New approaches for track reconstruction in LHCb's Vertex Locator

J. Albrecht [1], B. Couturier [2], C. Hasse [1,2]

D. Bourgeois [3], V. Coco [2], N. Nolte [1,2], S. Ponce [2]

[1] TU Dortmund, [2] CERN, [3] EPFL

On behalf of the LHCb Collaboration

LHCb Upgrade 1

- The LHCb detector is a single arm forward spectrometer to study b and c hadrons
- Significant detector upgrade is planned during Long Shutdown 2 (2018-2020)
- From 2021 onwards:
  - usage of purely software based trigger system
  - full online reconstruction at 30MHz

Motivation

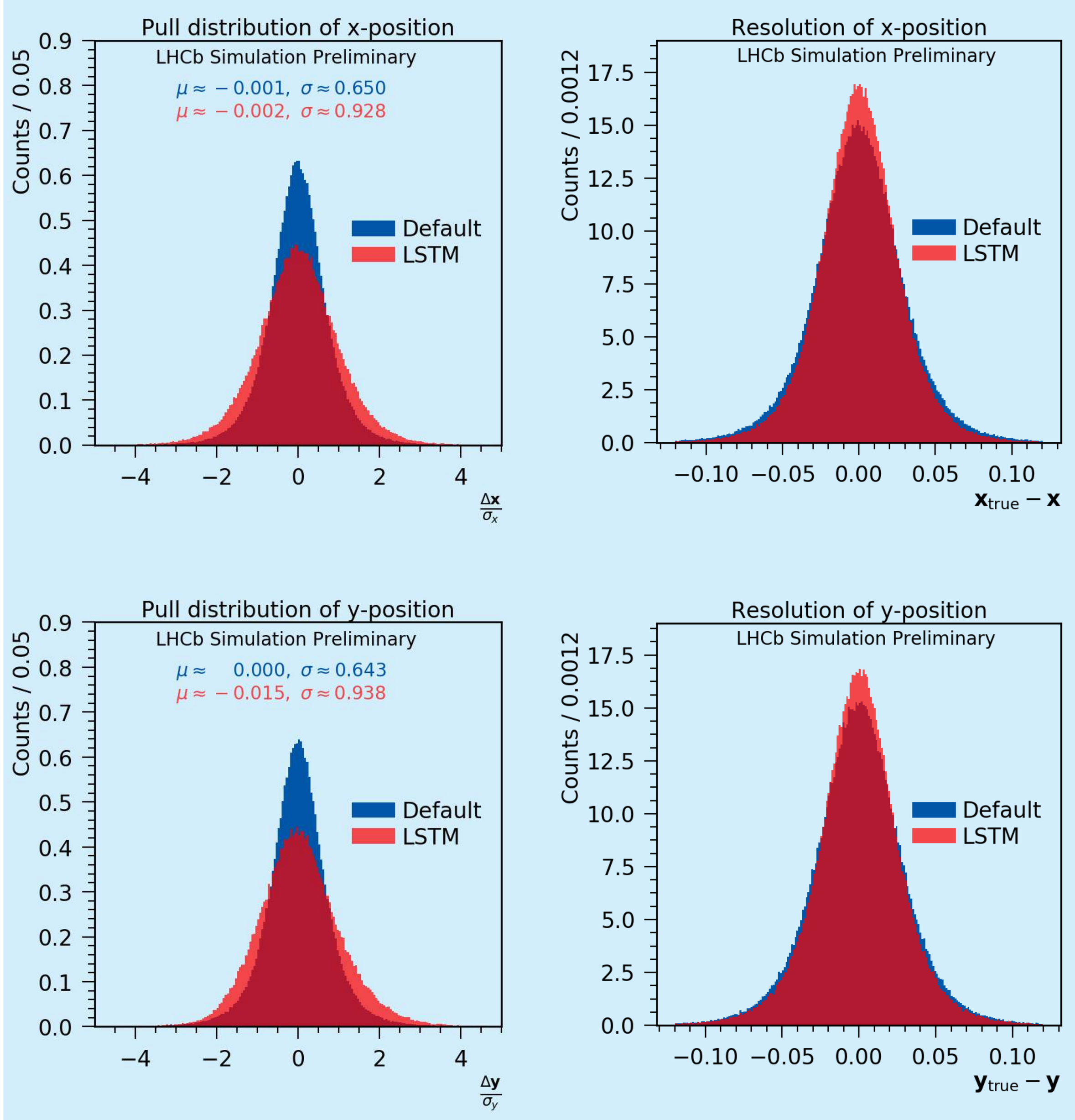
A 30MHz trigger throughput might require IP cuts on VELO tracks. A better approach would be a cut on IP and its uncertainty, which is challenging solely based on VELO information.

- IP calculation is based on CTB prediction, IP uncertainty requires known uncertainty on CTB position
- Kalman Filter (KF) needs momentum information to predict uncertainty correctly
- Due to missing momentum information, KF uses a fixed value performing well on average

Can machine learning based approaches help to improve this?

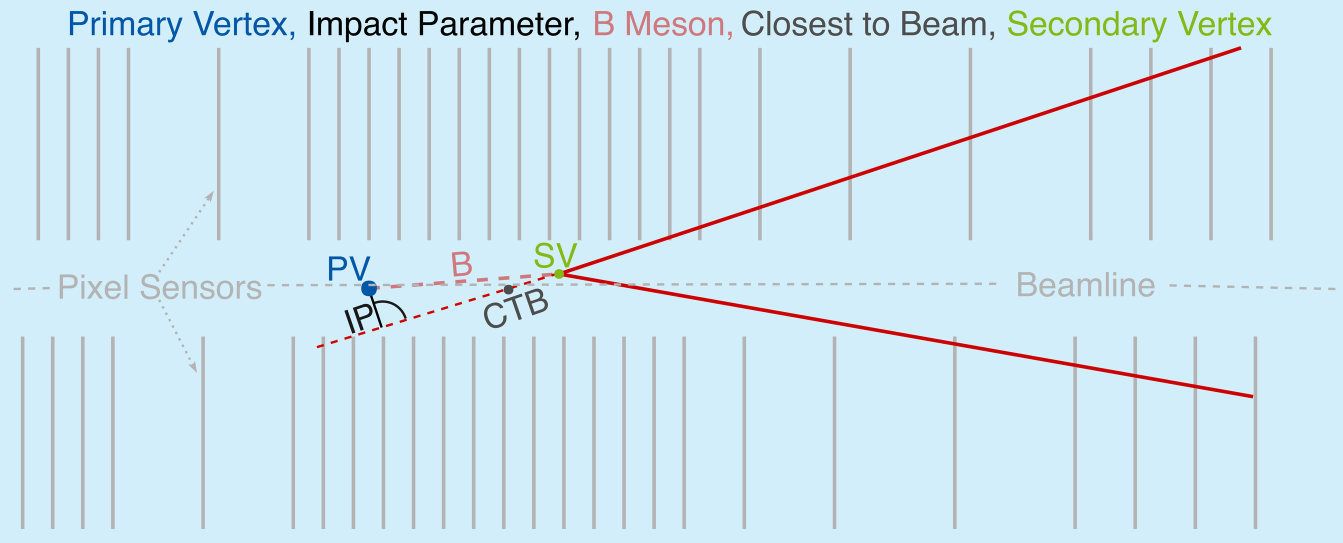
Results

Comparison of the (x,y)-resolution and (x,y)-pull distribution between the LSTM based model and the current default VELO Kalman Filter.



Track Reconstruction in the Vertex Locator

- The VErtex LOcator (VELO) is the closest subdetector to the collision point
- Only straight tracks as VELO is located outside of the magnetic field
- Reconstruct tracks via track forwarding from the outer to the inner region
- Simplified Kalman Filter to account for multiple scattering and predict a track's closest to beam (CTB) position

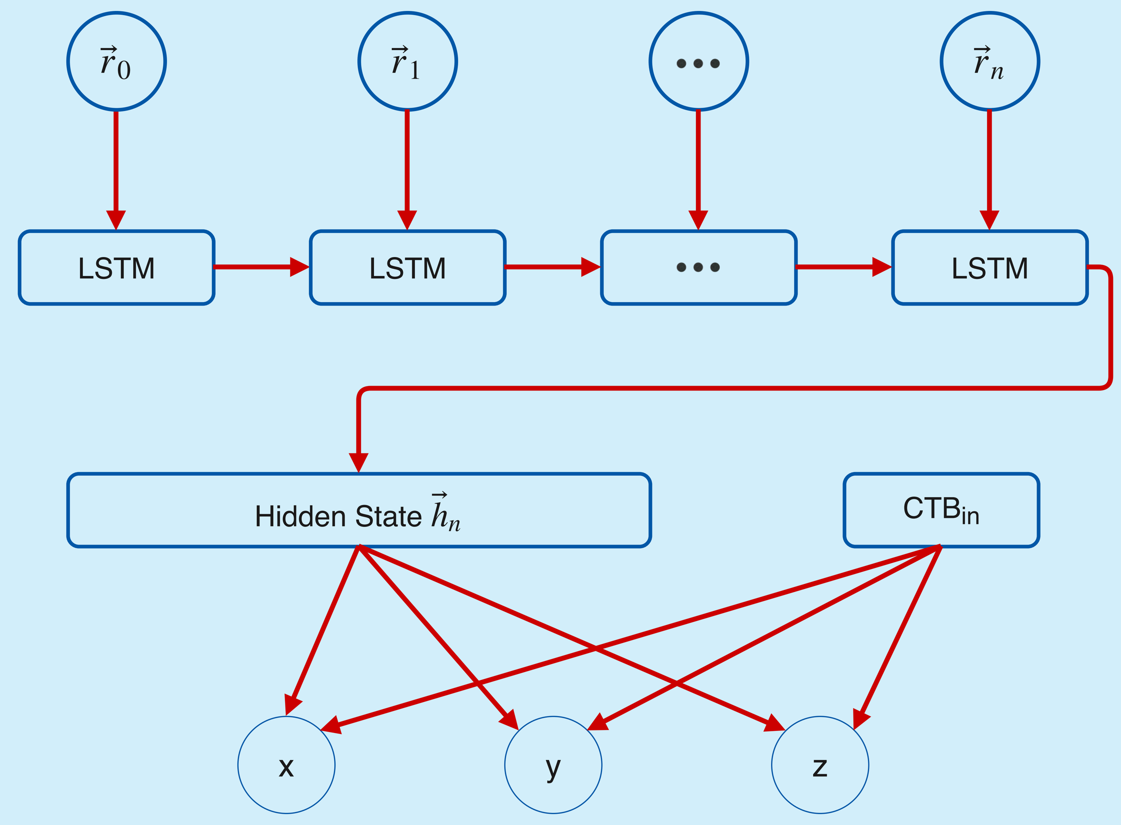


Outline of the VELO including an exemplary B decay

Idea and Method

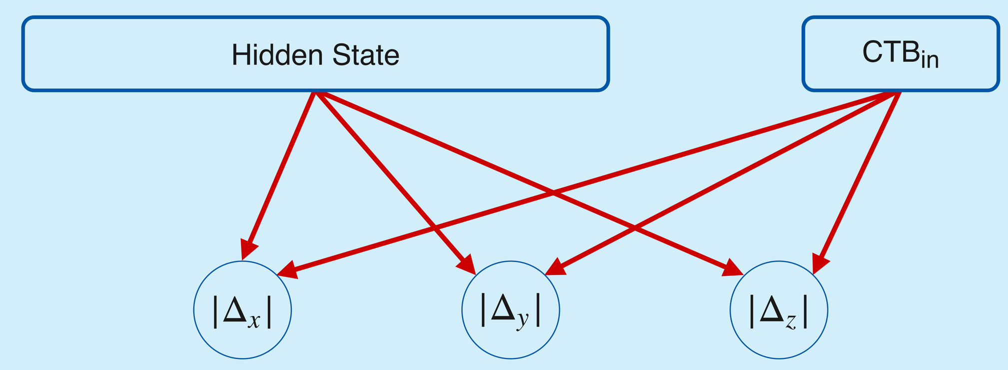
Given the straight line fit performed during the VELO track finding, compute the residuals $\vec{r}_i = (\Delta x_i, \Delta y_i, z_i)$ of each hit in respect to the straight line. Assuming the residuals are primarily due to multiple scattering, the magnitude is inversely proportional to the particle's momentum. **The goal is to design a neural network which is able to use this correlation to better predict the position and uncertainty of the CTB position.**

- We use a LSTM [1] based architecture to model the sequence like behavior of a track
- The LSTM's hidden state is kept rather small at 16 entries to ensure fast inference speeds
- For a track with n hits the LSTM processes the residuals $\vec{r}_0, \ldots, \vec{r}_n$ and outputs its last hidden state $\vec{h}_n$
- $\vec{h}_n$ concatenated with the CTB state from the straight line fit $\text{CTB}_{\text{in}} = (x, y, z, \frac{\Delta x}{\Delta z}, \frac{\Delta y}{\Delta z})$ and processed by a single $[21 \times 3]$ fully connected layer



Model architecture to predict CTB position

- Given the network's predictions and the true CTB positions the absolute deviations are given by $|\Delta_{x,y,z}| = |(x,y,z)_{true} - (x,y,z)|$
- We freeze the weights of the above network and add an additional $[21 \times 3]$ fully connected layer
- This network is trained to, on average, predict the absolute deviation $|\Delta_{x,y,z}| \approx \langle|\Delta_{x,y,z}|\rangle$
- Assuming the uncertainties are Gaussian distributed their standard deviation is $\sigma_{x,y,z} = \sqrt{\pi/2}\langle|\Delta_{x,y,z}|\rangle$



Model extension to predict the uncertainty on the CTB position

Conclusion and Outlook

We have a presented a machine learning based approach to estimate the position and uncertainty of a VELO track's closest to beam state. The resolution of this prediction as well as its ability to estimate the uncertainty is shown to be superior. While this is not a production ready solution yet, these preliminary results are promising and indicate that

a machine learning based approach might provide an alternative to the simplified Kalman Filter.

References

[1] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Comput.* 9.8 (Nov. 1997), pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.
[2] C. Patrignani et al. "Review of Particle Physics". In: *Chin. Phys.* C40.10 (2016), p. 100001. DOI: 10.1088/1674-1137/40/10/100001.
[3] Adam Paszke et al. "Automatic differentiation in PyTorch". In: *NIPS-W*. 2017.

The authors want to thank the open source comunity for providing and maintaining many helpful machine learning tools