VeloPix Pattern Recognition and Linear Track Fit

Laurence Carson

University of Santiago de Compostela

Velo Upgrade Meeting,
CERN, 28 April 2010

Overview

- Cheated Pattern Recognition
- Linear Track Fit
- Real Pattern Recognition

Cheated Pattern Recognition

To allow first studies of IP resolution etc., and to create framework for real pattern recognition, an algorithm in Brunel to do cheated patrecog for VeloPix is needed.

The cheated patrecog itself simply involves getting the list of LHCb IDs corresponding to each MC particle. This list can then be used as the input to a track fit in the same way as a list found using real patrecog.

The cheated patrecog for VeloPix is implemented in CheatedVeloPixPat, which can be found in the Rec/RecoUpgradeMC package. The initial output of the cheated patrecog is a track object where the track states have the MC truth values for the x,y and z, and dummy values for the tx and ty.

Linear Track Fit

This "baby track" is passed to a linear fitter (PatVeloPixFitLHCbIDs in Rec/RecoUpgrade), which updates the tx and ty values (and the covariance matrix) according to the result of the fit.

To take multiple scattering into account in the linear fit, each hit is assigned a weight as follows:

$$weight = \frac{1}{hit\ variance + MSError},$$

where MSError is zero for the first measurement, and is then incremented by a fixed value[1] of stepError with each following hit.

Currently the stepError is set to the value used in the RZ fit in PatVeloSpaceTracking, which is 0.002 mm^2. As the typical variance is $\simeq$0.00015 mm^2, the contribution from the first hit is dominant. Studies of the IP pulls can be used to tune stepError.

[1] Momentum-dependent formula not used as no momentum estimate exists here.

Linear Track Fit

A standard least squares linear fit is then made with these weights, i.e. minimising the weighted χ^2

$$\sum_i w_i * r_i^2,$$

where w_i are the weights and r_i the residuals. The minimisation can be done analytically since the fit is linear.
The results of the linear fit give the track slopes in x and y, and also the intercepts, which are used to calculate the IP.

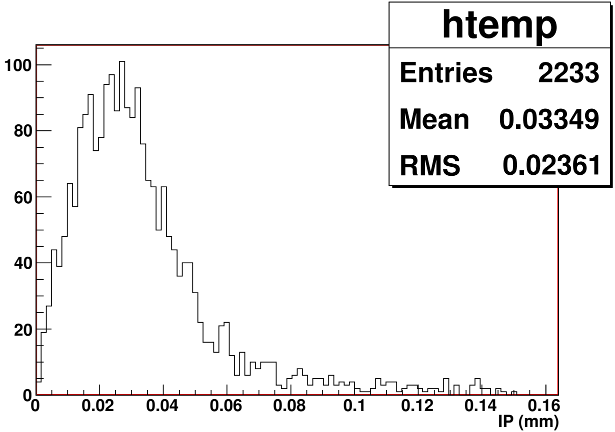
Currently the results for the slope and intercept values seem reasonable (see next slides), however the expressions for the errors on the slope and intercept still need some corrections.

Dataset

Use minimum bias events created with MUL (DDDB tag velopix-mul-20091116). Can be found under /castor/cern.ch/grid/lhcb/MC/Upgrade/DIGI/.

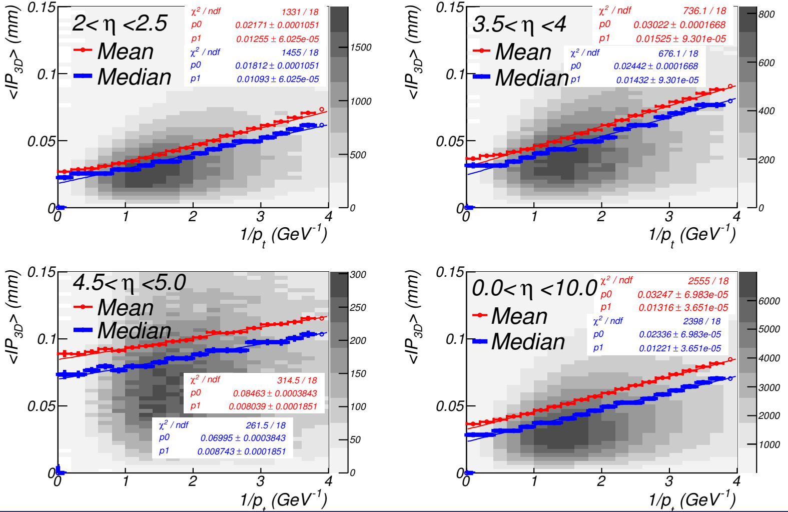
Run over 20,000 events and perform linear fit on VeloPix tracks. Make NTuple of relevant quantities (position, IP, slopes etc.), including reconstructed values and true values.

IP Distribution for $p_T > 5$ GeV



Cut is made on true p_T value. No cut made on η for this plot.

IP Resolution as a Function of p_T^{-1} in Different η Ranges

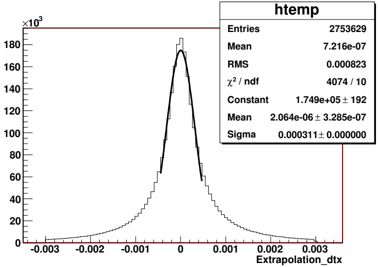


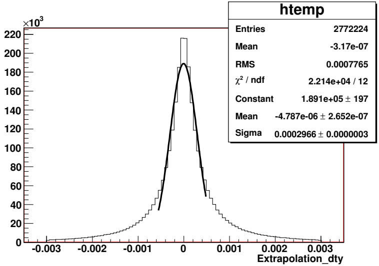
IP Resolution Summary

IP resolution seems to be within expectations. Resolution behaves linearly with p_T^{-1} and is better at low η.

η Range	IP resolution (μm) at high p_T
[2.0,2.5]	21.7
[3.5,4.0]	30.2
[4.5,5.0]	84.6
All	32.5

Resolution on Slopes





Core width for slope resolution is $\simeq 3 * 10^{-4}$. RMS of distribution is several times higher. Double Gaussian fit could be used to widen fit range.

Real Pattern Recognition

Long-term goal is to produce an algorithm for patrecog in VeloPix without the cheating.

Clearly this algorithm will be very different from the one used in the current Velo as e.g. will no longer have two-step R and Φ patrecog.

Where to find ideas from?

- Look at patrecog from BTeV, as geometry similar to VeloPix.
- GPDs use pixels, but their patrecog unlikely to be too much use due to different geometry of layers (not at fixed z).
- Other suggestions very welcome!

Conclusions and Next Steps

- Algorithm now exists to perform cheated pattern recognition and linear track fit for VeloPix in Brunel.
- Performance was tested using min. bias events in MUL.
- IP resolution (η-averaged) for high p_T tracks is $\simeq$30 μm.
- Resolution on track slopes is $\simeq 3 * 10^{-4}$ (core Gaussian).
- Calculation of errors from linear fit needs to be correctly implemented.
- Long term goal is real pattern recognition.