News and progress from the ACTS workshop

Diptaparna Biswas

Universität Siegen

HEP Experimental High Energy Physics

CPPS Center for Particle Physics Siegen

STAR Sensing and Sensibility
Transcending Disciplines for a Responsible Future

TrackOpt Meeting
27th April 2026

ACTS workshop @ NIKHEF

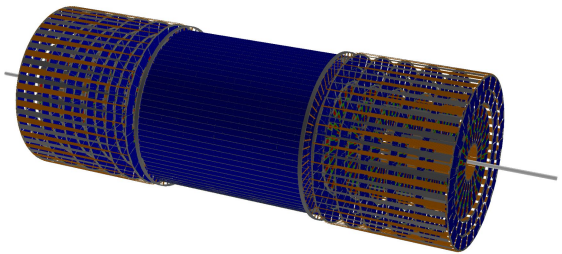
- Hackathon-style mini workshop.
  - ➢ Few talks, more time for discussion.
  - ➢ Sit with the experts, and do work.
- Activities planned beforehand:
  - ➢ Include the information of the first hit of each reconstructed track.
  - ➢ Develop a robust simulation setup for dataset generation.

ACTS: *A Common Tracking Software*

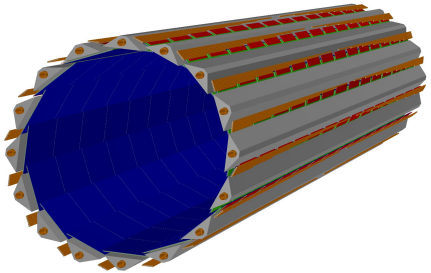
- Evolved from *ATLAS Common Tracking Software*
- Key Features:
  - ➢ Flexible tracking geometry description.
  - ➢ Simple and efficient event data model.
  - ➢ Algorithms for seed finding, track propagation, track fitting and primary vertexing.
- Designed in modern C++ (C++17/20): emphasis on parallel execution.
- Existing primary vertexing algorithms in ACTS:
  - ➢ Iterative Vertex Finder (IVF)
  - ➢ Adaptive Multi-Vertex Finder (AMVF)

ODD: Open Data Detector

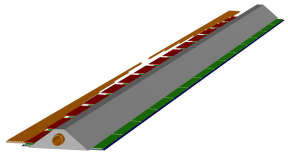
- Designed for algorithm research and development.
  - ➢ Based on the detector used in the TrackML challenge.
  - ➢ Provides a template for an HL-LHC detector, inspired by ATLAS Inner Tracker (ITk) upgrade.
- ODD serves as a crucial platform for testing and benchmarking ACTS track reconstruction algorithms. It provides a full reconstruction chain.
  - ➢ We extend the ACTS+ODD ecosystem for our secondary vertexing research for HL-LHC.



The full Open Data Detector



Innermost layer (r = 36 mm)



A single stave

News #1: The ColliderML dataset

- ODD has been extended to include calorimeters (ECL and HCL).
  - ➢ Plans to include muon system as well.
- A official simulated dataset is now available on Hugging Face.
  - ➢ Includes both inner tracker and calorimeter information.
  - ➢ Contains everything we need for our secondary vertexing research.
  - ➢ Available in ML friendly parquet format.

The dataset is organized into multiple configurations, each representing a combination of:

- ➢ Physics process (e.g., ttbar, ggf, dihiggs)
- ➢ Pileup condition (pu0 = no PU, pu200 = HL-LHC PU)
- ➢ Object type (particles, tracker_hits, calo_hits, tracks)

More info on the ColliderML dataset

- Provides standard data-ingestion pipeline.
  - ➢ Eliminates (most of) the problem we had in sharing our dataset with computer scientists.
  - ➢ Comes with a dedicated python package to manage this huge (~ 20 TB) dataset.
    - For download:

```
colliderml download --channels ttbar --pileup pu0 --objects particles,tracker_hits,calo_hits,tracks --max-events 200
```

    - For ingestion:

```
from colliderml.core import load_tables, collect_tables
cfg = {
    "dataset_id": "CERN/ColliderML-Release-1",
    "channels": "ttbar",
    "pileup": "pu0",
    "objects": ["particles", "tracker_hits", "calo_hits", "tracks"],
    "split": "train",
    "lazy": False,
    "max_events": 200,
}
tables = load_tables(cfg)
frames = collect_tables(tables)  # dict[str, pl.DataFrame]
# e.g. frames["particles"], frames["tracker_hits"] — one row per event, list columns
```

- Also compatible with the Hugging Face *datasets* package.

```
from datasets import load_dataset
ds = load_dataset("CERN/ColliderML-Release-1", "ttbar_pu0_particles", split="train", streaming=True)
```

News #2: pyACTS

- Install (part of) the ACTS package using a single command:
  - ➢ `pip install pyacts`
- Lets you write your own “algorithm” in pure python:
  - ➢ Perfect for quick prototyping, specially when the expensive “tool” is written in C++.

```cpp
ProcessCode CppTrackFinder::execute(const AlgorithmContext& ctx) const {
  const auto& spacepoints = m_spacepoints(ctx);
  ProtoTrackContainer prototracks{};
  m_prototracks(ctx, std::move(prototracks));
  return ProcessCode::SUCCESS;
```

```python
def execute(self, ctx):
    spacepoints = self.spacepoints(ctx.eventStore)
    prototracks = acts.examples.ProtoTrackContainer()
    self.prototracks(ctx, prototracks)
    return acts.examples.ProcessCode.SUCCESS
```

More info on pyACTS

- pyACTS is designed to work with the ColliderML dataset.
  - ➢ Support for reading parquet file is coming soon!
  - ➢ It can already read ROOT files using uproot.
  - ➢ Lets us build flexible data-ingestion pipeline.
    - ■ For example, perform “track-pair billoir fit” by calling appropriate “tool” on-the-fly.
- Already created my first pull request to the ACTS project to improve the uproot based ROOT file readers.

Planned work #1: Include the information of the first hit

- The hits used to reconstruct a track are stored as a linked list.
  - ➢ The head points to the last hit.
    - This is required because of the way Kalman Filter and ambiguity resolution works.
  - ➢ Needs a special step to create the backward-links.
    - The ACTS expats at the workshop immediately pointed out this technicality.
- Plans (as of now):
  - ➢ Locally modify the TrackSummaryWriter to include this info. ✅
    - This won't be merged into the upstream ACTS project.
  - ➢ Information of all the hits is already present in the ColliderML dataset.
    - Trivial to get the last hit associated with a track, as part of data-ingestion pipeline.

Planned work #2: A robust simulation setup

- Currently, we were running our ODD simulation on OMNI cluster in this way:
  - ➢ Use Geant4 and other dependencies from CVMFS.
    - ■ Broken CVMFS access was the greatest hurdle.
    - ■ Spent countless hours to fix that, but with only partial temporary success.
  - ➢ Work in a EL9 container.
    - ■ OMNI cluster runs on EL8, which is not supported by LCG stack.
    - ■ Manually compiling all the dependencies for EL8 was practically impossible.
- Suggested workaround: Migrate to the new spack based package system.
  - ➢ Absolutely no CVMFS access needed.
    - ■ The Geant4 data files (~ 5 GB) are also downloaded locally.
  - ➢ Full ACTS+ODD chain is now running on a Ubuntu based OS on my laptop. ✅
    - ■ Next step: implement the same on OMNI cluster.

Next steps

- Implement on-the-fly track-pair Billoir fit as part of the data-ingestion pipeline to use the official ClusterML dataset for our vertexing research.
  - ➢ Using pyACTS or by some other means.
- Implement spack-based ACTS+ODD full gem+sim+reco pipeline on OMNI.
  - ➢ Fully offline, no CVMFS access, preferably without using any container.
- Resume the actual ML research using the HyperGraph based approach.
  - ➢ Already got several ideas from DPG:
    - ■ CAT Finder and BAT Finder at Belle II.