Real-time luminosity and beam spot analysis with FPGA-reconstructed hits from the LHCb vertex detector

VERTEX 2025: 33rd International Workshop on Vertex Detectors

Speaker:

Giulio Cordova

giulio.cordova@cern.ch

Scuola Normale Superiore (SNS)
Istituto Nazionale di Fisica Nucleare (INFN)

August 28, 2025
Knoxville, Tennesse, USA

Authors:

Giulio Cordova

Daniele Passaro

Elena Graverini

Federico Lazzari

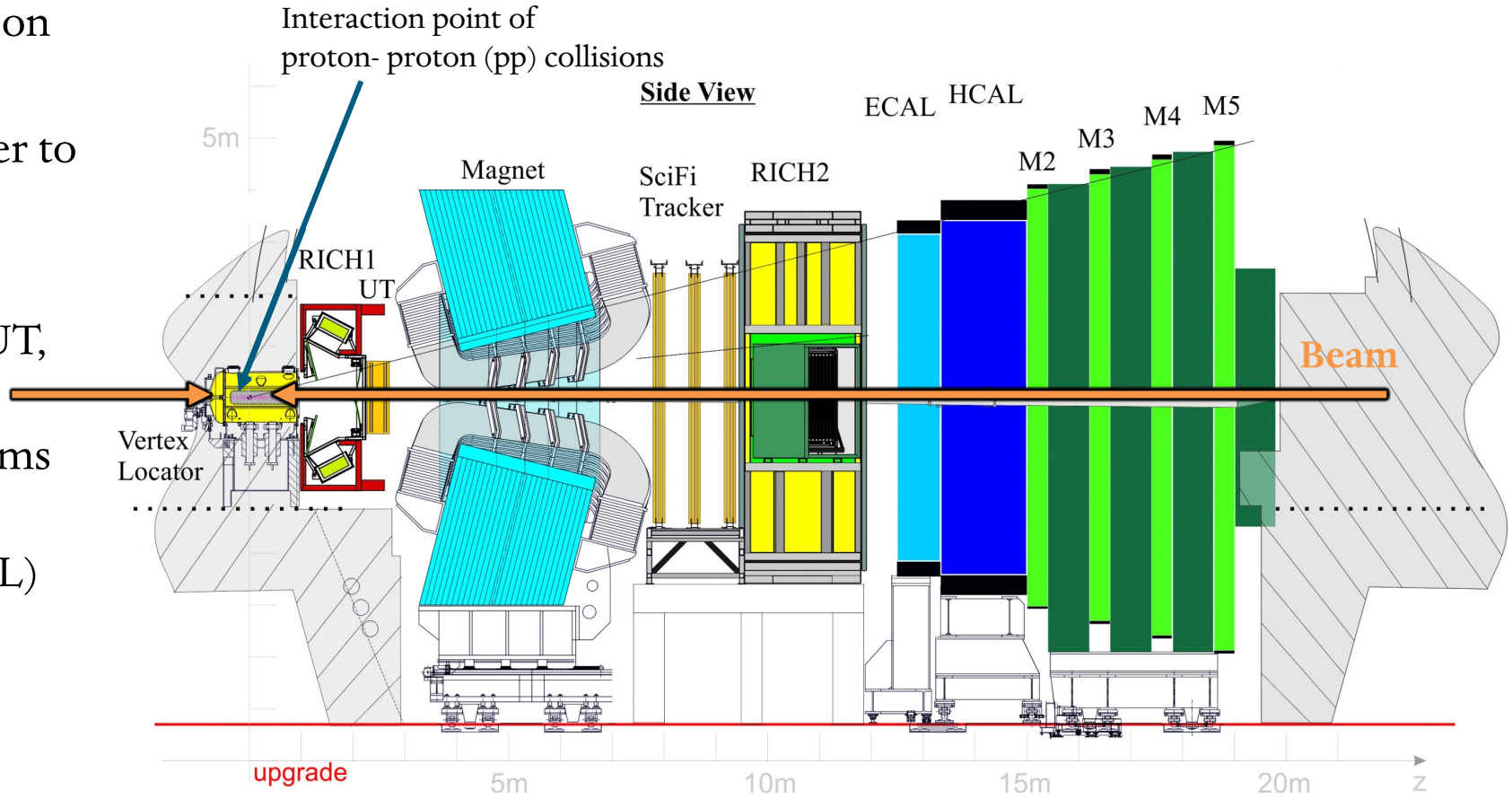
Michael J. Morello

Giovanni Punzi

Introduction

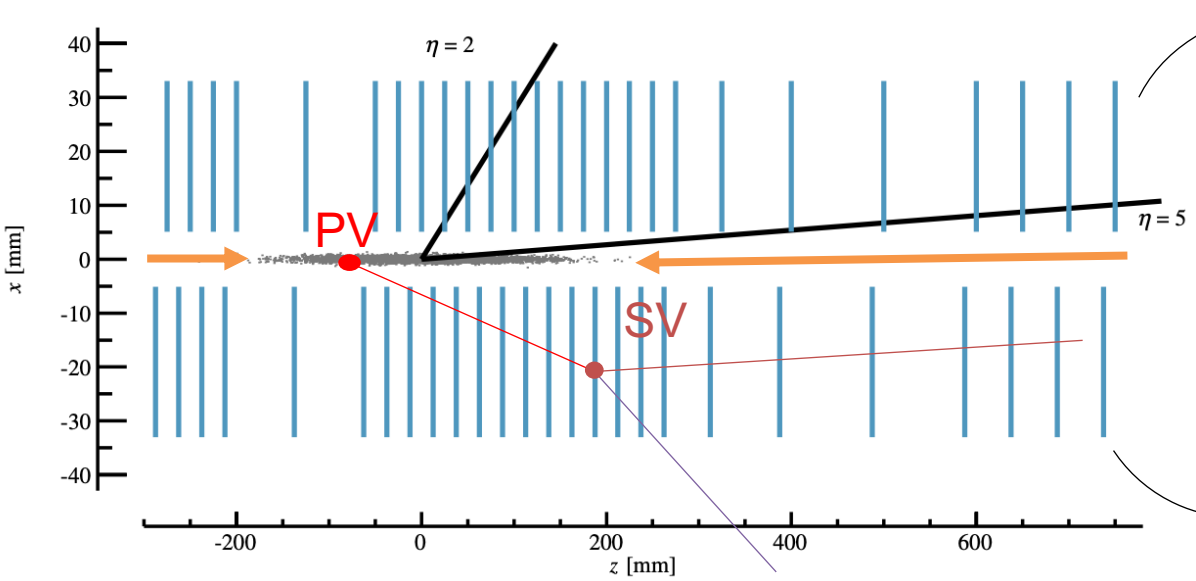
The LHCb detector

- One of the 4 main experiments on the LHC
- Single-arm forward spectrometer to study heavy flavour decays
- Multiple subdetectors:
  - Tracking stations (VELO, UT, SciFi)
  - Particle identification systems (RICH)
  - Calorimeters (ECAL, HCAL)
  - Muon Stations (M2,3,4,5)

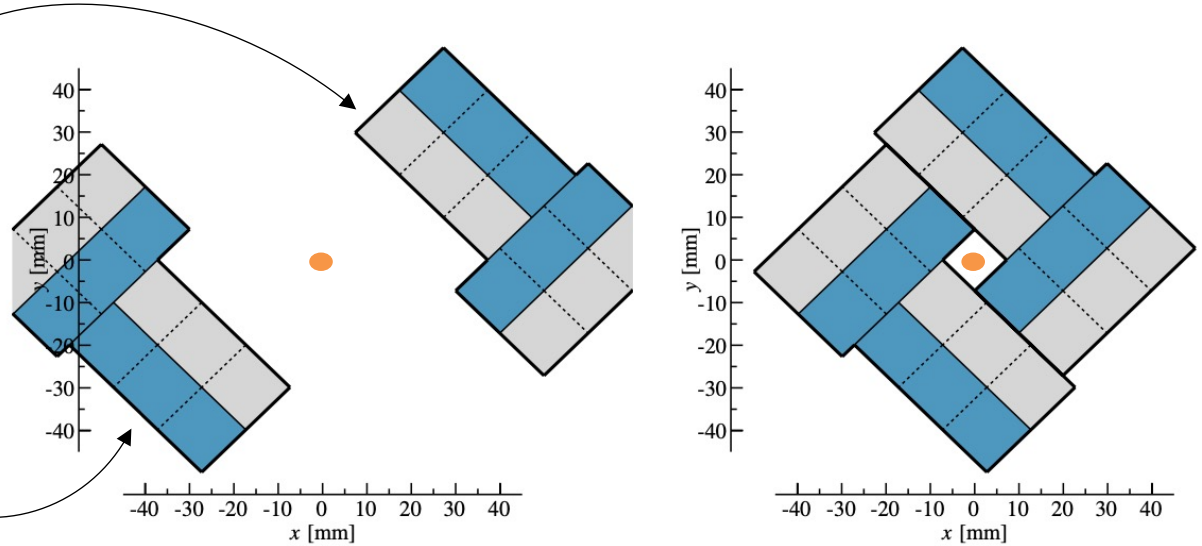


The Vertex Locator (VELO)

- Silicon pixel detector and the innermost subdetector at LHCb
- *D* and *B* mesons decay O(100 μm - 10 mm) away from primary vertices (PVs)
- The VELO reconstructs the vertices of the particles with a resolution of O(10 μm)
- It has 26 stations composed of pairs of retractable modules
- It opens and closes *by design:* it needs to be as near as possible to the beams when they are stable



Top view of the 26 station (52 modules) in the x-z plane

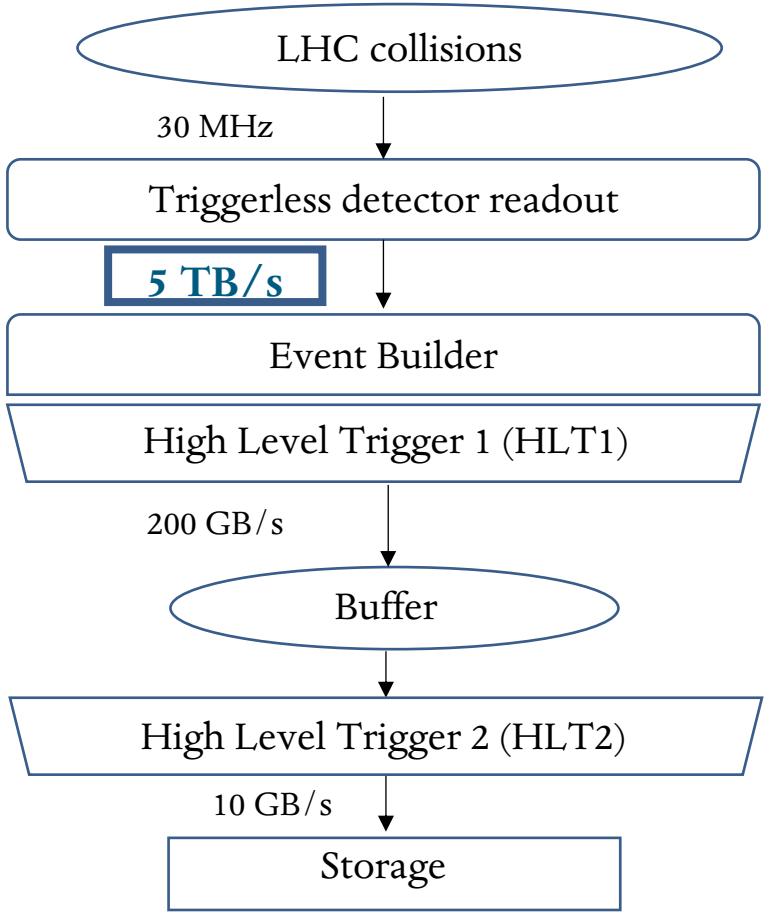


Front view of a single VELO station in x-y plane: two retractable halves (modules)

The real-time software trigger of LHCb

Flavour physics penalised by conventional hardware threshold-based selections (p_T, muons…)

→ In 2022, LHCb switched to reconstructing *all* events **in detail** in software, in real time at 30 MHz



Readout of all events handled by FPGAs

Processing a stream of **5 TB/s** requires significant computing power

Addressed with set of GPUs for parallelisation in the first reconstruction stage (Event Builder @ HLT1)

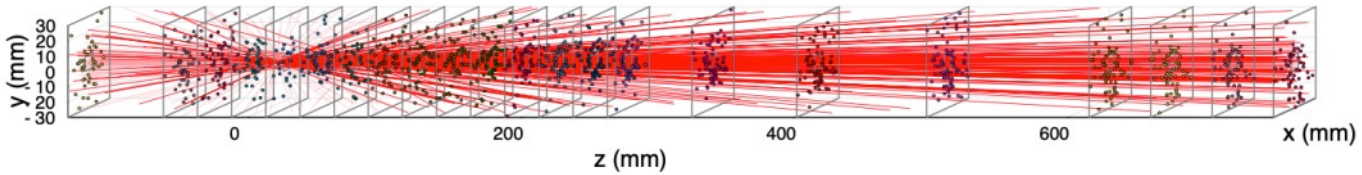
Final reconstruction uses conventional CPUs (HLT2)

Events stored for physics analysis

Towards High-Luminosity LHC

- **Processing data**: key challenge at LHCb
- **Heterogeneous computing** in the first reconstruction stage to **reduce the throughput**

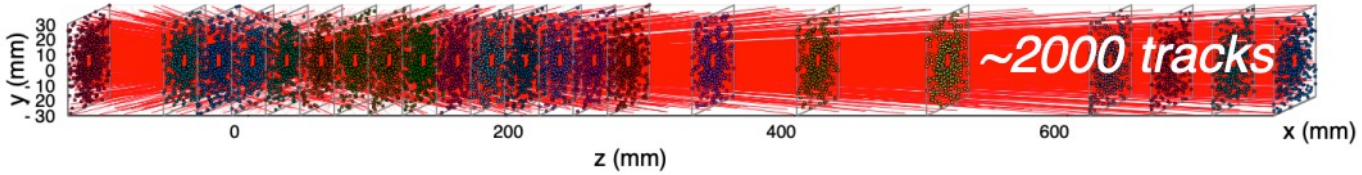
Run 3: pile-up ~6



Today

- High-Luminosity LHC: increase of the average number of pp interactions per event (pile-up)

Upgrade II: pile-up ~40



Future

- **Problem:** Conventional reconstruction techniques may become a limiting factor
- Move even more tasks to suitable devices may allow to free HLT processing power
- Executing a first reconstruction swiftly "on the fly" as the data is being read
- **Possible solution:** Reconstructing **primitives** with FPGAs during detector readout

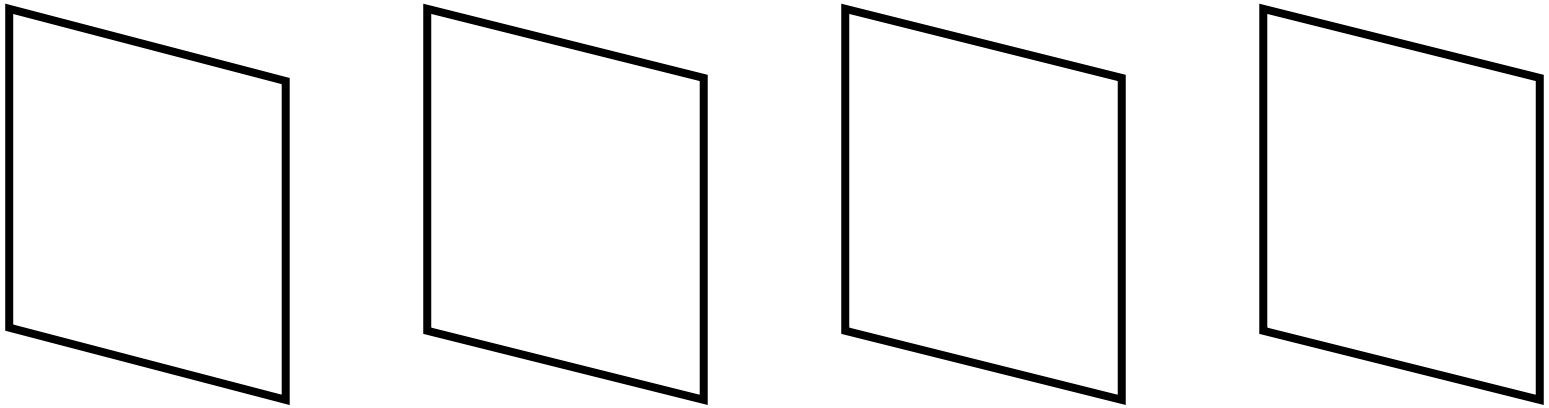
Primitives

What are primitives?

- Primitives: quantities evaluated during detector readout ⟶ produce higher level objects
- Primitives can be raw data or intermediate quantities calculated from raw data

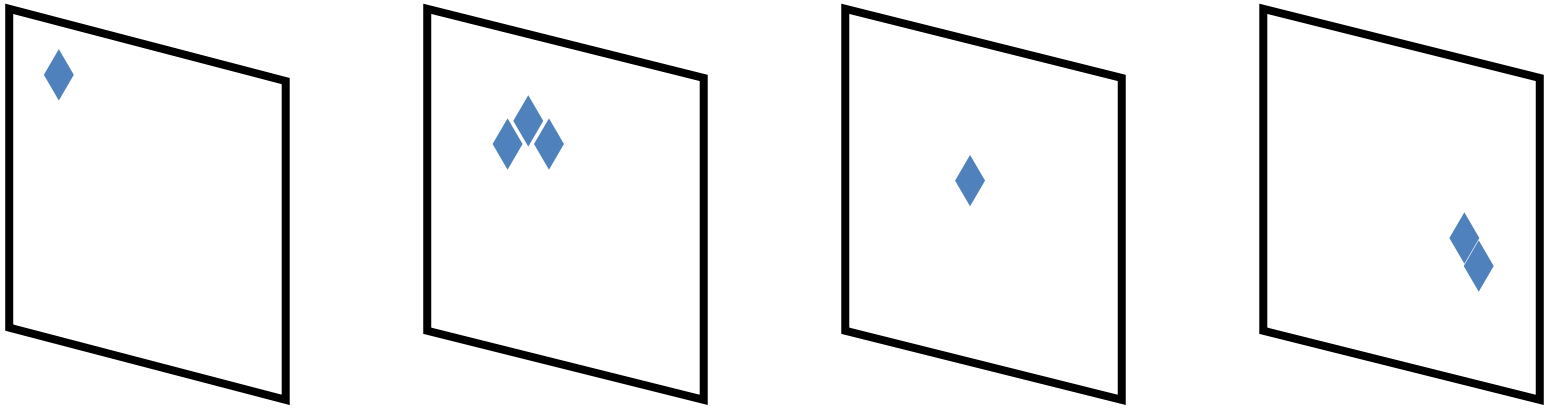
What are primitives?

- Primitives: quantities evaluated during detector readout ⟶ produce higher level objects
- Primitives can be raw data or intermediate quantities calculated from raw data
- In a tracking example with four pixel detector stations:



What are primitives?

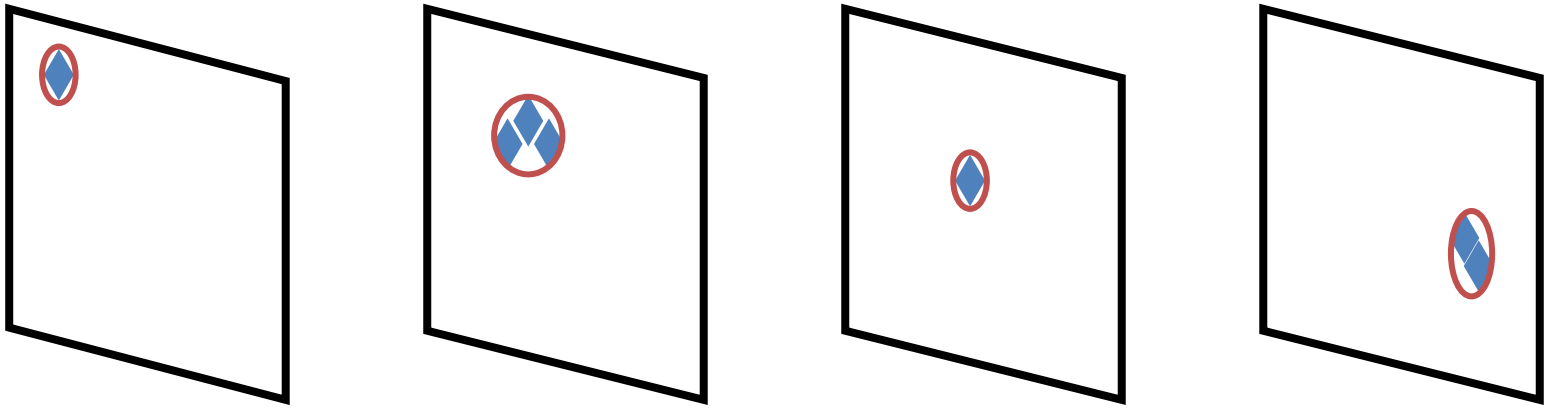
- Primitives: quantities evaluated during detector readout ⟶ produce higher level objects
- Primitives can be raw data or intermediate quantities calculated from raw data
- In a tracking example with four pixel detector stations:



Active channels

What are primitives?

- Primitives: quantities evaluated during detector readout ⟶ produce higher level objects
- Primitives can be raw data or intermediate quantities calculated from raw data
- In a tracking example with four pixel detector stations:



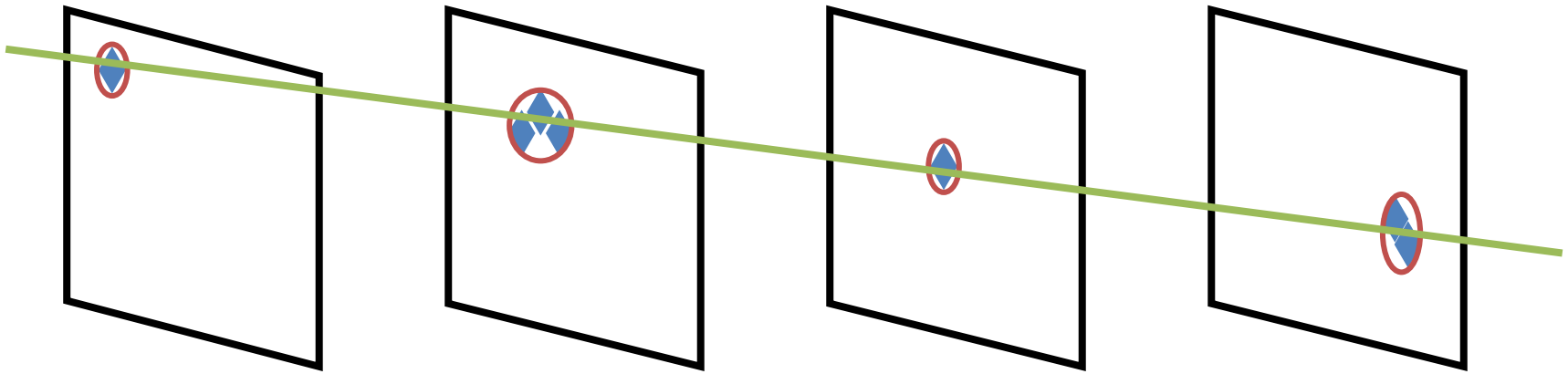
Active channels

↓

List of hits (clusters)

What are primitives?

- Primitives: quantities evaluated during detector readout ⟶ produce higher level objects
- Primitives can be raw data or intermediate quantities calculated from raw data
- In a tracking example with four pixel detector stations:



Active channels

↓

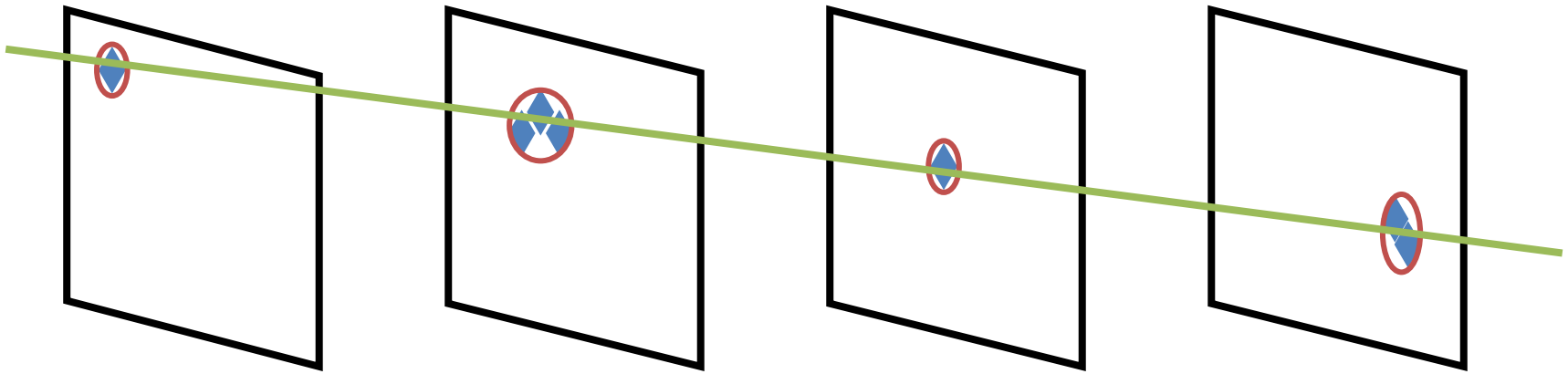
List of hits (clusters)

↓

Tracks

What are primitives?

- Primitives: quantities evaluated during detector readout ⟶ produce higher level objects
- Primitives can be raw data or intermediate quantities calculated from raw data
- In a tracking example with four pixel detector stations:



Active channels

↓

List of hits (clusters)

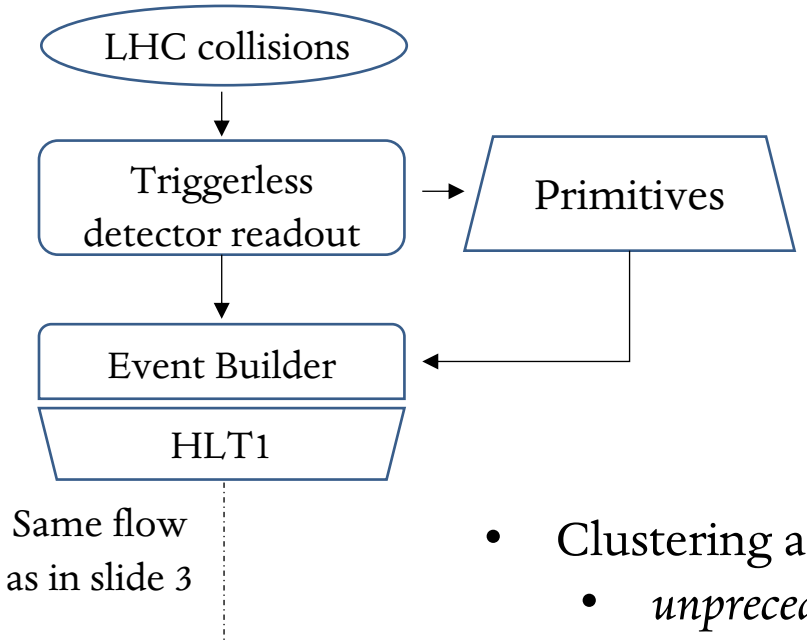
↓

Tracks

- At LHCb: a 2D clustering architecture implemented on FPGA operates already in this Run
- Reconstruct hits on the VELO directly during detector readout **at the collision rate of 30 MHz** (real-time)

Primitives: a new paradigm that speeds up the reconstruction process

Raw Data → Primitives → Higher level objects



- Complex primitives - such as clusters (hits) - are used to accelerate the High Level Triggers
  - The reconstruction starts from pre-processed data:
  - Possibility to drop some raw data and reduce bandwidth needs
  - **Clustering frees 11% of the resources in HLT1**

10.1109/TNS.2023.3273600

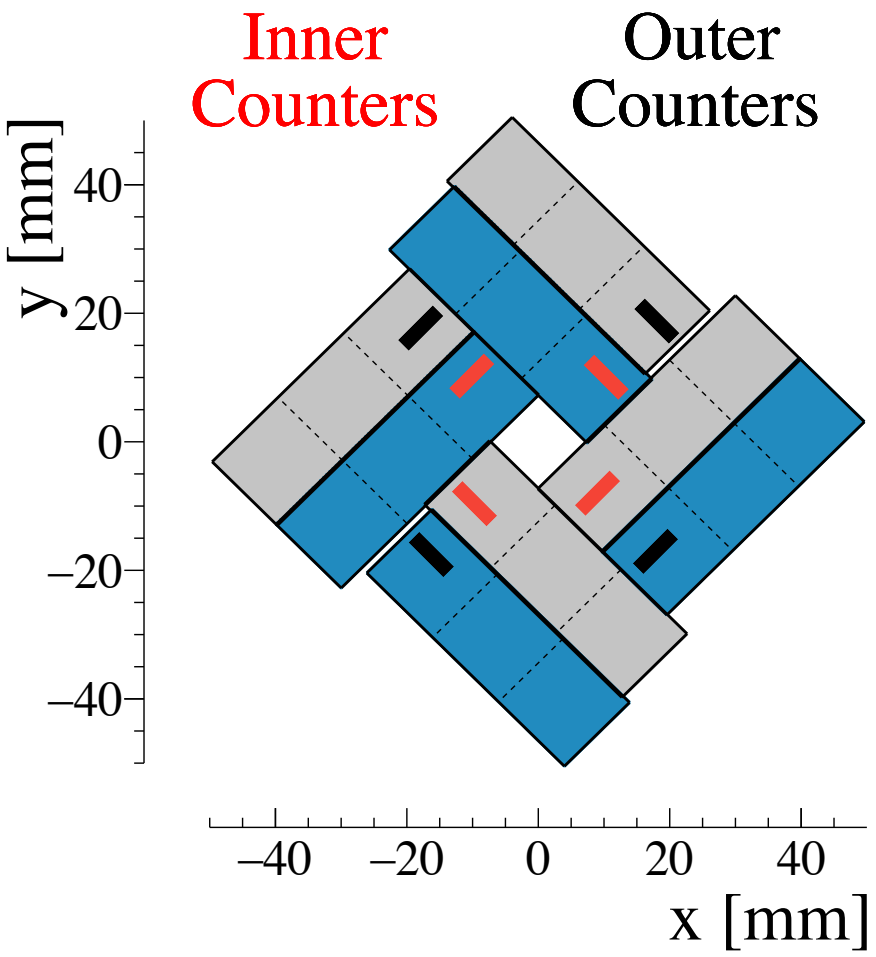
- Clustering architecture provides hit positions for **every collision**
  - *unprecedented* flow of 10^{11} hits per second
  - **High statistics hit sample** that can be used to calculate more complex quantities

➢ Availability of clusters in real time allows to define new primitives starting from clusters

Constructing other primitives: the VELO cluster counters

- Only simple calculations can be performed during readout
- Counting clusters: simple operation, implemented in firmware
- Use cluster counters in specific VELO regions as additional primitives

- Implemented multiple counters in each VELO station
- 8 counters per station, total of 208 counters
- Each counter accumulates a certain number of events N_{events}
- Reading an *average* number of cluster counts per event
- Background (from non pp collisions) can be subtracted in real time



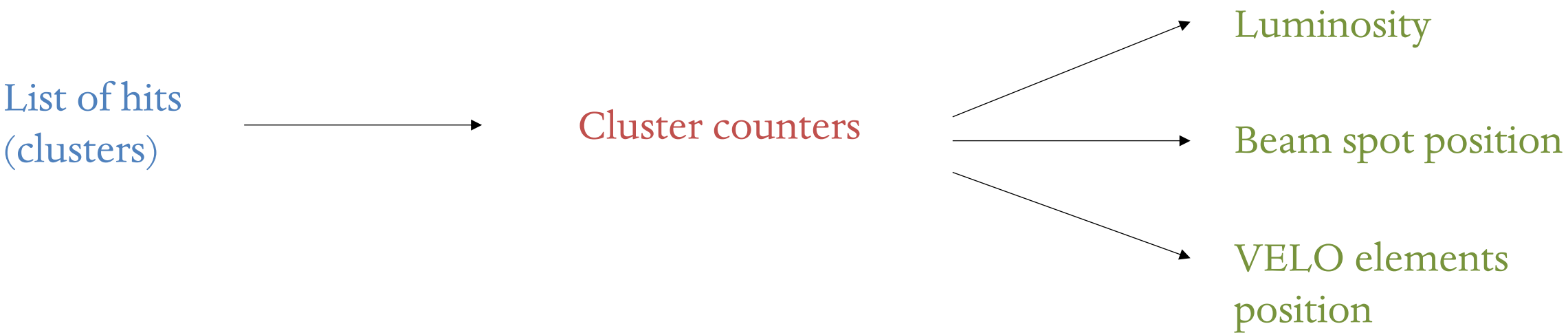
Newly defined primitives: $C^i_{avg} = \frac{C^i_{TOT} - C^i_{bkg}}{N_{events}}$

More complex quantities reconstructed in real time

The clusters are available at the collision rate of ~30 MHz

The counters are implemented in firmware
→ available at ~30 MHz

These new primitives allow the computation of useful quantities in **real time**



➢ **How is this different from conventional reconstruction techniques?**

- These quantities are estimated *before* the event is reconstructed
- This is possible because high-statistic hit samples are available at an early stage in data acquisition chain
- Normally, these quantities require to run much more complex algorithms (tracking)

Luminosity

What is luminosity and why is it useful?

Definition:

$$L = \frac{\text{Number of inelastic } pp \text{ collision per sec}}{\text{inelastic } pp \text{ cross section}} = f\, N_b \frac{\mu}{\sigma_{pp@LHCb}}$$

- f: LHC revolution frequency
- N_b: Number of colliding bunches
- μ: Inelastic collisions per evt (pile-up)
- $\sigma_{pp@LHCb}$: Inelastic cross section

Luminosity is needed for:

- cross section measurements
- performing luminosity levelling → **Needed in real time**

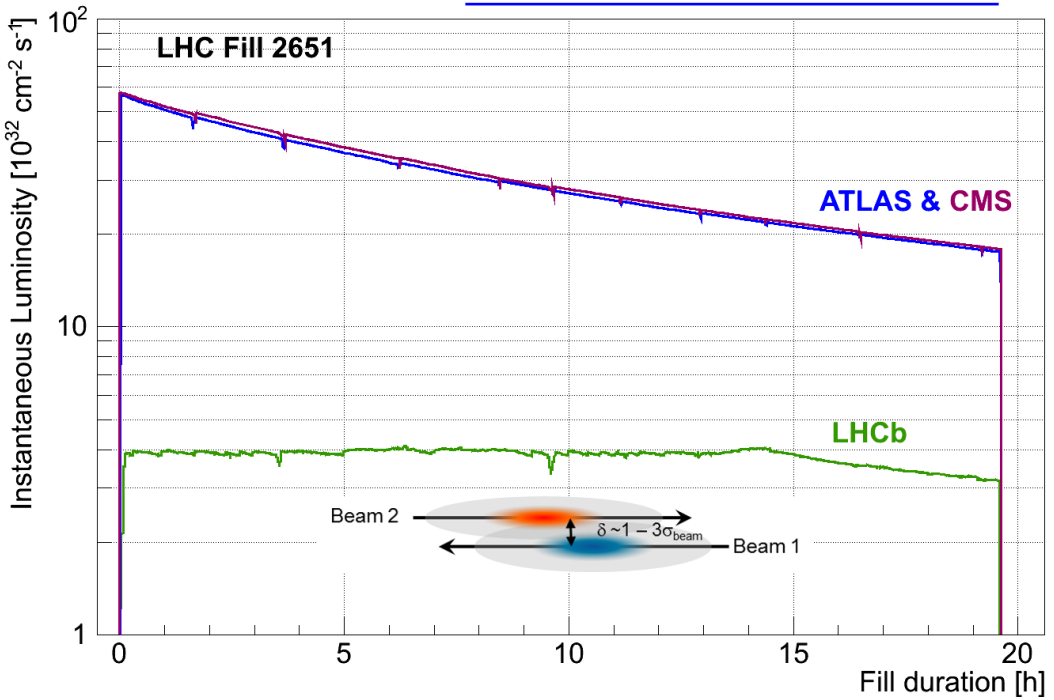
Measurement: using quantities linearly correlated with pile-up

$$L = f\, N_b \frac{\mu_{vis}}{\sigma_{vis}}$$

Using number of visible interactions/event
= number of vertices, tracks, hits on detector…

➢ **Cluster counters** are suitable observables to measure the luminosity

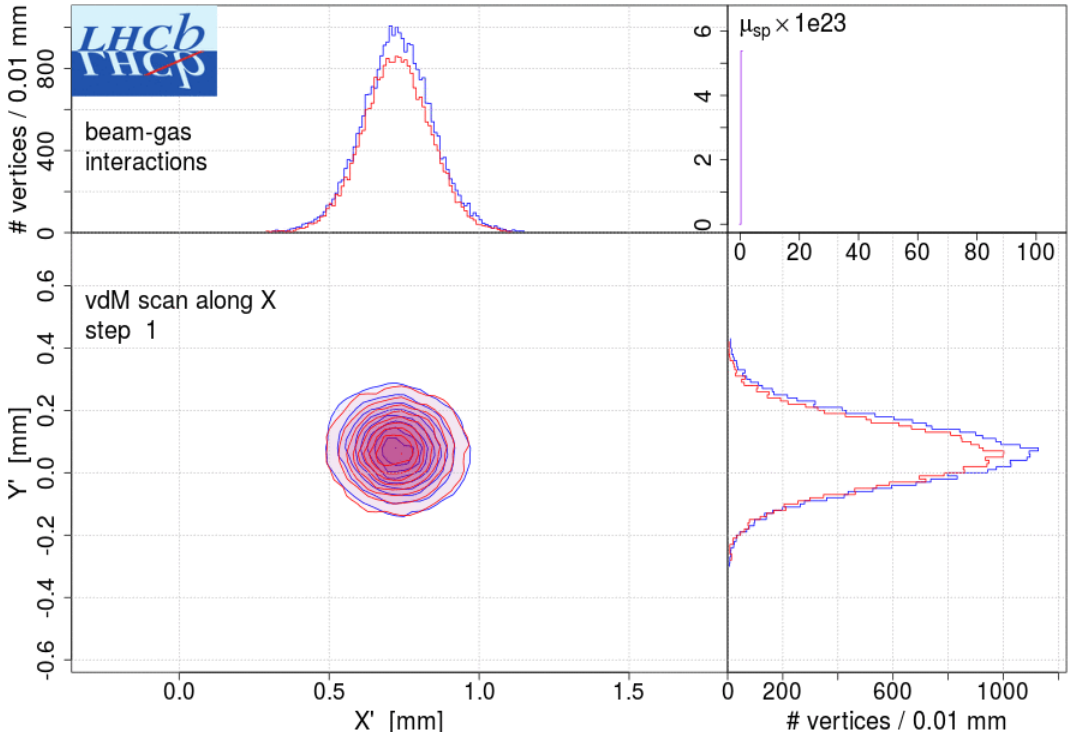
10.1142/S0217751X15300227



How can we obtain luminosity from counters rates?

- Luminosity is proportional to μ_{vis} with a proportionality factor $1/\sigma_{vis}$
- σ_{vis} is computed during van der Meer (vdM) scans
- These scans involve shifting the two colliding beams and measuring the rates varying the relative distances $\Delta x, \Delta y$
- The visible cross section can be computed for each counter using the formula

$$\sigma^i_{vis} = \iint \frac{\mu^i_{vis}(\Delta x, \Delta y)}{N_1 N_2} d\Delta x d\Delta y$$



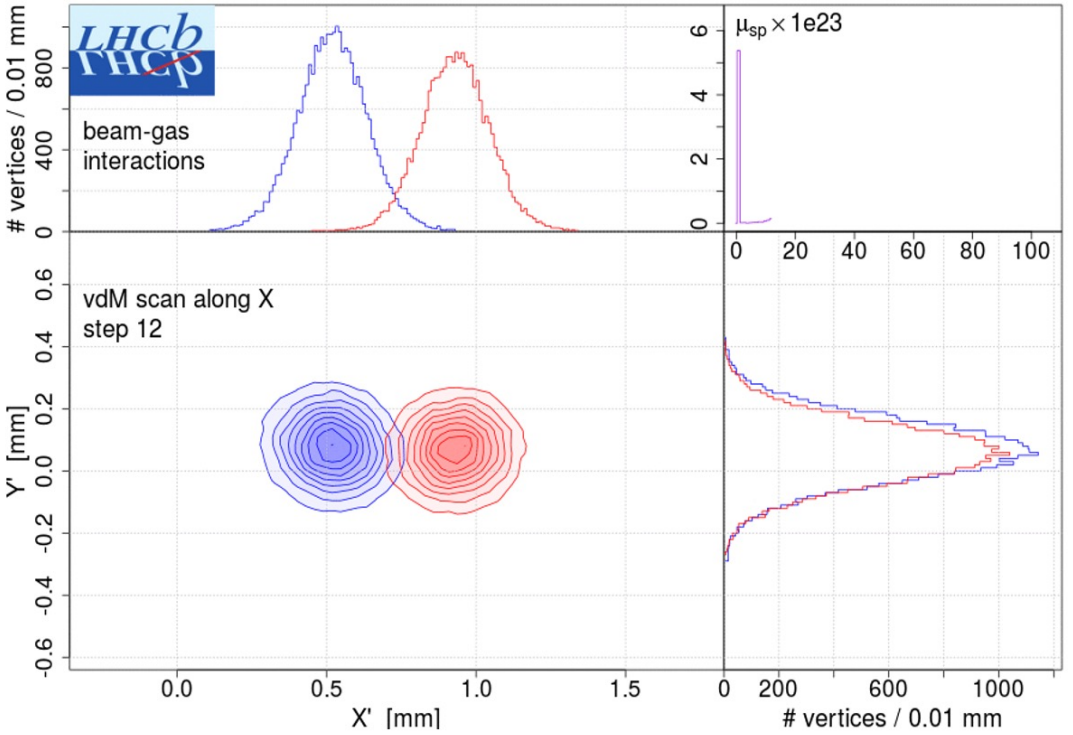
Gif: courtesy of Vladislav Balagura

- Each counter produces an independent luminosity measurement $L_i = f\ N_b \frac{\mu^i_{vis}}{\sigma^i_{vis}}$

How can we obtain luminosity from counters rates?

- Luminosity is proportional to μ_{vis} with a proportionality factor $1/\sigma_{vis}$
- σ_{vis} is computed during van der Meer (vdM) scans
- These scans involve shifting the two colliding beams and measuring the rates varying the relative distances $\Delta x, \Delta y$
- The visible cross section can be computed for each counter using the formula

$$\sigma_{vis}^{i} = \iint \frac{\mu_{vis}^{i}(\Delta x, \Delta y)}{N_1 N_2} d\Delta x d\Delta y$$



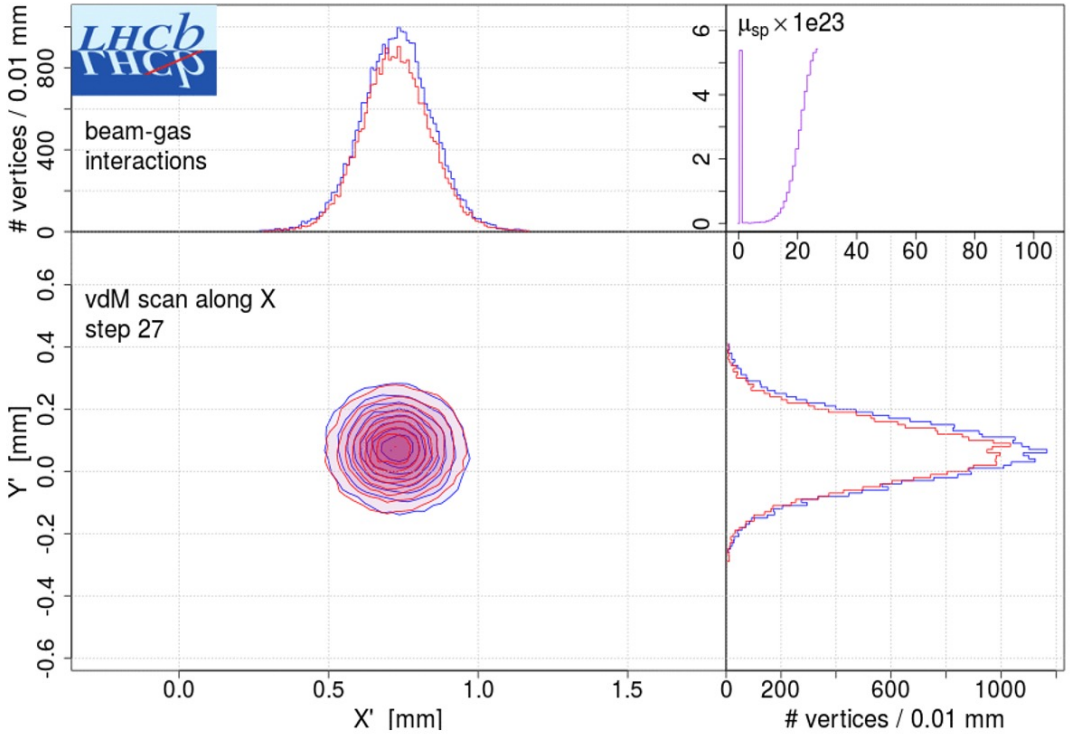
Gif: courtesy of Vladislav Balagura

- Each counter produces an independent luminosity measurement $L_i = f\ N_b \frac{\mu_{vis}^{i}}{\sigma_{vis}^{i}}$

How can we obtain luminosity from counters rates?

- Luminosity is proportional to μ_{vis} with a proportionality factor $1/\sigma_{vis}$
- σ_{vis} is computed during van der Meer (vdM) scans
- These scans involve shifting the two colliding beams and measuring the rates varying the relative distances $\Delta x, \Delta y$
- The visible cross section can be computed for each counter using the formula

$$\sigma^i_{vis} = \iint \frac{\mu^i_{vis}(\Delta x, \Delta y)}{N_1 N_2} d\Delta x d\Delta y$$



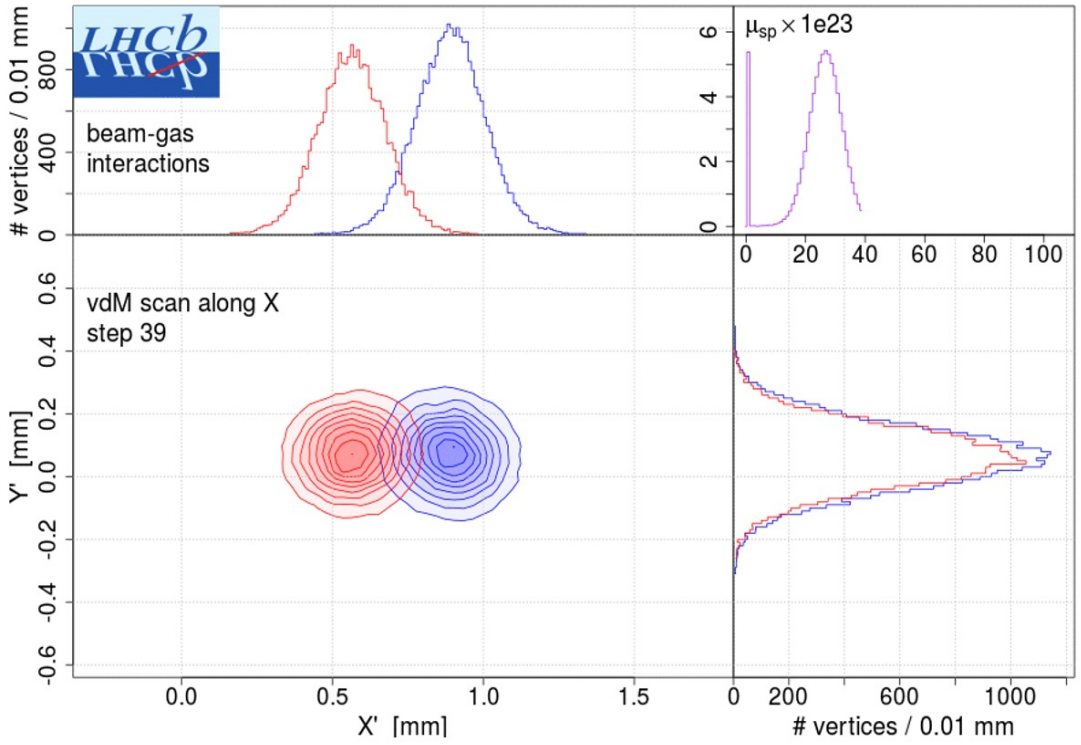
Gif: courtesy of Vladislav Balagura

- Each counter produces an independent luminosity measurement $L_i = f\ N_b \frac{\mu^i_{vis}}{\sigma^i_{vis}}$

How can we obtain luminosity from counters rates?

- Luminosity is proportional to μ_{vis} with a proportionality factor $1/\sigma_{vis}$
- σ_{vis} is computed during van der Meer (vdM) scans
- These scans involve shifting the two colliding beams and measuring the rates varying the relative distances $\Delta x, \Delta y$
- The visible cross section can be computed for each counter using the formula

$$\sigma^i_{vis} = \iint \frac{\mu^i_{vis}(\Delta x, \Delta y)}{N_1 N_2} d\Delta x d\Delta y$$



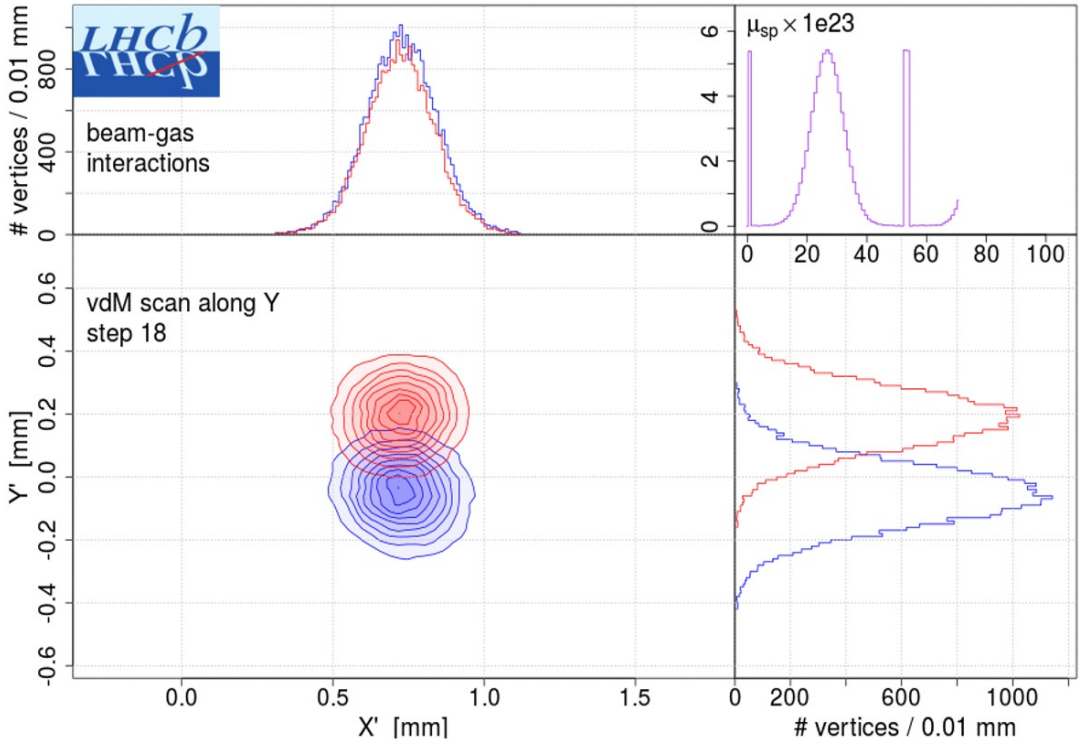
Gif: courtesy of Vladislav Balagura

- Each counter produces an independent luminosity measurement $L_i = f\ N_b \frac{\mu^i_{vis}}{\sigma^i_{vis}}$

How can we obtain luminosity from counters rates?

- Luminosity is proportional to μ_{vis} with a proportionality factor $1/\sigma_{vis}$
- σ_{vis} is computed during van der Meer (vdM) scans
- These scans involve shifting the two colliding beams and measuring the rates varying the relative distances $\Delta x, \Delta y$
- The visible cross section can be computed for each counter using the formula

$$\sigma^i_{vis} = \iint \frac{\mu^i_{vis}(\Delta x, \Delta y)}{N_1 N_2} d\Delta x d\Delta y$$



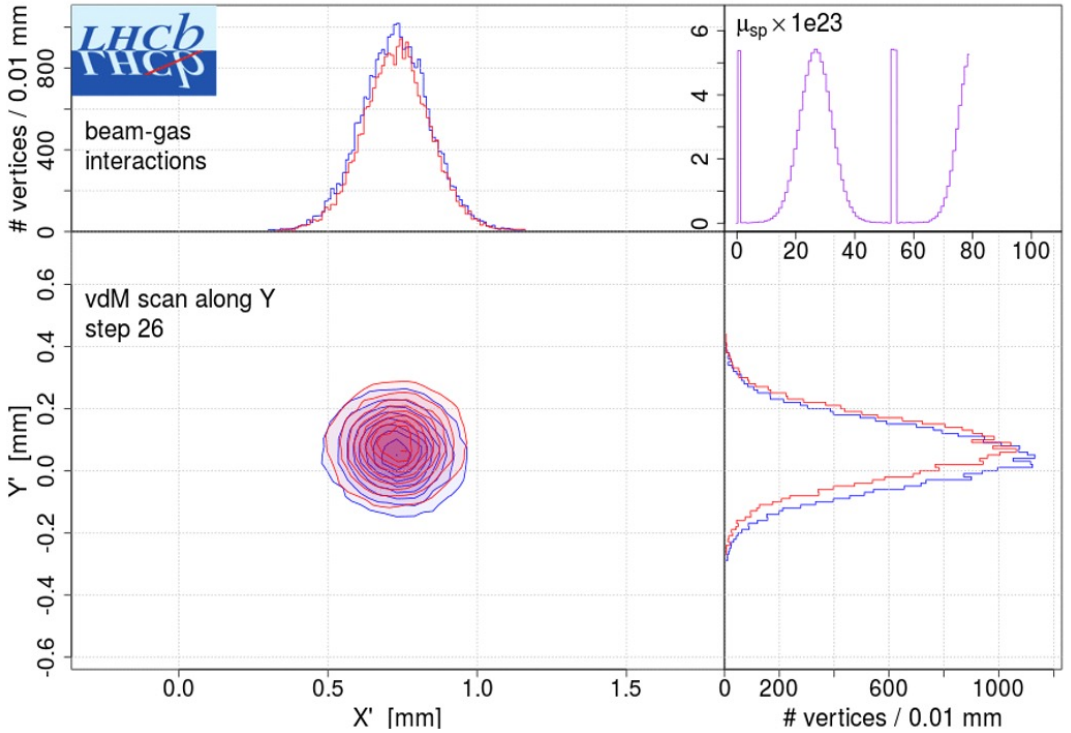
Gif: courtesy of Vladislav Balagura

- Each counter produces an independent luminosity measurement $L_i = f\ N_b \frac{\mu^i_{vis}}{\sigma^i_{vis}}$

How can we obtain luminosity from counters rates?

- Luminosity is proportional to μ_{vis} with a proportionality factor $1/\sigma_{vis}$
- σ_{vis} is computed during van der Meer (vdM) scans
- These scans involve shifting the two colliding beams and measuring the rates varying the relative distances $\Delta x, \Delta y$
- The visible cross section can be computed for each counter using the formula

$$\sigma^i_{vis} = \iint \frac{\mu^i_{vis}(\Delta x, \Delta y)}{N_1 N_2} d\Delta x d\Delta y$$



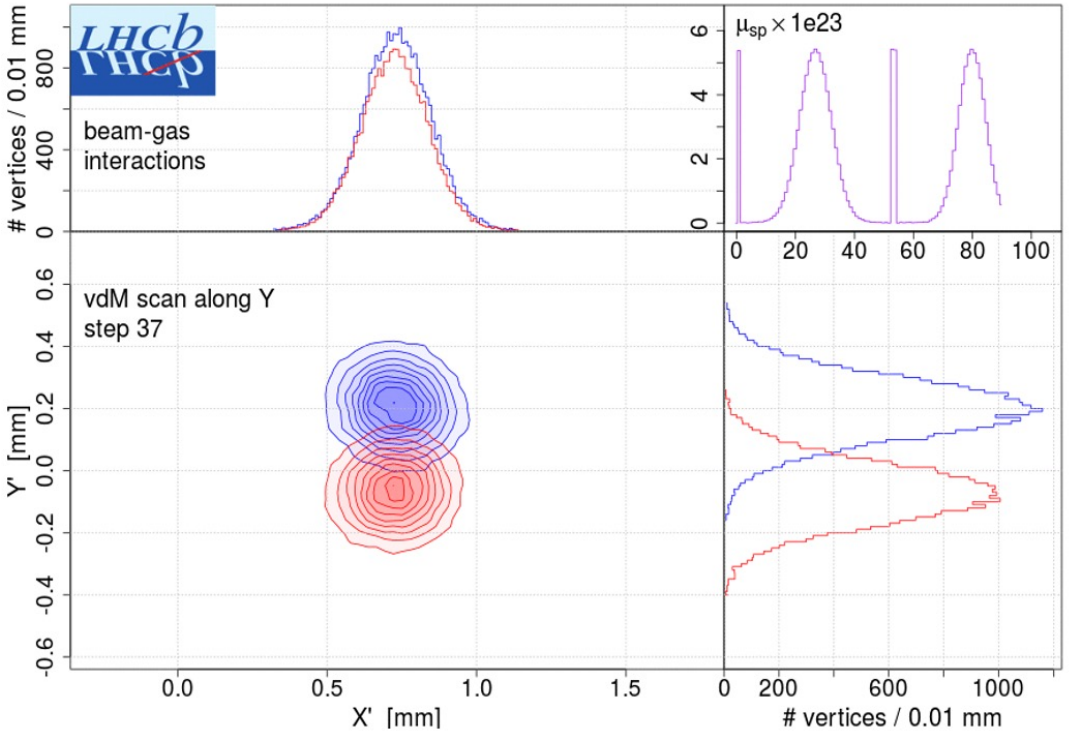
Gif: courtesy of Vladislav Balagura

- Each counter produces an independent luminosity measurement $L_i = f\ N_b \frac{\mu^i_{vis}}{\sigma^i_{vis}}$

How can we obtain luminosity from counters rates?

- Luminosity is proportional to μ_{vis} with a proportionality factor $1/\sigma_{vis}$
- σ_{vis} is computed during van der Meer (vdM) scans
- These scans involve shifting the two colliding beams and measuring the rates varying the relative distances $\Delta x, \Delta y$
- The visible cross section can be computed for each counter using the formula

$$\sigma_{vis}^{i} = \iint \frac{\mu_{vis}^{i}(\Delta x, \Delta y)}{N_1 N_2} d\Delta x d\Delta y$$



Gif: courtesy of Vladislav Balagura

- Each counter produces an independent luminosity measurement $L_i = f\ N_b \frac{\mu_{vis}^{i}}{\sigma_{vis}^{i}}$

A global luminosity estimator

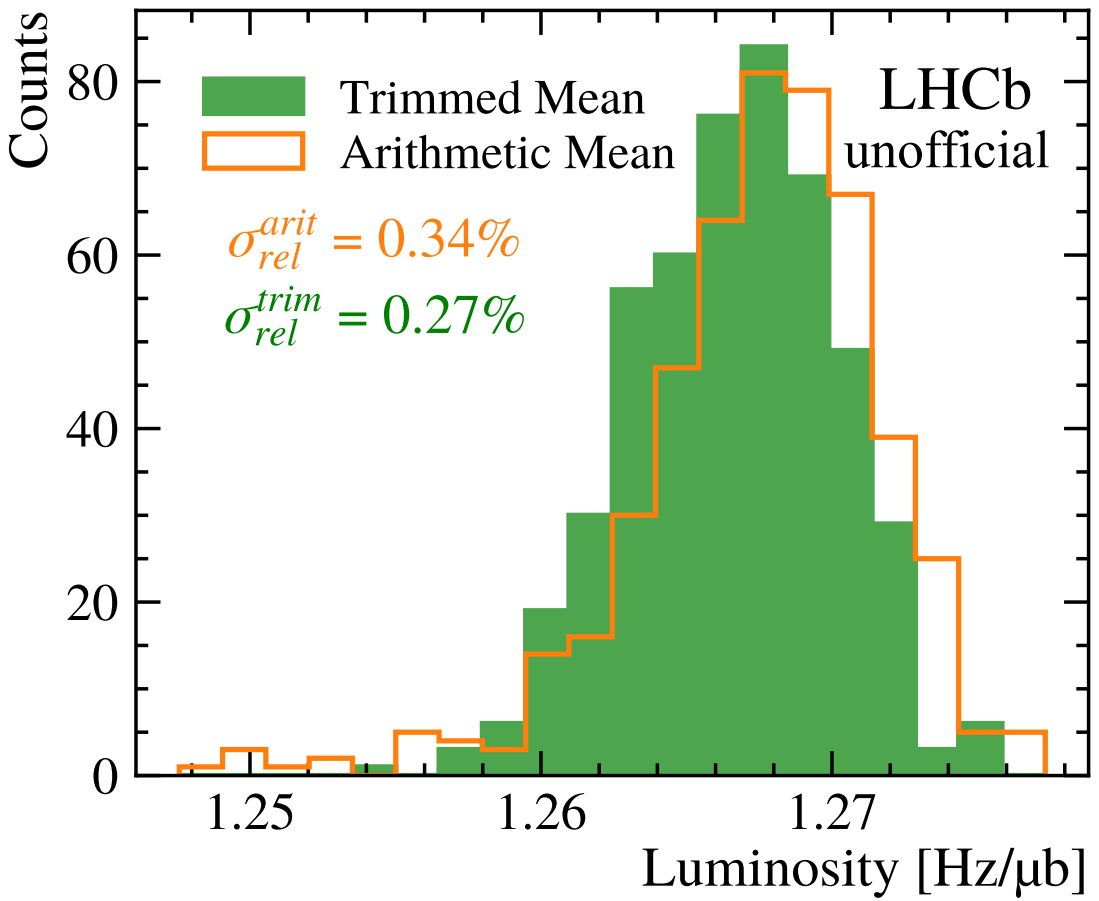
- Each counter produces an independent luminosity measurement
- Several choices to combine them in a global estimator:
  - Arithmetic Mean
  - Median
  - Trimmed Mean
  - …

- The presence of outliers cannot be excluded: need a robust estimator, i.e. less sensible to values outside the expected distributions

➢ **Trimmed mean** is an inherently robust estimator

- Performance evaluated during period of constant luminosity

➢ Relative uncertainty of trimmed mean: 0.27% (20% better than the arithmetic mean)



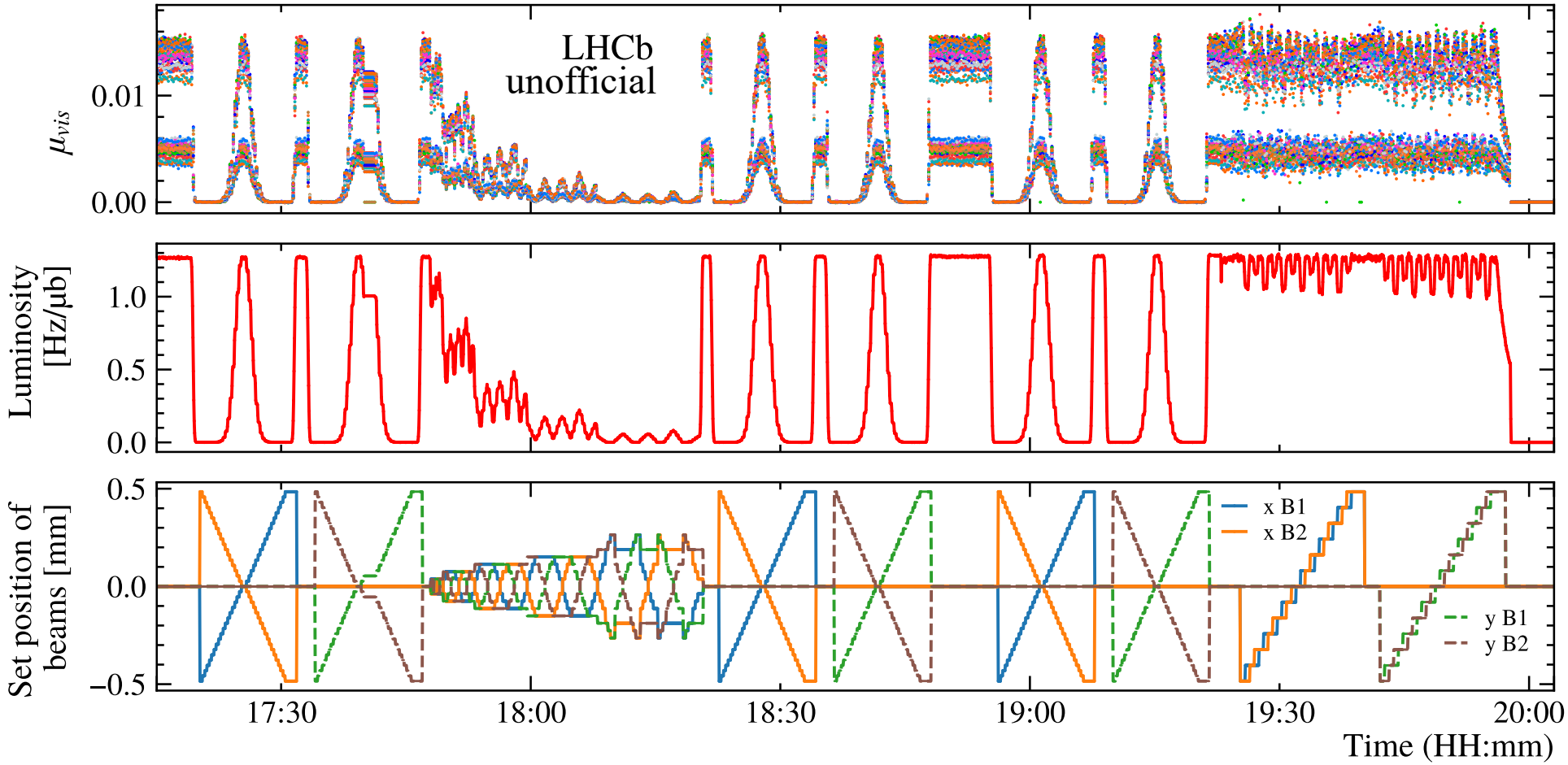
LHCb-DP-2025-006, in preparation

Summary of the 2024 vdM scan

Rates of all the available counters

Luminosity estimator (trimmed mean)

Displacement of the two beams

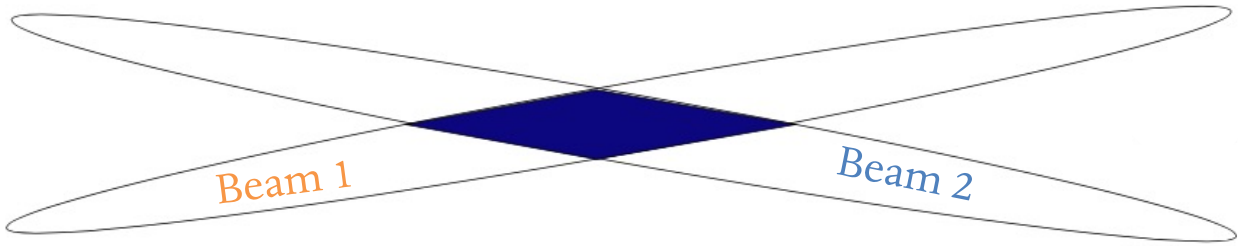


LHCb-DP-2025-006, in preparation

Position of the Beam Spot and VELO elements

What is the Beam Spot (luminous region) and why is it useful to know its position in real-time

Definition of Beam Spot: the beams overlap region, where collisions happen



Useful to know in real time:

- to correct luminosity estimation
- in the future this can be used as input to speed tracking up

Usually, the beam spot is computed using the positions of the primary vertices, which are reconstructed from tracks

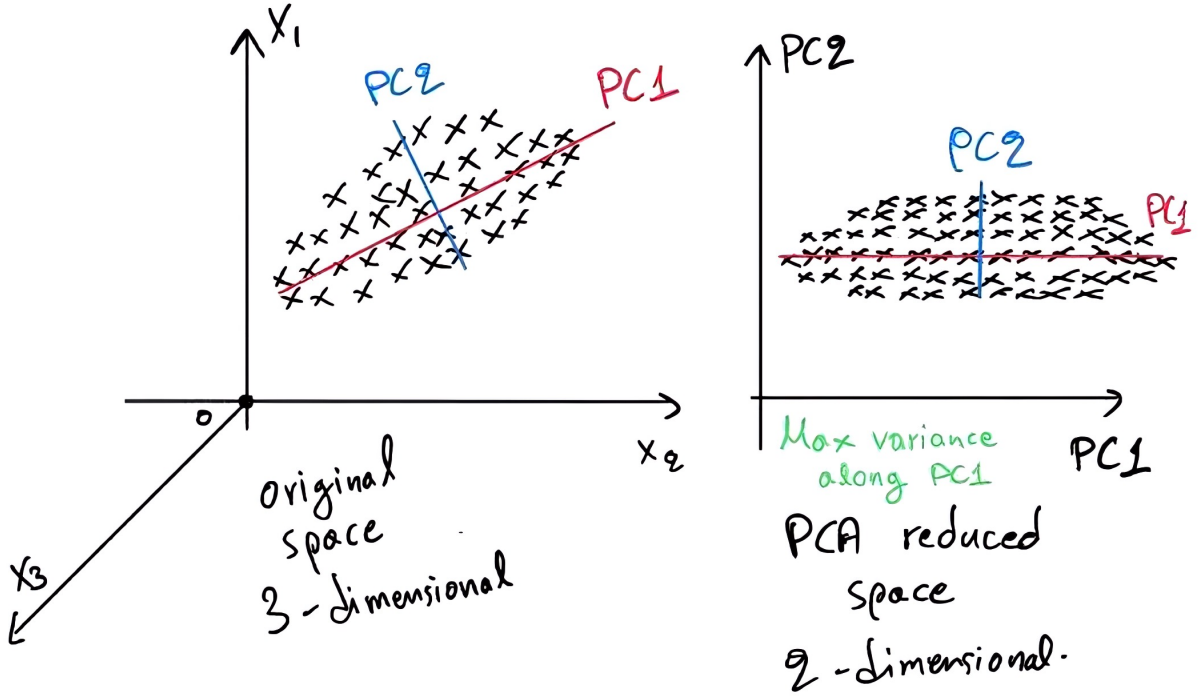
- Problem: Procedure that requires lots of computational power and slows down the reconstruction (computed only for a subset of events)
- Possible solution: Cluster counters provide an opportunity to construct a linear estimator of the position of the beam spot **with no help from tracks**

Principal Component Analysis (PCA)

- Idea: fit the counters c_k with a linear function, e.g. $\hat{x} = \sum_k v_k c_k$ with weights v_k computed in MC
- Problem: high dimensionality of the counters (208) and few points to fit available in MC (order of 10)
- Solution: dimensionality reduction technique → Principal Component Analysis (PCA)

Definition: Orthogonal linear transformation for a new set of coordinates such that the directions (principal components) capturing the largest variation in the data can be easily identified.

- **Evaluation of the Principal Components (PCs):** diagonalizing the covariance matrix of the dataset:
  - Eigenvectors $\vec{w}_k$ of the covariance matrix form the new basis
- **Transformation:** from a vector $\vec{c}$ in the original space to the k-th PC with a scalar product $t_k = \vec{c} \cdot \vec{w}_k$



Position estimator as linear function of the counters

- Define the estimator as a linear combination of PCs, instead of counters
- $\hat{x} = \alpha_1 t_1 + \alpha_2 t_2 + \ ... + \alpha_n t_n$, with $t_k = \vec{c} \cdot \vec{w}_k = \sum_i c_i w_{k,i}$ being the k-th PC
- The estimate of the position $\hat{x}$ is a linear function of the counters, regardless of the number of components used

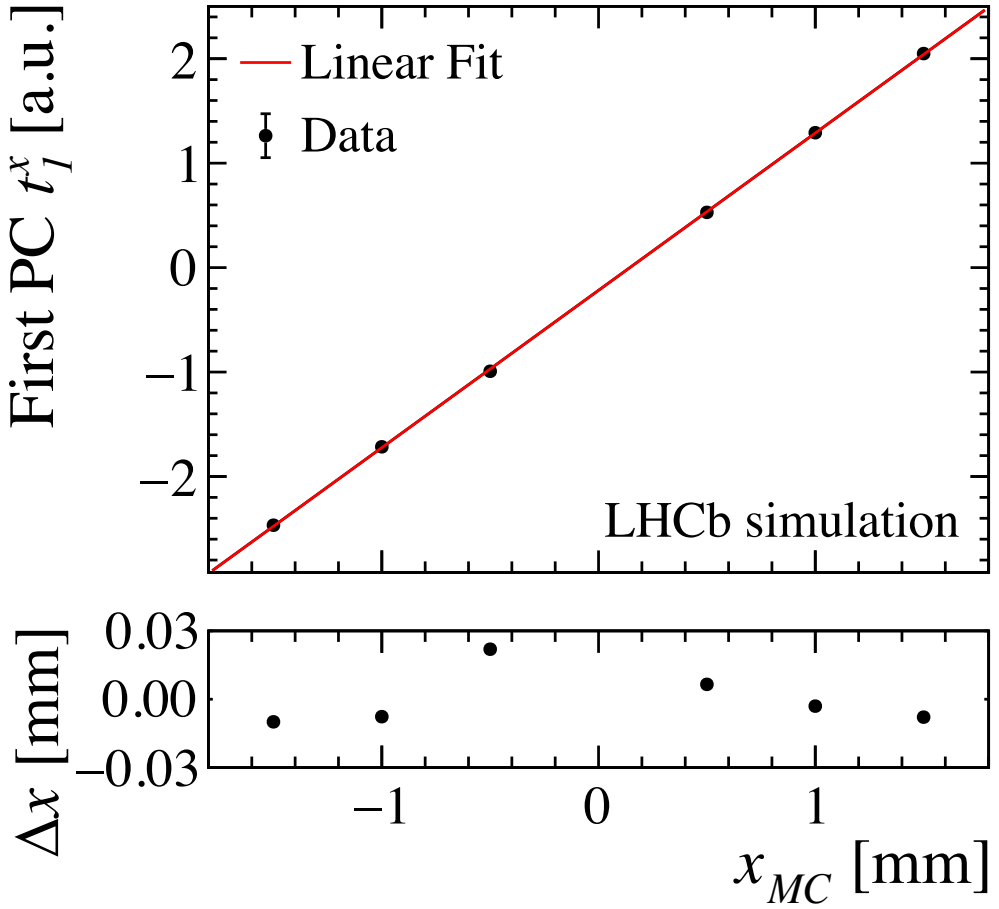
$$\hat{x} = \alpha_1 \sum_i c_i w_{1,i} + \alpha_2 \sum_i c_i w_{2,i} + \ ... + \alpha_n \sum_i c_i w_{n,i}$$

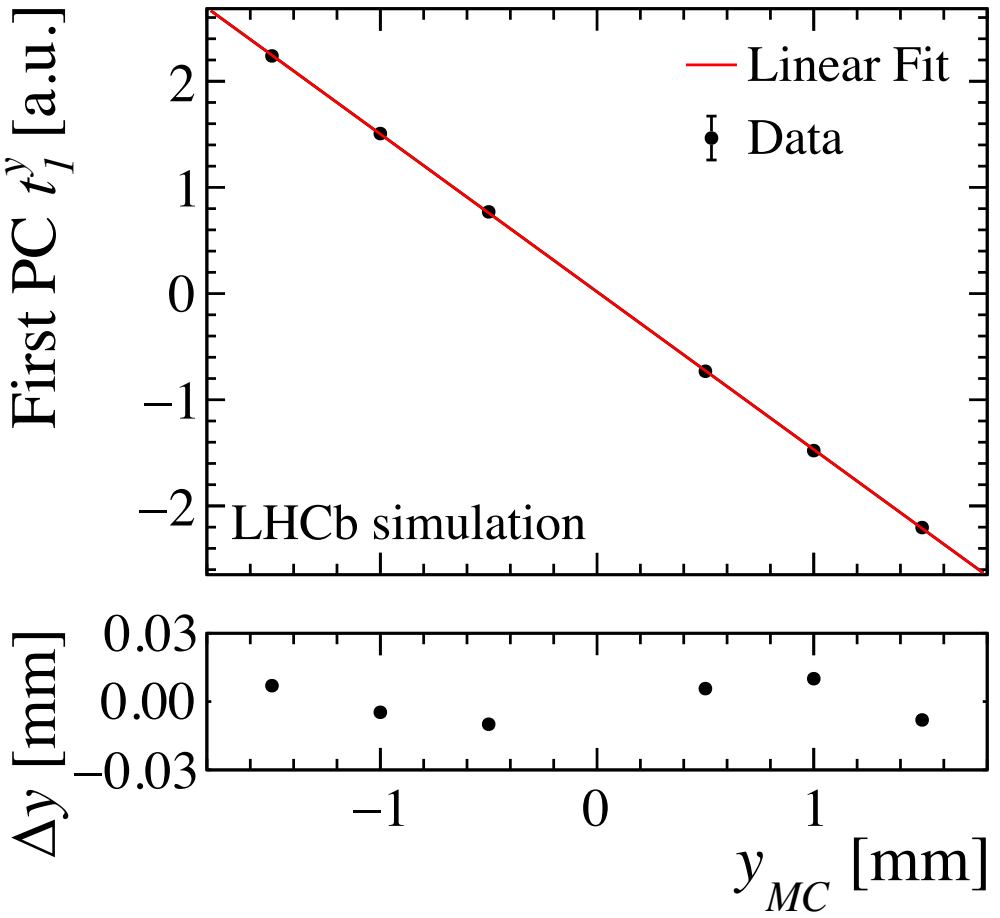
$$= \sum_i c_i \cdot \sum_k \alpha_k w_{k,i} = \sum_i c_i v_i = \vec{c} \cdot \vec{v}$$

➢ How many PCs are needed to calculate an estimate of the position?

➢ The estimators of the $\hat{x}$ and $\hat{y}$ position only need one PC, while $\hat{z}$ needs more due to larger variations

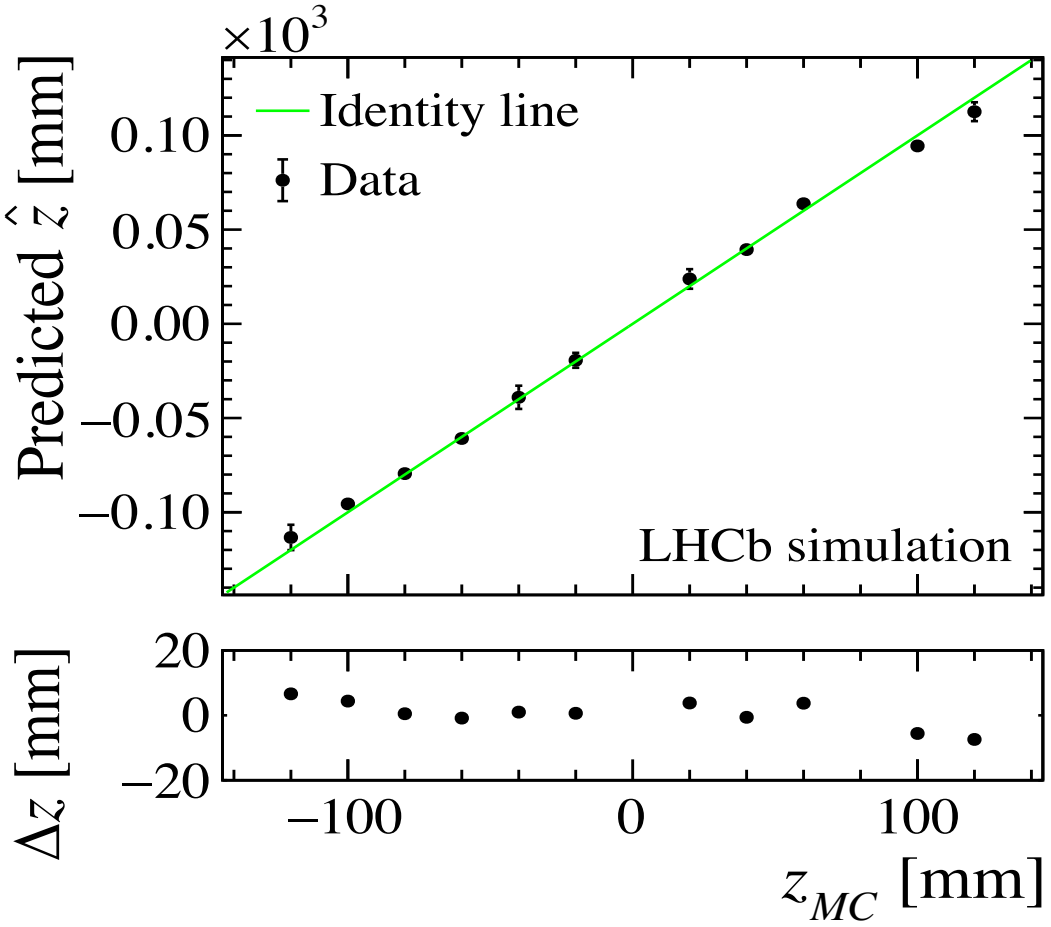
Results on Monte Carlo for x and y position estimators

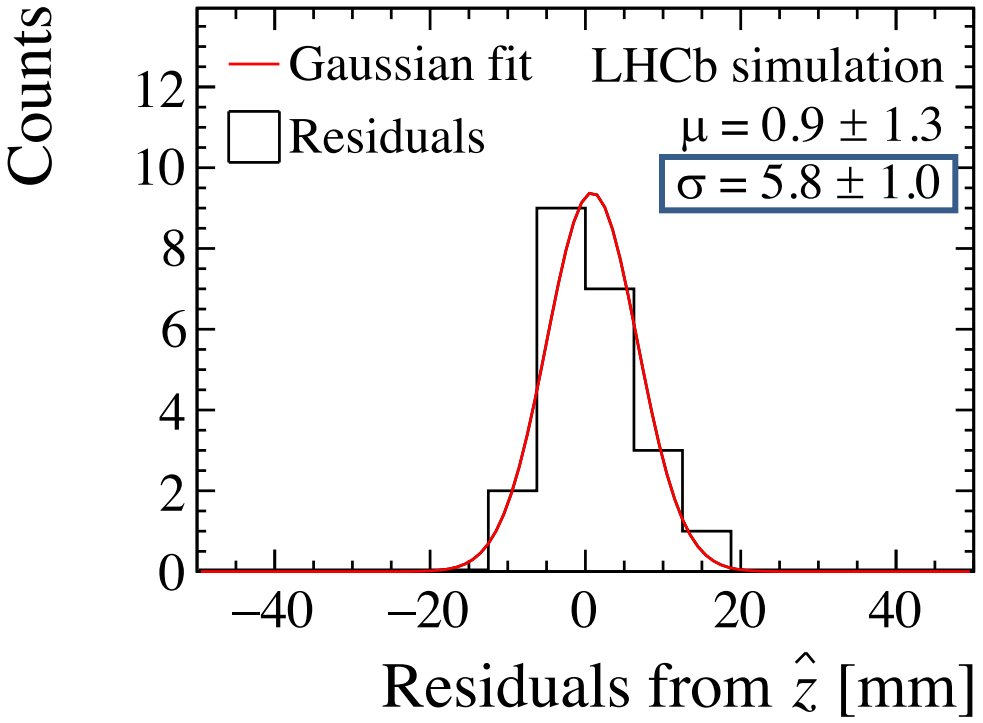




- Linearity of the first component in the transversal beam directions

Results on Monte Carlo for z position estimator





- The z position is estimated using 30 PCs and is an acceptable linear model
- The resolution of 5.8 mm depends on the amount of data on which the counters average is calculated

The closed form of the estimators is very simple and robust

Linearity w.r.t. the pile-up μ

Expected, since the counters are linear w.r.t. luminosity

- Scalar product divided by average number of counters value per evt

Outliers

Counters sometimes do not behave as expected (e.g sudden drops in efficiency or general detector malfunctions)

- Counters too far from the median of all counters are replaced with the median value: **robust procedure**

Analytical form of the estimators:

$$\hat{x} = a_x \frac{\vec{w}_x \vec{c}}{\sum_i c_i} + b_x \qquad \hat{y} = a_y \frac{\vec{w}_y \vec{c}}{\sum_i c_i} + b_y \qquad \hat{z} = \frac{\hat{v}_z \vec{c}}{\sum_i c_i} + b_z$$

This is just a scalar product with some simple multiplications and additions

- **Very simple operation that can be executed very fast without the need for extensive resources**

The closed form of the estimators is very simple and robust

Outliers

Counters sometimes do not behave as expected (e.g sudden drops in efficiency or general detector malfunctions)

- Counters too far from the median of all counters are replaced with the median value: **robust procedure**

Linearity w.r.t. the pile-up μ

Expected, since the counters are linear w.r.t. luminosity

- Scalar product divided by average number of counters value per evt

Analytical form of the estimators:

$$\hat{x} = a_x \frac{\vec{w}_x \vec{c}}{\sum_i c_i} + b_x \qquad \hat{y} = a_y \frac{\vec{w}_y \vec{c}}{\sum_i c_i} + b_y \qquad \hat{z} = \frac{\hat{v}_z \vec{c}}{\sum_i c_i} + b_z$$

This is just a scalar product with some simple multiplications and additions

- **Very simple operation that can be executed very fast without the need for extensive resources**

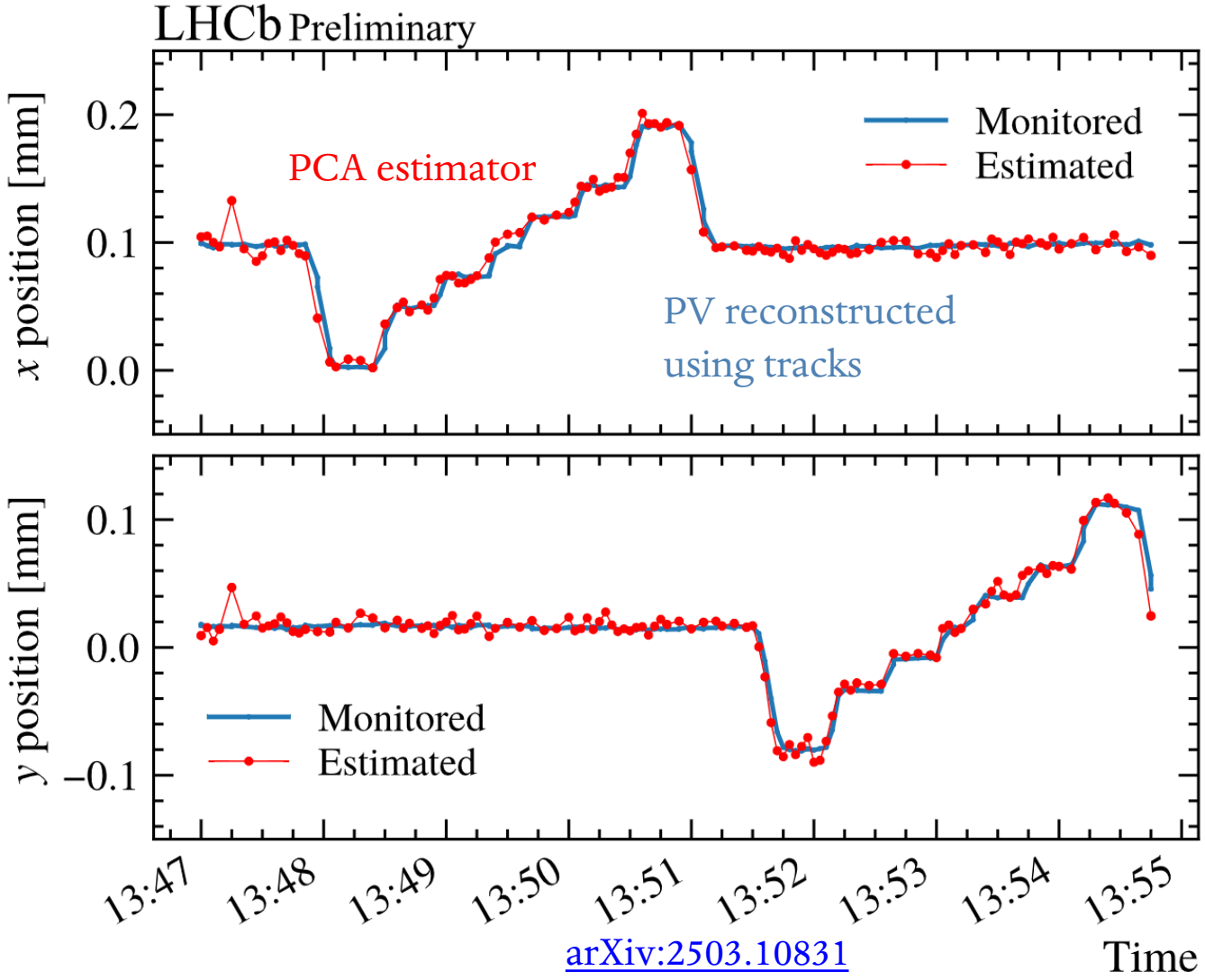
Test on collision data during Length Scale Calibration (LSC)

Test performed on collision data during Length Scale Calibration (LSC) of emittance scan of 06/04/2024

LSC: procedure that involves shifting the beams head-to-head from a position $-a$ to a position $+a$ both in x and y direction

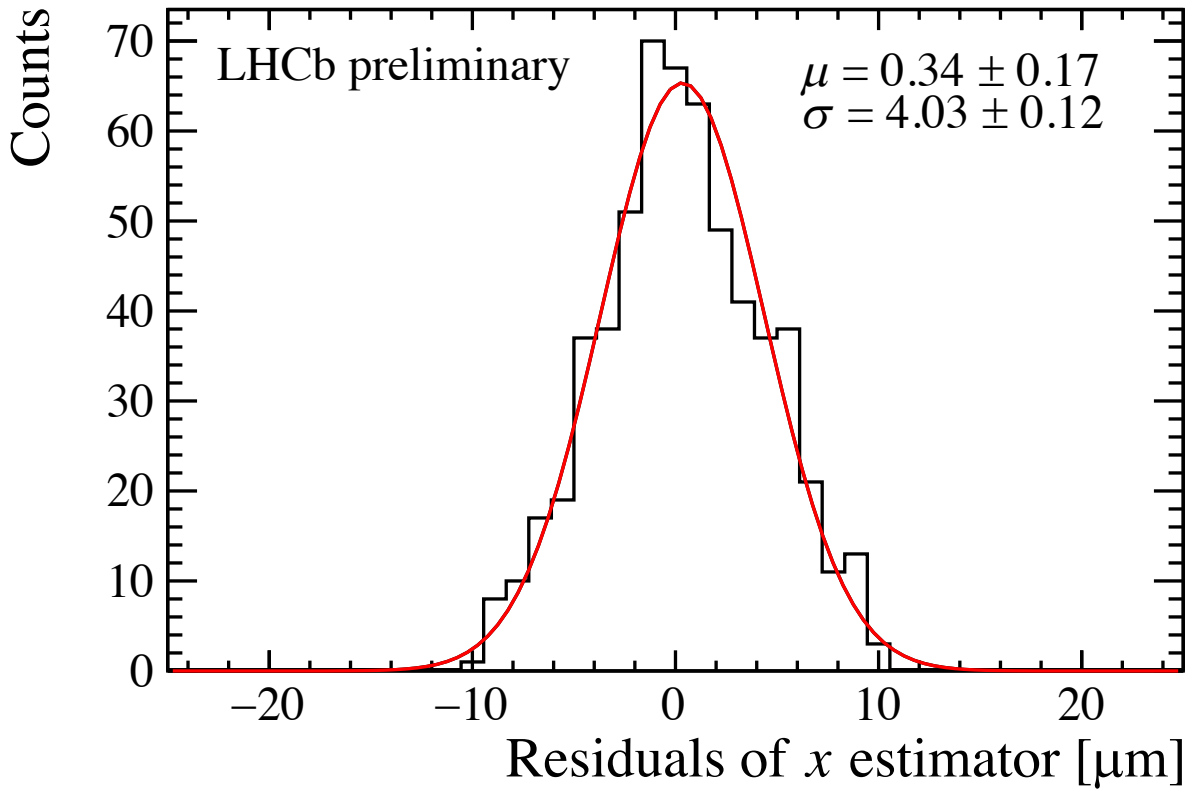
Blue lines are LHCb official monitoring tool based on track reconstruction

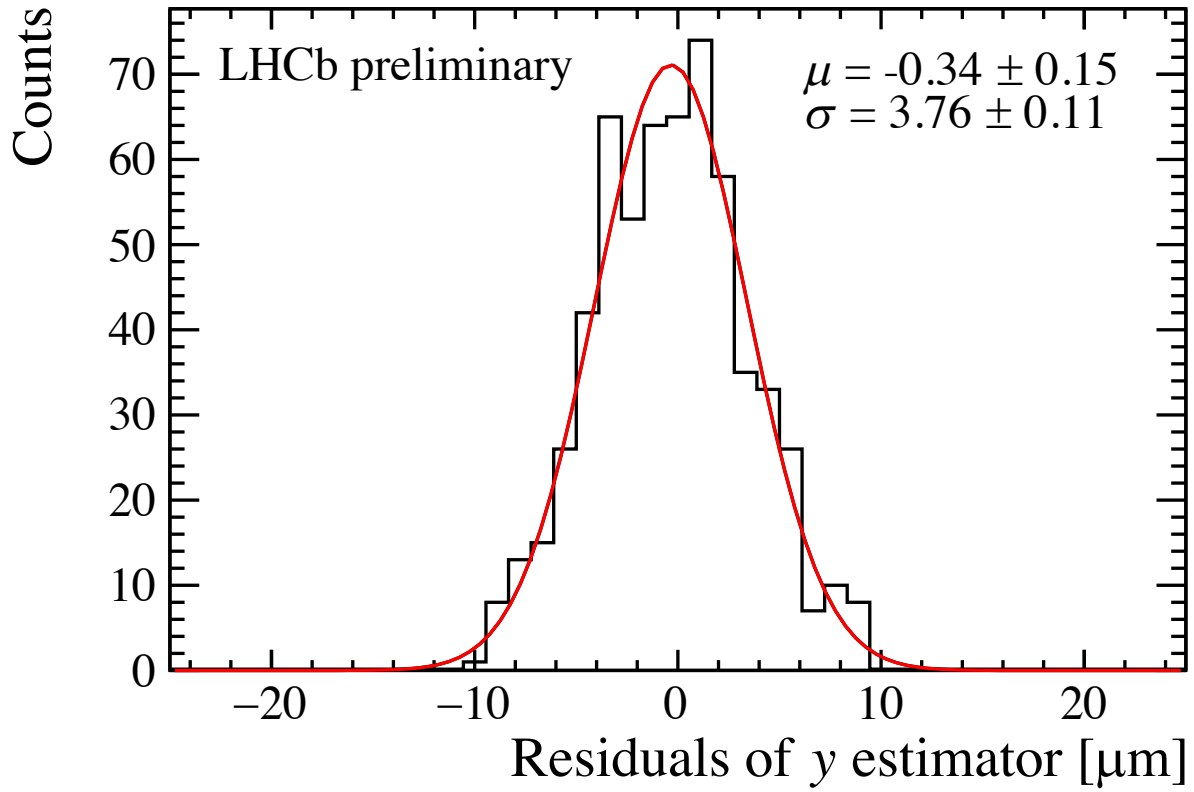
Red lines are the estimates with PCA method



arXiv:2503.10831

Resolution of estimators on real collision data





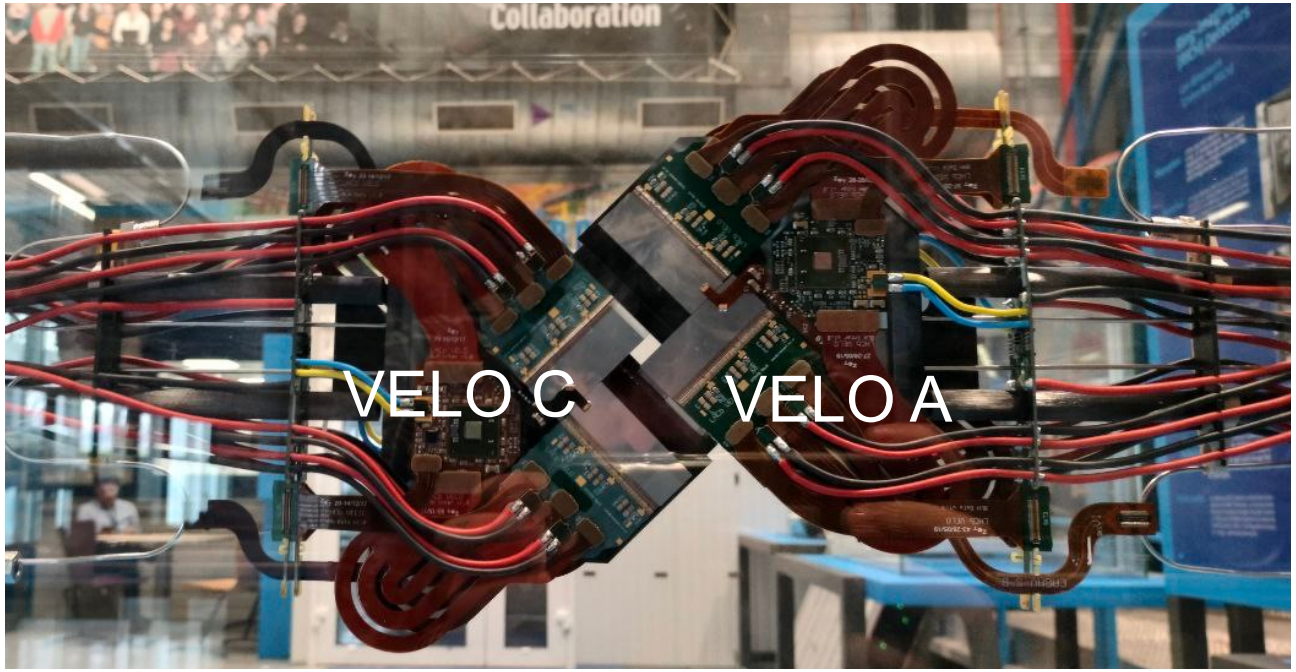
Resolution of ~**4 μm** for a measurement in real-time using only VELO primitives

The statistics needed to collect this data is achieved in **18 ms** in nominal data taking conditions (pile-up ~5.5, 30 MHz)

Monitoring the position of the VELO halves

Reminder from slide 2

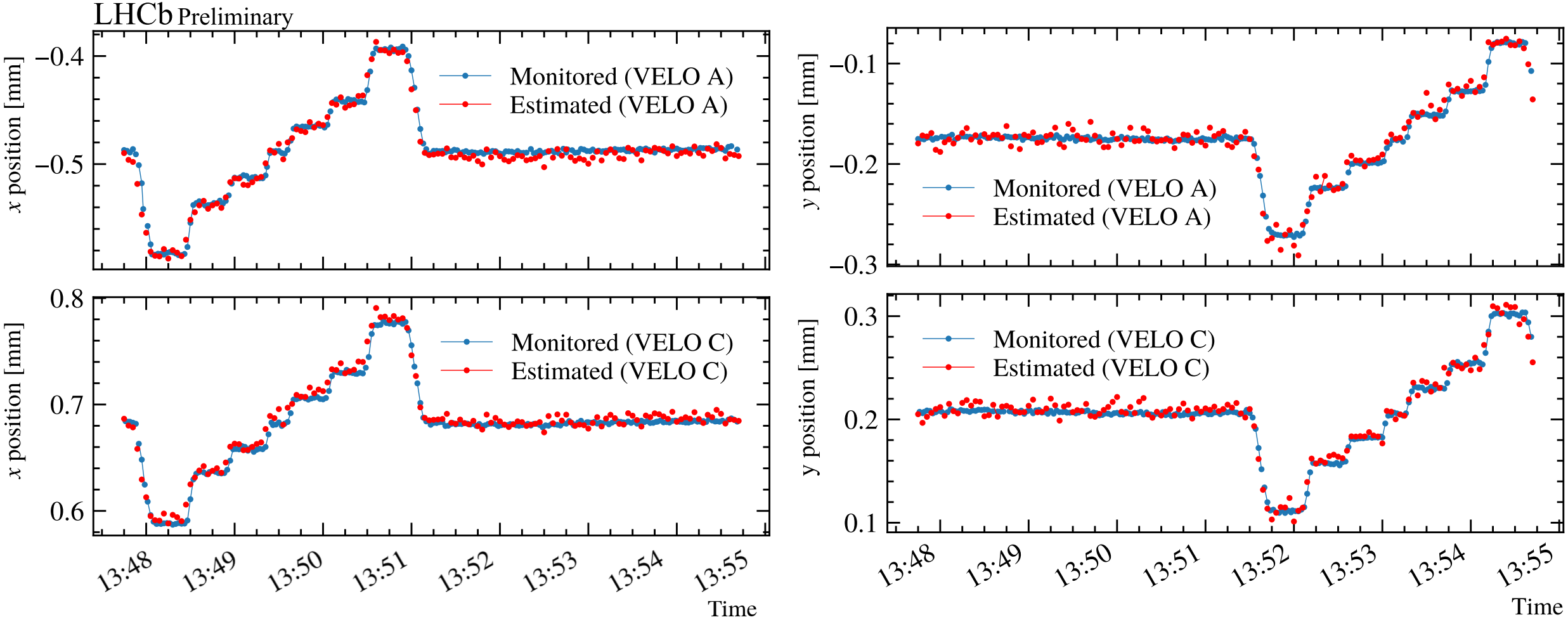
- The VELO is supposed to move *by design:*
  - Two mechanical halves (VELO A and VELO C)
  - They stay open until a stable condition of the beams is reached
  - They stay close for the duration of the fill



- The position of the beam spot is measured w.r.t. a fixed system (the VELO)
- If the beam spot does not move, we can implement estimators in an analogous way to monitor the position of the two VELO sides

- Defining new estimators for the position of VELO A in x,y coordinates and that of VELO C in x,y coordinates
- The vector of counters refers only to the counters in one specific half, and the weights also have reduced dimensionality (104 instead of 208)
- The vectors of weights are recomputed in MC

Overview of the estimators during *pp* collisions

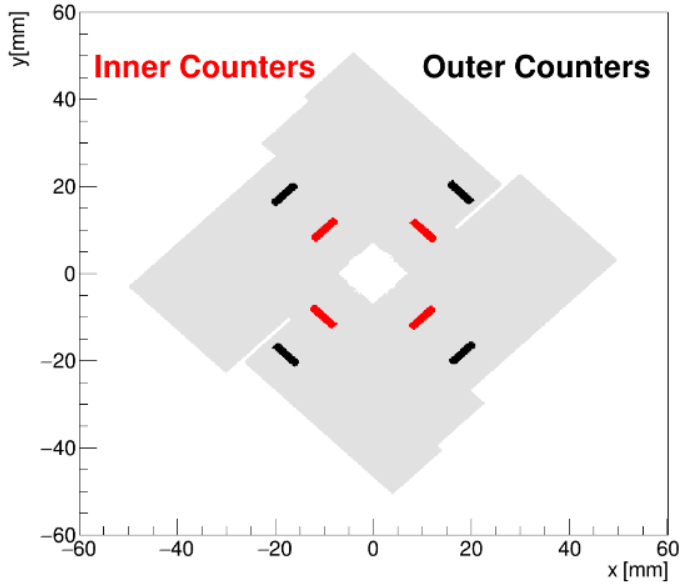


Resolution between 4 μm and 7 μm for the VELO-half position estimators

Outlook and Conclusions

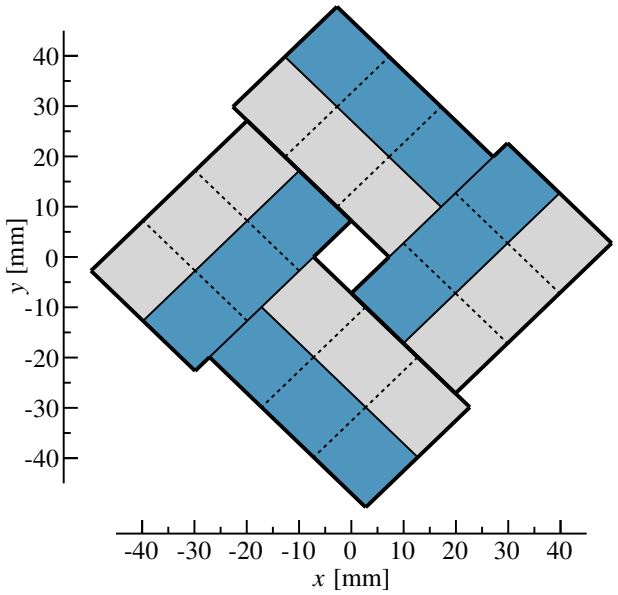
Outlook

- Until now only 1% of the VELO sensor is used for the counters



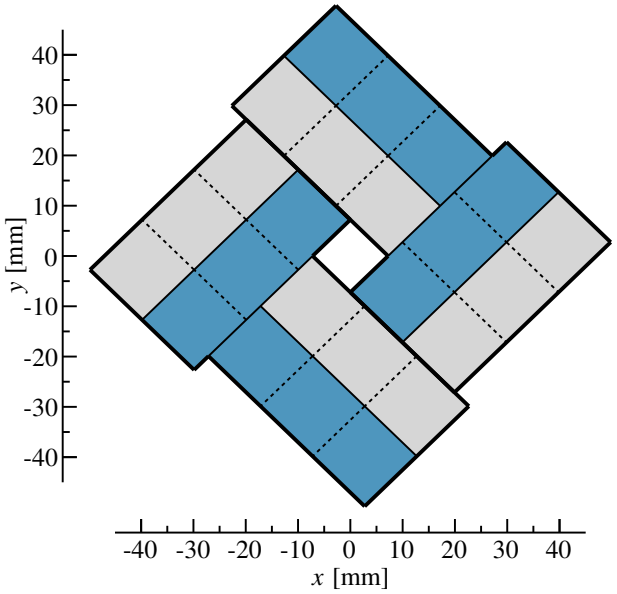
Outlook

- Until now only 1% of the VELO sensor is used for the counters
- Implemented new set of 624 configurable counters that can cover up to the whole silicon sensor



Outlook

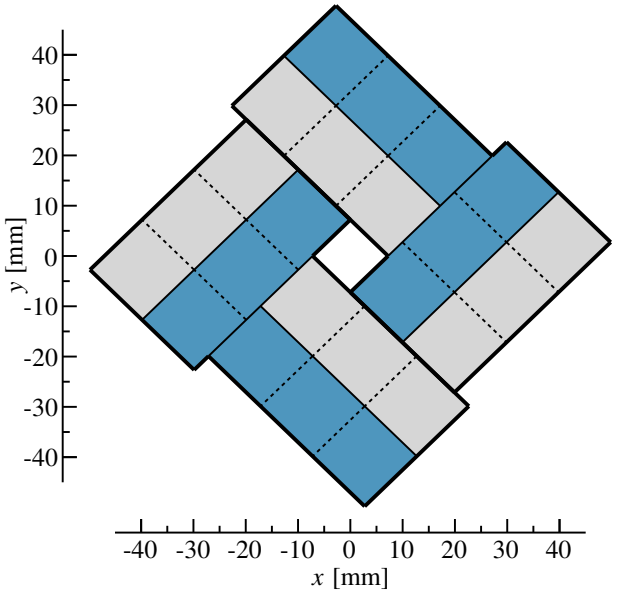
- Until now only 1% of the VELO sensor is used for the counters
- Implemented new set of 624 configurable counters that can cover up to the whole silicon sensor



Resolution of position estimators expected to grow by a factor 3

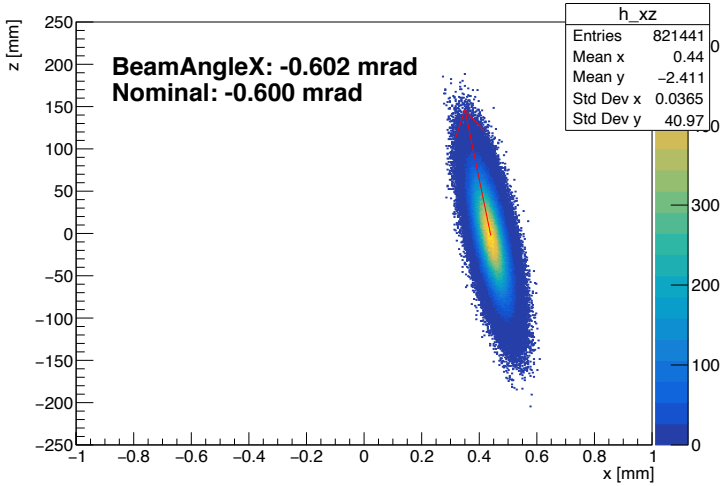
Outlook

- Until now only 1% of the VELO sensor is used for the counters
- Implemented new set of 624 configurable counters that can cover up to the whole silicon sensor



Resolution of position estimators expected to grow by a factor 3

New quantities to estimate: inclination of the beam spot in both the *xz* and *yz* planes



➢ **Goal**: construct a tool for measuring all beam spot parameters, including spread in the three directions

New estimators tested in MC and await to be validated on data

Summary and results

- Counters of clusters were implemented in the VELO firmware
- An appropriate counters combination provides a real-time estimate of useful quantities
  - The trimmed mean luminosity estimator has a relative statistical precision of 0.27%
  - The PCA method is used to combine counters to measure beam spot and VELO halves position
  - For the first time, a measurement of the beam spot position is performed **without any help from tracks**, with a resolution of 4 μm / 18 ms
  - VELO halves position are also measured in real-time, with a resolution ranging from 4 μm to 7 μm

➢Take home message

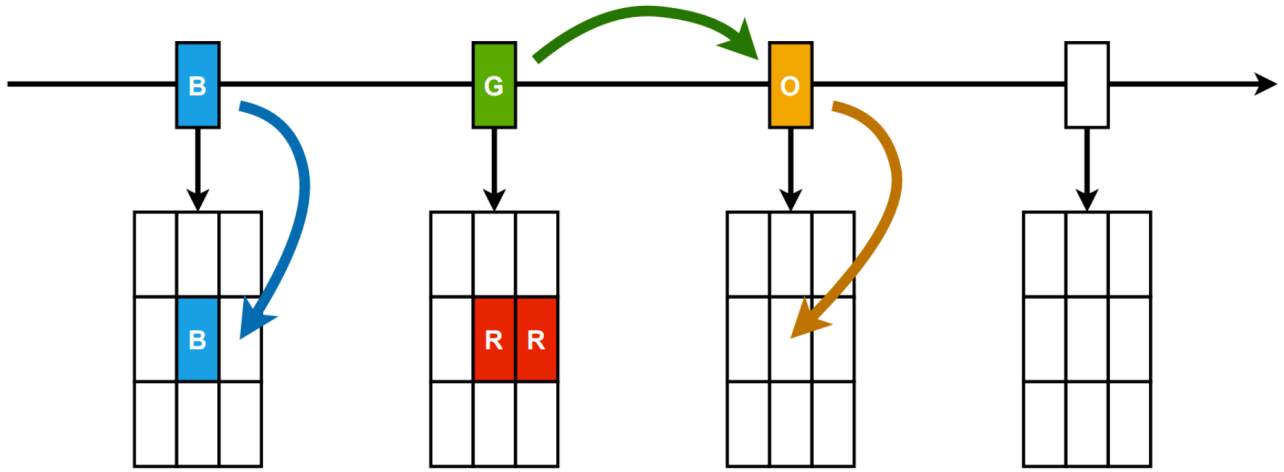
- High-rate primitives produced at low level from a vertex detector should be exploited in view of bandwidth challenges in future runs
- Here, only quantities that could be constructed with the available resources were explored, but the potential of this kind of approach is very wide and can be leveraged

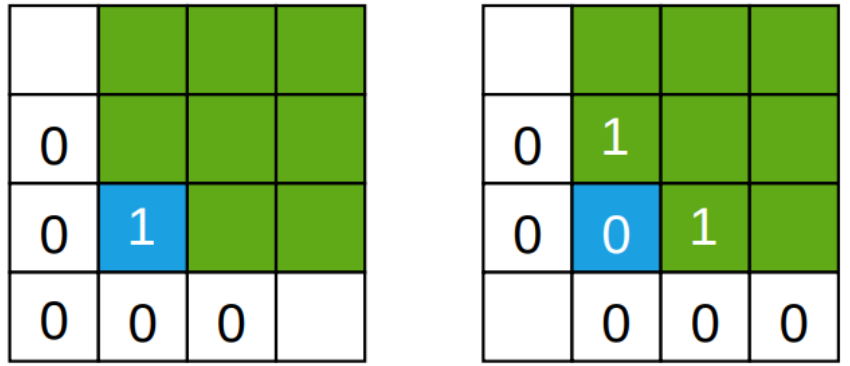
Backup slides

How does the clustering algorithm operate?

- The readout format of the VELO are SuperPixels (SPs) → 4 x 2 contiguous pixels
- Clusters for isolated SPs are resolved with Look Up Tables
- Clusters for non-isolated SPs are resolved with:

Matrix filling for identifying cluster candidates → Cluster recognition through pattern matching



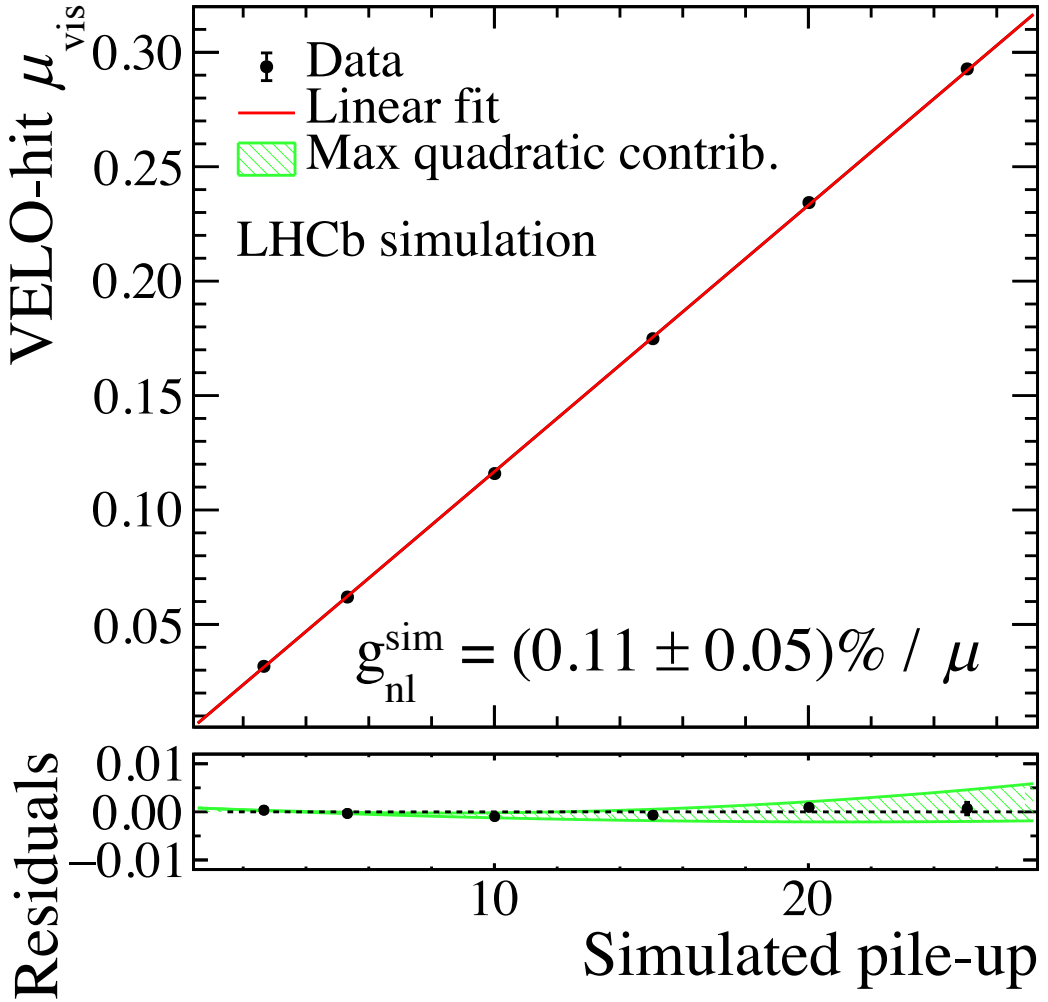


The cluster counters are linear in luminosity

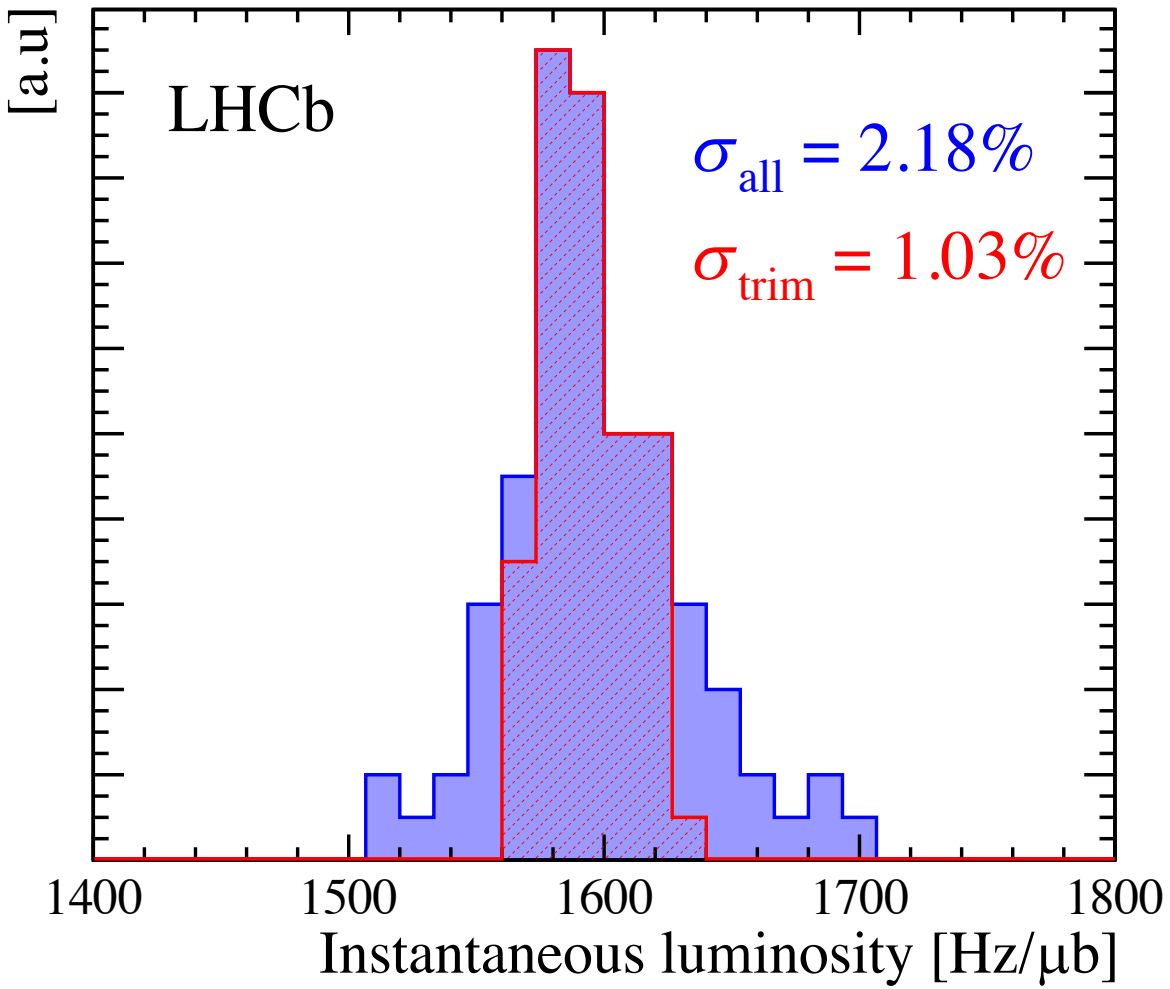
- Luminosity counters must be linear in the pile-up μ of LHCb

$$L = \frac{\mu_{vis}}{\sigma_{vis}} f = \frac{\mu}{\sigma_{pp@LHCb}} f$$

- The VELO cluster counters behave correctly scaling linearly with the mean number of interaction per bunch crossing μ

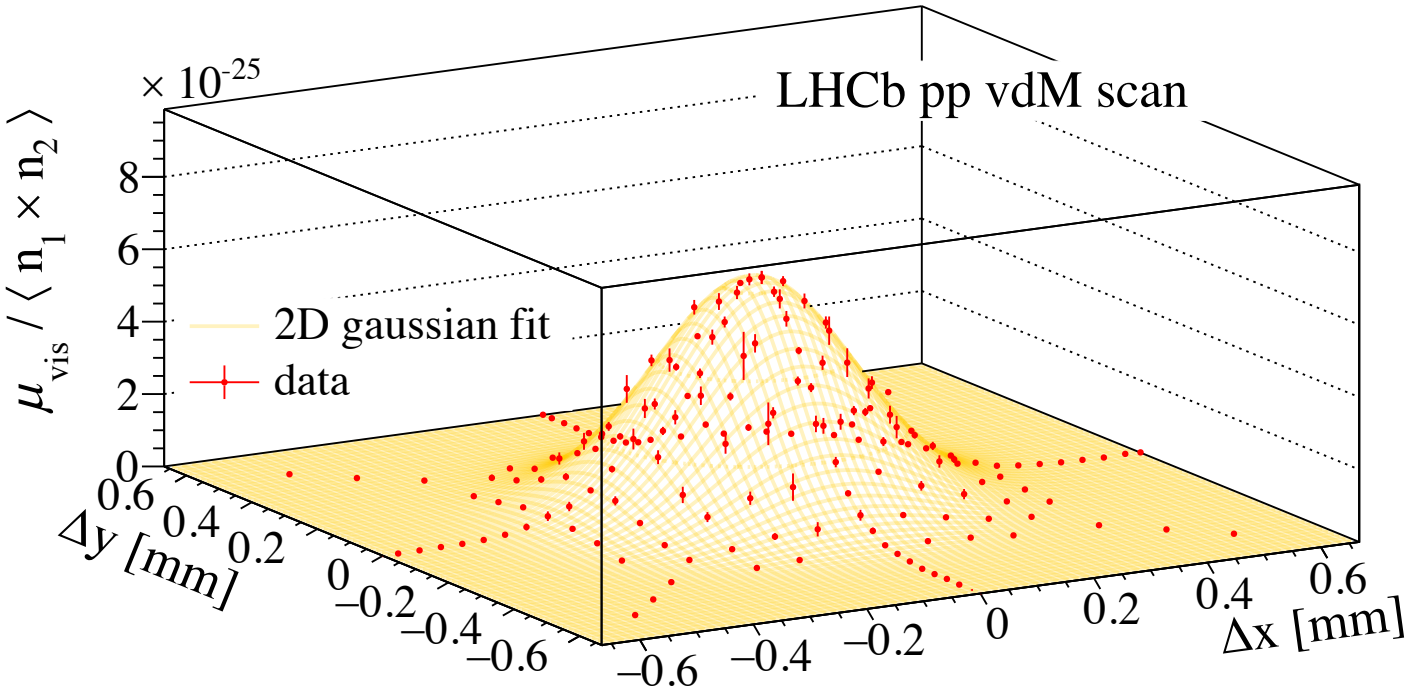


How are the different luminosity measurement distributed?



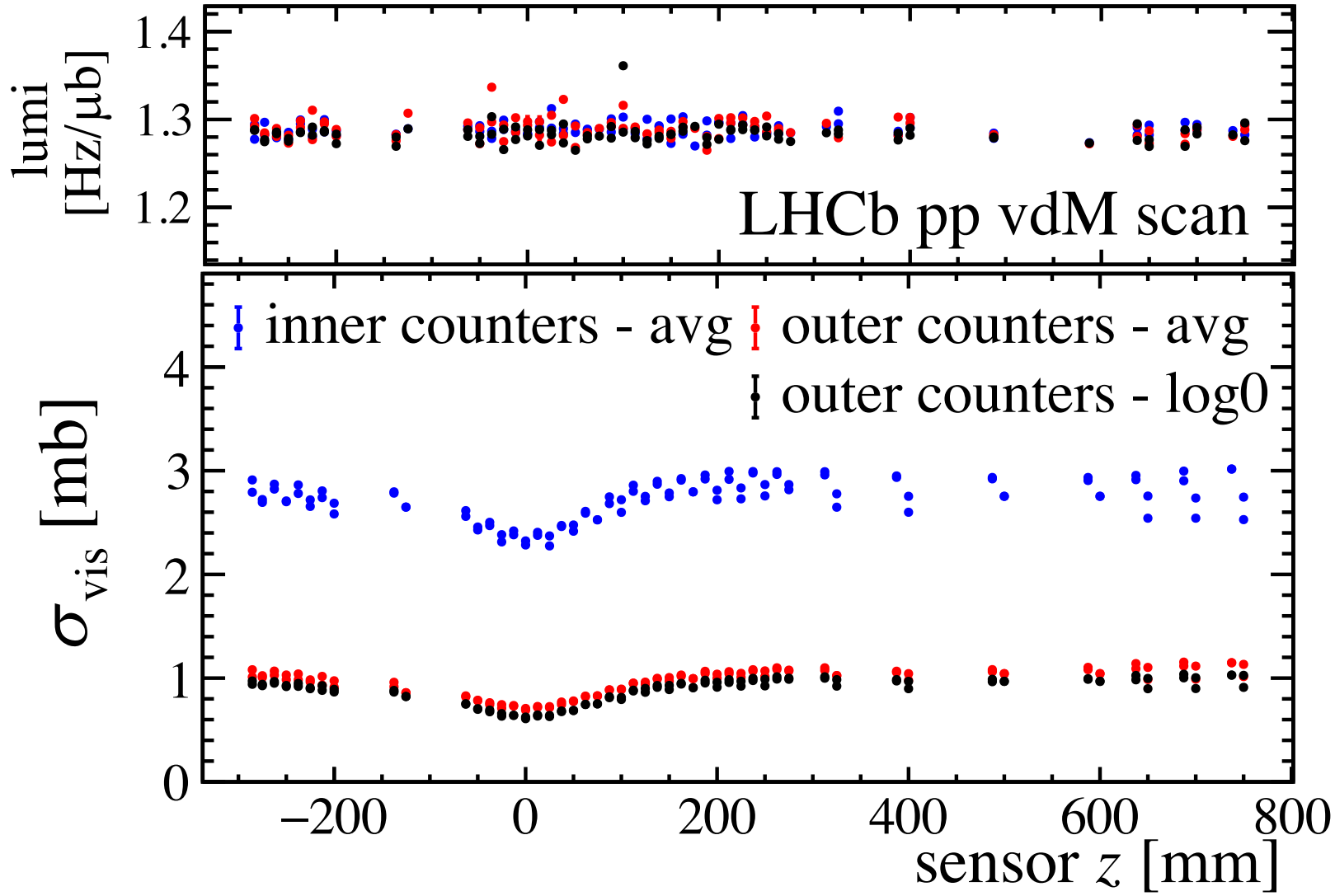
A bivariate Gaussian is fitted to the data to evaluate the integral, allowing to calculate σ_{vis}

- Assuming that the distribution for $\mu_{vis}(\Delta x, \Delta y)$ is a bivariate gaussian, its integral can be computed with a chi-square fit
- $\sigma_{vis} = 2\pi A \sigma_x \sigma_y \sqrt{1-\rho^2}$
- Calculation done for every available counter



➢ **Each counter provides a different luminosity measurement: need to combine 208 measurements in a single estimator**

Cross section of each counter



“Explainability” of the coefficients estimated with the PCA: asymmetry between up-down and left-right

“Explainability” of the coefficients estimated with the PCA: asymmetry between down-up and right-left