First Tracking Performance Results from the ATLAS Fast TracKer (FTK)

Benjamin Hooberman
University of Illinois at Urbana-Champaign on behalf of the ATLAS Collaboration
Overview

• **ATLAS Fast TracKer (FTK)** is a custom electronics system that performs fast tracking with Associative Memory ASICs & FPGAs, for use in trigger decisions.

• Installation and commissioning in ATLAS is underway → **targeting Run 3 physics**

• In 2018, two FTK vertical *Slices* covering $\eta$-$\phi$ *Towers* were installed and tracks were collected → **assess and optimize tracking performance** and **validate firmware**

• These slides present **first tracking performance results from FTK Slice data**, as well as **expectations for full FTK system** based on simulation.
Motivation: Triggering on Tracks

- Dense environment in proton-proton collisions due to pile-up interactions:

- Reconstructing tracks allows to determine the positions of the pile-up interactions and remove particles originating from them

- Software full-event tracking too slow for trigger
  - Level-1 (hardware) trigger accept rate ~ 100 kHz and latency constraint ~ 100 µs

- Hardware-based solution is needed for full-event tracking in the trigger → FTK
The ATLAS Detector

- O(100M) channels in ATLAS SCT, Pixel and IBL detectors
The ATLAS Detector
FTK | CDT 2018, Seattle | Karolos Potamianos | March 20, 2018

And its Inner Detector

- O(100M) channels in ATLAS SCT, Pixel and IBL detectors

- ~100M pixel and strip electronic readout channels

inputs to FTK:
- stereo + axial 1D hits
- 2D pixel hits

April 2, 2019
Connecting the Dots
The ATLAS Detector

- O(100M) channels in ATLAS SCT, Pixel and IBL detectors
- • Data from calorimeters, muon detector, and tracker is sent to TDAQ system

Connecting the Dots
Tracking in TDAQ System

- 2-tiered trigger system:
  - Level-1 (hardware): no tracking
  - HLT (software): tracking only in Regions-of-Interest (ROI)

• O(100M) channels in ATLAS SCT, Pixel and IBL detectors

TRT

- Tracker
  - Calo
  - Muon
  - Tracker
  - ~100 kHz
  - FTK
  - Storage
  - 1 kHz (for offline analysis)

~40 MHz
The Fast TracKer (FTK)

- FTK uses AM chips and FPGAs to reconstruct tracks with $p_T > 1$ GeV over full detector
Tracking is performed in two stages:
- 1\textsuperscript{st} stage considers 8 tracker layers
- 2\textsuperscript{nd} stage extends 8-layer tracks to all 12 layers
FTK → High-Level Trigger

- FTK tracks provided at start of HLT
- Use directly or refit quickly in HLT
• **Step 1) Parallelize:** FTK processing is performed simultaneously in 64 independent $\eta$-$\Phi$ “Towers”
FTK Track Processing Strategy

- **Step 1) Parallelize:** FTK processing is performed simultaneously in 64 independent $\eta$-$\Phi$ “Towers”
  - 2 towers were installed in 2018 → commission FTK with collisions data
• **Step 2) Track-finding**: pixel & strip hits are grouped into coarse **superstrips** and compared to ~1B pre-computed track **patterns** in AM at the same time
  - Patterns are trained using ~1B fully-simulated muons
Step 3) Track-fitting: track parameters are estimated from hit positions using a linear approximation to full helix fit

- **Fit constants** for each sector (defined by set of ~1 cm$^2$ Silicon modules, one in each layer), evaluated from ~1B fully-simulated muons
The FTK Hardware

strip and pixel data
The FTK Hardware

- **Data Formatter (DF) board** clusters the strip and pixel hits from tracker
strip and pixel data

- **1st tracking stage:** AUX board uses Associative Memory chips to match clustered hits to 8-layer patterns
The FTK Hardware

- 2nd tracking stage: Second Stage Boards extend 8-layer tracks to 12-layer tracks and perform duplicate removal.
The FTK Hardware

- **FLIC board** formats tracks and sends them to **Read Out System**

April 2, 2019  Connecting the Dots

strip and pixel data
Data Sample

- Use special high pile-up $\mu = 82$ commissioning run collected October 2018
- Tower22 slice ran stably for ~2 hours and outputted tracks to ATLAS special data stream for trigger development and rate predictions
- Using **single Data Formatter** with **partial coverage** of Tower22
- Collected ~0.5M FTK tracks
- **FTK tracks with Insertable B-layer (IBL) hits were excluded**, due to a FTK module ordering problem that caused incorrect hit positions in the run (cause is understood and the fix is being implemented)

April 2, 2019
Connecting the Dots
FTK Simulation

- Implemented full functional emulation of FTK in C++ (FTKSim)
  - Used to train sectors & constants and patterns and validate firmware with bit-level comparisons to FTK tracks
  - Very slow! 600 HS06 seconds per event

- For large-scale MC sample productions, developing parameterized FastSim approach with weights and smearings
  - Track parameter resolutions extracted from FTKSim and modeled with Double-Gaussian resolution functions
  - Good modeling of core and tails of resolution function
First Commissioning Results

- 1st tracking stage is outputting tracks
- Validation of AUX firmware
• Collected sample 0.5M 12-layer FTK tracks matched to offline tracks
• Quantify the fraction of FTK tracks with a matched offline track within $\Delta R < 0.02$
• >95% of FTK tracks have a nearby matched offline track $\rightarrow$ FTK is reconstructing good 12-layer tracks
• Compare FTK / FTK full simulation / FTK refit tracks to matched offline tracks with $\Delta R<0.02$

• FTK reconstructs tracks with correct momentum and direction!

• FTK refit improves resolutions by $\sim$10-20% (full helix fit vs. linear approx. and additional hit position corrections, e.g. Lorentz angle)
Coping with Changing Conditions

- FTK performance depends on conditions that can shift during a run
  - e.g. beamspot displacements, typically limited to $\Delta x < 100 \mu m$ over run
- In Run 3, need strategy to quickly adapt to changing conditions
Step-by-Step Efficiencies

- Quantify tracking efficiencies for several steps of FTK track processing:
  - Efficiency for track to fall within a **defined sector**
    - Does not depend on beamspot position → **only one set of Sectors & Constants is needed**
  - Efficiency for track to fall within a **defined pattern**
    - Limited pattern coverage is main source of inefficiency
  - Efficiencies after **1st** and **2nd** tracking stages
    - ~1% inefficiencies from hits/holes requirements, 8-layer→12-layer extrapolation, duplicate removal
• Study dependence by generating patterns with various beamspot x positions and plotting tracking efficiency vs. actual beamspot x in simulation

• Can maintain max efficiency with a set of pattern banks with generated beamspot x positions every 0.5 mm
FTK for Long-Lived Particles (LLPs)

- LLPs are theoretically well-motivated but not yet exhaustively explored
  - Excellent physics target for Run-3, in which energy and lumi won’t increase by large factors
- FTK allows to trigger directly on displaced tracks that are characteristic signatures
- Developed specialized pattern bank with 30% of patterns dedicated to high $d_0$, high momentum tracks
- **Able to extend coverage to large $d_0$ without degrading the prompt efficiency (by <1%)**

Using 30% of the FTK patterns for long-lived particles
Summary

• Collected sample of half a million tracks with FTK Slice covering \( \eta - \phi \) region of ATLAS detector

• Presented first tracking performance results
  – FTK is producing good tracks with reasonable track parameters

• Laying the groundwork for Run-3 physics…
  – Commissioning Fast Simulation
  – Coping with changing beamspot position
  – Preparing specialized patterns for long-lived particles

• Developing Phase-II upgrade Hardware Track Trigger
  – See talk by Richard Brenner in next session…