SVD hits and alignment

Peter Kvasnicka
Peter.Kvasnicka@mff.cuni.cz

Charles University in Prague

8th Belle II VXD workshop
Trieste, 9-11 September 2015

Outline

Current activities in SVD simulation and reconstruction

Refactoring of SVD digitization and clustering

- Introduce new SVDDigit format
  - The new SVDDigits carry the full set (3 or 6) APV25 samples.
  - Clustering becomes trivial
  - Code gets simpler and data smaller.
- Remove u/v duplicity in the code
  - Define u- and v- projections of SVD::SensorInfo
  - Makes shorter code
  - Improves maintainability
- Improve waveform fitter
  - The fitter is likely to eat up any speed improvements of the re-factoring.
  - Decrease time estimate bias
  - A rough fitter is OK, provided it is robust and allows error estimation.
  - Ability to detect time difference is more important than time measurement.
  - There is currently no support for waveform handling.

Pending tasks in SVD reconstruction

Waiting to be started

See Andrzej's talk for more, this is more a personal selection

- Combining u- and v- hits
  - This requires some trial-and-error work
  - How to resolve/pass on ambiguities
  - How to resolve overlaps
- Handling reconstruction bias and uncertainties
  - There is an effort by Peter KodyÅ¡ to tabulate things by clsuter observables and track direction
  - Also applies to Lorentz shifts
- Handling large clusters
  - u-/v- combination can help to merge torn clusters
  - Trial and error: difficult to define a cost-efficient extent of the task beforehand.
- Digitization
  - We have a symbolic model of current generation in strips: can we do better?
  - At least, we need to calibrate the simulation properly (capacitances, diffusivity etc.)
  - Giacomo Caria undertook the calibration effort forcefully.

SVD alignment and misalignment

There was a general VXD alignment talk by Tadeas yesterday

- Tadeas covered the current status of the VXD alignment task.
- I will only talk about SVD-specific issues and efforts in alignment and misalignment.

Alignment in VXD

- In VXD, alignment comes into play when we construct tracks from hits on sensors, that is, when we want to transform hit coordinates from sensor coordinates to laboratory frame.
  - Up to SVDClusters (= reconstructed hits), all simulation and reconstruction uses local sensor coordinates (with the exception of local magnetic field values)
  - The local-to-global transformation is used to construct RecoHits.
  - RecoHits are non-persistent objects used in tracking. They combine hit data with information about the plane where the hit is sitting.
- Alignment corrections are applied to RecoHits: they update the position of the RecoHit's sensor plane. It uses data to correct geometry information.
- Misalignment is applied to RecoHits: it changes hit data in the RecoHit. It is a simulation-based procedure using MC track information from TrueHits.
  - In this talk, misalignment is a specific procedure to facilitate alignment studies, and not a general lack of alignment.

Alignment in the SVD

- The alignment people use renowned algorithms to find alignment corrections based on sufficiently large samples of tracks.
- In this sense, SVD does not need its own alignment, though it would be prudent to look at specific alignment problems in the SVD.
- However, there are subdetector-dependent information that SVD must define for alignment to work optimally for the SVD.
  - Alignables - properties of structures that affect position of active parts of sensors (in particular, design tolerances of SVD structure, deformability of sensors and support structures etc.)
  - Structural hierarchy helps to define optimal parametrisation of alignment corrections. It has to be different from the half-shell/layer/ladder/sensor scheme and must reflect actual mechanical design of SVD.

Misalignment

Misalignment is an MC-truth based procedure used to economize full-simulation requirements of alignment studies.

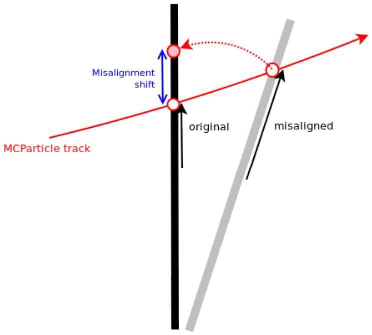


Figure : Shift measurements to create the effect of a different geometry.

The procedure:

- Use a different geometry in simulation and reconstruct in nominal geometry to test alignment procedures.
- For economy, it is convenient to repeatedly distort a simulation in nominal geometry to mimic effects of misaligned geometries.
  - Simulate in nominal geometry
  - Shift hits on sensors to imitate a different geometry
- To apply misalignment, we need to:
  - Calculate the correction to 3D transformation for the sensor plane.
  - Calculate the shift of the cluster position (this requires MC information as it depends on track direction)

Sensor deformation

The mechanism of applying (mis-)alignment to RecoHits is used to implement sensor deformations. Sensor deformations are not implemented in geometry, they are implemented as corrections in the RecoHits.

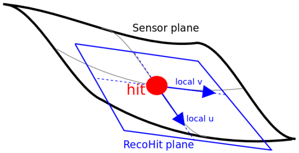


Figure : RecoHit plane is tangent to the deformed sensor.

The RecoHit

- The RecoHit carries information about (local) hit position in a RecoHit plane (SensorPlane).
- Trackers ask RecoHit about the position of the hit, giving track status (intersection with RecoHit plane, direction etc.) with the query.
- We can therefore use the RecoHits to apply a variety of position- and track-dependent corrections.

Deformed sensors

- For a properly parameterized sensor plane deformation, we construct the RecoHit plane as a tangent plane to the deformed sensor plane at the hit position.
- Similarly, we use a position-dependent misalignment transoformation in misalignment to shift hits on a deformed sensor plane.

RecoHit corrections

RecoHits allow to apply a variety of position- and track-dependent corrections. Here are some of them:

RecoHit corrections

- Coordinate directions in slanted sensors: position-dependent
- Hit reconstruction biases and hit position errors: track direction- and position-dependent
- Lorentz shifts (position-dependent)
- Misalignment shifts (position-dependent on deformed sensors, MC-track dependent)
- Alignment corrections (position-dependent on deformed sensors)

What information is needed for SVD (mis-)alignment

The following information is needed for SVD (mis-)alignment

- Degrees of freedom of the structure: which constraints define the position of an SVD sensor in space and what are there tolerances?
- For which coordinates will there be an initial measurement (such as shifts of glued parts etc.)
- Which components are deformable?
  - Sensors, we know
  - Other parts? Ribs?
- For deformable components, what are deformation constraints? Supports, glued edges etc.

From this information, we can:

- define a proper structural hierarchy and parameterise the alignment
- properly parameterize structural deformations
- design realistic alignment studies.

Towards the 2016 VXD testbeam

General testbeam tasks

- The 2016 VXD testbeam at DESY is intended to be most of all a system test.
- Our most important task is to ensure that the SVD data processing chain works: that we can correctly acquire and process data.
- We also want to see that the DQM tools provide relevant information on data quality in reasonable time.

More detailed

Things that must be improved compared to the 2014 testbeam:

Logbook and data handling This needs some consideration before the test, to reduce manual intervention/failure to reasonable amounts.

Configuration data We need a database solution working, to handle run-dependent configuration data (magnetic field, beam parameters, alignment etc.)

DQM Some work is needed to be done on DQM plots. They proved to be very useful in the previous beam test, and we will want them to be better, in particular for initial stages of the beam test.

Conclusions

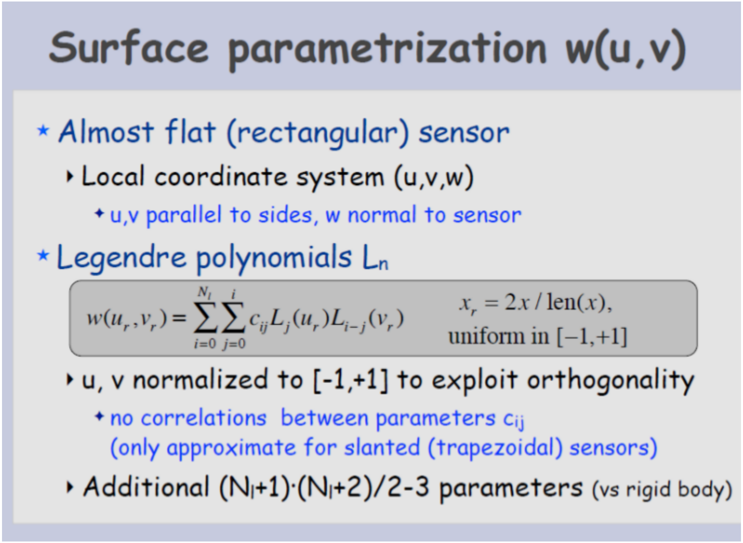
- SVD software now undergoes re-factoring, including change of SVDDigit format and waveform fitter.
- SVD alignment is basically working (Tadeas), but some input is required from SVD mechanics people to optimize it.
- SVD calibration effort has been started by Giacomo Caria.
- Data handling and DQM are the most important software tasks for the upcoming VXD testbeam.

Thank you for attention

Questions? Comments?

Background material

Example of a surface parametrisation



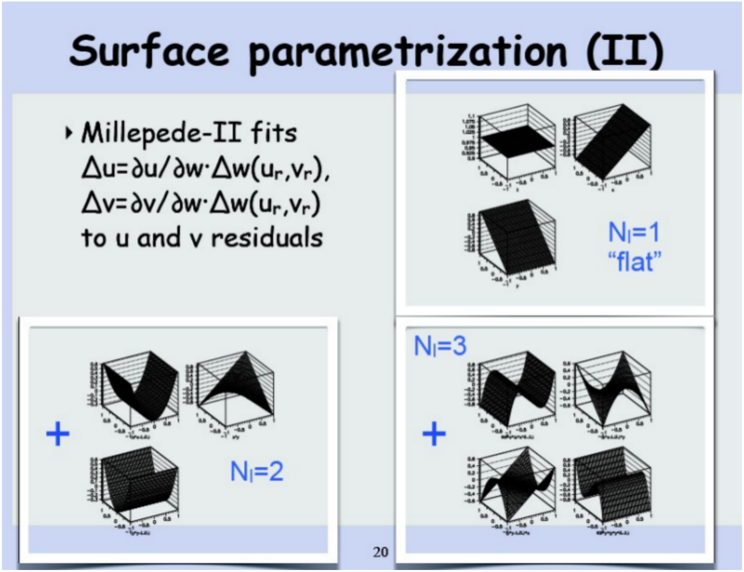
Surface parametrization w(u,v)

- Almost flat (rectangular) sensor
  - Local coordinate system (u,v,w)
    - u,v parallel to sides, w normal to sensor
- Legendre polynomials L_n

$$w(u_r,v_r) = \sum_{i=0}^{N_l}\sum_{j=0}^{i} c_{ij} L_j(u_r) L_{i-j}(v_r) \qquad x_r = 2x/\mathrm{len}(x), \text{ uniform in } [-1,+1]$$

  - u, v normalized to [-1,+1] to exploit orthogonality
    - no correlations between parameters c_{ij} (only approximate for slanted (trapezoidal) sensors)
  - Additional $(N_l+1)\cdot(N_l+2)/2-3$ parameters (vs rigid body)

Example of a surface parametrisation



Surface parametrization (II)

- Millepede-II fits $\Delta u = \partial u/\partial w \cdot \Delta w(u_r, v_r)$, $\Delta v = \partial v/\partial w \cdot \Delta w(u_r, v_r)$ to u and v residuals

$N_l=1$ "flat"

+ $N_l=2$

$N_l=3$ +

20