Proceedings of CTD 2020
PROC-CTD2020-06
September 21, 2020

ACTS Vertexing and Deep Learning Vertex Finding

Bastian Schlag

On behalf of the ACTS developers,
CERN/JGU Mainz

ABSTRACT

The reconstruction of particle trajectories and their associated vertices is an essential task in the event reconstruction of most high energy physics experiments. In order to maintain or even improve upon the current performance of tracking and vertexing algorithms under the upcoming challenges of increasing energies and ever increasing luminosities in the future, major software upgrades are required. Based on the well-tested ATLAS tracking and vertexing software, ACTS (*A Common Tracking Software*) aims to provide a modern, experiment-independent set of track- and vertex reconstruction software, specifically designed for parallel execution. Exploiting modern software concepts, thread-safe implementations of iterative and multi-adaptive primary vertex finding algorithms, as well as a full Billoir vertex fitter, Z-Scan- and Gaussian track density seed finder, are available in ACTS and being deployed in the multi-threaded version of the ATLAS software framework AthenaMT. In addition to these computationally optimized reimplementations of classical primary vertexing algorithms, all of which have been validated against the original ATLAS implementations, ACTS provides a solid code base for evaluating new approaches to primary vertex finding, such as applications of sophisticated deep learning methods. Associating tracks to the correct vertex candidate is a crucial step in vertexing and will become even more important in the high-pileup environments expected for HL-LHC or FCC-hh in order not to merge close-by vertices. Learning a track representation in an embedding space in such a way that tracks emerging from a common vertex are close together while tracks from neighboring vertices are further separated from one another allows for the determination of a similarity score between a pair of tracks. Constructing undirected, edge-weighted graphs from these results allows the subsequent usage of classical graph algorithms or graph neural networks for clustering tracks to vertex candidates. The current status of the ACTS vertexing as well as new results on deep learning approaches to vertex finding will be presented in this talk.

PRESENTED AT

Connecting the Dots Workshop (CTD 2020)
April 20-30, 2020

1 Introduction

Future upgrades and improvements of particle accelerators, like the high-luminosity upgrade of the LHC (HL-LHC, [1]), will impose new and never before seen challenges to the respective experiments and their event processing software.
High pile-up scenarios with expected $\langle\mu\rangle \simeq 200$ for the HL-LHC or even higher values for future particle accelerators like the FCC-hh [2] will lead to extremely high track multiplicities and thus extreme conditions, especially for track- and vertex reconstruction software tools. Major software upgrades will therefore be required in order to maintain or even improve upon the current physics performance while being able to fully exploit the available computational resources. A fully functional, multi-threading capable vertexing suite and novel approaches to vertex (seed) finding within the detector-independent track- and vertex-reconstruction software ACTS will be presented in the following.

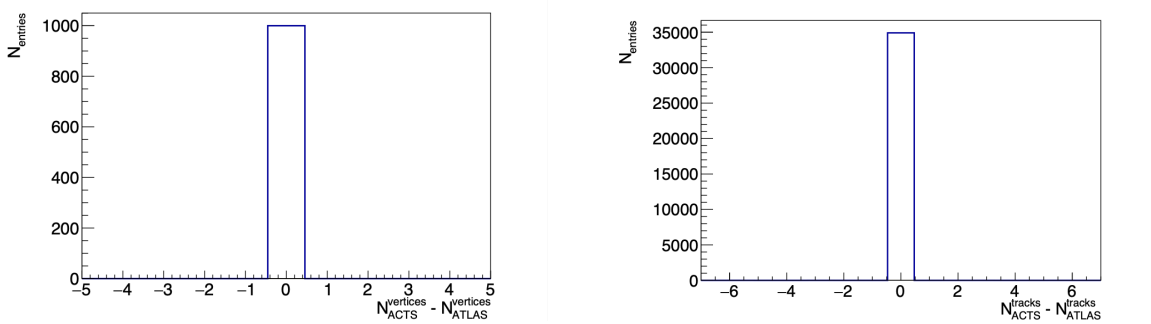
2 ACTS Vertexing

2.1 The ACTS Project

ACTS (*A Common Tracking Software*) is a detector-independent track- and vertex reconstruction toolkit, aiming to provide a modern, highly-performant and open source tracking software for particle physics experiments. It is designed for parallel code execution and is based on the ATLAS track reconstruction software [5], which has been rigorously tested over the last 15 years and always served to highest satisfaction. In order to ensure high code quality, the ACTS coding guidelines demand in addition to modern C++ and thread-safe code with const-correctness and stateless engines also the minimization of virtual inheritance and externals dependencies.
Apart from computationally optimized re-implementations of ATLAS tracking- and vertexing code, ACTS offers a solid code base for evaluating novel approaches to track- and vertex reconstruction.

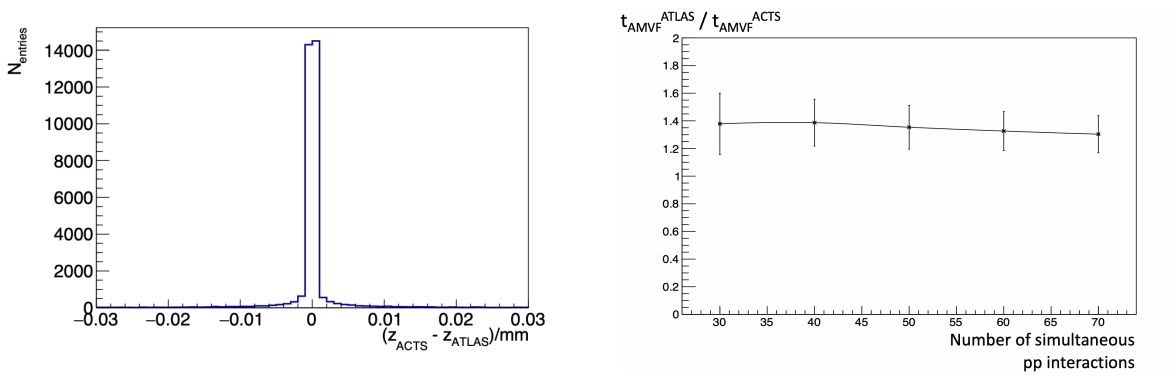
2.2 The ACTS Vertexing Suite

Several ATLAS primary vertexing algorithms have been re-implemented in ACTS and numerically validated with respect to the original ATLAS implementations. Two major vertexing approaches are available in ACTS: A so-called *fitting-after-finding* approach, manifested through the `IterativeVertexFinder` (IVF) with a *Billoir* vertex fitter [3] and a *finding-through-fitting* approach, manifested through an adaptive multi-vertex fitter deployed by the `AdaptiveMultiVertexFinder` (AMVF) [4].
The modular design of the vertexing software allows for easy exchange of various components, such as the easy usage of different vertex seed finders. Currently, the ACTS vertexing suite comprises three different vertex seed finder, the `ZScanVertexFinder`, the `GaussianTrackDensityVertexFinder` and a newly developed vertex seed finder (see Section 2.3.2), the `GaussianGridTrackDensityVertexFinder`. In addition, also tools for linearizing tracks, estimating impact parameters and deterministic annealing are available in the ACTS vertexing suite.
A relative comparison between the ACTS and ATLAS implementation in terms of number of reconstructed vertices and number of tracks associated to a vertex for the `AdaptiveMultiVertexFinder` together with the `GaussianTrackDensityVertexFinder` as an example is shown in Fig. 1. A perfect agreement on reconstructed objects on all tested events can be seen.
Fig. 2a shows the AMVF vertex z-position resolution with respect to the original ATLAS implementation where a very good agreement within micrometers is observable. Small deviations between the two algorithms are expected here as the ACTS vertexing deploys the *ACTS Propagator* with the *ACTS EigenStepper* which varies in its internal implementation from its ATLAS counterpart.
Although no algorithmic changes have been made (yet) to the ACTS vertexing algorithms, a significant speedup with respect to the ATLAS version of $t^{\mathrm{ATLAS}}_{\mathrm{AMVF}}/t^{\mathrm{ACTS}}_{\mathrm{AMVF}} \approx 1.4$ can be seen in Fig. 2b for the AMVF implementation, exclusively arising from various code improvements in the ACTS AMVF implementation.



(a) (b)

Figure 1: Relative comparisons of the ACTS Adaptive Multi Vertex Finder with respect to the original ATLAS implementation on $\langle\mu\rangle = 60$ $t\bar{t}$-data in terms of number of reconstructed vertices (a) and number of tracks associated to a vertex (b) with perfect agreement.



(a) (b)

Figure 2: Vertex z-position resolution where almost perfect agreement is met (a) and CPU performance speedup (b) of the ACTS Adaptive Multi Vertex Finder with respect to the original ATLAS AMVF implementation on $\langle\mu\rangle = 60$ $t\bar{t}$-data.

2.3 ACTS Vertexing Developments

As ACTS provides a solid test bed for developing and testing novel tracking- and vertexing methods, several studies on new vertexing techniques are currently being conducted and will be presented in the following.

2.3.1 Track Linearization using the ACTS Propagator

The track linearization tool that is currently available in ACTS as the `HelicalTrackLinearizer` is based on the ATLAS *LinearizedTrackFactory* [5] which assumes an underlying helical track model for linearizing tracks. The linearized track parameters

$$\vec{q} = \vec{q}(\vec{r}, \vec{p}) = A\vec{r} + B\vec{p} + \vec{c}_0 \tag{1}$$

are a function of the global position $\vec{r}$, momentum $\vec{p}$, a constant term $\vec{c}_0$ and the Jacobians A and B which represent the transformations from the $\vec{r}$ and $\vec{p}$ space to the track parameters space $\vec{q}$, respectively. Assuming helical tracks, A and B can be analytically derived and computed. Since this assumption is only a special case for detectors with an ideal solenoid magnetic field and thus not robust in all detector regions, a generalization of linearizing tracks using Eq.(1) is currently being investigated in ACTS, potentially harmonizing primary and secondary vertexing with common math kernels. This process will involve the generalization of the

Jacobians A and B which can be computed in all detector regions and made available to the vertex finders by the *ACTS Propagator*. Since time propagation is fully integrated in the *ACTS Propagator* and the time component t is thus available also in the Jacobians A and B, the global position $\vec{r}$ in Eq.(1) will be expanded to $\vec{r} \rightarrow \mathbf{x} = (\vec{r}, t)$ and Eq.(1) becomes

$$\vec{q} = \vec{q}(\mathbf{x}, \vec{p}) = A(t)\mathbf{x} + B(t)\vec{p} + \vec{c}_0. \tag{2}$$

Besides being able to harmonize primary and secondary vertexing, the generalized track linearization equation Eq.(2) will allow time-dependent vertex fitting which can help to resolve spatially superimposed but temporally distinct vertices.

2.3.2 Gaussian Grid Track Density Vertex Finder

The *Gaussian Track Density Seed Finder* presented in [6] models tracks as two-dimensional Gaussian distributions in the $(d_0 - z_0)$ plane in the form of

$$P(r, z) = \frac{1}{2\pi\sqrt{|\Sigma|}} e^{-\frac{1}{2}((r-d_0),(z-z_0))^T \Sigma^{-1}((r-d_0),(z-z_0))}. \tag{3}$$

The track density distributions, which are centered around the respective impact parameter points (d_0, z_0) with their shapes determined by the track covariance submatrices

$$\Sigma = \begin{pmatrix} \sigma^2(d_0) & \sigma(d_0, z_0) \\ \sigma(d_0, z_0) & \sigma^2(z_0) \end{pmatrix}, \tag{4}$$

are superimposed and the maximum along the beam axis is considered the vertex seed position. Based on the assumption that the seed position is located in the vicinity of a track, the summed track density values and their first and second derivatives are evaluated along the beam axis for every single track, allowing for subsequent Δz-steps in the direction of the density maximum if the second derivative is negative (and hence the track position close to a density maximum).

This approach has proven to give excellent physics performances [6], however, since it relies on the calculation of continuous track density values which are algorithmically hard to cache, it can become computationally very expensive, especially for high track multiplicity environments.

In order to maintain the excellent physics performance while mitigating the CPU performance issues for high pile-up events, the ACTS *Gaussian Grid Track Density Vertex Finder* models tracks as two-dimensional Gaussian density distributions on a grid, similarly centered around (d_0, z_0) with their shapes defined by Σ. Fig. 3 shows an example of three single track density distributions in the $(d_0 - z_0)$ plane grid. Since primary vertices are expected along the beam axis, and hence only the density distribution along this axis is of interest, only the respective one-dimensional track contribution vectors (red) need to be calculated. The superposition of all track density vectors along the beam axis results in the overall density beam axis vector from which the maximum bin can easily be determined and its center z-position considered the vertex position.

During the iterative processes in vertex finding, the track density contribution vectors need to be calculated and summed only once in the very first iteration and can be easily cached. In subsequent iterations, where single seed tracks have possibly been removed, the former contributions of the removed tracks can just be deducted from the overall beam axis density vector and a new maximum can be found.

This approach avoids the (possibly massive amount of) recalculation of track densities during vertex finding and thus results in a significant speedup with respect to the original continuous version [6] of this algorithm, as can be seen in Fig. 4b. Fig. 4a shows the vertex z-position resolution of this approach where very good agreement on micrometer level is observable. The resulting position resolutions and speedups are subject to density grid bin width optimization, where coarser bin sizes yield even higher speedups with lower vertex position accuracy.

In addition to the seed vertex z-position, also the seed width σ_z can be estimated exploiting the full width at half maximum (FWHM) of the track density peak according to the relation

$$\sigma_z = \frac{\text{FWHM}}{\sqrt{8\ln(2)}}. \tag{5}$$

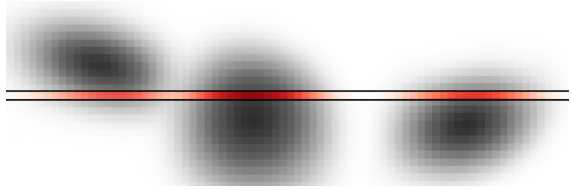
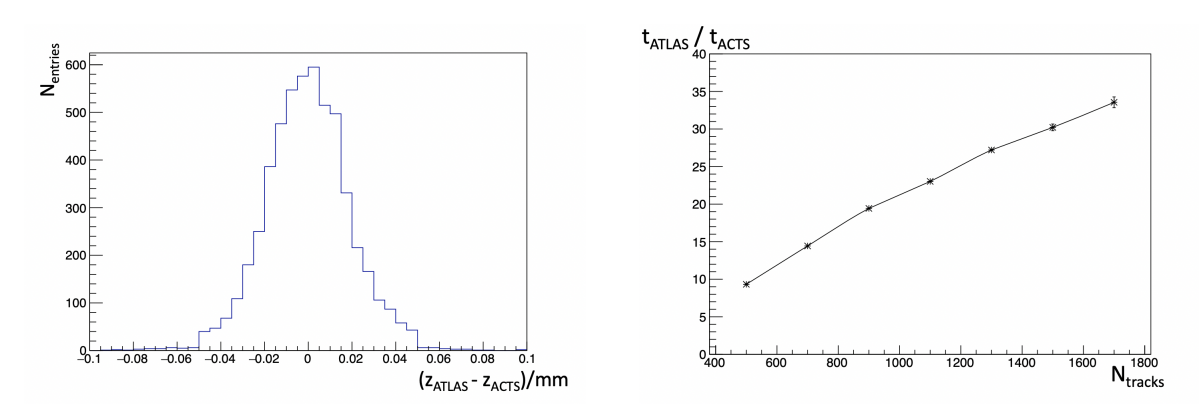


Figure 3: Example of three single Gaussian track density distributions in the $d_0 - z_0$-impact parameter grid. The two horizontal black lines indicate the beam axis vector at which the track density contributions (red) are calculated.



(a) (b)

Figure 4: Vertex z-position resolution (a) and CPU performance speedup (b) of the Gaussian Grid Track Density Vertex Finder with respect to the original continuous implementation of this algorithm [6].

2.3.3 Siamese Neural Network Vertex Finder

Vertex finding is the process of grouping reconstructed particle tracks into disjoint sets of tracks, believed to have originated from a common interaction vertex, and will become more and more challenging for high pile-up environments at future particle accelerators. A novel approach to vertex finding, based on a so-called Siamese Neural Network (SNN) [7], is currently being investigated in ACTS.

As shown in Fig. 5, the SNN utilizes a single instance of a simple feed-forward *Track embedding network* N twice with shared weights. N takes a 20-dimensional track vector (5 perigee parameters + 15 flattened covariance matrix entries) x_i for track i as input and returns an embedding representation vector h_i of this track which is consecutively fed, together with the embedding vector h_j of track j, into a distance function $d(h_i, h_j)$. d can now be used to determine the SNN output, a similarity score s that equals 1 if the tracks are *similar* (i.e. they stem from the same vertex) or 0 if they do not originate from the same vertex.

Iterating through an event of z_0-sorted tracks, the similarity scores of all combinations of track pairs within a certain z_0-window can be determined using the SNN and an adjacency matrix M can be constructed, where M_{ij} represents the score that track i and j originated from a common vertex (see Fig.6a). The resulting block structure in M can be exploited to identify each block with a set of tracks stemming from the same vertex and hence find vertices. Fig. 6b shows the resulting number of reconstructed vertices for different numbers of true vertices on truth smeared track events by making use of this simple approach of counting non-zero blocks in the adjacency matrix. Non-zero interconnections between single blocks can lead to vertex merging and thus a more elaborated strategy for reconstructing vertices from the adjacency matrix, utilizing *graphs*, will be pursued. Graphs, where nodes are identified as tracks and edges as similarity scores can easily be constructed from adjacency matrices and clustered into subgraphs (i.e. vertices) using classical as well as machine learning-based graph clustering algorithms such as Graph Neural Network.

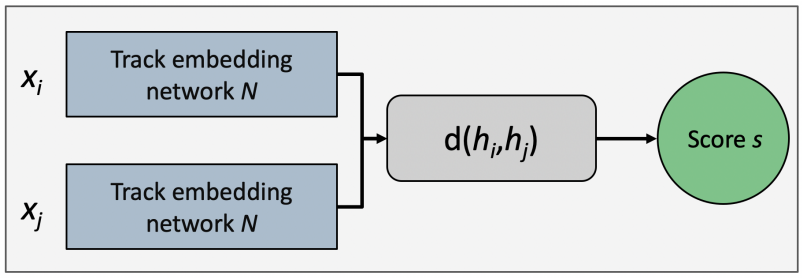
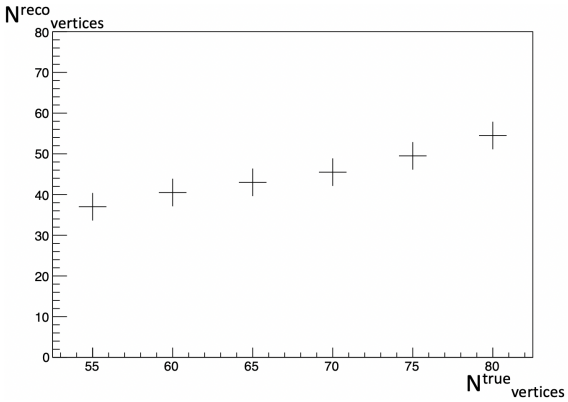


Figure 5: Siamese neural network for vertex finding which takes a pair of track vectors (x_i, x_j) as input, finds an embedding representation (h_i, h_j) that minimizes a metric d in such a way that it outputs a score s indicating if the tracks originated from the same vertex ($s = 1$) or not ($s = 0$).

$$\begin{bmatrix} 1 & 0.96 & 0.93 & 0.95 & 0 & 0 & 0 \\ 0 & 1 & 0.98 & 0.94 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.92 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0.97 & 0.92 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0.94 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

(a)



(b)

Figure 6: (a) Adjacency matrix constructed using a SNN with two visible vertex candidates with four (blue) and three (red) tracks, respectively. (b) Number of found vertices from counting non-zero blocks in the adjacency matrix on ACTS truth smeared track events.

3 Conclusions

ACTS offers a modern, MT-capable vertexing suite with various highly performant primary vertexing tools, validated against their original ATLAS implementations. Ongoing vertex reconstruction developments include both, classical developments such as a generalized track linearization and a novel extremely fast grid density vertex finder, as well as very promising machine learning approaches to vertex finding that will be extended to even more powerful graph methods in the future.

References

[1] **Apollinari, G. and Béjar Alonso, I. and Brüning, O. and Fessia, P. and Lamont, M. and Rossi, L. and Tavian, L.**, High-Luminosity Large Hadron Collider (HL-LHC): Technical Design Report (2017)

[2] **Abada, A. and others**, FCC-hh: The Hadron Collider: Future Circular Collider Conceptual Design Report Volume 3 (2019)

[3] **Billoir, Pierre and Qian, S.** Fast vertex fitting with a local parametrization of tracks (1992)

[4] **G. Piacquadio and K. Prokofiev and A. Wildauer**
Primary vertex reconstruction in the ATLAS experiment at LHC
IOP Publishing, 032033 (2008), DOI: 10.1088/1742-6596/119/3/032033.

[5] **ATLAS Collaboration**, The new ATLAS track reconstruction (NEWT)
J. Phys.: Conf. Ser., 032014 (2008), DOI: 10.1088/1742-6596/119/3/032014.

[6] **ATLAS Collaboration**, Development of ATLAS Primary Vertex Reconstruction for LHC Run 3,
ATL-PHYS-PUB-2019-015 (2019)

[7] **Gregory R. Koch**, Siamese Neural Networks for One-Shot Image Recognition (2015)