Report on Big 12 Faculty Fellowship

Gary M. Wysin, Department of Physics, Kansas State University

Physics Education Technology Simulations in Statistical Physics

Hosted by Paul Beale, July 11 – July 29, 2010

Department of Physics, University of Colorado, Boulder

I visited the Physics Department of University of Colorado, Boulder, during three weeks in July 2010 to discuss and work on simulations in statistical physics, especially as a contributor to the PhET project. The Physics Education Technology (PhET) Project at University of Colorado is a highly regarded and highly accessed set of free online interactive simulations in physics, chemistry, biology, mathematics and earth science. From phet.colorado.edu a use can download and run Java and Flash animation programs covering a wide variety of science topics. The programs are designed to teach many science concepts by encouraging the users to experiment and see visual results immediately. I planned to work on a new simulation and get some ideas about how these are constructed. I also met several members of the PhET team and the Physics Education Research group, including Mike Dubson, Kathy Perkins, and John Blanco. I spent my time there working primarily on two things: (1) giving some assistance and commentary on how to improve a "Collisions" simulation, written by Mike Dubson, and (2) writing an initial program for the simulation of what happens inside a real magnet, whose PhET name is "Inside Magnets."

The Collisions simulation is a Flash program the Mike Dubson has written to show the basic concepts needed to analyze collisions of up to five circular masses moving in two dimensions. The user can click to place balls into the system, and set their masses and initial velocities, and then click Run to see the time evolution. The balls collide with the walls and with each other. One can check the law of conservation of momentum and other basic principles. One can force the balls to move in only one dimension, and check obvious results such as the collision of two balls of equal masses (they just trade velocities). Another feature included is to allow not only for energy conserving *elastic* collisions, but also for ones where an elasticity parameter $0 \leq e \leq 1$ is present. When this parameter is less than 1, some kinetic energy gets transformed

to other non-mechanical forms; the collisions are then *inelastic*. Primarily, I discussed some bugs that were occuring in the limit of zero elasticity, which is a singular limit of the equations of motion. In that limit, two colliding balls must emerge after the collision with zero relative velocity (along a line connecting their centers when at contact). They can still have a common velocity perpendicular to their contact line after the collision. I did not do any programming for this problem, but rather, gave some ideas about how to fix some bugs where the balls could incorrectly pass through the walls of the system. Mostly, the programming needed an improved treatment of the time sequence of collisions that takes place when the elasticity goes to zero.

I spent most of my time designing a program for simulating the motion of magnetic dipoles inside a real magnet, whose working PhET name is now "Inside Magnets." This was done with daily discussions with Paul Beale and more random discussions with the others. The idea is to show how a magnetic phase transition occurs, where the atomic magnetic dipoles are organized at low temperatures, and become disorganized at high temperature. The intention is also to show how the resulting magnetic field lines around a small bar magnet change as the temperature is changed, or, as another strong magnet is placed near the first one. We also wanted to show how any externally applied magnetic field would affect the internal organization of the dipoles inside the magnet, and the associated process of magnetization reversal.

For my own interest and research, I already have experience using the UNIX X11 graphics library to make some simple simulations.[1] In one program (xmc) a simple magnetic model known as the XY-Heisenberg model is used to illustrate the effects of temperature changes in a magnet, and the so-called "vortex-unbinding" phase transition. But that earlier program uses a model involving three-dimensional dipoles, whereas, we though it would be clearer to have a program using only two dimensional dipoles, being much easier to display and interpret on a screen. Also we were not interested in advanced concepts such as vortices, within the PhET simulation. So I developed a new program for showing the thermal effects inside a magnet, based on the "planar-rotor-model," where the dipoles are represented as unit-length arrows

[1] See my X11 simulation programs at www.phys.ksu.edu/personal/wysin

with only xy-components. This initial version of the program is written using the X11 graphics system; later the code will be converted into Flash or Java for the PhET project. The initial version will be available on my web site, under the name "rotor." At its current form, it should be understandable starting at the undergraduate level for students in physics and other sciences.

In this new program, the plane-rotor dipoles interact with each other by "exchange-interactions" that try to keep them parallel to their neighbors. They can interact with an externally applied field. I also included the so-called "dipole-dipole" interaction, where each dipole generates its own magnetic field that acts on all the others. Finally, they have some dynamics that needs to include the randomizing effects of temperature T. We wanted to display a dynamics that is fast and looks physically realistic and that really produces physically realistic processes. Therefore I worked to include two possibilities for the dynamics: Monte Carlo, which updates the system in a random way but consistent with thermal equilibrium, and: Langevin dynamics, which is an evolution in a real time variable, but with random forces of the right statistical averages, so that the system evolves towards thermal equilibrium. The Monte Carlo method is easy to implement and works well, except that the updating looks jerky, rather than smooth, and MC is not considered a true dynamics in a real time variable. I took more of my time to implement the Langevin dynamics, because it required a time integration technique that I was not familiar with. After eliminating some obvious bugs, the correct distribution of forces (really, torques) was achieved, and the system was found to evolve to the correct equilibrium distribution (based on the average energy per dipole being proportional to the temperature). The Langevin dynamics has some advantages over Monte Carlo, namely, if there is a kind of oscillatory motion or resonance present, it will be correcly displayed. Monte Carlo, on the other hand, quickly damps out any kind of oscillations. One disadvantage of the Langevin dynamics, however, is that the dynamics requires a "damping parameter," and its value determines the rate at which thermal equilibrium will be approached. These are all details that would not be expected to be displayed or explained to the user, but whose values need to be chosen by the PhET developers. Currently the program includes both Monte Carlo and Langevin dynamics possibilities, and exactly

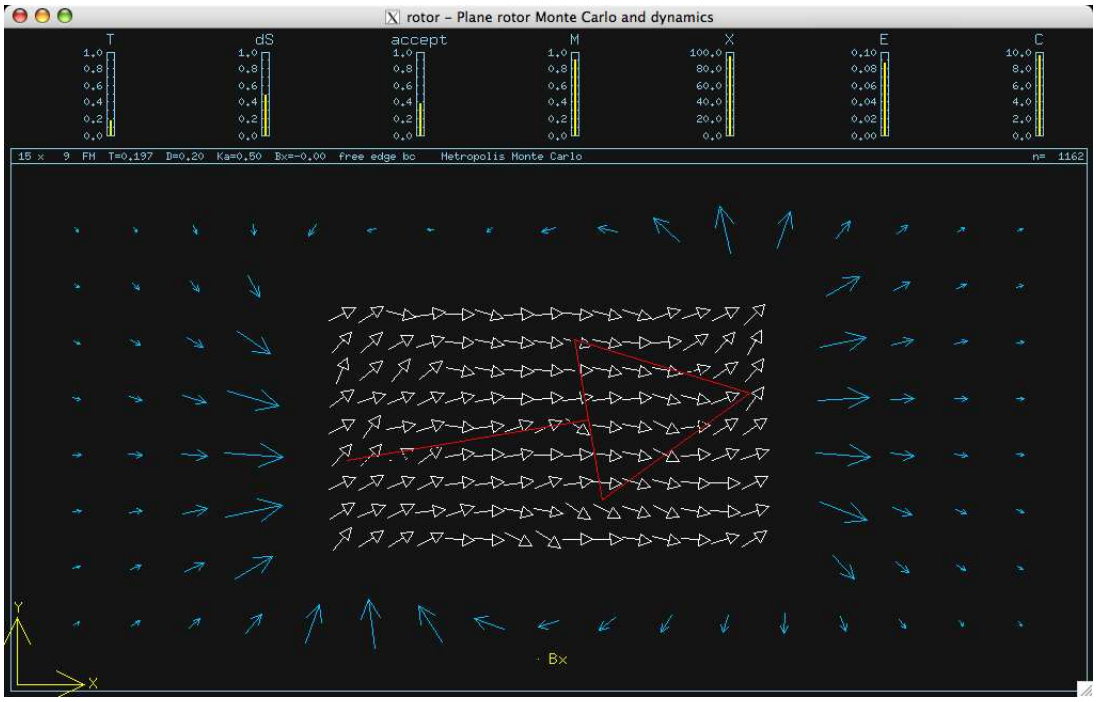

Figure 1: The rotor program. A view of the magnet at low temperature, where the dipoles are well-organized and there is a large net magnetization (the large red arrow). The blue arrows farther out are the magnetic field caused by this magnet.

what will be used in the final published PhET simulation is yet to be determined, after further discussions with the team.

The final simulation program that I wrote (rotor) can be summarized as follows: The program displays how the dipoles inside a magnet form domains (aligned regions) and how the domains move and re-arrange under the action of applied fields and changes in temperature. It displays the magnetic field lines around the magnet, which change as the internal dipole arrangement changes. The net magnetization (average of all the dipoles) is shown as an arrow that grows as the magnet becomes more aligned or magnetized. The user can "demagnetize" the magnet, simulating hitting it with a hammer (which de-organizes the dipoles), by re-initiating with a random dipole configuration. It will subsequently re-organize to a magnetized state (Fig. 1), if the temperature is low enough, or if there is an applied field of sufficient strength

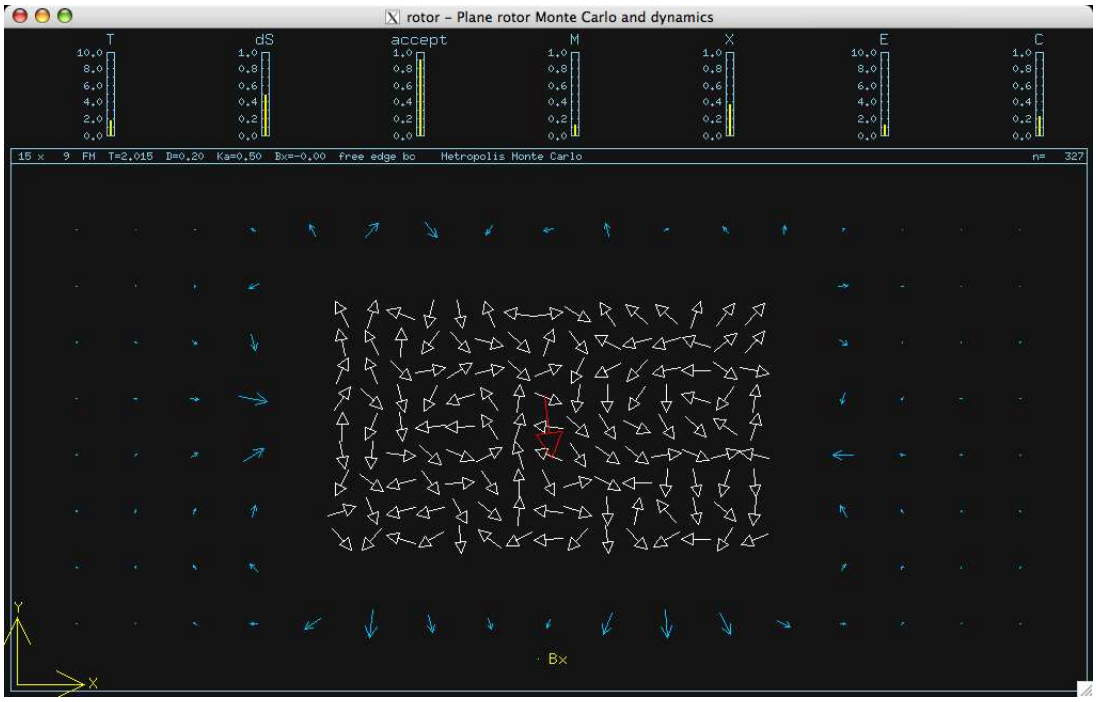

Figure 2: A view of the magnet at high temperature, where the dipoles become more disordered, and the net magnetization is small (the red arrow).

present. The user can also see what happens to a magnetized magnet if heated–the dipoles will become randomly aligned, hence, the magnet gets demagnetized (Fig. 2). Other effects, such as precessional motion of the magnetization in an applied field, can be observed in the Langevin dynamics. The user will get a good impression of what is happening within a magnet, and especially, see how its "magnetization" is really not a constant, but depends on what you do to the magnet.

Overall, the project was a success, as in a very short time I was able to implement essentially everything that is interesting in the description of a magnet at an introductory level. Eventually, the PhET developers will convert the program to Flash or Java, and it will be published to the PhET web site. Then it can be used in introductory science classes starting around the middle school to high school level. In the meantime, I may think whether I might get interested in other new simulations in statistical physics.