VeloPix in LHCb software: Reconstruction

or making tracks at $2 \times 10^{33} cm^{-2} s^{-1}$ with the VeloPix

Victor Coco

Syracuse University

April 28, 2010

VeloPix 2D hits and measurements

Or what the LHCb software knows about VeloPix hit

For Pattern Recognition

- Need 2D (+z) measurments to fully exploit the point-like aspect (avoid ambiguites).
- For the moment pattern recognition is cheated (see Laurence talk) $\rightarrow$ simplified hit management.
- Hit2D class, hitmanager for VeloPix etc. skeltons ready, specific implementation will be defined by pattern recognition.

For Track fitting

- Pixel measurment are intrinsically 2D (+z) but LHCb tracking framework only deals with 1D (+z) measurment.
- Native 2D (+z) will be implemented at one point by the tracking group in a longer term.
- For now we use 1D + 1D measurments (a la muon), assuming in first approximation that x and y errors are independant.

Requirements for having forward tracks

A little bit of technical stuff...

Functionnalites that were implemented

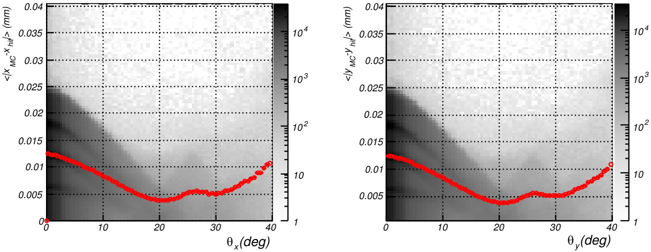
- Adaptation of the Detector Element to get fast access to pixel parameters.
- MC association, VeloPixLiteCluster $\rightarrow$ MCHit, VeloPixLiteCluster $\rightarrow$ MCParticle. Was needed both for cheated pattern recognition and monitoring.
- New `VeloPixLiteMeasurement` class, creating X and Y measurement out of the LiteCluster in a form recognized by the LHCb implementation of Kalman Filter.
- For Kalman filter projection step: projection from measurment space to track state space. Easier than for R/ϕ geometry since measurment space $\equiv$ track state space.
- For Kalman filter prediction step: evaluation of the errors measurments (details in next slide)

- A bunch of modified base classes to let know standard tools about VeloPix case
- Three new packages:
  - `VeloPix/VeloPixTools` that contains the error calculation tool.
  - `Rec/RecoUpgrade` that contains everything that does not requiers MC information (linear fit, pre-fitting track treatment for the different use case, etc...). Not VeloPix specific.
  - `Rec/RecoUpgradeMC` that contains everything that requiers MC information (track to MC Particles association, monitoring with MC, cheated pattern recognition, etc...). Not VeloPix specific.

Cluster errors parametrisation

The `VeloPixClusterPosition` tool

- Tool based on VeloClusterPostition (see. LHCb-2007-151): for a given angle between the track and the sensor, the tool return the error.
- Good agreement between digitization algorithm and test beam results with 300μm sensor (see. Velo Upgrade meeting 3rd Feb. 2010)
- Assuming it is still correct with 150μm sensor, we extract parametrisation (5th order polynom) of the cluster error as function of the angle from simulation to feed the tool.



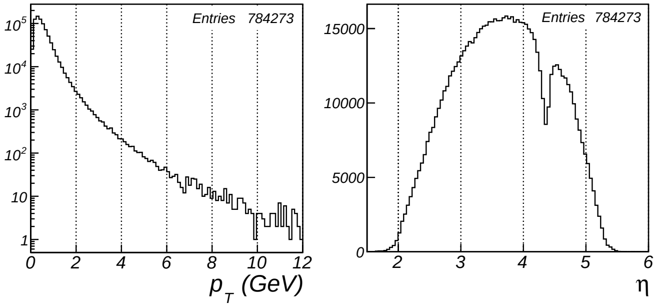
1. **RAW bank decoding**: Extract list of VeloPixLiteClusters from the RAW banks.
2. **Pattern recognition**: Get LiteClusters compatible with a track. *For the time being cheated using MC information*.
3. **Linear fit**: first evaluation of the track parameters, enough for *Velo standalone* and needed as *input of the kalman filter* (see Laurence talk today).
4. **Forward pattern recognition**: Seeding, Matching... this is standard, mainly reuse existing tools.
5. **Forward fit**: *Fit the track using Kalman filter*.

Event content

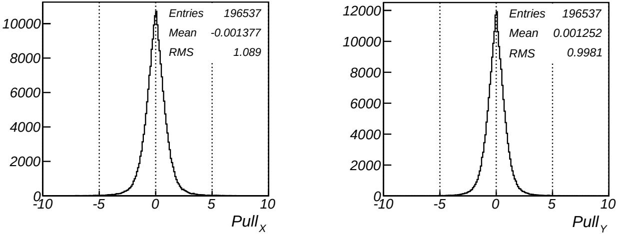
Simple variables to get an idea of what is reconstructed...

For the following, keep in mind:

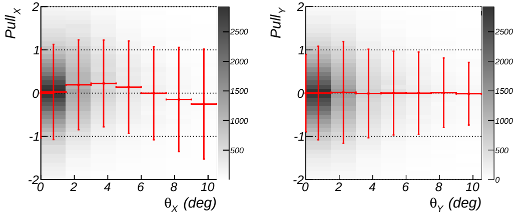
- Geometry used is rather "unrealistic" and untunned (10 sensor U shape with the central ladder shifted in... production was made for electronic studies)
- First pixel in X and Y not at the same distance.
- Z position and transverse shape not optimal to match with forward acceptance (reason for gap arround 20mrad?)
- Because of overlap between RFFoil and guard ring of the central ladder there might be a underestimastion of material effect for tracks in horizontal plane.
- ...



- Pull distribution in X and Y are well centered on 0 and have a RMS ~ 1.

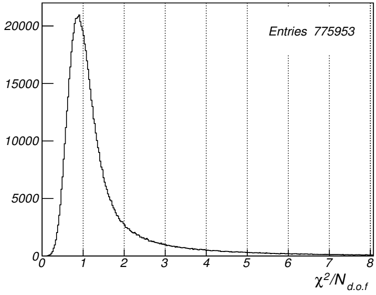


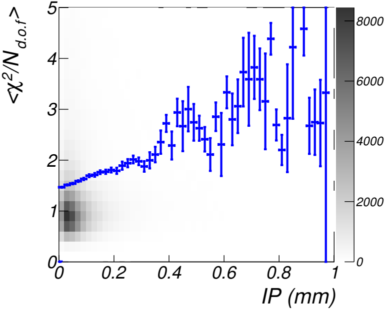
- Pull vs. θ shows small non linearity in X and possible oversestimate of errors in Y (In red central value is Mean of Pull per θ bin and error bar is RMS).



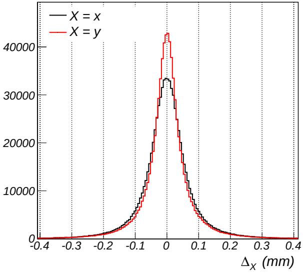
- Looks ok in first approximation, will be reviewed with strawman implementation.

- $\chi^2/N_{d.o.f}$ peaks close to 1. Might need to tune a bit the error evaluation or can be biased by the fact that each VeloPix measurements are counted twice while they already contains information on the second direction.
- As a crosscheck $<\chi^2/N_{d.o.f}>$ is well correlated with the IP of the track w.r.t to it's production point (known by MC).

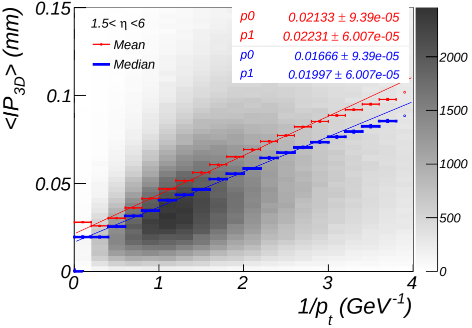




- From first track state, the track is extrapolate to $z_{production}$
- Difference between true and extrapolated x, y positions in this plane (Δ_X, Δ_Y)
- Difference between true and track state measured slope (Δ_{T_x}, Δ_{T_y})



- Impact parameter between the production point and track is computed.
- Since distribution per bins of $1/p_T$ has large tails, both mean and median (that is less sensitive to tails) are mentionned.



- It varies a lot as function of η range (see backups)
  - $\eta \in [3.5; 4]$: $< |IP_{3D}| >\sim 17(\mu m) + 19(\mu m)/pt(GeV)$,
  - $\eta \in [4.5; 5]$: $< |IP_{3D}| >\sim 47(\mu m) + 20(\mu m)/pt(GeV)$

Short term

Toward a Brunel release with VeloPix tracking support

What's done

All classes and packages needed for having **VeloPix standalone tracking** are commited (ie. can be used from nightly). Configurables need a simple hack to be used (see. VeloPix Simulations Twiki)

- All `LHCb` basic classes modifications are released within `LHCb v30r1`
- All `Rec` basic classes modifications will be released in `Rec v9r2`
- New packages will be released in `Rec v9r2` and `Lbcom v9r2`

What's need to be done

- All classes and packages needed for having **VeloPix in full tracking** are written.
- They are still to be commited (still some packaging work to do).
- Configurables needs some more work to get correct sequence out of the box.

If full tracking and configurables are not in time for `v9r2` releases they will be available in the `lhcb_head` nightly with instruction of how to run in VeloPix Simulations Twiki.

Longer term

Toward a full simulation of VeloPix

Pattern recognition

- The base classes to deal with 2D hits are not commited but skeletons are ready.
- A basic (not CPU time friendly) algorithm might be written as reference.
- If you want to test a tricky idea of fast pattern recognition... feel free! The framework is there, support too.

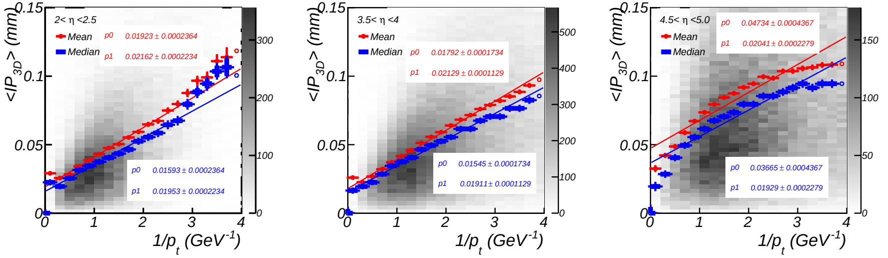
Track fit

- Update errors estimation and parametrisation.
- Write some monitoring to easily compare the different options.
- Tune the track fit (use of full/simplified geometry, number of outliers, use of material correction, etc...)
- If you want to test a tricky idea of improved fitting... fell free! The framework is there, support too.

Trigger

- In longer term impact of VeloPix on trigger is an important feature.
- If you want to test a tricky idea of fast and efficient triggering... feel free! The framework is there, support too.

BACKUP



- $\eta \in [2; 2.5]$: $< |IP_{3D}| >\sim 19(\mu m) + 21(\mu m)/pt(GeV)$,
- $\eta \in [3.5; 4]$: $< |IP_{3D}| >\sim 17(\mu m) + 19(\mu m)/pt(GeV)$,
- $\eta \in [4.5; 5]$: $< |IP_{3D}| >\sim 47(\mu m) + 20(\mu m)/pt(GeV)$

- $\eta \in [2; 2.5]$: $< |IP_{3D}| >\sim 16(\mu m) + 19(\mu m)/pt(GeV)$,
- $\eta \in [3.5; 4]$: $< |IP_{3D}| >\sim 15(\mu m) + 19(\mu m)/pt(GeV)$,
- $\eta \in [4.5; 5]$: $< |IP_{3D}| >\sim 37(\mu m) + 19(\mu m)/pt(GeV)$