Computer Physics Communications 57 (1989) 251–254
North-Holland

A VECTORIZED MONTE CARLO DETECTOR SIMULATION PROGRAM FOR ELECTROMAGNETIC INTERACTIONS

S. YOUSSEF [1]

Supercomputer Computations Research Institute, Florida State University, Tallahassee, FL 32306-4052, USA

W. MARTIN, T.C. WAN [2] and S. WILDERMAN

Laboratory for Scientific Computations, University of Michigan, Ann Arbor, MI 48109, USA

MC4 is a detector simulation program combining a vectorized ray-tracing algorithm with a vectorized version of the electromagnetic interaction routines from GEANT3. The implementation of ray tracing is able to represent moderately complex geometries such as single calorimeter modules or test-beam situations. Results from MC4 are compared with EGS4 simulations and with experimental results. Timing results are given for scalar machines and on a vector supercomputer. Production applications and applications to future versions of the GEANT code are discussed.

1. Introduction

The vector computers available at the Supercomputer Computations Research Institute (SCRI), the five NSF supercomputer centers *, Argonne National Laboratory, KEK, Rutherford Appleton Laboratory, CERN, Lawrence Livermore Laboratories, Los Alamos National Laboratory and at many universities represent a large potential source of computing capability which could be used for experimental high energy physics **. In particular vector computers have been shown to be suitable for detailed particle transport Monte Carlo simulations [1–3] with the significant proviso that the basic structure of most scalar Monte Carlo simulations must be completely changed. At SCRI, we are collaborating with CERN and the University of Michigan in an ongoing effort to change the structure of the well-known GEANT code [4] so that it is amenable to vector computations [5]. Part of this effort has been coalesced into a stand alone simulation package designed to take full advantage of vectorization for moderately complex geometric situations such as single calorimeter modules or test beam calorimeters. In addition to production use, a substantial part of the motivation for MC4 is to use the code as a test bed for developing new simulation ideas including the use of variance reduction methods, correlated sampling [6] and the use of massively parallel computers.

2. Major components of MC4

The two main components of MC4 are a vectorized implementation of an algorithm for ray tracing and point location, and a vectorized version of the electromagnetic interaction routines from GEANT3 [4].

Ray tracing and point location are purely geometric problems which occur in computer graph-

[1] Partially funded by the US Department of Energy through contract number DE-FC05-85ER250000.

[2] Current address: EVEREX Co., Fremont, CA 94538, USA.

* The Cornell National Supercomputer Facility (CNSF) at Cornell University; the John von Neuman Center (JvNC) at Princeton; the National Center for Supercomputer Applications (NCSA) at the University of Illinois at Urbana–Champagne; the Pittsburgh Supercomputer Center (PSC) at Carnegie–Mellon University and the San Diego Supercomputer Center (SDSC).

** Many of these facilities are being upgraded with a new generation of vector computer such as the Cray YMP, Cray 3 and Hitachi S-820.

ics, accelerator design, computer aided design and nuclear reactor simulation as well as in detector simulation in high energy physics. Point location is the problem of finding the volume containing a given point and ray tracing is the problem of breaking a given curve into pieces each in a single volume. The curve may be a straight line (e.g. for computer graphics) or a more complicated path for a charged particle in a magnetic field. The ray tracing algorithm adopted for MC4 is well suited for detector simulation, because of an efficient treatment of repeated similar objects, rapid point location which is independent of the total number of volumes, and because it is is amenable to the fine-grained parallelism needed for vectorization [7–9]. In full generality, this ray tracing algorithm allows an input "primitive" volume P to be defined by

$$P = \left\{ X \in \mathbb{R}^d : \forall j : 1 \leq j \leq n : l_j \leq f_j(X) < h_j \right\},$$

where $l_1, \ldots, l_n$, $h_1, \ldots, h_n$ are real numbers and where each of the continuous functions f_j maps $\mathbb{R}^d$ into the reals [9]. These primitives can then be freely combined in a similar spirit as the constructive solid geometry algorithms used in computer aided design [10]. Since MC4 is only designed to simulate a few calorimeter modules at a time, the ray-tracing algorithm is implemented with the restriction that $d = n = 3$, $f_1 = x$, $f_2 = y$ and $f_3 = z$. However, the primitive boxes may be rotated, translated and freely combined. For example, the typical trapezoidal calorimeter module shell can be constructed from six overlapping boxes. This implementation is sufficient to represent most calorimeter modules or test beam situations in any desired amount of detail.

To simulate the interactions of electrons, positrons and photons, the electromagnetic interaction routines from GEANT3 were adopted and vectorized. Although this involves substantially rewriting the programs, the simulated interactions and sampling logic are preserved. Thus, as in GEANT3, each of the electromagnetic cross-sections are simulated in arbitrary mixtures of materials.

The overall structure of the program is shown in fig. 1. A complete geometric situation is represented as a tree, where each node of the the tree is a set of similar primitives. In the case of MC4, each set of similar primitives must be a set of (possibly overlapping) boxes with the same rotation with respect to the origin. A program "LatMak" creates each node of the tree from an input sequence of boxes. "GeoMak" then constructs a geometry file containing a complete description of the entire geometry including all of the data structures needed for ray tracing. Finally, a main simulation program produces output events in a detector-independent format. Auxiliary programs are available for making non-graphics displays of the contents of geometry files and events files. A plotting utility allows histograms to be made from any event file. For example, one can build the geometry for a liquid argon calorimeter, simulate a beam of electrons entering the front face and plot the signal in the argon without every having to write a FORTRAN program. A simple graphics package is being prepared to show two dimensional slices ("catscans") and particle interactions.

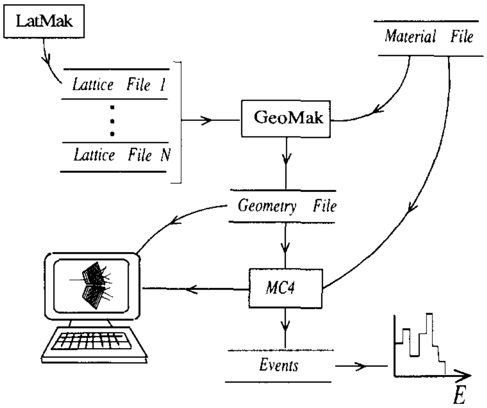

Fig. 1. The large scale structure of the MC4 simulation code. Each session with "LatMak" produces a node of the tree representing a complete geometric situation. Each node consists of a sequence of possibly overlapping boxes and is stored in a "lattice file.". "GeoMak" reads each such "lattice file" and produces a geometry file representing a completely geometry. "MC4" then reads this file and produces simulated events. A material file defines a prearranged set of materials to choose from.

Digitization in MC4 is handled in the simplest way that is useful for calorimeter simulation. Energy deposition is simply accumulated for each of the input primitive volumes used to construct the geometry. In addition, there is an optional interface (similar to GUSTEP in GEANT) if a more detailed treatment is desired.

The MC4 code has two versions, one running under VAX/VMS and the other running on an ETA10 under Unix system V. The VAX/VMS version is written in FORTRAN 77 * and is " vectorized" in the sense that the computationally intensive routines are written with small FORTRAN loops which translate almost automatically into vector instructions. This code could be used as a starting point for vectorization on any of the vector supercomputers. Unfortunately, the vectorizing preprocessor available on the ETA10 was not reliable enough to do the translation and so the translation to explicit vector instructions was done by hand. The resulting FTN200 (Fortran 77 with vector extensions) code runs on the Cyber 205 or ETA10 series computers.

3. Comparison with EGS4 and with experiment

Before MC4 can be used for production simulations, it must pass an extensive battery of tests to demonstrate that the results are reliable. Figure 2 shows the results from one such benchmark taken from the EGS manuals [1]. The test consists of 44 and 177 MeV photons incident on varying thicknesses of lead followed immediately by a 0.5 cm thick polystyrene scintillator. A photon is said to have "converted" in the lead if at least 60 keV is deposited in the scintillator. Figure 2 shows the conversion probability as a function of lead thickness for both photon energies. The results from MC4 compares well with both EGS4 and with the experimental results [12]. Space limitations preclude reporting on further test cases but these tests will be published elsewhere.

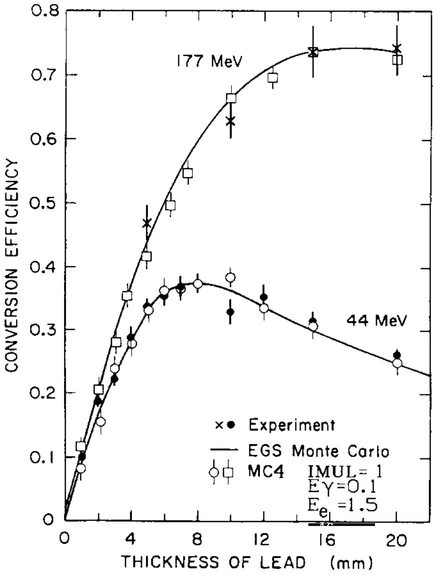

Fig. 2. A comparison of experimental results with EGS4 [14] with an additional comparison to MC4 results. In this test, 44 and 170 MeV photons are incident upon a lead sheet of varying thickness followed by a sheet of polystyrene scintillator. A photon "converts" if at least 60 keV of energy is deposited in the scintillator.

4. Timing benchmarks

One of the main motivations for writing MC4 is to demonstrate that a production level detector simulation program can take full advantage of vector computers. Some simple benchmarks indicate that this is indeed the case.

For a typical characteristic timing benchmark, consider 50 GeV electrons incident upon a calorimeter consisting of a uranium plate (3 mm) and a liquid argon gap (6 mm) repeated twenty times and thus containing the shower to within a few percent. For low energy cut offs of 1.5 MeV for electrons and 0.1 MeV for photons, MC4 uses 11 seconds per GeV on (for reference) a VAX 11/780. For comparison, the analogous simulation using a hand coded version of EGS4 takes 8 seconds per GeV on a VAX 11/780. On one of the four ETA10G processors, the vectorized version of MC4 uses 35 milliseconds per GeV and is thus a factor of 310 faster than the scalar code on a VAX 11/780. A factor of 12–16 of this improvement in performance is due to vectorization. For

* With the exception of the use of the VMS INCLUDE statement and the use of lower case.

this benchmark on the ETA10, the time spent in different sections of the code is: EM interaction routines – 43%, ray tracing – 34%, stack operations – 15%, digitization – 7%, i/o – 0.5% and miscellaneous – 0.5%. Less than 5% of the CPU time is spent in scalar routines.

For a more realistic simulation, consider adding G10 boards in the above calorimeter and enclosing it in a trapezoidal steel box. In this case, the vectorized version of MC4 runs 30% slower. This is mostly due to the additional G10 boards and not because the calorimeter is enclosed in a box. Even though these results are promising, the most important uses of MC4 will be in simulating calorimeter modules in great detail including, for example, spacers, notches, bolts, etc. MC4 is designed to be efficient for such detailed simulations, but this remains to be demonstrated by explicit examples. Of course, the speed of the vector version of the program depends on there being enough particles to make sufficiently large vector operations. Thus, at energies below approximately 10 GeV, MC4 loses much of the gains from vectorization. However, low energy simulations can be performed at high speed by simply simulating several events together.

There are several possible avenues to improving the speed of MC4 even further. These include improving the handling of electrons crossing boundaries, simulating more than one shower at a time to increase the vector sizes, using a safety radius, and using large pages on the ETA10. In total, these improvements represent at most an additional factor of 2 increase in speed.

5. Applications and future work

The natural application of MC4 is in producing detailed response maps of calorimeter modules, comparable to a test beam study of a calorimeter *. We are planning such simulations for calorimeter modules from the D0 experiment at Fermilab and the ALEPH experiment at CERN. MC4 is also meant to function as a test bed for further improvements in simulation. One possibility is to implement correlated sampling [6] in MC4 which would allow exact detailed simulation results to be extracted from a parameterization. As an additional benefit, the representation of "primitive" objects (see section 2) used in MC4 is useful for solving the problem of detecting "illegal" volume intersections in GEANT which are difficult to detect otherwise. This project has a high priority since it could be useful for many GEANT users.

* Coincidentally, MC4 on 1 processor of an ETA10G produces events at about the same rate as the D0 test beam at the muon area at Fermilab, i.e. about a 1 Hz event simulation rate for 30 GeV electron showers.

Acknowledgements

We thank Forrest Brown, Rene Brun and Ken Miura for helpful comments.

References

[1] F.B. Brown and W. Martin, Prog. Nucl. Energy 14 (1985).

[2] F.B. Brown, Trans. Am. Nucl. Soc. 54 (1986).

[3] K. Miura, Comput. Phys. Commun. 45 (1987) 127.

[4] R. Brun, F. Bruyant, M. Maire, A.C. McPherson and P. Zanarini, CERN report DD/EE/84-1 (1986).

[5] M.J. Corden, C.H. Georgiopoulos, R. Brun, F. Bruyant and J.-L. Dekeyser, Comput. Phys. Commun. 57 (1989) 268.

[6] R. Rubinstein, Simulation and the Monte Carlo Method (Wiley, New York, 1981).
S. Youssef, in: Proc. Workshop on Calorimetry for the SSC, University of Alabama, Tuscaloose, Alabama (1989).

[7] S. Youssef, Computer Vision, Graphics and Image Processing 34 (1986) 125.

[8] S. Youssef, in: Proc. Workshop on Detector Simulation for the SSC, ed. L. Price, Argonne National Laboratory (1988).

[9] S. Youssef, SCRI technical report FSU-SCRI-88-71 (August 1988).

[10] See, for example: M. Mäntylä, An Introduction to Solid Modeling (Computer Science Press, Maryland, 1988).

[11] R.L. Ford and W.R. Nelson, Stanford Linear Accelerator Center report SLAC-210 (1978).
W.R. Nelson, H. Hirayama and D.W.O. Rogers, Stanford Linear Accelerator Center report SLAC-265 (1985); see also W.R. Nelson in: Proc. Int. Workshop on Detector Simulation, KFA Jülich, Germany, 11–13 October 1988.

[12] P. Darriulat, E. Gygi, M. Holder, K.T. McDonald, H.G. Pugh, F. Schneider and K. Tittel, Nucl. Instrum. Methods 129 (1975) 105.