccc, Athena software, Regional reconstruction, Detector Geometry

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Διπλωματική Εργασία

**Επέκταση των δυνατοτήτων του συστήματος ανίχνευσης τροχιών στο στάδιο Event Filter του πειράματος ATLAS στο HL-LHC**

**Ελένη Ξωχέλλη**

# Περίληψη

Οι υπολογιστικές ανάγκες σε πειράματα Φυσικής Υψηλών Ενεργειών, όπως το ATLAS, αυξάνονται συνεχώς, σε μια προσπάθεια προσαρμογής στις νέες τεχνολογίες και εξερεύνησης νέας φυσικής. Με την επερχόμενη αναβάθμιση του HL-LHC, το σύστημα Event Filter (EF) του ATLAS καλείται να ανακτήσει τις ιδιότητες των φορτισμένων σωματιδίων και να λάβει αποφάσεις σε πραγματικό χρόνο με υψηλότερο ρυθμό από ποτέ. Στο πλαίσιο της προετοιμασίας για αυτή την αναβάθμιση, πολλαπλοί κλάδοι ανάπτυξης εργάζονται για την δημιουργία μιας βάσης σύγκρισης για την αξιολόγηση διαφορετικών λύσεων σε επίπεδο υλικού (hardware) και τη λήψη των τελικών αποφάσεων για την υλοποίηση.

Η παρούσα Διπλωματική Εργασία επιχειρεί να αντιμετωπίσει τις προκλήσεις της ανίχνευσης τροχιών στο EF, διευκολύνοντας την ενσωμάτωση λύσεων ανοικτού κώδικα από το "A Common Tracking System" (ACTS) στα υπάρχοντα συστήματα Trigger. Υπό την ομπρέλα του ACTS, τα έργα λογισμικού Traccc και detray παρέχουν εναλλακτικές λύσεις ανίχνευσης τροχιών αξιοποιώντας GPU.

Το πρώτο μισό αυτής της εργασίας περιγράφει λεπτομερώς την υλοποίηση μιας μετατροπής της γεωμετρίας ανιχνευτή από την έκδοση του ACTS για CPU σε μια μορφή detray, η οποία προσφέρει ένα σχεδιασμό φιλικό προς GPU. Αυτό αποτελεί ένα πρώτο βήμα προς ένα ευρύτερο σχέδιο ενσωμάτωσης, καθιστώντας τις λύσεις εντοπισμού με χρήση GPU πιο προσιτές στο υποσύστημα Trigger. Στο δεύτερο μέρος της, η παρούσα Διπλωματική Εργασία συνδυάζει την τοπική ανίχνευση τροχιών του Athena με τους αλγριθμους εντοπισμού Traccc. Το λογισμικό Athena ορίζει τις περιοχές του ανιχνευτή που ενδεχομένως περιέχουν υποατομικά σωματίδια μιονίων, ενώ το Traccc είναι υπεύθυνο για τους αλγορίθμους ανακατασκευής των τροχιών τους. Αυτή η πειραματική υλοποίηση παρήγαγε κάποια πολλά υποσχόμενα πρώτα αποτελέσματα, ενθαρρύνοντας περαιτέρω πειραματισμό.

Συνολικά, η εργασία αυτή συνεισφέρει στις προσπάθειες ανίχνευσης συμβάντων με χρήση GPU (GPU-EF), την αξιολόγηση και βελτιστοποίηση νέων μεθόδων εκτέλεσης. Και οι δύο

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

υλοποιήσεις διερευνούν νέες δυνατοτήτες για το στάδιο του Event Filter και συστατικά στοιχεία ευρύτερων πρωτοβουλιών ανάπτυξης των ομάδων του Trigger και του ACTS R&D, στα πλαίσια τωνπροετοιμασιών για την αναβάθμιση HL-LHC.

**Λέξεις-κλειδιά:**

Event Filter, Ανακατασκευή τροχιών, ACTS, Traccc, λογισμικό Athena, Τοπική ανίχνευση τροχιών, Γεωμετρία ανίχνευσης τροχιών

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

# Table of contents





Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

# List of figures





Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

# Abbreviations

| | |
|---|---|
| ACTS | A Common Tracking Software |
| AOD | Analysis Object Data |
| ATLAS | A Toroidal LHC ApparatuS |
| CPU | Central Processing Unit |
| EF | Event Filter |
| GPU | Graphics Processing Unit |
| HL-LHC | High Luminosity - Large Hadron Collider |
| HLT | High Level Trigger |
| ITk | Inner Tracker |
| IDTPM | Inner Detector Tracking Performance Monitoring |
| ID | Inner Detector |
| LHC | Large Hadron Collider |
| ODD | Open Data Detector |
| RDO | Raw Data Object |
| RoI | Region of Interest |



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

# Chapter 1

# Introduction

## 1.1 Research Context

High-energy physics experiments, such as those conducted at the ATLAS detector, rely on robust systems to process and filter collision data. Real-time decision-making on vast amounts of data demands high computational capabilities on both the software- and hardware-side, especially as advances in detector technologies continue to increase the event rate.

The ATLAS experiment utilizes the Athena software framework and CPUs for the intense tracking reconstruction, namely to recover the properties of a charged particle from measurements caused by the interaction with sensitive detector material. As part of the open-source 'A Common Tracking Software' (ACTS) initiative, various R&D projects are exploring GPU-friendly tracking solutions. This work aims to enhance the Event Filter's capabilities by incorporating ACTS' GPU solutions to meet trigger requirements.

The first part of this work focuses on the development of an ACTS plugin that enables direct conversion between two detector descriptions: one for CPU-based software and the other for GPU-based software. The plugin serves as part of a larger plan for integrating ACTS' GPU-friendly solutions into ATLAS software.

In the second part of this Thesis, the ACTS GPU tracking demonstrator, Traccc, is used within the Athena reconstruction process. This work focuses on a proof-of-concept exploration of Regions of Interest (RoIs), where detector regions are filtered as RoIs, and Traccc is tested for regional tracking.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

## 1.2 Motivation

The upcoming High-Luminosity Large Hadron Collider (HL-LHC) upgrade [1] will increase the collisions' rate for the LHC, requiring detectors such as ATLAS to process and analyze a larger volume of data. The HL-LHC is scheduled to start operations around 2030, increasing the collider's luminosity, allowing the collection of more collision data, and placing pressure on existing software and hardware infrastructures. This directly affects the ATLAS trigger system, necessitating more efficient and modern solutions that will prevent it from becoming overwhelmed.

Tracking is one of the biggest computational challenges for the trigger system, as it is a crucial stage of Event Filtering. Tracking algorithms are inherently complex and resource-intensive in this real-time environment. Since the data must be processed in parallel for the tracks to be reconstructed, one way to enhance the computational capabilities of the system is by utilizing Graphical Processing Units (GPUs). GPUs are exceptionally well-suited for this type of parallel computation by handling vast amounts of data divided into smaller tasks. Studying the feasibility of GPU usage in the trigger system has been an ongoing effort, driven by multiple projects within ATLAS.

From a software perspective in trigger, Athena [2] has been a robust solution for the system since 2001, but its underlying technologies have become increasingly outdated over the years. In 2017, a significant modernization effort was initiated, allowing the enhanced Athena-MT framework to leverage multithreading technologies. Further modernizing Athena by extending parallelism to include GPU acceleration could pave the way for embracing the changes that will be brought about by the HL-LHC, enhancing its capabilities.

As a step towards evolving trigger software, ACTS's traccc [3] initiative could allow the exploitation of GPUs in the tracking trigger. This merge would bring much-needed capabilities to trigger, while additionally supporting the general effort to align software for trigger and offline processing. Working towards this broader goal, we could enable a more maintainable and efficient software ecosystem.

## 1.3 Thesis Contribution

As shown in Figure 1.1 this work focuses on tracking geometry and tracking strategies to bring GPU software closer to Athena. In close collaboration with ACTS R&D and Trigger

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

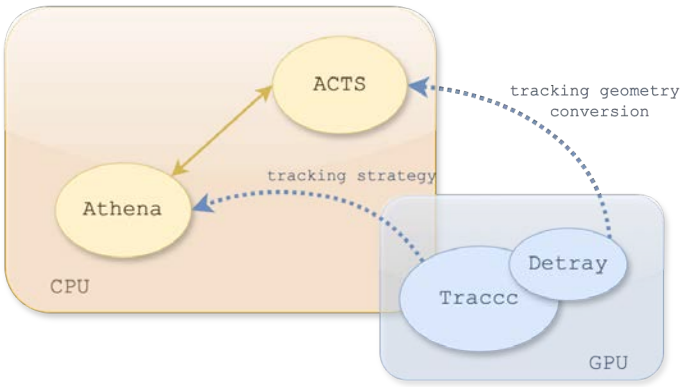


Figure 1.1: Mapping of both projects within the integration process for EF with GPUs.

teams, it aims to provide integration of existing systems and experiment with strategies that enhance Event Filter capabilities.

The creation of a detray plugin in ACTS is the first contribution of this Thesis. The plugin offers a direct translation between the tracking geometry of ACTS and detray, as a first step to a broader integration plan. The relevant merged pull requests for the two-part conversion of geometry and surface grids can be found in the ACTS Gitlab repository:

- https://github.com/acts-project/acts/pull/3299
- https://github.com/acts-project/acts/pull/3608

The second part of this Thesis focuses on implementing a proof-of-concept for regional tracking using Traccc within Athena. This experimental work provides a complete pipeline and initial results, contributing to the GPU EF tracking pipeline.

## 1.4 Thesis structure

The rest of this Thesis is organized as follows: Chapter 2 discusses the current and upcoming status of the experiment, providing context for the development environment and the motivation for software solutions that support future upgrades. In Chapter 3, the implementation of the ACTS plugin for Detray is described, starting with the relevant software packages and predecessors, leading to the new conversion method for geometry description. Chapter 4 details the process of using Traccc, an open-source GPU-focused software, for regional tracking within the ATLAS reconstruction software environment. The results of both studies are summarized in Chapter 5, where potential areas for further development and improvement are also highlighted.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

# Chapter 2

# Background

## 2.1 Status of the LHC and ATLAS experiment

### 2.1.1 The Large Hadron Collider

The Large Hadron Collider (LHC) [4] is the world's largest and most powerful particle accelerator. Its 27-kilometer circular tunnel is located 100 meters underground beneath the France-Switzerland border. Built by the European Organization for Nuclear Research (CERN), its primary goal is to explore high energy physics and address fundamental questions regarding the Standard Model (SM) of particle physics and beyond.

To facilitate the operation of the LHC, smaller-scale particle accelerators inject pre- accelerated protons or heavy ions into the system, see Figure 2.1. In the accelerator complex, machines progressively increase the particle beam's energy. At CERN, the particle beams are ultimately accelerated at the LHC to a record energy of 6.8 TeV per beam. In total, there are nine accelerators and two decelerators, with many of these accelerators also hosting experimental halls for lower-energy experiments.

Within the LHC ring, two beams of hadronic particles—either protons or lead nuclei—travel in separate pipes and in opposite directions. These beams are directed toward each other at specific collision points, where detectors record their interactions. As the counter-circulating particle beams collide, their combined energy reaches 13.6 TeV.

Processing and analyzing the vast amount of data produced from the particle collisions has been a significant challenge. At the four collision points, the detectors -ATLAS, CMS, LHCb, and ALICE- study these collisions from different perspectives. Each detector operates



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

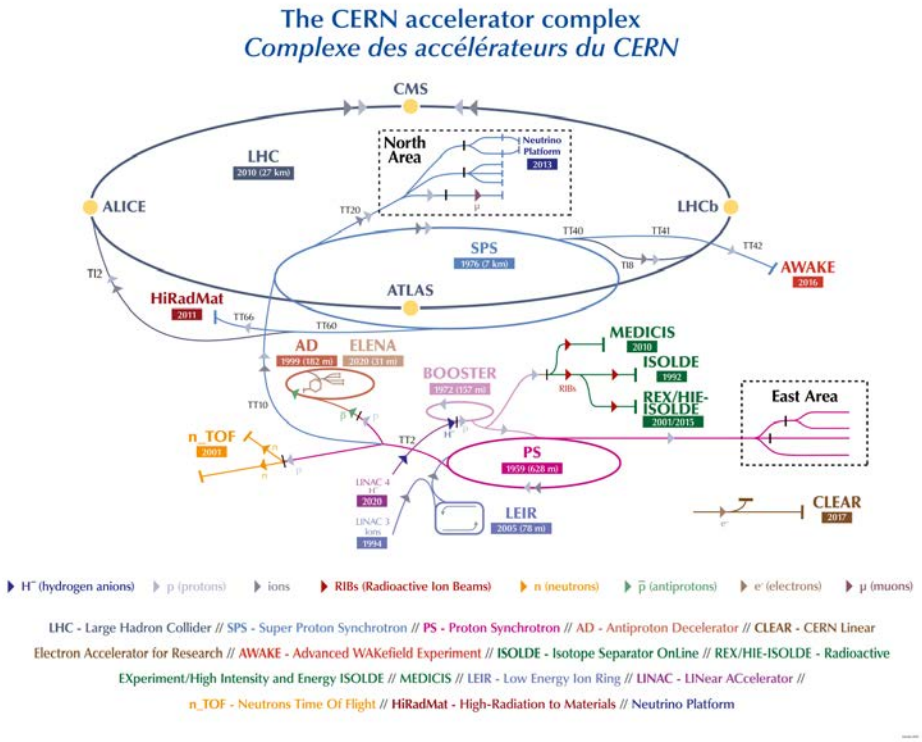


Figure 2.1: The CERN accelerator complex, layout in 2022 [5].

independently, using unique subdetectors and software. Since beginning operations in 2008, the LHC has contributed to the validation of numerous predictions of the Standard Model and led to the discovery of the Higgs boson in 2012 [6].

### 2.1.2 The ATLAS experiment

At the LHC, ATLAS (A Toroidal LHC ApparatuS) [8] is one of two general-purpose experiments alongside CMS (Compact Muon Solenoid). Both investigate a broad range of physics, while each experiment developed its own sub-detectors and magnet system design.

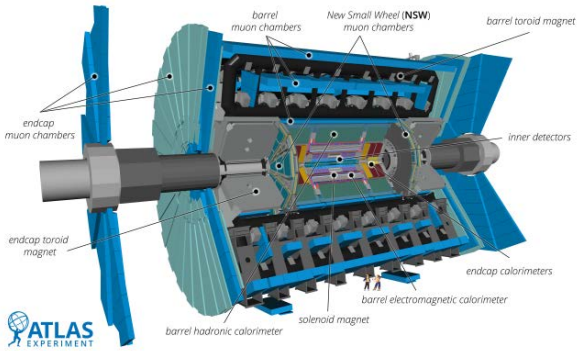


Figure 2.2: ATLAS experiment structure from Run-3 [7].

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

ATLAS is the largest detector ever built for a particle collider, with cylindrical dimensions of 46 m in length and 25 m in diameter, and is located in a cavern 100 m underground. The weight of the detector is approximately 7000 tonnes.

Particle beams from the LHC are made to collide at the center of the ATLAS detector. As we see in Figure 2.2, with respect to the interaction point, the detector is forward-backward symmetric. Its six different subsystems for particle detection are arranged in concentric layers around the collision point. In each layer of the detector outgoing particles from the collisions leave different traces, allowing for efficient particle identification and accurate measurements of energy and momentum.

**Inner detector** The innermost component of the ATLAS Inner detector [9] is the Pixel Detector, situated at a distance of 3.3 cm from the LHC beam, see Figure 2.3. The Pixel Detector consists of four compact layers of silicon pixels, comprising over 92 million pixels and 20,000 detector elements. The second layer of the inner detector is the Semiconductor Tracker (SCT), which contains 6 million "microstrips" of silicon sensors. The SCT is responsible for the detection and reconstruction of particle tracks throughout the collisions. The Transition Radiation Tracker(TRT) is situated around the previous two components and provides insights on particle types by utilizing gas-filled drift tubes to create detectable electrical signals.

In preparation for HL-LHC, the Inner Detector upgrade began during Long Shutdown 2 (2019–2022). While the SCT and TRT remain unchanged during the current LHC Run 3, the Pixel Detector was upgraded as part of the new ATLAS Inner Tracker (ITk) project. By the time HL-LHC operations start, the entire Inner Detector will be fully replaced by the new ITk.

**Calorimeter** Calorimeters are designed to absorb the majority of particles produced in collisions and measure their energy. The first level, known as the Liquid Argon(LAr) electromagnetic Calorimeter, uses multiple layers of metal to stop incoming particles and determines the energy by measuring the ionisation of liquid argon. LAr is surrounded by the Tile Hadronic Calorimeter, which consists of steel layers and plastic scintillating tiles that absorb particles and produce a measurable electric current when showered with particles, respectively.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

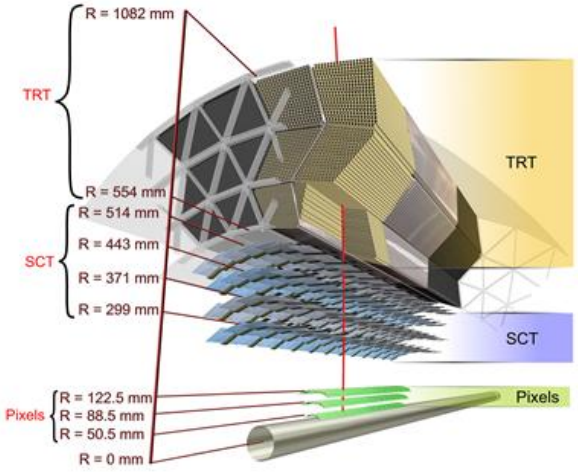


Figure 2.3: Inner Detector Layout [10].

**Muon Spectrometer** The ATLAS outer layer is made of muon gas-filled chambers that gather information on muon particles as they pass through. The chambers are positioned cylindrically around the beam axis and in the end-cap regions. Among them, the Monitored Drift Tubes (MDTs) are responsible for precision tracking at the barrel and end-cap regions, while the Cathode Strip Chambers (CSCs) provide precision tracking in the forward regions. Resistive Plate Chambers (RPCs) and Thin Gap Chambers (TGCs) are designed for fast responses and send critical information to the trigger system for event selection in real-time.

### 2.1.3 Trigger System and Data Acquisition

The trigger system [11] in the ATLAS experiment is divided into two separate stages, namely Level 1 (L1) and High Level Tigger (HLT), which are responsible for early data selection and filtering of collision events, as shown in Figure 2.4.

The L1 System is a hardware trigger located on the detector. Using custom electronics, L1 triggers on reduced granularity information from either the calorimeters (L1Calo) or the muon detectors (L1Muon). The L1 Topo system makes topological selections using information from the L1Calo and L1Muon reconstructed objects. The trigger decision is finally made in the Central Trigger Processor (CTP), taking into account information from L1Calo, L1Muon and L1Topo. Events accepted by L1, are temporarily kept in storage buffers, filtered in less than 2.5 microseconds after the event occurs, and sent to the HLT. This next level also receives

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

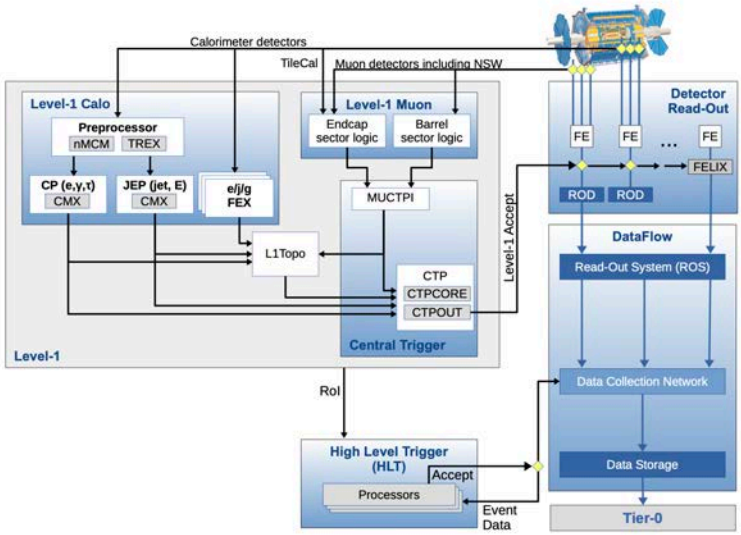


Figure 2.4: Functional diagram of the ATLAS Trigger and Data Acquisition system in Run 3 [12].

information on Regions of Interest (RoIs) -regions of the detector where candidate trigger objects have been identified per event.

The HLT is a software-based system where online algorithms reconstruct the events with much more precision and detail, using a large farm of 40.000 CPU cores. It can accept up to 100,000 events per second from L1 and manages to select about 1000 events to be stored for offline analysis.

Data Acquisition (DAQ) supports the above operations and is in charge of transporting the data from custom subdetectors electronics to offline storage after the trigger decision. It can handle an input data rate of up to several petabytes per second before triggers reduce the data volume, so advanced data compression techniques are crucial for efficient data management. At the same time, a robust monitoring system with on-detector and off-detector electronics verifies the health of its various components, minimizing the risk of data loss.

**Athena Software** Athena [11] is used online in the ATLAS HLT and is responsible for managing ATLAS production workflows, including event generation, simulation, reconstruction and derivation production. As a software framework, it is based on the Gaudi framework, a package for data processing applications for High Energy Physics. The Athena code is hosted and developed in Gitlab [13], where different lines of development can be merged and synchronized.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

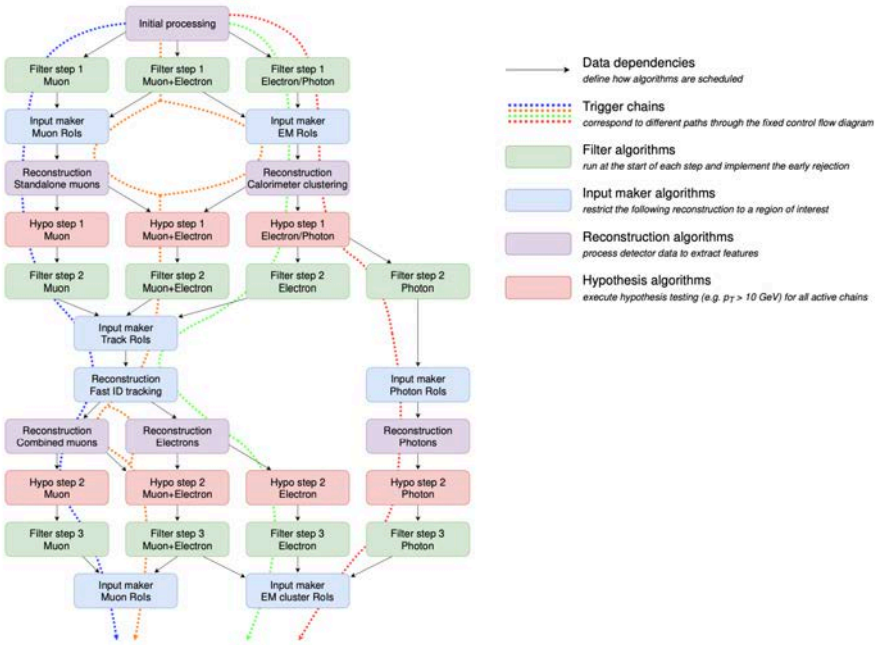


Figure 2.5: Algorithm Control Flow Logic example [11].

Athena jobs are defined using chains of algorithms, where each one processes and produces data to and from the StoreGate, Athena's transient store. The three main methods for each algorithm are:

- *initialize*: runs once after configuration, at the start of the job,
- *execute*: runs once per event,
- *finalize*: runs once at the end of the job.

Algorithms are additionally assisted by tools. Contrary to algorithms,, tools do not require an *execute* method. Therefore their specialized methods can be executed as many times as needed per event providing more freedom to the developer. In cases where the tool is defined as public (rather than private), multiple algorithms can share it. In contrast to tools, services are global, being visible to all tools and algorithms in a given job. All the above algorithms, tools and services are written in C++, while the configuration layer is written in Python. This way the developer can easily set up the framework, including choosing the algorithms and data involved in a job, and run it as a Python process.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Depending on the needs of a task and the available resources, Athena can be run in different modes:

- *Serial*: Simplest mode where the job runs as a single process on a single core.
- *Multi-process (AthenaMP) [14]*: After the *initialize* method the process is forked into multiple processes with each one independently handling a batch of events
- *Multithreaded (AthenaMT) [15]*: Fully multithreaded operation with inter- and intra-event concurrency.

The HLT configuration determines the control-flow logic for algorithm execution. This is an additional level of scheduling, along with the basic execution steering based on input and output data. As illustrated in Figure 2.5, algorithms follow a specific sequence: Filter, Input Maker, Reconstruction, Hypothesis. At the same time, multiple filters can coexist, leading to multiple execution paths. Each path then corresponds to a single trigger chain that can be enabled independently.

## 2.2 Upcoming Innovations and Upgrades

### 2.2.1 HL-LHC

In order to fully exploit the LHC physics potential, in accordance with the European Strategy for Particle Physics [16] CERN has put in place the High-Luminosity LHC (HL-LHC) project [17] [1]. The project aims to increase luminosity with a peak of $5 \times 10^{34}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$ . For this upgrade, the essential hardware installations and machine configuration will be implemented during the third long shutdown (LS3) scheduled for 2026-2030. These major changes are occurring in close collaboration with the upgrade projects in detectors of the LHC.

Existing collider parameters are being optimized by harnessing innovative technologies to efficiently and safely increase the collision rate at the HL-LHC. To enhance the concentration of bunches, more powerful magnets will be installed on either side of the ATLAS and CMS experiments. Although protons are lost during collisions, dynamic adjustments to the beam focusing system will maintain a constant collision rate. To improve the overlap between bunches and further enhance collision efficiency, 16 crab cavities will be installed near

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

the ATLAS and CMS experiments to align the bunches by tilting them sideways. With this increased intensity of the beams, the machine protection systems also need to be upgraded. This will include replacing 60 existing collimators and adding 20 new ones.

### 2.2.2 ATLAS ITk

As part of the upgrades in preparation for the HL-LHC, the ATLAS experiment will undergo several changes [18], including the replacement of the current ATLAS Inner Detector with a new system. The new ATLAS Inner Tracker (ITk) detector is designed to maintain, or even improve, tracking performance under more challenging operational conditions with increased pile-up events, collision rate and radiation damage. The ITk layout comprises a silicon Pixel Detector in the inner part, with layers close to the interaction point, and a silicon microstrips outer part at larger radii, the Strip Detector. A quadrant of this layout is visible in Figure 2.6. Active elements are shown in blue for the Strip Detector and in red for the Pixel Detector.

The Strip detector [19] includes a barrel region and two identical end-cap regions. Barrel layers and end-cap disks are positioned for maximizing track hits and optimal reconstruction. Each strip module is made of a power board, a sensor, and one or two hybrid circuits. Support for this system is provided by additional local structures made of carbon-fiber composites with integrated electrical and cooling interface.

The Pixel Detector [20] is further divided into three distinctive areas. The Inner System (IS) includes two barrel layers and an end-cap ring layer within an Inner Support tube, allowing replacement when deteriorated. The Outer Barrel (OB) system has three layers of modules in a larger radii and the Outer End-cap (OE) consists of three sets of rings in each side of the OB. With the improved layout the highly granular hermetic coverage is extended from $|\eta|$ =2.5 to $|\eta|$ =4 in the Pixel System to improve pattern recognition.

### 2.2.3 Event Filter Tracking Demonstrator

The upgraded trigger system will consist of a hardware-based Level-0 (L0) trigger followed by a software-based Event Filter (EF). L0 will receive data in rate of 40 MHz and forward 1 MHz to EF, that will further reduce it to 10 kHz. To achieve this, EF will need to use sophisticated event selection algorithms with complex software infrastructures. The baseline for this upgrade is described in the Phase-II ATLAS Trigger and Data Acquisition

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

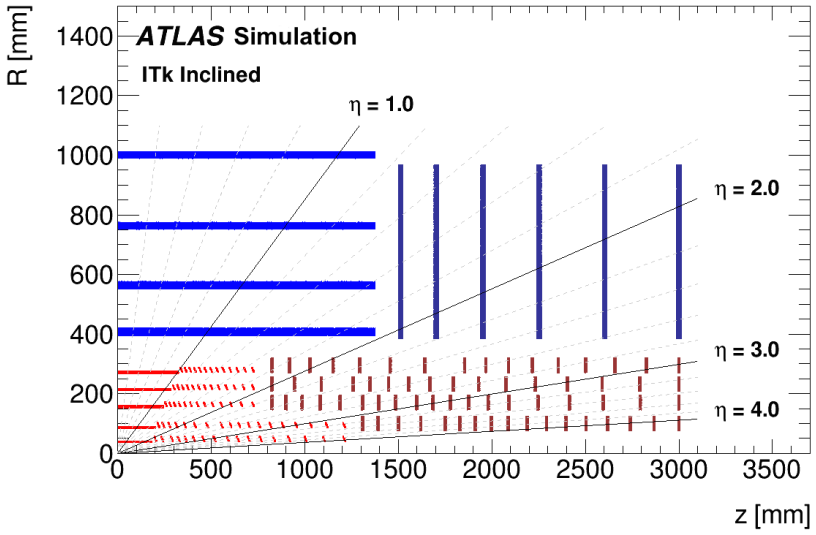


Figure 2.6: Inner Tracker Layout [19]. The horizontal axis is the axis along the beam line with zero being the interaction point. The vertical axis is the radius measured from the Interaction Point.

System (TDAQ) Technical Design Report [21]. The goal set for EF is a flexible, heterogeneous, commercial system that could contain accelerators in addition to CPUs, delivered by the end of LS3.

In order to provide a system that meets all requirements, the decision making process involves the development of a series of demonstrators of tracking using CPUs, hardware accelerators (GPUs or FPGAs), or a combination of them. The demonstrators will be compared in terms of technical feasibility, estimated tracking performance, operational procedures, opportunities for improvement, risks, and resource requirements [22]. Different steps of the ITk track reconstruction chain can be executed in either type of hardware in EF nodes or servers. For GPUs specifically, a previous demonstrator on ATLAS ID for Run-3 has proven that the software can be successfully adapted to other architectures for a full evaluation, but the cost-effectiveness of adding GPUs to the EF for Run-4 is not yet clear.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

# Chapter 3

# Detray Plugin

As part of the integration of Traccc into Athena, this chapter presents the intermediate work of converting the ACTS geometry description to a detray geometry description. We discuss the implementation of an ACTS plugin responsible for this translation, along with its predecessor, the conversion process, and an evaluation of accuracy.

## 3.1 Towards A Common Tracking Software

### 3.1.1 ACTS

A Common Tracking Software (ACTS) [23] is an open-source C++ toolkit implemented for particle track reconstruction in high-energy physics experiments. The toolkit includes algorithms on track propagation, fitting, seed finding, vertexing and track simulation. It targets modern many-core CPUs, offering parallel execution of algorithms and thread safety.

Although the ACTS toolkit is based on the track reconstruction currently used by the ATLAS experiment, it is being purposely developed as a detector-agnostic toolkit that alows customization for experiments. Although the tuning of algorithms depends heavily on detector-specific details, there are many similarities among reconstruction algorithms used in different experiments. In this way, ACTS explores a compromise, bridging abstract software design that favors easy integration with more specialized solutions for better performance results.

The project is hosted on GitHub, encouraging contributions and fostering a wide community. The creation of an effective development environment has been a primary goal enabling R&D efforts in areas such as machine learning and heterogeneous architectures. The toolkit has minimal dependencies, and its functionality is easily extendable via plugins. For



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

instance, for geometry description both manual construction and input from TGeo (ROOT Geometry Package) and Detector Description for High-Energy Physics (DD4hep)[24] software are available using the corresponding plugins. The detray plugin explained in detail in the next sub-chapters is another instance of an ACTS open-source implementation that provides interface with a different software.

### 3.1.2 ACTS R&D

ACTS provides an R&D environment, independent from the main repository, designed for the rapid development of smaller projects with distinct objectives. Apart from enhancing track reconstruction with the use of machine learning, using hardware accelerators to run common track reconstruction algorithms is one of the main objectives of ACTS R&D. As part of this ecosystem, the *algebra plugin* [25] provides an abstraction layer for matrix and vector operations essential for track reconstruction, *vecmem* [26] simplifies GPU memory management enabling an efficient common model across devices, and *covfie* [27] [28], the co-processor vector field library, supports different vector field implementations.

To further enhance efficiency, the *detray* project [29] [30] aims to offer a GPU-friendly, compile-time polymorphic library for describing the tracking geometry of a detector, by utilizing a shallow memory layout and variadic template programming. Within detray, vecmem handles memory management between host and device, eliminating duplicate code and reducing complexity.

Traccc [3] [32] [33] integrates all the aforementioned libraries into a unified framework, serving as a GPU-based track reconstruction demonstrator and the central repository for R&D efforts in GPU parallelization. This relationship is illustrated in Figure 3.1. The demonstrator includes a variety of algorithms for clusterization, seeding, track finding, track fitting, and ambiguity resolution, while supporting multiple technologies, including CPUs and GPUs via CUDA and SYCL.

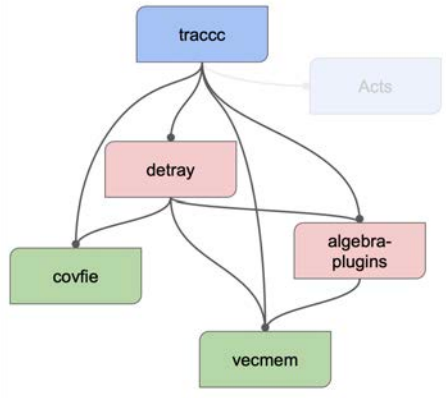


Figure 3.1: ACTS R&D Parallelization projects [31]

In the development of this new tracking software, the Open Data detector (ODD) [34] plays a crucial

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

role as a testing geometry baseline. Developing new approaches for a specific detector system and software can be challenging or overly specialized. To address this, ODD is designed as a generic, HL-LHC style tracking detector. Its geometry can be defined using DD4Hep to create a virtual yet realistic model, which can then be translated to ACTS, enabling streamlined development and testing.

## 3.2 Previous geometry conversion implementation

**Tracking Geometry** For track reconstruction in high-energy physics, complex detector components and layout must be accurately described within the software framework. The level of abstraction in the description of the tracking geometry strikes a balance between tracking precision and computational resource efficiency.

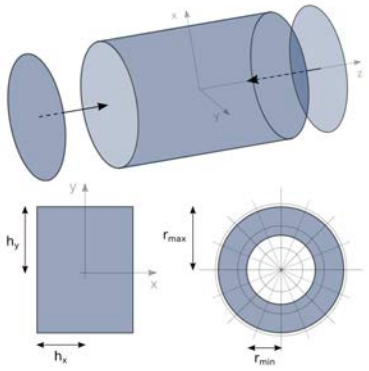


Figure 3.2: A detray cylinder volume and its boundary surfaces [29].

In the models described in this Thesis, the detector is naturally subdivided into volumes, which are defined by their boundary surfaces, as depicted in Figure 3.2. Every volume, regardless of its shape, is constructed using surfaces as the fundamental building blocks. The surfaces separating neighboring volumes are often called portals. In general, the surface and volume types and the handling of sensitive detector modules are software-dependent. Frameworks like Detray and ACTS provide extra accelerator data structures for efficient geometry navigation by implementing “neighborhood” groupings, such as surface grids. In addition, information about the material properties of detector components is considered a crucial part of geometry description.

**Previous Implementation** Since the ACTS and the Detray geometry descriptions differ, converting one detector object to the other initially required a multi-step process. In the first design of such a conversion, starting from ACTS, the detector description of an `Acts::Detector` object was processed and exported in a detray-friendly format for detray to access, read, and convert to a detray detector. The JavaScript Object Notation (JSON) file format, with the nlohmann library, was chosen for this process of exporting geometry data

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

because it is well-suited for hierarchical structures while remaining human-readable. All the geometry `nholmann::json` files are the result of a single `toJsonDetray` call. Internally, different `toJson` functions construct the JSON structure depending on which data type they process, e.g. a surface.

For the conversion to a detray detector object, ACTS exported a maximum of three files detailing different detector components: geometry, surface grids, and material. Only the geometry file was essential for detray object creation, so the other two files on material and acceleration structures were considered optional. Each type of file contained a header and a data section and was tied to a separate data path implementation, which allowed detector building and testing to be more configurable depending on the number and type of input files available at any time.

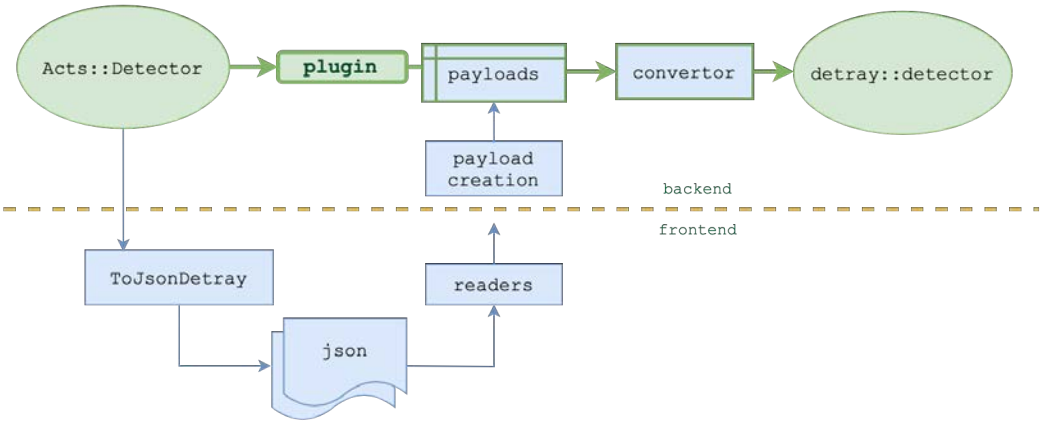


Figure 3.3: Workflow comparison. The previous implementation is noted with blue colors. In the new design in green, payloads and final conversion are still in use.

On the detray side, the front-end implementation included all the functions to convert data from IO formats to suitable data types for detector building. These data types were introduced as an intermediate data representation named *payloads*, which offered a simple way to hierarchically collect data. They also defined a boundary between the front- and back-end parts of the detray object creation. For this setup, every detector component required a data type for data representation and a corresponding pair of readers and writers for IO. The conversion to *payloads* was highly automated because the library provided `from_json` and `to_json` function overloading. When a reader function accessed the JSON file during execution, the respective `from_json` functions were automatically selected for each corresponding data type contained in the file.

After a detector builder was initialized and registered readers had performed the neces-

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

sary conversion to *payloads*, the detector building process proceeded by volume and was responsible for creating the final detector object.

**Motivation for change** The above JSON approach successfully provided a standardized format for storing and transferring geometry data, but in-memory conversion was meant to be the final design. File I/O can be replaced to allow for a better interface with no file dependencies or overhead. In addition, a direct geometry conversion resolves precision and data integrity concerns and guarantees version control and synchronization.

In contrast to ACTS, detray does not support full geometry description conversion to a detray detector from TGeo, or DD4hep. A direct conversion between the two geometries resolves this limitation by better exposing ACTS broad geometry functionalities to detray without the need for duplicate code or changes in detray software.

Overall, in-memory conversion combined with shared full geometry description sets a solid basis for testing and performance comparisons between ACTS and traccc. It also contributes to the effort to integrate traccc into ACTS and eventually ACTS with traccc into Athena. An efficient geometry interface as part of a full pipeline connecting traccc to Athena is crucial for investigating the potential benefits of traccc as a GPU-focused solution in EF tracking.

## 3.3 Designing an interface

**Compiling detray as part of ACTS** The first step in creating a direct pipeline from ACTS geometry description to detray is integrating the two systems. Multiple versions of this unification were implemented throughout the development of the project. Originally, the two were completely separated, so the first studies were carried out using detray just as a component within ACTS. The final design, though, is in unison with the broader plan to incorporate traccc into ACTS, and detray into traccc, allowing ACTS to access detray as a result of this chain.

In accordance with ACTS practices, detray was integrated into third-party configurations, and a corresponding flag was added to the CMake variables of the project. This configuration enables users to select whether detray should be included in the build process, maintaining flexibility and consistency across the system. This detray flag was replaced by a more comprehensive traccc flag at a later stage of development.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Special attention was paid to detray-related dependencies during the integration process. A series of libraries and tools were incorporated into ACTS to ensure seamless use and testing of the combined software. These components are essential for building the detray project, but they were often optional, had different versions, or were missing entirely from ACTS. Conflicts arising from these dependencies had to be carefully resolved to ensure maintainability of the integration.

**Facilitating differences in geometry description** Detray emulates only parts of the ACTS tracking detector geometry and navigation model. In both frameworks, the detector is subdivided into volumes that contain surfaces and serve as the main building blocks for geometry description. In ACTS, these surfaces are not placed directly under the volume structure but instead are organized into a layer substructure. Portals in detray are considered surface types rather than separate structures, favoring structural simplicity. This naturally affects the way iterations within the detector objects are designed.

In order to support multiple shapes, ACTS uses runtime polymorphism, while detray makes an effort to flatten the geometry tree by storing geometric data outside volume and surface classes and rather into specialized containers. Technically, the main classes for volumes and surfaces are indexing structures, and all geometric data reside in these containers. In order for memory to be flatter, less fragmented, and easier to relocate, detray needs to manage a high number of links and indices to keep references correct. A big part of converting the ACTS tracking detector into a detray detector is related to this architectural difference between the two.

## 3.4 Core Functionality

**Structural design choices** The geometry conversion functionality from ACTS to Detray is designed as an optional feature of ACTS, available when ACTS is compiled with Detray-related options enabled. Under Plugins/Detray in the ACTS project, the new conversion code structure mirrors the three-way conversion process of its predecessor. Geometry, surface grids, and material are translated in sequence with consecutive conversion function calls, where geometry remains the only component necessary for object creation out of the three. All conversion implementations are distributed across three source files and their corresponding header definitions.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

In interfacing the two software programs, the ACTS plugin retains many detray code characteristics, including the templated data types of detray. Throughout the new design, ideas for managing complex detector geometry objects are borrowed from both sides of the previous conversion process: from ACTS to an analytical JSON structure and back again to a well-defined detray object. This ensures compatibility while leveraging the strengths of the previous methodology.

The front-end implementation of readers and writers for file I/O described in the previous section is no longer relevant under the new design. As a result, the readers per data type are not chosen automatically by JSON functionalities, and new methods are made responsible for correlating each data type from ACTS to detray. The intermediate data representation of *payloads* in detray proved particularly crucial in the conversion implementation as the minimal building block in the progression from the `Acts::Detector` object to the `detray::detector` object. In this way, the file I/O became obsolete. For debugging and testing purposes only, the plugin allows exporting the geometry data to a three-file JSON format. The deployment focuses overall on providing a maintainable and versatile solution for geometry translation.

**Using existing detray functionalities** Under the previous workflow, *payload* generation was a transitional phase after the JSON reading process and before the final detray detector object creation. This structure is used in the new design in a similar way. The final detector building function that returns a detector object given the *payloads* can also be utilized "as is" by simply calling the detector builder with the newly created *payloads* as an argument.

Additionally, since the final object is of the same type as in the previous version, detector geometry consistency checks remain compatible and can be used as a verification tool for the new design.

**Geometry conversion** Geometry translation for detray begins with the `Acts::Detector` object information. While this object and its constituents (Detector Volumes, Surfaces, Portals) are harvested, *payloads* are created, filled, and stored. The main loop traverses the detector object in a depth-first fashion entering each volume the object contains and each surface or portal contained in the volume. At every step at the end of each individual access, an equivalent *payload* is returned, e.g.: a detray surface payload is returned after entering and processing an ACTS surface.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

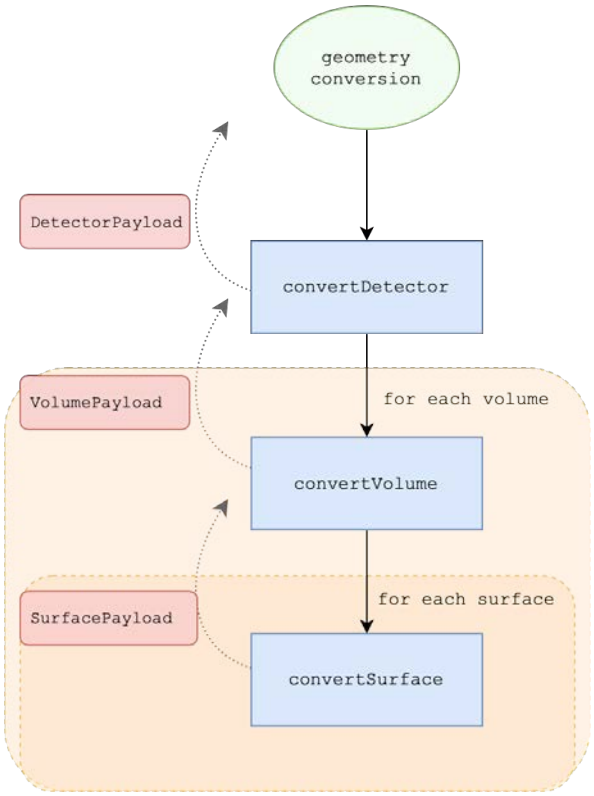


Figure 3.4: Recursive payload building for geometry. The main loop traverses the detector object in a depth-first fashion. At every function call a payload is created and returned.

As we see in Figure 3.5 the main recursive traversal of an ACTS detector objects starts from the `convertDetector` function. It iterates over every volume within the detector, calls `convertVolume` for each, and constructs a vector from the collected *volume payloads*. This vector will eventually be used by the detector builder. Within the `convertVolume` function `convertSurface` and `convertPortals` are called in a similar manner for surfaces and portals within each volume to construct a *surface payload* vector for each of them. It is worth noting that since surfaces and portals are different structurally in the ACTS framework, they are handled by separate functions in the detray plugin, but the same type of detray *surface payload* is still created by both calls. During portal and surface creation, additional function calls are responsible for creating *mask* and *transform payloads*.

The above iterative process guarantees that the given ACTS detector object is examined fully and in such a manner that *payloads* are directly placed in their correct and properly nested positions within the resulting detray object. Therefore, the final "aggregated-payload" is by construction hierarchical and can be converted into a complete detray detector object without further changes.

**Surface grids conversion** Conversion for surface grids was designed as an optional addition aiming at improving detray performance. A detector without surface grids would fall back to the slowest of all navigation policies, using a trial-and-error approach for all available surfaces. The conversion implementation takes care of translating grid axes, bins, and surface grid information into a format compatible with detray. Conversion happens recursively and hierarchically, following a similar logic to the geometry conversion explained before:

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

the functions are called in a depth-first manner to create and fill payloads, and finally feed the composite payload to the detector builder. As the reader can observe in Figure 3.5, the main function call that encompasses all the others is called `convertSurfaceGrids` and is responsible for checking all volumes within the detector object. Afterwards, a series of functions iterate over all possible grid configurations, grids, and axis while ensuring validity and taking care of any essential data processing.

The key helper functions are `convertAxis` and `convertGrid`. The first creates `detray::io::axis_payloads`, considering boundary types, binning, and bin edges and the second handles 1D and 2D grids, populates detray payloads, and adjusts indices. Furthermore, `convertImpl` determines if axes should be swapped for 2D grids, `convert` identifies the grid type and ensures validity, and `unrollConvert` uses template recursion to iterate over all grid configurations and collect all payloads.

The workflow for payload creation mimics the logic previously used by ACTS to export a JSON file and results in the same data type. A final `detector_grids_payload`, containing vectors of `grid_payload`, is now created to be used by the surface grid reader and the detector builder provided by the detray framework.

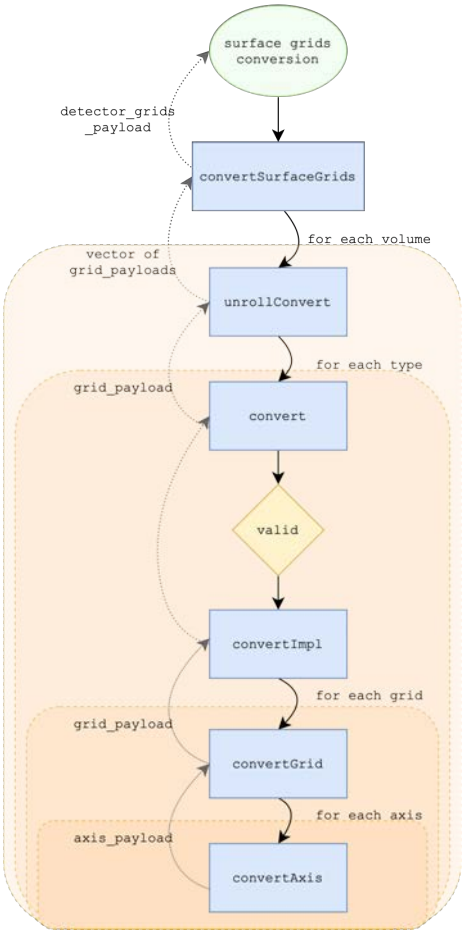


Figure 3.5: Recursive payload building for surface grids. A series of functions iterate over all possible grid configurations, grids, and axes within the Acts::Detector object

**Testing and verification** In the early stages of development, the most direct method of verification involved working with a known geometry, creating a new route to export incomplete data, and comparing it to a reference output. At the point in development when the conversion resulted in a complete object, the detector geometry could be automatically examined using Detray's consistency checks, as it was fully compatible with existing functionalities.

A versatile solution for experimenting with interfacing ACTS and Detray was provided

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

by Python bindings through Pybind. Since ACTS already offers Python run examples and supports Pybind11, a complete pipeline could easily be created to connect all available tools. Following ACTS existing Python examples, a full description of the geometry was generated using DD4Hep, loaded into ACTS, converted by the ACTS plugin, and ported to Detray. Although Pybind is no longer required for interfacing, this workflow was highly useful for testing and validation, serving as a test case that demonstrates the plugin's current functionality and intended future use.

## 3.5 Evaluation

**Conversion accuracy** The accuracy of a tracking geometry is crucial for the quality of the tracking results. Any detector representation, with all its sensitive and passive elements, has to be correct and precise to reconstruct tracks succesfully while avoiding track loss and fake tracks. Incorrect positioning or missing elements can introduce biases and inefficiencies in the reconstruction, leading to false physics results. Validating geometry accuracy across multiple software ensures that they remain consistent. This is particularly important since geometry descriptions are produced by different frameworks following separate conventions, structure, and definitions. In the case of the detray plugin, a correct and complete conversion from ACTS to detray geometry description should produce a precise and accurate geometry representation.

The plots below present results in the $\eta$ and $\varphi$, that are often used to describe particle directions in detectors like ATLAS, since detector performance and particle interactions with material depend on these variables. The $\varphi$ angle describes a particle's rotation around the beam axis, while pseudorapidity, $\eta$, describes how far a particle's trajectory is tilted relative to the beamline. The ratio bottom panel in the plots expresses the ratio among the first and second variable values and should be close to 1.00 to indicate good agreement.

In Figure 3.6, the number of hits remains consistent between Detray and ACTS across different regions of the ITk detector in the $\eta$ direction, and the ratio remains close to 1. The matching between the hits of two geometries verifies that the sensitive elements have been properly converted. A deviation in this plot would suggest navigation problems, wrong placement or missing elements, directly affecting physics performance.

The shape of the hits distribution in this diagram is characteristic of ITk geometry as

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

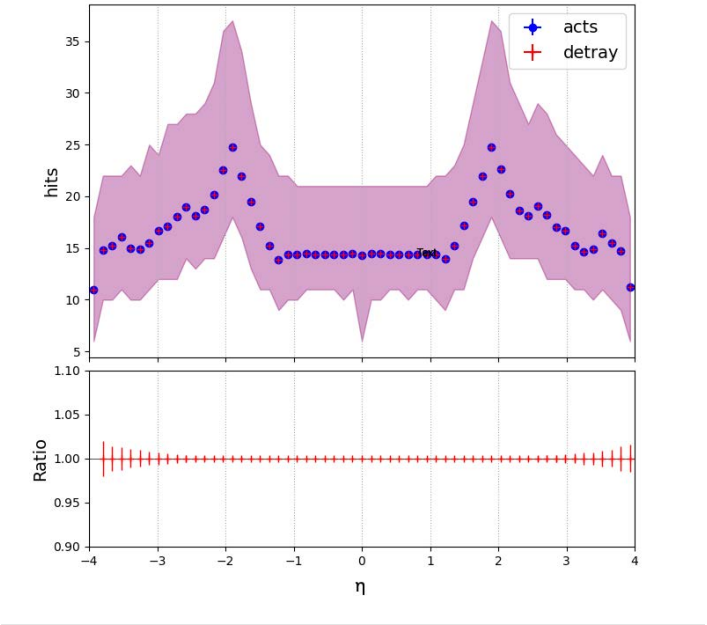


Figure 3.6: ACTS vs detray hits in eta. The number of hits remains consistent between Detray and ACTS across different regions of the ITk detector in the η direction.

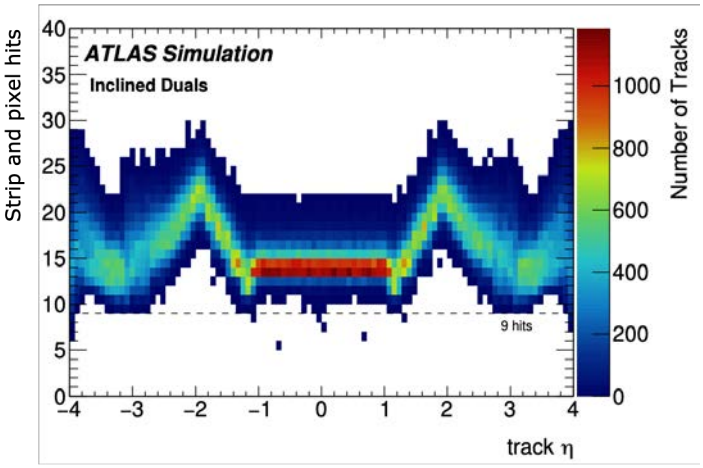


Figure 3.7: The total number of strip plus pixel measurements (hits) on track as a function of η in ITk [20].

shown in Figure 3.7. In accordance with the ITk technical design report, the number of pixel plus strip measurements on the track is above 9 at all η, except when |η| is very close to 4.

To verify that material distribution is consistent across separate geometries, t_x0 vs η plots compare the estimates of the amount of material that particles traverse as simulated by different methods. Any mismatches in this plot would be the result of a faulty conversion in material or geometry. As shown in Figure 3.8, in all three —ACTS, detray and Geant4—

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

the material properties show minimal discrepancies. Therefore, detray offers a good approximation of Geant4, maintaining accuracy in a less complex model.

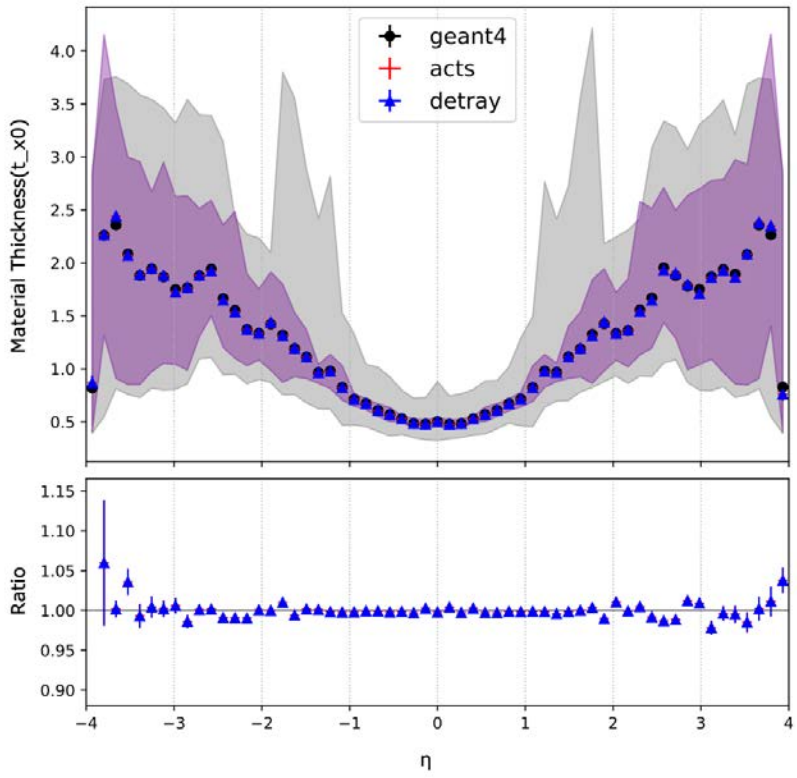


Figure 3.8: Comparing Geant4, ACTS and detray material distribution in η direction. The consistency across the different tracking geometries indicates successful translation from ACTS to detray. Geant4, as a more detailed model, exhibits higher variability in the results represented by the wider gray band. The shorter purple band describes the ACTS and detray material variability.

**Computational Performance** In the second stage of geometry integration, we can observe improvements in performance. The inclusion of surface-grid translation exposes the grouping optimization already implemented by detray and allows for performance acceleration in geometry navigation. When enabled, the surface grid conversion from ACTS to detray leads to a speed improvement of 5 times (5x) as measured per event track for ODD geometry on the CPU.

As shown in Figures 3.9 and 3.10 by the overlap of values in both η and φ, the addition of surface grids has no negative effect on the hits recorded in ITk. While surface grids introduce better computational performance, the sensitive elements are represented with equal accuracy in the optimized version of the geometry.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

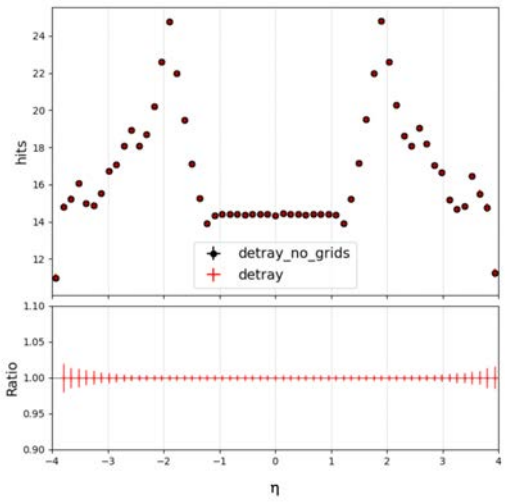


Figure 3.9: ITk detray hits with vs without surface grids in η overlap successfully

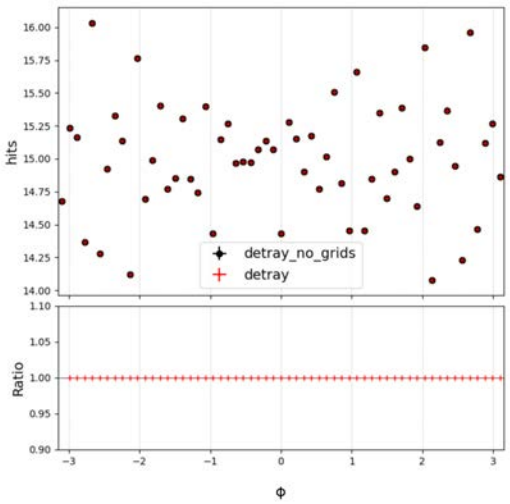


Figure 3.10: ITk detray hits with vs without surface grids in φ overlap successfully

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

# Chapter 4

# Regional Tracking using Traccc in Athena

Regional tracking in Athena and full-scan GPU tracking using Traccc software have been verified and extensively tested. This chapter focuses on their combination, presenting a proof-of-concept implementation that integrates regional tracking in trigger with R&D software solutions for GPUs, motivated by the EF demonstrator efforts. It details the setup, methodology and early results of this work.

## 4.1 Experimental Setup

**Regions of Interest (RoIs)** The HLT system is responsible for making rapid trigger decisions by analyzing a huge volume of data in real time. Track reconstruction in HLT can be performed within a Region of Interest (regional tracking) or using the full detector (full-scan). Using Regions of Interest (RoIs) enables the HLT to provide early decisions with only a subset of the total data volume.

Each RoI refers to a well-defined geometrical region within the detector and contains the signature of a physics object with which the HLT system can decide whether the event fulfills any of the pre-defined physics selection requirements. These regions are shaped as shown in Figure 4.1. The z range is sometimes extended because the z position is unknown before tracking. In these case RoIs are wedge-shaped, extending in z along the full 450 mm interaction region at the beam line and opening out in $\eta$ and $\varphi$, as shown in Figure 4.1(b).



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

The L1 trigger system defines these regions using coarse-granularity data from the muon spectrometer or calorimeters and passes this information to the HLT for reconstruction. The use of RoIs in the current ATLAS trigger system reduces the amount of data transferred for processing by the HLT to between 2% and 6% of the total data volume [36]. A similar concept can be employed at the HL-LHC. In that case, RoIs would be defined by the L0 trigger system based on the information from the calorimeters and the muon system and passed on the EF system.

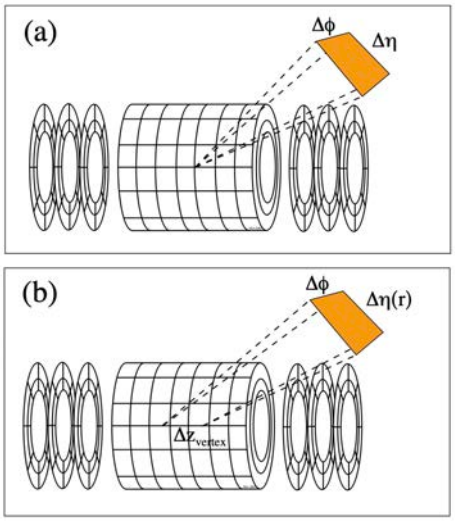


Figure 4.1: Examples of RoI definition [35], represented as a region in η and φ. The RoI in b) accounts for the uncertainty in the z position and is used for SCT and Pixels.

**Event Filter Tracking** In track reconstruction, measurements from the interaction of a charged particle with the detector components are used to recover the particle's properties. When these particles pass through the tracking detectors, they cause ionisation and leave small signals (hits) in the individual channels of the silicon-based detectors, which are then grouped into clusters. Since the particles travel through a magnetic field, their paths curve, and by analysing this curvature, the detector can determine their momentum. Finally, each resulting track contains detailed information about the particle's trajectory, including this parameters (e.g., curvature, momentum, and direction) and the associated uncertainties.

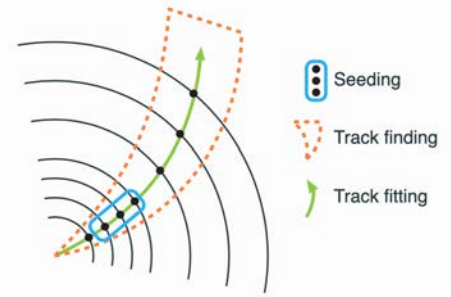


Figure 4.2: Tracking steps visualization [37].

The EF track reconstruction chain comprises the preparation of the input ITk data, followed by track finding. In preparation for tracking, ITk raw data are decoded and combined into clusters — a process called clustering — where each cluster represents particle intersections with local sensor elements. These clusters are then converted into three-dimensional space points, or SPs, through space-point formation, giving them global positions. From triplets of space points, the first track seeds are defined in a step known as track seeding. The seeds are extended by searching for adjacent clusters, forming candidate tracks, and their trajectories are refined during track finding and fitting, as shown in Figure 4.2. Finally, to

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

avoid the inclusion of fake tracks, candidates are passed through an ambiguity solver, which resolves conflicts and ensures only high-quality tracks are retained.

**Motivation for experimentation** As discussed earlier, with the upcoming HL-LHC upgrade, the tracking system must deliver exceptional performance in the ATLAS trigger while overcoming significant new challenges. The increased collision rate, expected to reach up to 200 pile-up events per bunch crossing, will make pattern recognition far more computationally intensive and susceptible to fake tracks than in current data taking. To address this, the EF system is being upgraded to improve both regional tracking and full-scan tracking.

In preparation for the EF demonstrator, multiple branches of code development are exploring heterogeneous computing solutions [22], organized into different pipelines and teams. One such pipeline, the GPU EF pipeline, is experimenting with the integration of Athena HLT software and Traccc, a GPU-optimized tracking software designed for high-performance pattern recognition and track reconstruction. Traccc offers a complete tracking reconstruction chain, from clustering to ambiguity resolution. The integration of this software within the ATLAS tracking system is currently under active investigation.

The full integration of Traccc into Athena via ACTS is under development, a part of which is the work described in Chapter 3. Independent of this developments, the EF GPU group, in collaboration with Traccc developers, has successfully established an environment where Traccc operates as part of an Athena job. For these jobs, GPU execution is enabled by defining Traccc tracking as an additional algorithm in the algorithm execution sequence, called `TrackingAlg`. In addition, a series of conversion methods were developed for geometry and tracks, to ensure seamless integration between the different systems and data types. Building on this experimental setup, the work described in this chapter focuses on using Athena-defined RoIs along with hits as input to perform regional tracking with Traccc on GPUs.

**Athena Configuration with RoIs: Experimental Setup** General purpose reconstruction is performed in an Athena environment with the `reco_tf.py` algorithm [2]. This algorithm can process either raw detector data or simulated input. Generally, the HLT outputs raw data in *Byte Stream* format, which can be converted into *Raw Data Objects* (RDO) for a structured C++ representation. For simulated events, in a standard dataflow, a program such as Geant4 provides a complete detector geometry and material description. The simulation of

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

particle interactions produces a detailed record of hits within the detector. These hits are then converted into RDOs, ensuring consistency with the real data format of the detector. After reconstruction, `reco_tf` produces *Analysis Object Data* (AOD) files.

For RoI-based studies with Traccc, a correctly configured Athena job builds upon this reconstruction process. The goal is to run an Athena job with RoI creation and additionally invoke Traccc for tracking following the job configurations and input conditions. To achieve this, some of the `reco_tf` arguments specified to define the steps of execution from RDO to AOD files are:

- `steering`: `doRDO_TRIG` and `doTRIGtoALL`, dividing the execution in two steps. The first step, from RDO to trigger, is necessary for RoI creation, and the second is where the TrackingAlg is called for tracking reconstruction.

- `input`: The RDO file used as input contains events from proton-proton collisions that produce pairs of top and anti-top quark pairs decaying to a final state containing one charged electron or muon.

- `postinclude`: `RAWtoALL:EFTracking.TrackingAlgConfig.TrackingAlgCfg:` is added for the TrackingAlg configuration file to be included in the `RAWtoALL` step.

- `geometryversion`: `default:ATLAS-P2-RUN4-03-00-00` referring to the ATLAS Detector geometry version used for the creation of the input file.

- `autoConfiguration`: "everything" for easier configuration

- `preExec:` used for flags, specifically the: `all:flags.Trigger.enableL1CaloPhase1=False` flag for turning off L1 Calo reconstruction, since it is not needed in the context of muon RoIs.

## 4.2 Methodology

In order to create a workflow for testing reconstruction, the Athena system provides a method for connecting and customizing components, including algorithms, services, and tools. While the components are written in C++, their configuration is provided in python

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

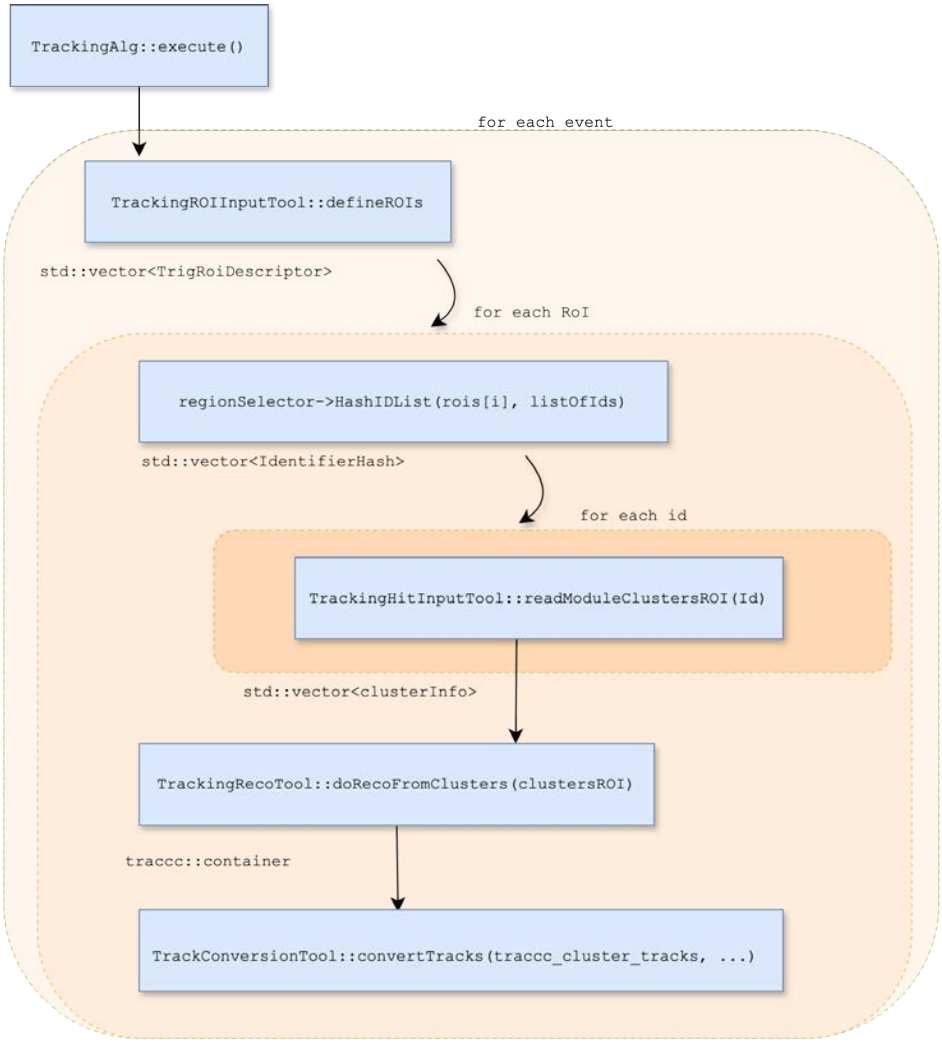


Figure 4.3: The modified `TrackingAlg` workflow for iteration over modules and Traccc scheduling per RoI.

files and stored in the `ComponentAccumulator`. Therefore, a job configuration is a result of merging multiple ComponentAccumulators.

The Traccc-Athena workflow provides an algorithm and helper tools for reading hits from an input file, loading the relevant geometry, calling the Traccc reconstruction, and converting the resulting tracks to a corresponding data format. In this chain of execution, Traccc reconstruction includes: track seeding, track finding, track fitting, and ambiguity resolution.

Based on this setup, running an Athena job adapted for regional tracking with Traccc involves introducing and modifying C++ tools and algorithms along with their respective Python configuration information.

The new design expands on the previous setup by incorporating existing Athena functionalities and new features, as well as modifying the main sequence of execution. The Athena

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

job is still built around the existing algorithm, `TrackingAlg`, but before tracking, RoIs are read and stored in an easily accessible vector. For every RoI in this vector, only the relevant hits are filtered out and passed on to a separate call for Traccc tracking.

For this implementation, a new tool is introduced called `TrackingROIInputTool`. It is responsible for assisting `TrackingAlg` in handling RoIs. This tool offers `initialise` and `finalise` methods, along with an additional method for execution.

- In `initialise`, a collection key is declared for accessing the relevant RoI collection—specifically `HLT_Roi_L2SAMuon` for muon RoIs.
- In the `defineRoIs` method the collection is accessed using a `ReadHandle` to ensure thread safety in standard Athena fashion. After verifying the validity and size of the collection, a simpler `std::vector<TrigRoiDescriptor>` is constructed and returned to the `Tracking Alg`.
- In `finalise`, the tool prints out a summary of cumulative statistics for all its calls.

To avoid checking all hits to determine whether they belong to a specific RoI, filtering is performed at the module level. Modules are building blocks of the ITk detector whose size varies from around 2 x 2 cm to 10 x 20 cm depending on their position. A module can contain anywhere from a few thousand to several hundred thousand channels. Therefore, an incoming charged particle traveling through the active area of a module typically produces many hits. Filtering at the module level is a standard practice that reduces complexity and overhead. All the modules that fall within an RoI are identified using the Athena `RegionSelector` tool [35].

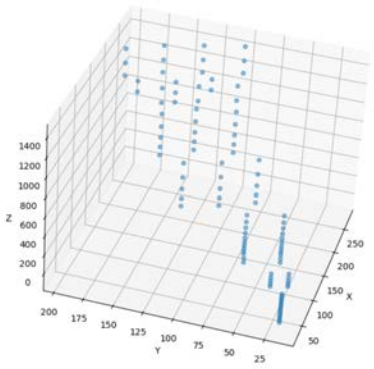


Figure 4.4: Distribution of modules found within an muon RoI, using RegionSelector in XYZ coordinates.

This tool was developed for rapid translation of physical regions within subdetectors into identifiers for grouping event data. In this way, the geometrical regions derived from L1 can be associated to Offline Identifiers and used within HLT, EF, or Offline software, whenever data needs to be accessed by region. The result of the module filtering process is visible in Figure 4.4, where modules of a specific region from a single event are visualized using their

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

XYZ coordinates. The z coordinate has a behaviour in the graph as expected for this type of RoI, exhibiting a range rather than a specific point, similar to Figure 4.1(b). For the inclusion of the `RegionSelector` tool, the Python configuration file was modified, and the relevant C++ headers were included in the main body of code.

In order to correlate the chosen modules to their hits the `TrackingHitInputTool` responsible for reading hits was enhanced with an extra input-handling method. The `TrackingHitInputTool::readModuleClustersROI` method accepts a `hashId` of a module as argument in order to read the relevant data. The reading process starts with accessing the *Inner Detector Pixel Container* and *Strip container* from the input file and afterwards the module-specific *Inner Detector Pixel Collection* and *Strip Collection* using the hash identifier. All RoI-hits within these collections can be processed, converted, and stored in a vector of types suitable for traccc tracking.

The changes described above in combination with the aforementioned `reco_tf` arguments allow for a different execution path in `TrackingAlg`. Initially, `TrackingAlg::execute()` is executed for every event, as is the case with all Athena algorithms. An iteration over every RoI and a nested iteration over each module within each RoI enable data filtering and the invocation of Traccc for regional tracking per event. Figure 4.3 shows the implementation of this process, detailing how the key functions interact and the resulting dataflow.

## 4.3 Evaluation

**Tracking Efficiency** Tracking efficiency quantifies how well the detector and algorithms identify real particle tracks. For a reconstruction algorithm, higher efficiency means that this algorithm is able to identify a larger fraction of tracks that represent actual particle interaction with the detector. The "truth" information used as a reference is produced by Monte Carlo simulation of particle interactions. The mathematical definition is the following:

$$\epsilon_{\text{tracking}} = \frac{N_{\text{reco \& truth-matched}}}{N_{\text{truth}}} \tag{4.1}$$

- $N_{\text{truth}}$ is the number of truth tracks associated with particles that passed through the detector, as predicted by simulation.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

- $N_{\text{reco \& truth-matched}}$ is the number of reconstructed tracks that are successfully matched to the corresponding truth tracks.

Tracking efficiency can be analyzed in relation to the particle's kinematic properties, such as $p_T$ and η, where The Transverse Momentum ($p_T$) describes the momentum of a particle in the plane perpendicular to the beamline.

**IDTPM** A robust method for measuring tracking performance is necessary for comparisons across different hardware options and different pipelines in EF-Tracking studies. The InDetTrackPerfMon (IDTPM) [38] package serves this purpose within the Athena framework, by producing a user-defined set of tracking performance validation histograms.

In IDTPM, each different tracking performance analysis is referred to as *TrackAnalysis* and is defined and run independently. All track analysis parameters are read from JSON configuration files so that the process is flexible and user-friendly. The mandatory parameters for a TrackAnalysis are the types of test, reference, and matching criteria. Additional properties are supported by a series of flags.

The two possible modes of TrackAnalysis execution are:

- *Trigger*: Each instance of IDTPM fills a separate monitoring histogram for each event. It creates test and reference tracks by looping over all RoIs for each trigger chain.
- *Offline*: Simple case of analysis with a single chain and full-scan, without internal loops.

The available types of matching between the truth and the reconstructed tracks are:

- *ΔR*: A reference track is considered to be matched to a test track if it falls within a cone of radius ΔR centered on the test track.
- *TruthMatch*: Each reference track matches a truth particle successfully if it can be linked to it with a high enough probability.
- *EFTruthMatch*: A test track and a reference track must both be linked to the same truth particle.

For the verification of the work in this chapter, the efficiency of the algorithm is evaluated as an offline algorithm using ΔR matching, as shown in Figure 4.1. The *TruthObjects*

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

specified are muons, since the focus of this work is on muon RoIs and the *OfflineTrkKey* flag is used to select the container of relevant tracks within the AOD output file.

Listing 4.1: Offline TrackAnalysis configuration for muons

```
"TrkAnaOfflMuon" : {
    "enabled" : true,
    "TestType"  : "Offline",
    "RefType"   : "Truth",
    "SelectTruthObject" : "highPTMuon",
    "MatchingType"  : "DeltaRMatch",
    "OfflineTrkKey" : "TracccTrackParticles"
}
```

**Analyzing the IDTPM results** Within the scope of this work, it is important to note that information on tracking efficiency is provided with respect to the complete truth information, as opposed to only the truth tracks around which a RoI is successfully constructed. The regional muon tracks are analyzed using reference of truth muon tracks from the full detector, thus complicating the evaluation. This work is also evaluated in an offline manner as a proof-of-concept for future trigger integration and developments. Using this setup, plots were produced by IDTPM, using the reco_tf output from almost 3000 events, detecting more than 1000 RoIs. All jobs were run in the *lxplus-gpu* server, using an NVIDIA T4 GPU.

As shown in Figure 4.5, the highest efficiency, with a peak of 0.6, appears in the central region $|\eta| < 1.0$ and drops for higher values of $|\eta|$. Starting from the center, close to 0, the decrease in efficiency could be related to L1 geometric coverage gaps around $|\eta| = 0$, that hold different detectors and structures. The next efficiency drop in the intermediate $|\eta|$ region corresponds to a transition region between the different components of the ITk detector, which is currently not well modeled in simulation. Therefore, the Traccc-Athena environment cannot currently reconstruct tracks in the transition region of the ITK detector for $1<|\eta|<2$. Larger values of $|\eta|$ are affected by the magnetic field, which is assumed to be uniform and doesdoes not correspond to the actual distribution of the magnetic field in the detector.

The plot in Figure 4.6 presents the tracking efficiency as a function of $p_T$ for $p_T > 10$ GeV. At low energies, muons are reconstructed with more difficulty because they often exhibit random deflections when passing through material (multiple scattering). At higher $p_T$, for

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

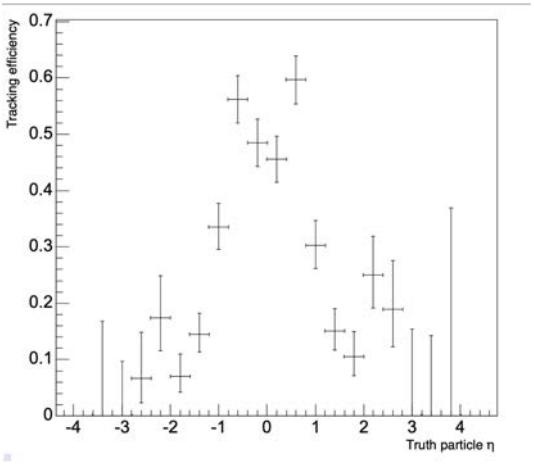


Figure 4.5: Efficiency of muon track reconstruction as a function of muon η when tracking is performed in muon RoIs.

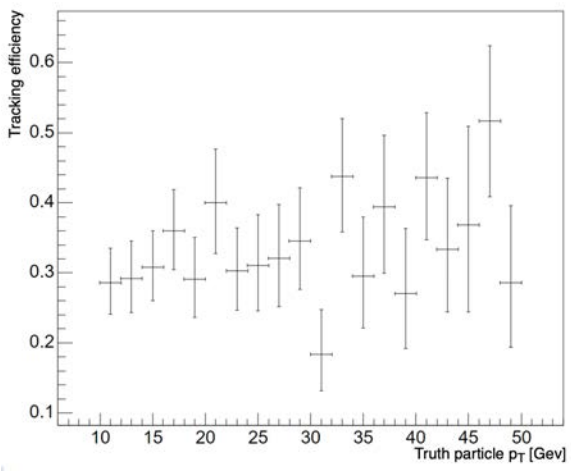


Figure 4.6: Efficiency of muon track reconstruction as a function of muon $p_T$ when tracking is performed in muon RoIs.

40-50 GeV, more energetic muons follow straighter paths, are less affected by the magnetic field, and can be reconstructed more easily.

**Taking into account RoI efficiency** With a deeper look at how efficiency is measured, it is possible to provide additional evaluation insights.

The IDTPM results described above, provide the *overall tracking efficiency*, which in the case of reconstructed tracks refers to regions, so it combines both RoI and track reconstruction steps:

$$\text{Total Efficiency} = \frac{N_{\text{truth-matched reco tracks in RoI}}}{N_{\text{truth}}} = \text{Track Reconstruction Efficiency} \times \text{RoI Efficiency}$$

where:

$$\text{Track Reconstruction Efficiency} = \frac{N_{\text{truth-matched reco tracks in RoI}}}{N_{\text{truth-matched RoI}}}$$

$$\text{RoI Efficiency} = \frac{N_{\text{truth-matched RoI}}}{N_{\text{truth}}}$$

- $N_{\text{truth-matched reco tracks in RoI}}$ is the number of reconstructed tracks in RoIs that are correctly matched to a truth muon.
- $N_{\text{truth-matched RoI}}$ is the number of RoIs that were identified around truth muons.
- $N_{\text{truth}}$ is the number of truth muons used as reference.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

The *Track Reconstruction efficiency* measures how well tracks are reconstructed from RoIs, given that an RoI was found, and *RoI efficiency* measures how well the algorithm identifies relevant RoIs based on truth muons, quantifying how many of the truth muons had an RoI identified around them.

The plots in Figures 4.5 and 4.6 show the *Total Efficiency* in η and $p_T$, affected by both RoI-finding and track reconstruction. Since RoI-finding is not a perfect process, any inefficiencies should be accounted for. Additionally, regional tracking is performed with fewer "neighboring" tracks than full tracking, which may introduce a bias in track reconstruction.

Future studies should measure individually the *Track Reconstruction Efficiency*, and the *Total Efficiency* and measure *RoI Efficiency* successfully. To accurately assess all the above, the execution should be modified to include the TrackingAlg in the trigger, with evaluation performed using Trigger Navigation within the monitoring tool.

**Final Considerations** This work is the first attempt in combining Traccc algorithms with Athena RoIs for tracking reconstruction. All figures and results conclude, as expected, that tracking performance is best when muons traverse the central region with higher $p_T$. The experimental setup offers early results with approximately 0.6 tracking efficiency in the central region. However, evaluating these types of efficiencies will present challenges until the setup is run as part of trigger software and is evaluated using Trigger Navigation and truth matching.

As a component of dynamically evolving systems, this work's biggest success is that it provides a foundation and allows for further studies to be performed iteratively, as the detector description, the magnetic field, or any additional features will evolve and improve in the upcoming months.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

# Chapter 5

# Conclusion

## 5.1 Summary

This Thesis contributes to the investigation of handling geometry and event data algorithms for heterogeneous architectures in trigger. The first part of the work has already been integrated into the ACTS framework, demonstrating its practical applicability. The second part presents a proof-of-concept for a potential pipeline enabling regional Traccc tracking within the Athena framework, paving the way for future advancements.

Both development efforts discussed in this Thesis aim to bridge the gap between Trigger and ACTS' R&D solutions for GPUs, and required heavy collaboration with both teams. The resulting software contributions are crucial components of broader efforts for system integration and testing. At the same time, these implementations are contributing to the EF GPU branch of development providing more data for an informed hardware decision on whether to use GPUs for HL-LHC EF tracking.

## 5.2 Future Work

The EF tracking demonstrator serves as the primary motivation for further research, defining the timeline and performance expectations for future improvements. EF tracking pipelines, as seen in Figure 5.1 are focused on developing complete, functional, and competitive tracking solutions. In parallel, three different types of hardware (CPU, GPU, FPGA) are being tested with classical tracking or machine learning algorithms. The current optimization stage is scheduled to last until the first quarter of 2026, enabling further development and



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

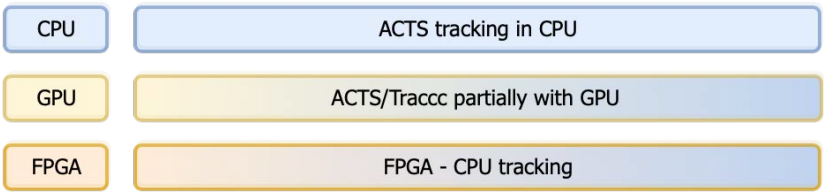


Figure 5.1: The three main branches of development currently for EF demonstrator.

testing before the final decision and later deployment.

The detray plugin described in this Thesis is already in use and constitutes a critical component of the EF GPU pipeline, aiding geometry translation for the potential use of Traccc. Independently of EF, the plugin is the first step towards a coherent integration of Traccc into Athena. These efforts are beyond the scope of this Thesis and is a goal the ACTS R&D team is consistently working on.

The implementation of Traccc regional tracking within Athena combines its components by building a complete chain of execution from Athena, to Traccc, to returning efficiency information with IDTPM, as the EF guidelines suggest. This work serves as a proof-of-concept, while future developments should include running and evaluating this algorithm in the trigger software. Studies can be additionally performed for the number of RoIs per Traccc call in GPU and different types of RoIs, expanding beyond muons. These more extensive computational and efficiency studies would provide the necessary insights for a final hardware decision within this development pipeline.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

# References


[1] ATLAS Collaboration. *High-Luminosity Large Hadron Collider (HL-LHC): Technical design report*. CERN Yellow Reports: Monographs. Geneva: CERN, 2020. DOI: 10.23731/CYRM-2020-0010. URL: https://cds.cern.ch/record/2749422.

[2] ATLAS Collaboration. *Athena Software Documentation*. https://atlassoftwaredocs.web.cern.ch/. 2023. URL: https://atlassoftwaredocs.web.cern.ch/.

[3] Y. B. et al. *ACTS Project - traccc*. 2020. URL: https://github.com/acts-project/traccc.

[4] Y. Baconnier, G. Brianti, P. Lebrun, A. G. Mathewson, R. Perin, and Y. Baconnier. *LHC: the Large Hadron Collider accelerator project*. Geneva: CERN, 1993. URL: https://cds.cern.ch/record/257706.

[5] E. Lopienska. "The CERN accelerator complex, layout in 2022. Complexe des accélérateurs du CERN en janvier 2022". In: (2022). General Photo. URL: https://cds.cern.ch/record/2800984.

[6] ATLAS Collaboration. "Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC". In: *Physics Letters B* 716.1 (2012), pp. 1–29. ISSN: 0370-2693. DOI: https://doi.org/10.1016/j.physletb.2012.08.020. URL: https://www.sciencedirect.com/science/article/pii/S037026931200857X.

[7] Bianchi, Riccardo Maria and ATLAS Collaboration. "ATLAS experiment schematic or layout illustration". General Photo. 2022. URL: https://cds.cern.ch/record/2837191.




Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

[8] ATLAS Collaboration. “The ATLAS Experiment at the CERN Large Hadron Collider”. In: *JINST* 3 (2008). Also published by CERN Geneva in 2010, S08003. DOI: 10.1088/1748-0221/3/08/S08003. URL: https://cds.cern.ch/record/1129811.

[9] ATLAS Collaboration. “The ATLAS Inner Detector commissioning and calibration”. In: *The European Physical Journal C* 70.3 (Aug. 2010), pp. 787–821. ISSN: 1434-6052. DOI: 10.1140/epjc/s10052-010-1366-7. URL: http://dx.doi.org/10.1140/epjc/s10052-010-1366-7.

[10] J. Pequenao. *Computer generated image of the ATLAS inner detector*. 2008. URL: https://cds.cern.ch/record/1095926.

[11] S. Alderweireldt et al. *The Run-3 ATLAS Trigger System*. Tech. rep. Geneva: CERN, 2022. URL: https://cds.cern.ch/record/2845056.

[12] ATLAS Collaboration. *Approved Plots for DAQ*. https://twiki.cern.ch/twiki/bin/view/AtlasPublic/ApprovedPlotsDAQ. CERN TWiki. Accessed:2025.

[13] ATLAS Collaboration. *Athena: ATLAS Software Framework*. https://gitlab.cern.ch/atlas/athena. Version Latest. 2023.

[14] P. Calafiura, C. Leggett, R. Seuster, V. Tsulaia, P. Van Gemmeren, and the ATLAS Collaboration. “Running ATLAS workloads within massively parallel distributed applications using Athena Multi Process framework AthenaMP”. In: *Journal of Physics: Conference Series* 664.7 (Dec. 2015), p. 072050. DOI: 10.1088/1742-6596/664/7/072050. URL: https://dx.doi.org/10.1088/1742-6596/664/7/072050.

[15] Bielski, Rafal on behalf of the ATLAS Collaboration. “ATLAS High Level Trigger within the multi-threaded software framework AthenaMT”. In: *Journal of Physics: Conference Series* 1525.1 (Apr. 2020), p. 012031. DOI: 10.1088/1742-6596/1525/1/012031. URL: https://dx.doi.org/10.1088/1742-6596/1525/1/012031.

[16] CERN Council. *European Strategy for Particle Physics*. Adopted at a special session at ministerial level in Lisbon. 2006. URL: http://cern.ch/council/en/EuropeanStrategy/ESParticlePhysics.html.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

[17] O. Brüning and L. Rossi. *The High Luminosity Large Hadron Collider*. WORLD SCIENTIFIC, 2015. DOI: 10.1142/9581. eprint: https://www.worldscientific.com/doi/pdf/10.1142/9581. URL: https://www.worldscientific.com/doi/abs/10.1142/9581.

[18] ATLAS Collaboration. *Letter of Intent for the Phase-II Upgrade of the ATLAS Experiment*. Tech. rep. Draft version for comments. Geneva: CERN, 2012. URL: https://cds.cern.ch/record/1502664.

[19] ATLAS Collaboration. *Technical Design Report for the ATLAS Inner Tracker Strip Detector*. Tech. rep. Geneva: CERN, 2017. URL: https://cds.cern.ch/record/2257755.

[20] ATLAS Collaboration. *Technical Design Report for the ATLAS Inner Tracker Pixel Detector*. Tech. rep. 2017. DOI: 10.17181/CERN.FOZZ.ZP3Q.

[21] ATLAS Collaboration. *Technical Design Report for the Phase-II Upgrade of the ATLAS TDAQ System*. Tech. rep. Geneva: CERN, 2017. DOI: 10.17181/CERN.2LBB.4IAL. URL: https://cds.cern.ch/record/2285584.

[22] ATLAS Collaboration. *Technical Design Report for the Phase-II Upgrade of the ATLAS Trigger and Data Acquisition System - Event Filter Tracking Amendment*. Tech. rep. Geneva: CERN, 2022. DOI: 10.17181/CERN.ZK85.5TDL. URL: https://cds.cern.ch/record/2802799.

[23] X. Ai et al. "A Common Tracking Software Project". In: *Computing and Software for Big Science* 6.1 (Apr. 2022). ISSN: 2510-2044. DOI: 10.1007/s41781-021-00078-8. URL: http://dx.doi.org/10.1007/s41781-021-00078-8.

[24] M. Frank, F. Gaede, C. Grefe, and P. Mato. "DD4hep: A Detector Description Toolkit for High Energy Physics Experiments". In: *Journal of Physics: Conference Series* 513 (June 2014), p. 022010. DOI: 10.1088/1742-6596/513/2/022010.

[25] ACTS-Developers. *algebra-plugins GitHub repository*. https://github.com/acts-project/algebra-plugins. 2022.

[26] K. A. et al. *ACTS Project - vecmem*. 2020. URL: https://github.com/acts-project/vecmem.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

[27] S. N. Swatman. *Covfie: A Compositional Vector Field Library*. URL: https://github.com/acts-project/covfie.

[28] S. N. Swatman, A.-L. Varbanescu, A. Pimentel, A. Salzburger, and A. Krasznahorkay. "Systematically Exploring High-Performance Representations of Vector Fields Through Compile-Time Composition". In: *Proceedings of the 2023 ACM/SPEC International Conference on Performance Engineering*. ICPE '23. Coimbra, Portugal: Association for Computing Machinery, 2023, pp. 55–66. ISBN: 9798400700682. DOI: 10.1145/3578244.3583723.

[29] A. Salzburger, J. Niermann, B. Yeo, and A. Krasznahorkay. "Detray: a compile time polymorphic tracking geometry description". In: *Journal of Physics: Conference Series* 2438 (Feb. 2023), p. 012026. DOI: 10.1088/1742-6596/2438/1/012026.

[30] ACTS-Developers. *detray GitHub repository*. https://github.com/acts-project/detray. 2022.

[31] A. Krasznahorkay. "traccc - a close to single-source track reconstruction demonstrator for CPU and GPU". Presented on behalf of the Acts Parallelization R&D Team at CHEP 2023, Norfolk, USA, 8–12 May 2023. 2023. URL: https://indico.jlab.org/event/459/contributions/11420/.

[32] *CTD2022: traccc - GPU Track reconstruction demonstrator for HEP*. Connecting the Dots, Princeton, New Jersey (USA), 31 May 2022 - 2 Jun 2022. May 31, 2022. DOI: 10.5281/ZENODO.8119504. URL: https://bib-pubdb1.desy.de/record/596106.

[33] *ACTS GPU Track Reconstruction Demonstrator for HEP*. Connecting the Dots 2022, Princton (USA), 31 May 2022 - 2 Jun 2022. May 31, 2022. URL: https://bib-pubdb1.desy.de/record/486142.

[34] P. Gessinger-Befurt, A. Salzburger, and J. Niermann. "The Open Data Detector Tracking System". In: *Journal of Physics: Conference Series* 2438 (Feb. 2023), p. 012110. DOI: 10.1088/1742-6596/2438/1/012110.

[35] A. Mello, S. Armstrong, and S. Brandt. *Region-of-Interest Selection for ATLAS High Level Trigger and Offline Software Environments*. Tech. rep. revised version number

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

1 submitted on 2003-06-19 11:52:04. Geneva: CERN, 2003. URL: https://cds.cern.ch/record/685465.

[36] ATLAS Collaboration. “The ATLAS inner detector trigger performance in pp collisions at 13 TeV during LHC Run 2”. In: *The European Physical Journal C* 82.3 (Mar. 2022). ISSN: 1434-6052. DOI: 10.1140/epjc/s10052-021-09920-0. URL: http://dx.doi.org/10.1140/epjc/s10052-021-09920-0.

[37] ACTS Collaboration. *ACTS Documentation*. https://acts.readthedocs.io/en/. 2023. URL: https://acts.readthedocs.io/en/.

[38] ATLAS Collaboration. *InDetTrackPerfMon - Inner Detector Track Performance Monitoring*. https://gitlab.cern.ch/atlas/athena/-/tree/main/InnerDetector/InDetValidation/InDetTrackPerfMon. 2024.

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 14:50:10 EEST - 216.73.216.250