Implementation of an Object Oriented Track Reconstruction Model into Multiple LHC Experiments

I. Gaines[1]*, S. Gonzalez*[2]*, S. Qian*[2α]

[1] Fermilab, Batavia, USA
[2] University of Wisconsin, Madison, USA

Abstract

An Object Oriented (OO) model[1] for track reconstruction by the Kalman filtering method has been designed for high energy physics experiments at high luminosity hadron colliders. The model has been coded in the C++ programming language and has been successfully implemented into the OO computing environments of both the CMS[2] and ATLAS[3] experiments at the future Large Hadron Collider (LHC) at CERN.

We shall report:
(1) how the OO model was adapted, with largely the same code, to different scenarios and serves the different reconstruction aims in different experiments (i.e. the level-2 trigger software for ATLAS and the offline software for CMS);
(2) how the OO model has been incorporated into different OO environments with a similar integration structure (demonstrating the ease of re-use of OO program);
(3) what are the OO model's performance, including execution time, memory usage, track finding efficiency and ghost rate, etc.
(4) additional physics performance based on use of the OO tracking model.

We shall also mention the experience and lessons learned from the implementation of the OO model into the general OO software framework of the experiments. In summary, our practice shows that the OO technology really makes the software development and the integration issues straightforward and convenient; this may be particularly beneficial for the general non-computer-professional physicists.

Keywords OO model, Track reconstruction, Kalman filtering, CMS, ATLAS

1. Introduction

Nowadays, Object Oriented (OO) technology has become more and more popular in high energy physics. Here we describe an OO model[4] for track reconstruction with simultaneous pattern recognition and track fitting by the Kalman filtering method. It was initially designed (using Booch methodology[5]) and coded in the C++ programming language in its first version in 1995 for the CMS experiment at the Large Hadron Collider (LHC) at CERN. After the C++ Standard Template Library (STL) became available, the model was re-designed at the beginning of 1996 in order to take advantage of the container classes provided by STL. In the fall of 1997, another partial re-design in the OO model was undertaken by introducing an abstract class that clarified and simplified the code. Since 1998 the OO model and C++ code has been successfully re-used, with only minor modification, in ATLAS, another major LHC experiment. This demonstrates one of the claimed advantages of the OO technique: ease of re-use. Here, the "re-use" is not only for just some particular classes, but also for the entire model. The modularity enforced by the OO design of the model made this re-use much easier, as the experiment-specific code is very localized. The first 4 years of history of the OO model (up to 4/1999) has been documented in [1,4].

[α] Speaker (Mailing address: CERN, CH-1211, Geneva 23, Switzerland)

Since then, the OO model has been tentatively integrated into the CMS OO software framework "ORCA" (Object oriented Reconstruction for CMS Analysis) and has been implemented in the ATLAS level-2 trigger OO reference software framework[6]. Again, the two implementations (into different OO environments) have shown a certain degree of similarity. As a consequence, any improvement in the code and experience in the implementation of the general OO environment for one experiment immediately benefits the other experiment.

In this short version of the full report[7], the new developments of the OO model and its implementation in the CMS and ATLAS experiments since CHEP'98 are explained in Section 2; some preliminary performance results are shown in Section 3; and a summary and future prospects are given in Section 4.

2. New developments on the OO model and its implementation since CHEP'98

The latest class diagram of the OO model for track reconstruction is show in Fig.1. The history (up to CHEP'98 in 9/1998) of the OO model's evolution and its implementation in CMS and ATLAS experiments are summarized in [1,4].

In early 1999, the highly modular FORTRAN code (which accounted for about 30% of total lines of code in the OO model and enabled us to concentrate our effort on the OO and C++ aspects of model in the first 4 years of development) was converted to C++ with the aid of the UNIX utility "f2c". Since then, all further development in the model has been with the pure C++ version. Nevertheless, the temporary use of legacy FORTRAN in methods of certain classes greatly increased the development speed of the initial prototype versions of the model.

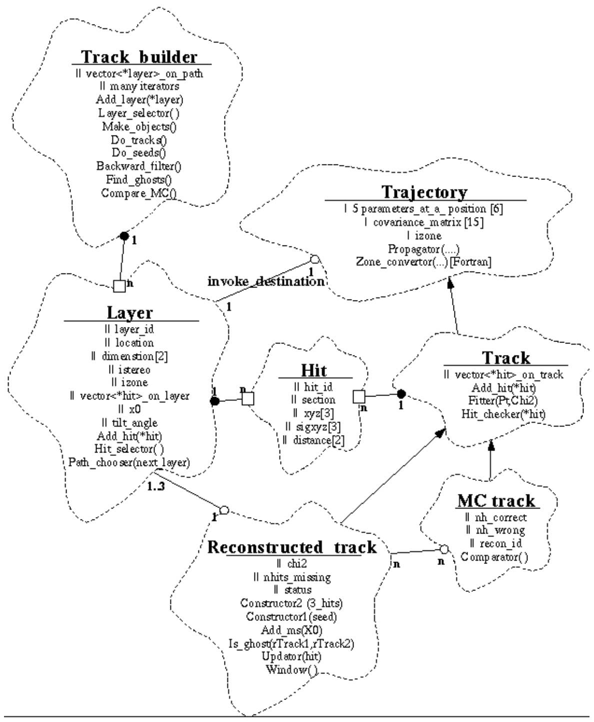
The original OO model was stand alone, as it predated the development of full reconstruction and analysis frameworks in both experiments. During 1999 CMS developed a powerful reconstruction framework known as ORCA. After all necessary functionality for input objects in ORCA became standardized in autumn of 1999, we integrated the OO tracker model into ORCA by inputting the ORCA reconstructed hits to the model and reconstructing tracks. ORCA is continuing to develop rapidly, and complete integration of the OO model will be done using the final ORCA detector and track classes.

In ATLAS, another seeding method using the seeds produced by a pixel reconstruction package, and starting the Kalman filtering process from the inner-most position outwards has been implemented. This results in a better performance of the OO model due to the preciseness of the pixel hits. For the 2nd scenario (Fig.2b of [7]) of the OO model, the performance of the model depends critically on the quality of the seeds; therefore, more seeding methods will be tested in order to achieve the optimum performance.

Despite the differences between the two experiments, we could put the I/O code in two encapsulated functions, so the implementation structures for both experiments are very similar.

3. Preliminary performance result

For the performance of the OO model, the ATLAS implementation has been tested extensively by using trigger simulation data as input. The standard data sets (for testing various algorithms) are single track (μ's, π's and electrons at different energies) events and B-physics events at low luminosity. The computer used is a 300 MHz Pentium II under Linux. We show a few typical results about: (1) efficiency, (2) momentum resolution, (3) execution speed (Table 1), (4) ghost reduction and (5) an example of B-physics study in the channel $B \rightarrow \pi\pi$ (Fig.2). Due to the limitation on the article length, the results (1)(2)(4) will appear in the long version [7] of this article in the proceedings of this conference.



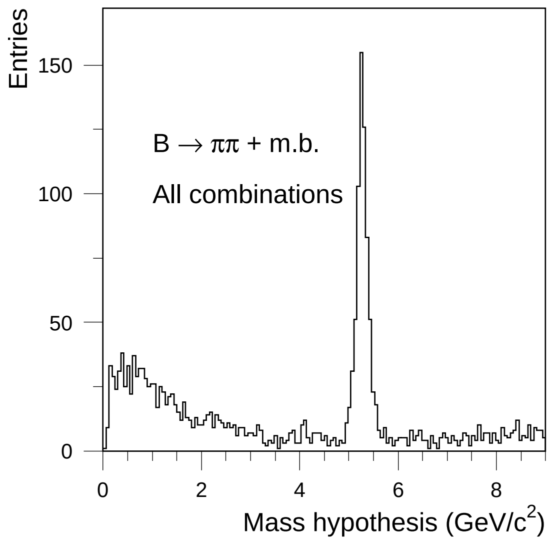


Figure 2: Reconstruction of the B → ππ mass hypothesis in the ATLAS Level-2 trigger using the Kalman filtering method

Figure 1: Class diagram of the OO tracker model

Data set	**Execution time per event (milli-second)**			**TRT seeds per event**			**Average execution time per TRT seed**
	min.	max.	average	min.	max.	average	(milli-second)
Electron without pile-up	1.2	5.6	~<1.9>	1	3	~<1.55>	~<1.8>
Dijet without pile-up	0.8	109	~<5.3>	1	14	~<2.21>	~<2.6>
B-physics at low luminosity	7.6	3544	~<172>	2	493	~<53.3>	~<2.6>

Table 1: Statistics of the benchmarking measurement

Many more results are described on Web sites [8].

The memory usage of the model is in the order of 10 MByte, which also depends on the data volume of the event, with high luminosity events occupying more memory than the low luminosity ones.

4. Summary and prospects

We have designed an object oriented model for track reconstruction in HEP experiments, coded it in the C++ programming language and preliminarily implemented it into both the CMS and ATLAS experiments on LHC. The main features of this model are:

(1) The class design is according to the OO paradigm and is based on the proven data concepts in HEP track reconstruction, so that it can be easily adopted by the non-expert class users.

(2) The OO model is closely related to the Kalman filtering track reconstruction package of the previous pure FORTRAN version. Many FORTRAN subroutines of the package have been originally re-used as member functions of various classes, and later converted into C++ in a straightforward manner.
(3) The STL, a powerful tool in C++ programming, has been extensively used in the OO model design and the C++ coding.
(4) The OO model is flexible enough to be re-used in multiple HEP experiments, with only the implementation of layer class different.

The preliminary results show that its memory usage is moderate, its track finding efficiency is satisfactory and its execution speed is approaching the requirement of level-2 trigger. We now have some powerful C++ debugging tools on hand, which will be very helpful for future development.

Next, for the CMS implementation, we will complete its integration into ORCA by investigating more efficient I/O functions, use the standard CMS classes, and then tune the performance. For ATLAS, we can improve the performance by investigating new seeding methods and exploring new reconstruction strategies. More sophisticated corrections (e.g. non-uniform magnetic field correction, etc.) can be gradually considered after a recently implemented energy loss correction for electrons.

More details of this OO model and its implementation (including the model design, the class and object diagrams, the performance results, the documentation and the presentations in various conferences and meetings, etc.) can be found on the Web site [4] which is updated regularly to include all new developments.

Acknowledgements

We are very grateful to Drs. S. Wynhoff and T. Todorov for their assistance on the integration of the OO model into ORCA, and to Drs. M. Sessler and J. Baines for their great help on the ATLAS implementation of the model into the level-2 B-physics trigger software.

Reference

[1] I. Gaines, P. LeBrun and S. Qian, CMS TN/96-122 (1996);
I. Gaines, T. Huehn and S. Qian, in the proceedings of “Computing in High Energy Physics (CHEP’97)”, Berlin, Germany (4/1997); also filed as CMS CR/1997-018 (1997);
I. Gaines and S. Qian, in the proceedings of “Computing in High Energy Physics (CHEP’98)”, Chicago, U.S.A. (9/1998); also filed as CMS CR/1998-023 (1998);
I. Gaines and S. Qian, in the proceedings of “6th Advanced Computing Conference in Physics Research (AIHENP’99)”, Crete, Greece (4/1999).
[2] CMS Technical Proposal, CERN/LHCC/94-38 (1994)
[3] ATLAS Technical Proposal, CERN/LHCC/94-43 (1994)
[4] http://cmsdoc.cern.ch/~sijin/oo.html or http://www-wisconsin.cern.ch/~sijinat/oo.html
[5] G. Booch, "Object-Oriented Analysis and Design", Benjamin-Cummings (1994)
[6] S. Qian and M. Sessler, ATLAS testbed note 026, (1998)
http://www.cern.ch/ATLAS/project/LVL2testbed/www/notes/026/tn026.ps
[7] I. Gaines, S. Gonzalez and S. Qian, the long version of this article in the proceedings of this conference
[8] http://www-wisconsin.cern.ch/~sijinat/lvl2/res.html
http://www-wisconsin.cern.ch/~atsaul/results/sctkalman
http://hepunx.rl.ac.uk/atlasuk/simulation/level2/Bphys/plots.html