

Software Tools for the 4th Concept

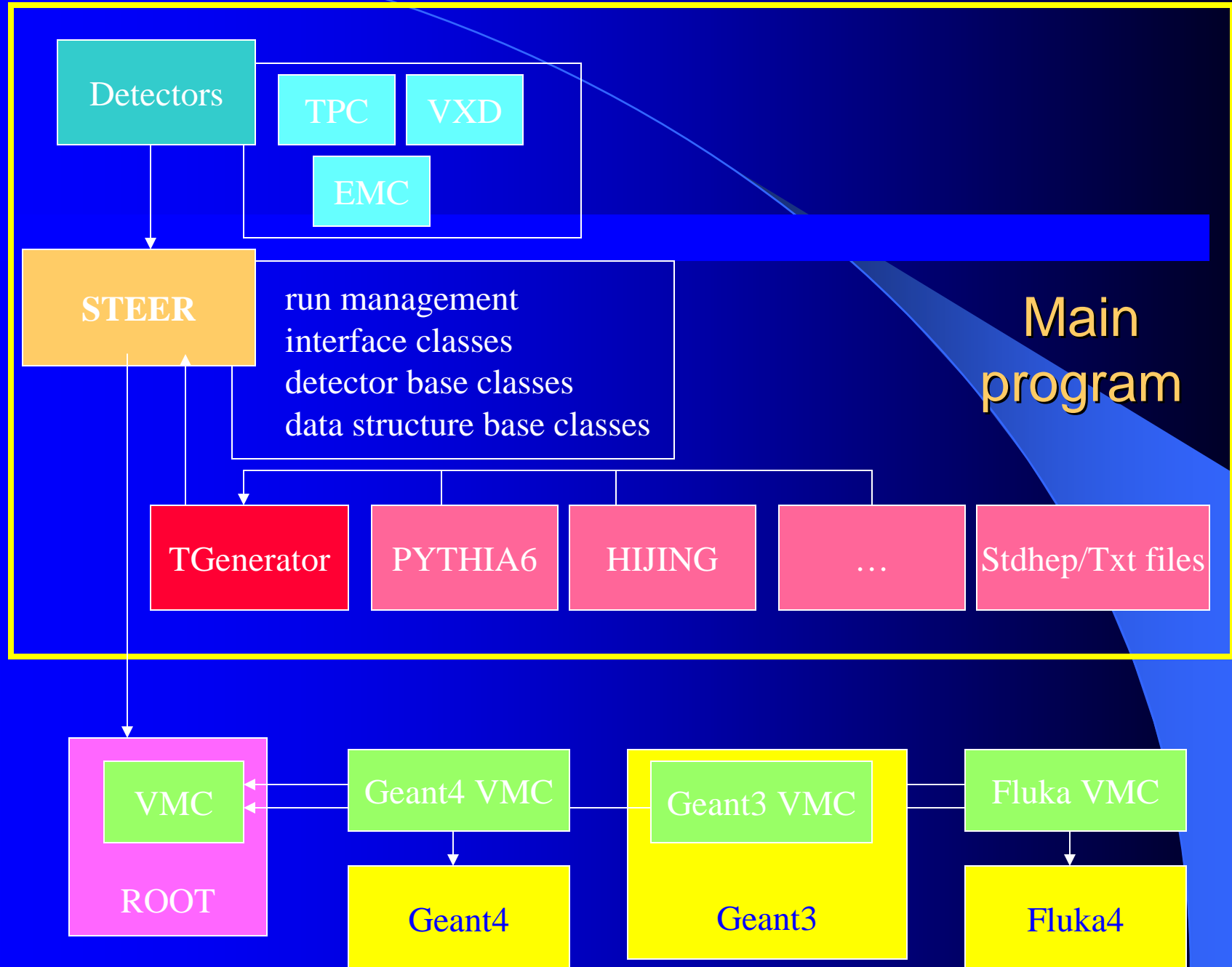
Outlook

- Aliroot/Ilcroot framework: a tool for the HEP community
- Digitization tools for Si based Detectors
- Track reconstruction in Ilcroot

Aliroot/Ilcroot Framework

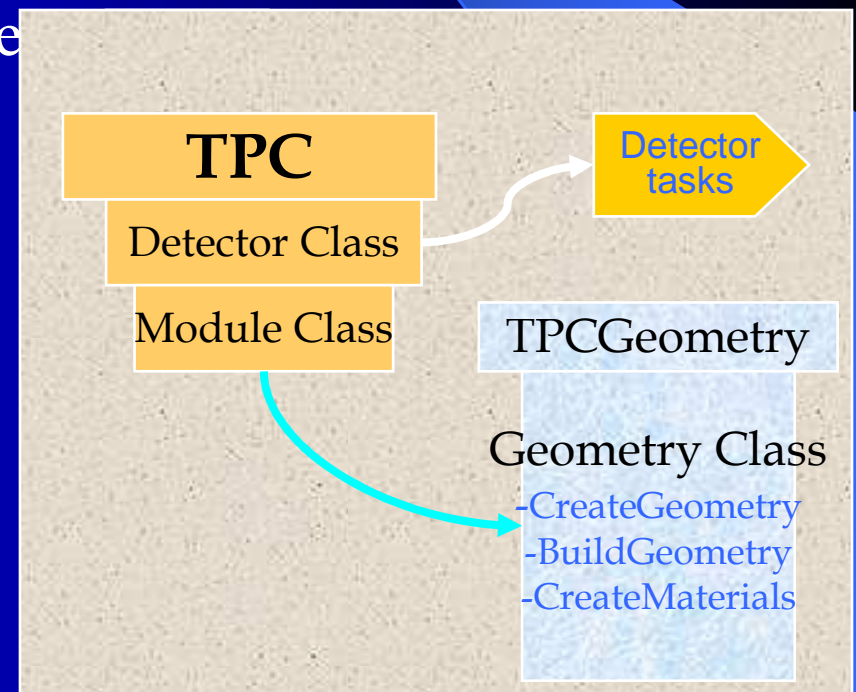
- Could it ever evolve into a general purpose entity for the HEP community (as ROOT)?
- Growing number of experiments have adopted it: Alice, Opera, CMB, (Meg), Panda, 4th Concept
- Main advantage: exchange modules among experiments
 - Generators (signal/background)
 - Detector simulation (MC step, digitization, P.R,...)
 - Software tools (vertexing, flavor tagging,)
- Other advantages:
 - Development and support

General Architecture




The Detector Class

- Base class for subdetectors modules.
- Both sensitive modules (detectors) and non-sensitive ones are described by this base class. This class
- supports the hit and digit trees produced by the simulation
- supports the objects produced by the reconstruction.
- This class is also responsible for building the geometry of the detectors and the event display.
- Several versions of the same detector are possible (choose at run time)



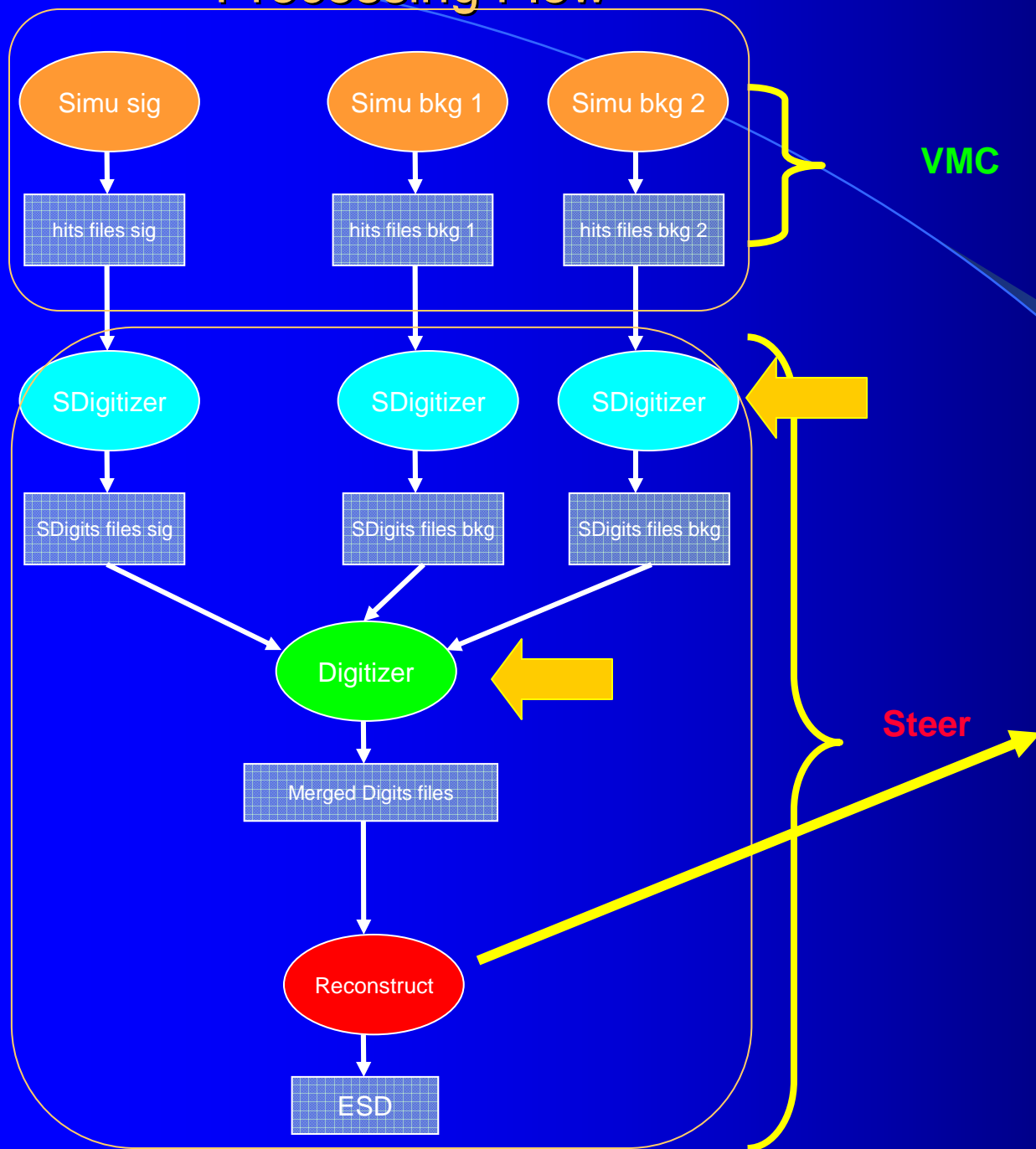
Is it feasible?

- Naked framework is already detector independent
 - Persistent object loaders (in highest efficient way)
 - Steering program
 - Generator interface
 - VMC interface
- Ilcroot adopted some of its code:
 - TPC, ECAL and Si from Aliroot
 - DCH from Megroot
- Ilcroot is growing along Aliroot
 - 1-2 code realinement/yr
 - Parametrize all the hard-coded stuff
- Preliminary discussion between Alice and 4th Concept for an *experiment-independent* framework (CERN – Dec. 2006)

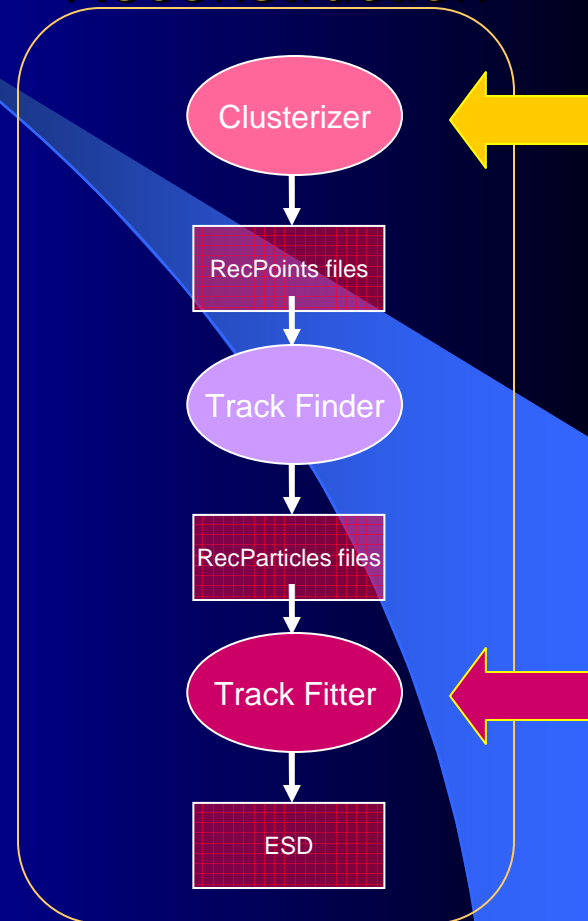
The background is a solid blue color with a subtle gradient. A thin, light blue curved line starts from the top left and sweeps across the upper right portion of the slide. On the right side, there is a vertical strip of a slightly different shade of blue.

Digitization and Clusterization of Si Detectors in Ilcroot

Processing Flow



Reconstruction



Technologies Implemented

- 3 detector species:
 - Silicon pixels (for VXD and FTD)
 - Silicon Strips (for Si Tracker and FTD)
 - Silicon Drift (not used at ILC)
- Pixel can have non constant size
- Strips can be stereo and on both sides
- Dead Regions are taken into account

SDigitization in Pixel Detector

- Get the Segmentation Model for each detector (from IlcVXDSegmentationSPD class)
- Get Calibration parameters (from IlcVXDCalibrationSPD class)
- Load background hits from file (if any)
- Loop over the hits and create a segment in Si in 3D
 - Step (from MC) along the line $>1 \mu\text{m}$ increments
 - Convert GeV to charge and get bias voltage:
$$el = dE*dt/3.6e-9 \quad dV = \text{thick}/\text{bias voltage}$$
 - Compute charge spreading:
$$\sigma_x = \sqrt{2k/e * T^\circ * dV * L}, \quad \sigma_z = fda * \sigma_{xy}$$
 - Spread charge across pixels using $\text{Erfc}(x, z, \sigma_x, \sigma_{xy})$
 - Charge pile-up is automatically taken into account

SDigitization in Pixels (cont'd)

- Add couplig effect between nearby pixels row-wise and column-swise (constant probability)
- Remove dead pixels (use signal map)

Digitization in Pixels

- Load SDigits from several files (signal or background)
- Merge signals belonging to the same channel
 - Non-linearity effects
 - Threshold
 - Saturation
- Add electronic noise
- Save Digits over threshold

Clusterization in Pixel Detector

- Create a initial cluster from adjacent pixels (no for diagonal)
- Subdivide the previous cluster in smaller $N \times N$ clusters
- Reconstruct Recpoints ant error matrix from coordinate average of the cluster
- Kalman filter picks up the best Recpoints

The Parameters for the VXD

- Size Pixel X = $20\text{ }\mu\text{m}$
- Size Pixel Z = $20\text{ }\mu\text{m}$
- Eccentricity = 0.85 (fda)
- Bias voltage = 18 V volts
- cr = 0% (coupling probability for row)
- cc = 4.7% (coupling probability for column)
- threshold = 3000 Electrons
- electronics noise = 0
- $T^\circ = 300\text{ }^\circ\text{K}$

SDigitization in Strips Detector

- Get the Segmentation Model for each detector (from IlcVXDSegmentationSSD class)
- Get Calibration parameters (from IlcVXDCalibrationSSD class)
- Load background hits from file (if any)
- Loop on the hits and create a segment in Si in 3D
 - Step along the line in equal size increments
 - Compute Drift time to p-side and n-side:
$$\text{tdrift}[0] = (y + (\text{seg} \rightarrow \text{Dy}() * 1.0\text{E-}4) / 2) / \text{GetDriftVelocity}(0);$$
$$\text{tdrift}[1] = ((\text{seg} \rightarrow \text{Dy}() * 1.0\text{E-}4) / 2 - y) / \text{GetDriftVelocity}(1);$$
 - Compute diffusion constant:
$$\text{sigma}[k] = \text{TMath::Sqrt}(2 * \text{GetDiffConst}(k) * \text{tdrift}[k]);$$
 - integrate the diffusion gaussian from -3σ to 3σ
 - Charge pile-up is automatically taken into account

SDigitization in Strips (cont'd)

- Add electronic noise per each side separately
 - `// noise is gaussian`
 - `noise = (Double_t) gRandom->Gaus(0,res->GetNoiseP().At(ix));`
 -
 - `// need to calibrate noise`
 - `noise *= (Double_t) res->GetGainP(ix);`
 -
 - `// noise comes in ADC channels from the calibration database`
 - `// It needs to be converted back to electronVolts`
 - `noise /= res->GetDEvToADC(1.);`
- Add coupling effect between nearby strips
 - different contribution from left and right neighbours
 - Proportional to nearby signals
- Remove dead pixels (use signal map)
- Convert total charge into signal (ADC count)
 - `if(k==0) signal /= res->GetGainP(ix);`
 - `else signal /= res->GetGainN(ix);`
 -
 - `// signal is converted in unit of ADC`
 - `signal = res->GetDEvToADC(fMapA2->GetSignal(k,ix));`

Clusterization in Strip Detector

- Create a initial cluster from adjacent strips (no for diagonal)
- Separate into Overlapped Clusters
 - Look for through in the analog signal shape
 - Split signal of parent clusters among daughter clusters
- Intersect stereo strips to get Recpoints from CoG of signals (and error matrix)
- Kalman filter picks up the best Recpoints

The Parameters for the Strips

- Strip size (p, n)
- Stereo angle (p-> 7.5 mrad, n->25.5 mrad)
- Ionization Energy in Si = 3.62×10^{-9}
- Hole diffusion constant (= $11 \text{ cm}^2/\text{sec}$)
- Electron diffusion constant (= $30 \text{ cm}^2/\text{sec}$)
- $v_{\text{drift}}^{\text{P}} (= 0.86 \times 10^6 \text{ cm/sec})$, $v_{\text{drift}}^{\text{N}} (= 2.28 \times 10^6 \text{ cm/sec})$
- Calibration constants
 - Gain
 - ADC conversion (1 ADC unit = 2.16 KeV)
- Coupling probabilities between strips (p and n)
- σ of gaussian noise (p AND n)
- threshold

Track Fitting in ILCARoot

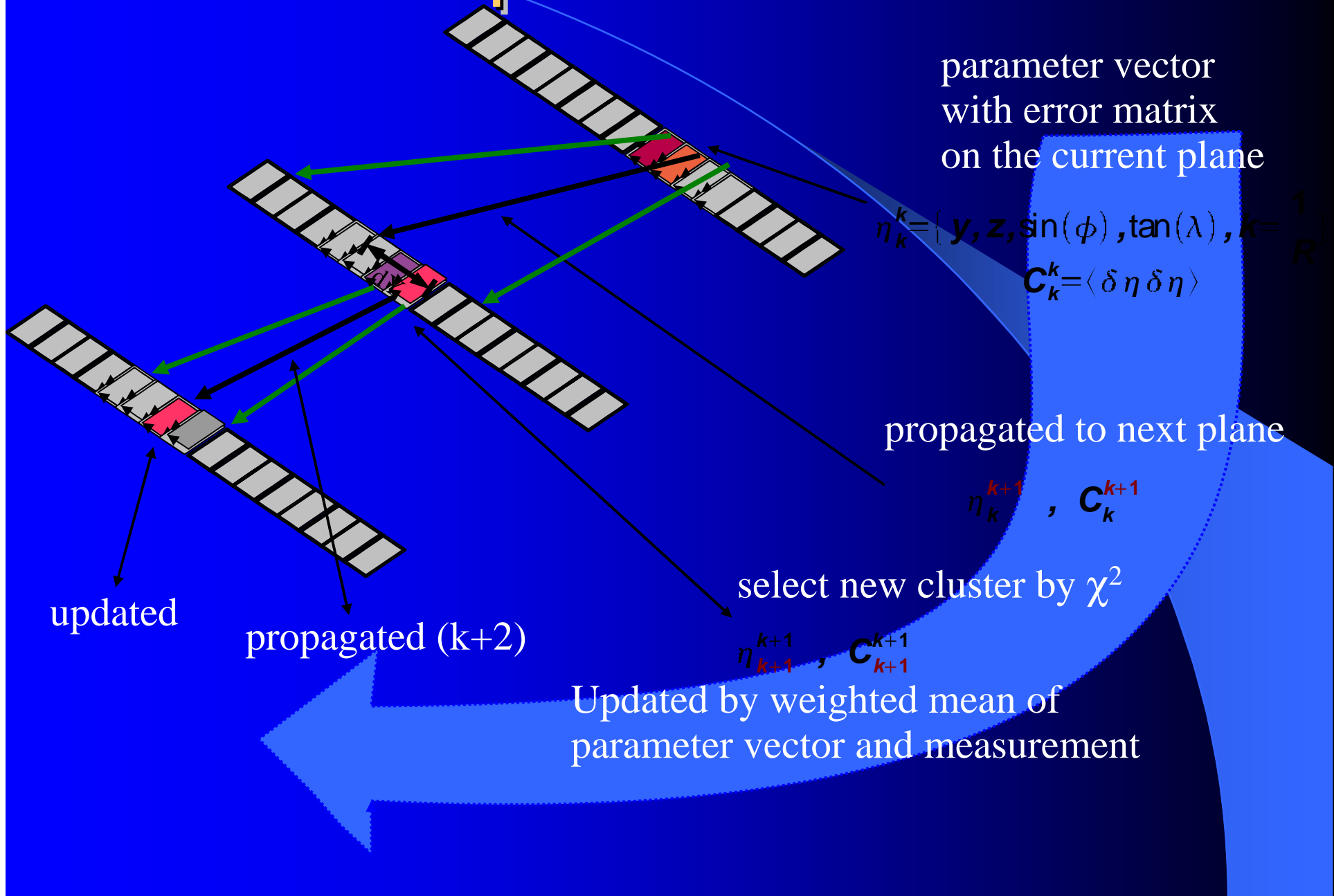
- Track finding and fitting is a global task: individual detectors collaborate
- It is performed after each detector has completed its local tasks (simulation, digitization, clusterization, etc.)
- It occurs in three phases:
 - Seeding in CT and fitting in VXD+CT+MUD
 - Standalone seeding and fitting in VXD
 - Standalone seeding and fitting in MUD

Not
implemented

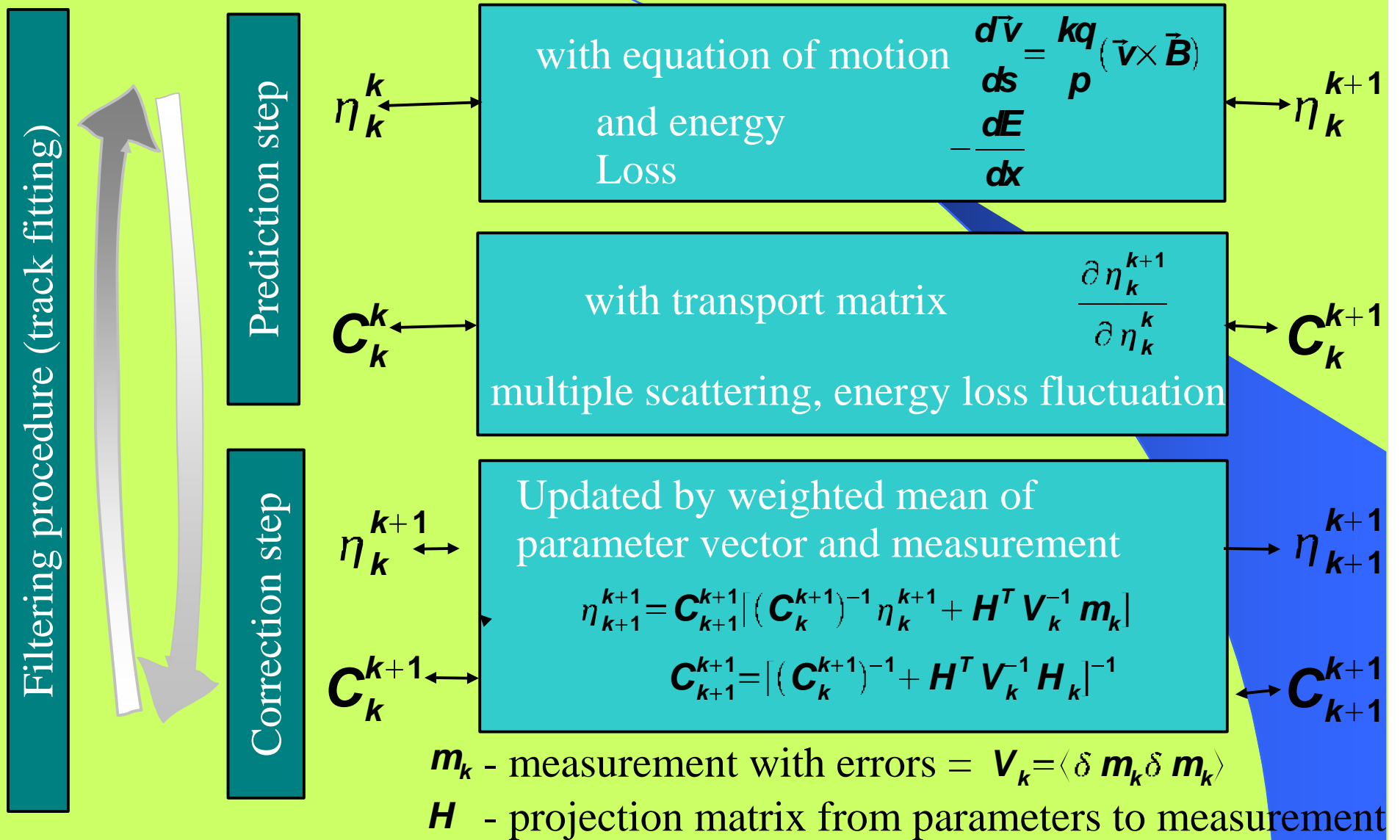
Kalman Filter (classic)

- Recursive least-squares estimation.
- Equivalent to global least-squares method including all correlations between measurements due to multiple scattering.
- Suitable for combined track finding and fitting
- Provides a natural way:
 - to take into account multiple scattering, magnetic field unhomogeneity
 - possibility to take into account mean energy losses
 - to extrapolate tracks from one sub-detector to another

Basic Principle of Kalman Filter



Basic Principle of Kalman Filter

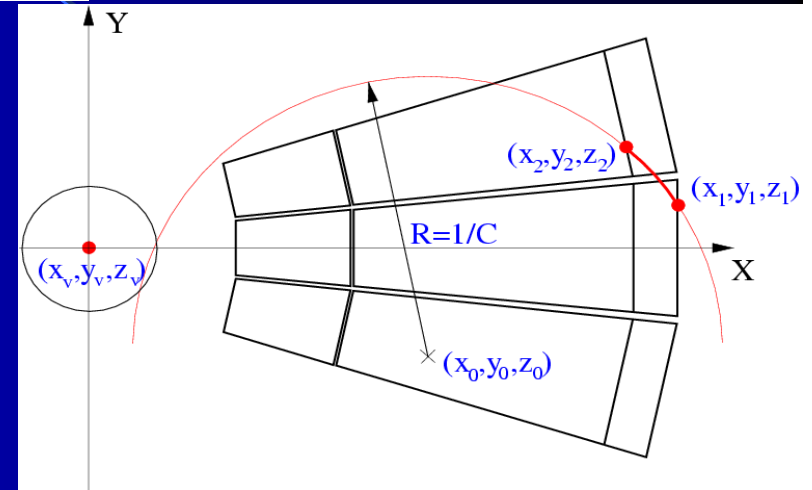


Seeding

Track Efficiency limited by efficiency of seeding!

Primary Seeding with vertex constrain

- Take 2 rows with gap 20 rows
- Check quality of track segment:
 - χ^2
 - number of found clusters
 - number of shared clusters



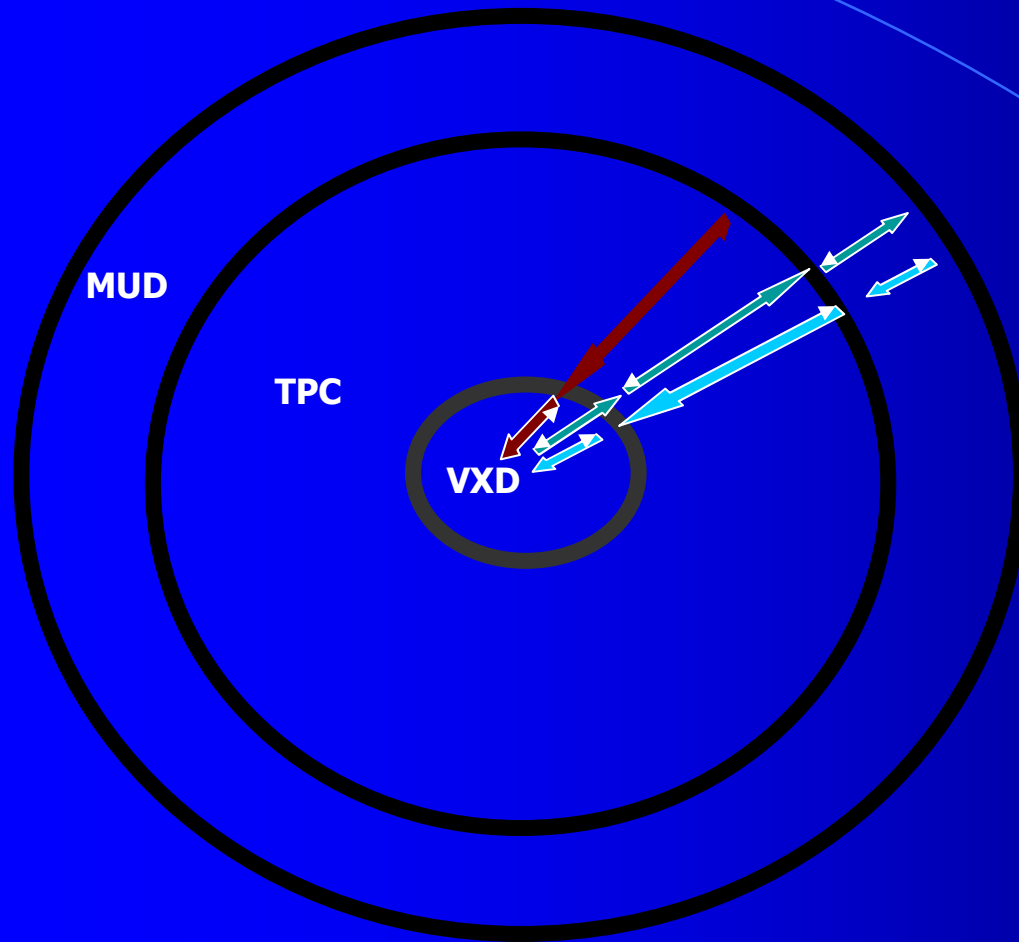
Secondary Seeding without vertex constrain

- Simple track follower
- Algorithm
 - Seeding between 3 pad-rows (with gaps 2 rows)
 - Check that nearest clusters available at prolongation
 - Find prolongation to inner radius to make 20 rows segment
 - Check quality of track segment

Parallel Tracking

- seedings with constraint + seedings without constraint at different radii (necessary for kinks) from outer to inner
- Tracking
 - Find for each track the prolongation to the next pad-row
 - Estimate the errors
 - Update track according current cluster parameters
 - (Possible refine clusters parameters with current track)
- Track several track-hypothesis in parallel
 - Allow cluster sharing between different track
- Remove-Overlap
- **Kinks and V0** fitted during the Kalman filtering

Tracking strategy – Primary tracks

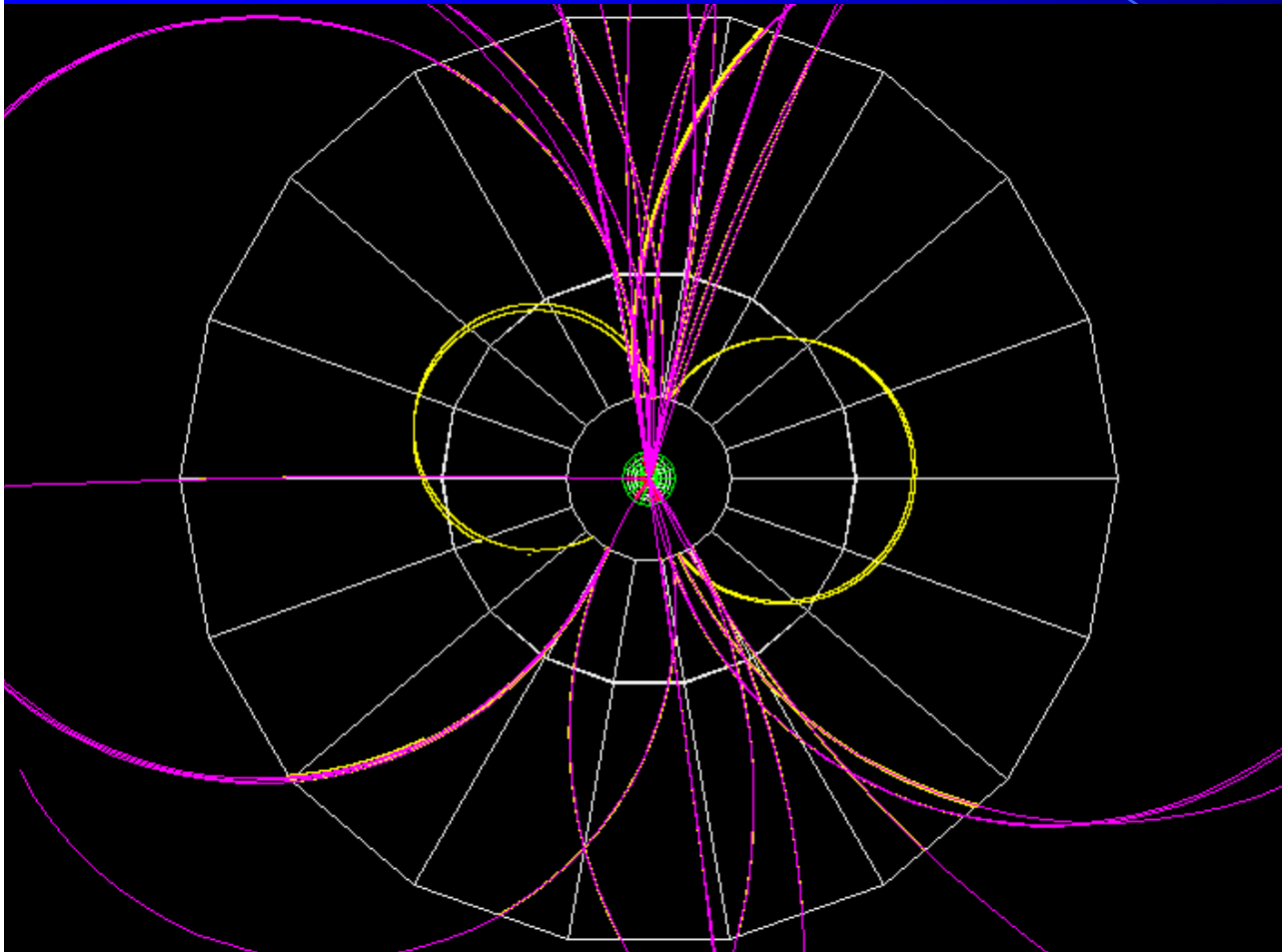


- Iterative process
 - Forward propagation towards to the vertex –TPC-ITS
 - Back propagation –VXD-TPC-MUD
 - Refit inward MUD-TPC-VXD
- Continuous seeding –track segment finding in all detectors

currently is implemented TPC+VXD+MUD barrel propagation

Reconstructed Tracks

$$e^+ e^- \rightarrow Z H \rightarrow \mu^+ \mu^- H$$



yellow - hits

purple -reconstructed
tracks

need to improve
low-pt tracks

VXD Standalone Tracker

- Uses Clusters leftover by Parallel Kalman Filter
- **Requires at least 4 hits to build a track**
- Seeding in VXD in two steps
 - Step 1: look for 3 RecPoints in a narrow row or 2 + the beampoint.
 - Step 2: prolongate to next layers each helix constructed from a seed.
- After finding clusters, all different combination of clusters are refitted with the Kalman Filter and the tracks with lowest χ^2 are selected.
- Finally, the process is repeated attempting to find tracks on an enlarged road constructed looping on the first point on different layers and all the subsequent layers.
- In 3.5 Tesla B-field $\rightarrow P_t > 20 \text{ MeV}$

Ilcroot Status

- VXD and HCAL: digitization + recpoints
- TPC and MUD: fastrecpoints
- Most of developments toward recommendation of TRC:
 - Si Central Tracker
 - He based Drift Chamber with cluster counting
 - FTD
- New ECAL with Dual Readout probably ready for LCWS07
- Analysis and jet reconstruction: see next talks