MM: 13/8/98

Tracking Software for the TDR

1 Task Definition

In this memo I want to review and evaluate the status of the tracking software, list the tasks to do for the TDR and start a discussion on the priorities of these tasks.

We have two main projects in the tracking software:

1. The Detector Simulation Package for the outer tracker.
   This includes implementing the detector geometry in GEANT and simulating the detector-response to interactions of traversing particles with the detector material. (HITS, DIGITIZATIONS)

2. The Track Reconstruction Package.
   The track reconstruction package uses the results of the digitizations as input in order to reconstruct the particle trajectories. The track reconstruction task can be split in separate parts:

   - Track Fitting
   - Track Finding:
     - Track seeding: find a starting point for reconstructing a track.
     - Track following: trace the track from its seed through the magnetic dipole field towards the track production vertex.

One can ask the question: "Why do we need to do these efforts now?" I believe that these efforts are essential in the preparation phase of the experiment for several reasons. Stated general terms, the following studies are needed:

- Studies related to the design of tracking detectors.
  Which detectors do we need? How many of them are needed? Where should they be positioned? How accurately must they measure? etc.
  The software needed to do these studies should be available when the detectors are designed. As the detector design gets more advanced, more detailed studies are required. In practice this means that these

studies are ongoing now and converge in a detector design in the TDR. They are relevant until mass production of the detectors starts.

- Studies related to the design of non-tracking detectors, which need reconstructed tracks as input. In this case the emphasis is not on how the tracks can be reconstructed, but more on the ultimate precision of the reconstructed trackparameters or perhaps on the efficiency for finding a track. Examples are the RICH detector and the ECAL (e-γ separation).

- Physics performance studies for the LHCb experiment. These studies test the physics capacity of the whole experiment. Also here, it is important that the efficiency and quality of track reconstruction in the software matches that of a well understood and calibrated detector. Since in many aspects the current tracking software is still very far from this state, we must use the Monte Carlo to mimic the final results. e.g. we assume that alignments calibrations, r-t relationships, etc. have been done.

- Preparations for the reconstruction of the real-data events. This software must be available at the start of datataking. The main effort here is presumable in the various calibrations procedures required to be able to understand and use the incoming data. Examples of these calibrations are: alignments, drift time calibrations, left-right ambiguity, understanding measurement errors, etc.

2 Status of Simulation Package

The current simulations of the inner and outer tracker detectors is based on the code written for the Letter of Intent. The emphasis of these simulations is on flexibility rather than on accuracy.

The software allows to change the layout of chambers, the stereo angles of monolayers, the wire pitch, the single wire resolutions etc., at the *reconstruction stage* in the program. This means that it is possible, by re-reconstructing a single sample of simulated events, to study many different experimental setups of the tracking stations.

At the flip side of the coin we have the facts that:

- Detectors are poorly simulated in the GEANT tracking. Stations simply consist of a box with a cover. No details can be implemented as this would lead to a loss in generality. (e.g. there is no material present inside the stations)
- In the GEANT simulation only hits at the entrance and exit points of the wire stations are saved as input for reconstruction. During the digitization step the responses of individual cells are derived from these entrance ande exit points. Problems related to this approach are:
  - curling tracks are treated incorrectly
  - no multiple scattering is implemented inside stations
  - only a vertical B field component is used to interpolate the track position inside monolayers in the station. Especially for lower momentum tracks this can lead to erroneous digitizations.

Currently, the drift chamber digitizations are kept simple: they consist of an identifier for the hit wire and an observed (signed!) drift measurement and error. Pile-up, either due to multiple interactions in a single bunch, or due to interactions in consecutive bunches, is not taken into account.
More details on the digitizations can be found in LHCb note: 98-044

2.1 Evaluation

The existing code has been extensively used and increased in complexity from 1994 (preparation of the LoI) until today. Adding any further detail to this should be preceded by the question whether it is not better to revise the whole setup. On the other hand I believe that in this area our group should not do any pioneering (OO) efforts. Instead our detector geometry description and the detector response should be adapted to the standard implementation adopted by LHCb for the timescale up to the TDR. We should actively contribute to a new ("SICB++") simulation program by specifying the requirements that the outer tracker simulation package needs from this program.

A concrete list of things to do for improving the outer tracker simulations and digitization package is:

- Decide (inside LHCb) what framework to use.

- Implement stations, monolayers, materials in the detector geometry.
- Update the hit structures.
- Update/Improve the digitizations accordingly.

3 Status of Track Reconstruction Package

The track reconstruction package is based on the implementation developed by HERA-B. Our current implementation consists of a track-fit part and on a part that performs local pattern recognition: i.e. looking for track segments inside a station.
Details on the implementation in LHCb can be found in LHCb note: 98-045.

3.1 The Trackfit

For studies in the LoI a global χ^2 minimization fit was performed to find the best matching trajectory over a number of detector hits. This approach suffered mainly from the fact that multiple scattering could not be implemented.

For the studies in the TP a Kalman filter based trackfit was implemented in which multiple scattering and energy loss were taken into account. Using this fit the reconstructed track parameters were specified with valid errors (momentum, direction and position measurements resulted in a Gaussian "pull" distribution centered around zero and with widths equal to one). This was essential for the physics performance studies that were based on these reconstructed track quantities.

The track reconstruction package requires input from two sides: the detector digitizations and the geometry database describing the amount of material in the setup.

- The digitizations used in the current fit are those of the outer tracker, inner tracker and of the vertex detector. These digitizations are first ordered in a sequence of decreasing z-position and are connected by the Kalman fit. Our current approach assumes that each subdetector performs a one dimensional measurement on a traversing track. This measurement can be: a drift time in the outer tracker, or a "center-of-gravity" measurement in the MSGC inner tracker or in silicon vertex detector. The measurement is projected onto the state vector of the track (x,y,s_x,s_y,q/p) and accordingly used to obtain a updated track vector.

However, in reality a silicon-strip in the vertex detector contains 2 dimensional information: e.g. a ϕ strip also measures, besides the ϕ-coordinate, very roughly (few centimeters) the r-coordinate. This is not yet taken into account. In practice this causes a problem in pattern recognition, but not in track fitting.

- A second input to the trackfit is the amount of material (dead or active) a track encounters travelling from hit i to hit j in the setup. In the existing code a dedicated database (a large common block) is kept to represent all material walls in LHCb that a track possible could encounter between i and j. The finite amount of material seen is accumulated in an infinitely thin slice, and the mean scattering angle of the particle is calculated. This scattering angle is then taken into account in the track parameter covariance matrix related to the transport of i to j.

  The advantage of such an material database implementation is that it can be used separately from the GEANT implementation of the detector. Also, it allows for a relatively fast application. However the disadvantage is that it is virtually impossible to keep the model realistic. The increasing complexity in the GEANT implementation of the LHCb detector can no longer be matched with the material implementation in the fit. This leads to incorrect covariances and thus also to incorrect χ^2 results on track parameters.

  The effect of thick scatterers is not implemented, nor is the process of large enerly loss (discrete bremsstrahlung), which is mainly relevant for electrons).

3.2 Pattern Recognition

The pattern recognition task can be subdivided in separate tasks:

- **Trackfinding.**
  Between a z positions of 8 meter and 10 meter the detector setup contains a field free region which is optimally suited to start the search

for tracks. The stations 8-10 are used for this purpose. The aim of trackfinding studies is to determine the efficiency with which track seeds can be found in our detector setup. Using this an optimal tracking station set-up in this area should be suggested.

- **Trackfollowing.**
  Once a starting point for a track is found downstream from the magnet, it must be traced upstream towards its production vertex, i.e. it must be followed through the magnetic field and into the silicon vertex detector, at the same time measuring the momentum and position of the track at each point.

  The trackfollowing task can be split up in two sub-tasks:

  - **local pattern recognition:** track elements inside a station are combined to form track elements. The search for trackelements includes solving left-right ambiguities which result from drift time measurements. The task here is to suggest a layer set-up providing optimal efficiency/ghost-track performace in the track-element search.
  - **global pattern recognition:** track elements are combined to make a full track through the experiment. In this case we have to demonstrate that high track following efficiency is obtained for various types of tracks (momentum dependence, angle dependence, particle ID dependence). Notoriously difficult tracks are K_s decay products that can have a production vertex at a large z-position (e.g. inside the magnet).

3.3 Evaluation

The track reconstruction package was used to optimise various items for the TP: the required optimal number of tracking stations and their position was suggested and the critical importance of multiple scattering in the trackfit was demonstrated. The momentum fit gave satisfactory results except in the cases of thick scatterers and hard radiation energy loss (i.e. mainly for electron tracks).

The most critical part in the *trackfit* code is the treatment of material. With the increasing complexity in the geometry description the method used to implement material in the fit is no longer practical. This part of the code must be replaced by a code which extracts information directly from the detector geometry implementation (i.e. GEANT).

The work on *track finding* must be started. The HERA-B implementation can be used as an example. The aim of this task is to provide a detector setup in terms of monolayer structure, stereo angle, which is ideally suited for track finding.

The *local pattern recognition* study is well underway. In the current version track elements are found in the stations. This is the first part of tracking software that is written in C++ and succesfully used in the SICB framework. There are two items that should be reconsidered:

- the interface to the track fit.
  The input to the local pattern recognition is provided by large arrays of hit-maps that are defined each time a track enters a station. The code is rather time and memory consuming and prone to errors. This interface would be improved as the trackfit is moved to C++ implementation.

- the real time performance of local pattern recognition.
  The program is so slow that it can never be used in practice to study more than a handful of events. Clearly, this is unacceptable.

A difficult task in the local pattern recognition is to solve the left-right ambiguities of the drift-time digitizations. This task is especially important as it gives input on the required monolayer structure of the stations.

The work on *global track following* has not started. This is perhaps the most complicated and time consuming effort in pattern recognition on the road to the TDR. The study will aim at obtaining the efficiency with which tracks can be propagated through the detector setup. Although this is a very important number it could be simulated with a not-final version of the code. This is the reason why I propose this task with a lower priority and aimed at longer timescale.

In summary, items in the current code that are prone to improvement are:

- for the trackfit:

  - revision of the implementation of scattering material.
  - implementation of thick scattering volumes

- improvement of the treatment of electrons

- for the local pattern recognition part:
  - improving the speed of the current algorithm
  - implementation of left-right ambiguity resolving task.

4 Towards the TDR

On the long term we have to migrate to an OO version of the code. However, on the short term we have to produce results almost continuously as this is required as the detector design evolutes. A possible solution for this is that we try to *gradually migrate* the software towards the OO version, rather then rewrite it from scratch. Newly developed software will be written in C++ and interfaced to the existing (non-OO) code. If so required, parts of old code can be replaced by C++ code when new functionality is required. The obvious disadvantage of this method is that there is no freedom to make an "ideally" structured redesign of the overall tracking code. The advantages, however, are that we can plug in and test new software in a continously running version of the program.

On the other hand, the general framework of the LHCb simulation and reconstruction program (SICB++) will also evolve and we have to give our requirements on this program to the LHCb computing group. Especially for the task of track reconstruction the program structure could be delicate, as the tracking software deals with many aspects of the program. (e.g. digitizations of many detectors, input from geometry databases, comparison of information between simulated and reconstructed objects, etc.)

The following contains a concrete list of things to do up to the TDR. For each task I have added an (gu-)estimated amount of time.
As these items reflect a personal opinion, they are open for discussion

- Simulation tasks:
  - Implement a GEANT simulation of the tracking stations (<u>3 months</u>)
  - Adapt the digitizations to the new GEANT setup (<u>1 month</u>)

- Reconstruction tasks:

- Write a C++ version of the Kalman Trackfit, revising the input from digitizations and the implementation of material (how?) (3 months)
  Implement thick-layer MS and discrete energy loss processes for electrons. (1 month)
- Improve the local pattern recognition algorithm taking into account left-right ambiguities. (3 months)
- Start the implementation of trackfinding in stations 8-10. (6 months)
- Implement a first version of global pattern recognition (track following from station to station). I don't think we need the full version here (although that would be nice). (3 months)

Adding up all these estimated times I get a period of 20 months f.t.e.

The following items I would propose to tackle after the TDR. They involve the various calibration procedures:

- chamber alignments
- drift-time relations.
- t0 calibration (including signal delay in wires)
- interaction time fitting (see HERA-B)
- time of flight calibration
- ...

5 URD for Track Reconstruction

In order to ensure that the interests of the tracking group are represented it is important to give our input in the form of a User Requirements Document (URD).