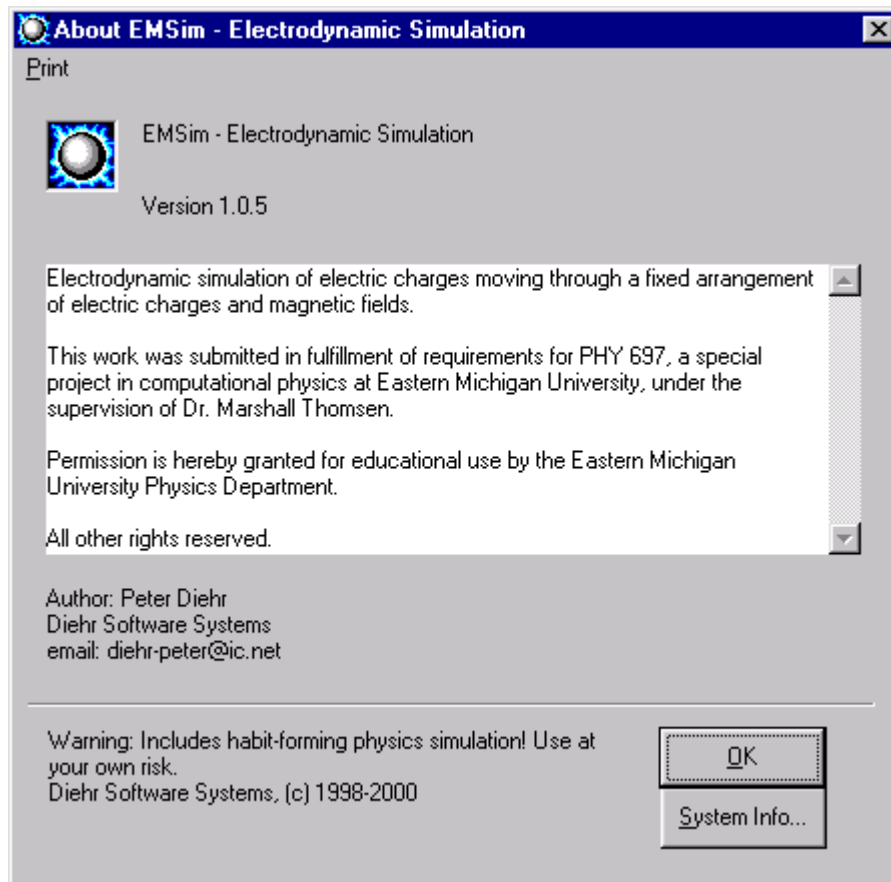


## Introduction



EMSim is an electrodynamic simulation program. It requires Windows 95 or later to run. It simulates the motion of electrically charged particles ("test charges") moving through a fixed arrangement of electromagnetic fields. These freely moving test charges obey the Lorentz force law,  $\vec{F} = Q(\vec{E} + \vec{v} \times \vec{B})$ , and Newton's second law of motion, in the form  $\vec{F} = m\vec{a}$ .

The resultant forces are summed for each test charge, and an integration process is used to find the test particle trajectories. The simulation setup allows test charges to optionally interact with each other.

Special Relativity is not taken into account; all interactions are taken to be instantaneous.

SI base units are used throughout.

## **Credits**

This work was designed and programmed by Peter Diehr, and submitted in fulfillment of requirements of PHY 697, a special project in computational physics at Eastern Michigan University, under the supervision of Dr. Marshall Thomsen. Programming was done using Microsoft Visual Basic 6.0.

Appreciation is expressed to Dr. Weidian Shen for helping me to learn electrodynamics. Thanks are given to Mark Diehr, for many constructive suggestions on the user interface, and for the development of the icons and the 3D image of a ball used by the trajectory viewer. Thanks are also given to Steve Swanson for many discussions of 3D graphics, and to Gary Robinson for much assistance in preparing the help files.

Permission is hereby granted to the Physics Department of Eastern Michigan University, for educational use of this program. All other rights are reserved to the author, for further information select "About" on the Help menu (press ^A).

Note: "Press ^A" means to hold down the CTRL key while pressing the A key.

## Structure of the Simulation

An object model is used throughout the design of EMSim. The main object is the electrodynamic simulation; the simulation object contains electric fields, magnetic fields, and test charges. The simulation object has properties such as the current time step size, and the number of simulation steps run. The simulation object is able to "tokenize" its properties, so that they can be externally saved, and later restored. The user interface saves the tokenized simulation as a text file with extension .ems.

Each test charge object includes a child electric field object, which holds its charge and current location, as well as a trajectory history object.

In order to interpret the output of the simulation, the user interface allows you to view the test charge trajectory histories, or save them for later analysis. Details are given in Simulation Operations below.

Coordinate-free notation fits naturally with the object structure of this simulation. A vector object is used to implement the mathematical requirements. Internal vector operations always use rectangular coordinates.

The user interface allows the user a free choice of rectangular, cylindrical, or spherical coordinates for all displays, as well as a choice of degree or radian notation for angles. All values are presented in SI units.

## Mathematical Notation

Parameters of equations appear in bold face type when they appear in the text.

$\vec{R}(\vec{P})$  Vectors are denoted by overbar arrows.

$\hat{A} = \vec{A}/|\vec{A}|$  Unit vectors are denoted by a hat.

$\langle (\vec{S} - \vec{P}) | \hat{A} \rangle$  Bra/ket notation is used for the inner product.

## Physical Model

Electric fields are defined by EField objects. The field is specified as a point charge of magnitude  $Q$ , and its location  $R$ . The field strength at  $P$  is then:

$$\vec{E}(P) = \frac{1}{4\pi\epsilon_0} \frac{Q}{|\vec{P} - \vec{R}|^3} (\vec{P} - \vec{R}).$$

Magnetic fields are defined by BField objects. The orientation of this field is given by a unit vector parallel to the axis,  $\hat{A}$ , and its location in space is fixed by giving a point  $S$  through which the axis passes.  $B_0$  is the field strength along the axis, and  $a$  is a radial decay rate; zero indicates a solenoidal field. The distance from location  $P$  to the axis is given by  $d$ :

$$\vec{R}(P) = \left\langle (\vec{S} - \vec{P}) \mid \hat{A} \right\rangle \hat{A} - (\vec{S} - \vec{P}),$$

$$d = |\vec{R}(P)|.$$

For a BField object with axis  $\hat{A}$ , axial field strength  $B_0$ , and radial decay constant  $a$ , the field strength at  $P$  is then:

$$\vec{B}(P) = B_0 \cdot \exp(-a \cdot d) \hat{A}.$$

Test charges are defined by TestCharge objects, which specify a point charge (magnitude and location), along with mass and initial velocity. You can also specify whether or not test charges are allowed to interact.

For each step of the simulation, each test charge is requested to tell the net force acting upon it. The test charge does this by requesting each of the EField, BField, and interacting TCharge objects to calculate the field at the current location of the test charge. These fields are summed (principle of superposition), and the Lorentz force law is used to find the net force acting upon this test charge. If the test charge magnitude is  $Q$ , with current velocity  $\vec{v}$ , and location  $P$ , then the net force is:

$$\vec{F}_{net}(P) = Q \cdot [\vec{E}_{net}(P) + \vec{v} \times \vec{B}_{net}(P)].$$

Newton's second law of motion is used to find the acceleration for this test charge of mass  $m$ :

$$\vec{a}(P) = \vec{F}_{net}(P) / m.$$

Updated velocity and position for each test charge is then calculated by an integration step. The user has a choice of Runge-Kutta integration methods.

## Runge-Kutta Integration Formulas

The first order Runge-Kutta integration formula is Euler's method with advanced velocity. After solving Newton's second law of motion for the acceleration, we simply apply our small time step,  $h$ , first to obtain a new velocity, and again to obtain a new location:

$$\begin{aligned} \mathbf{v}_{new} &= \mathbf{v} + h \cdot \mathbf{a}(P), \\ \mathbf{P}_{new} &= \mathbf{P} + h \cdot \mathbf{v}_{new}. \end{aligned}$$

The fourth order Runge-Kutta integration formulas are taken from Abramowitz and Stegun, "Handbook of Mathematical Functions", section 25.5.20. Though the error term is  $O(h^5)$  for the position, it is only second order in the velocity; this is due to the structure of the equations,  $y''=f(x,y,y')$ , which requires a double integration.

The fourth order forms require evaluation at intermediate steps to obtain values  $k_1$ ,  $k_2$ ,  $k_3$ , and  $k_4$ :

$$\begin{aligned} k_1 &= h \cdot f(x_n, y_n, y'_n), \\ k_2 &= h \cdot f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}h \cdot y'_n + \frac{h}{8}k_1, y'_n + \frac{1}{2}k_1), \\ k_3 &= h \cdot f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}h \cdot y'_n + \frac{h}{8}k_1, y'_n + \frac{1}{2}k_2), \\ k_4 &= h \cdot f(x_n + h, y_n + h \cdot y'_n + \frac{h}{2}k_3, y'_n + k_3). \end{aligned}$$

We advance the trajectory by the following formulas:

$$\begin{aligned} y_{n+1} &= y_n + h \cdot [y'_n + (k_1 + k_2 + k_3) / 6], \\ y'_{n+1} &= y'_n + (k_1 + 2 \cdot k_2 + 2 \cdot k_3 + k_4) / 6. \end{aligned}$$

The application of these formulas to each test charge is done by equating  $y''$  to the acceleration of that test charge, which makes  $f(x,y,y')$  the Lorentz force divided by the mass of that test charge.

## Simulation Operations

The program consists of a control panel, a visible log, and a trajectory viewer. The control panel supports:

- Loading a simulation from a previously saved .ems file
- Saving of the current simulation to an .ems file
- Running the current simulation
- Restarting the current simulation
- Modification of the current simulation
- Updating the Field Objects
- Saving the trajectory history of the current simulation to a .csv file
- Viewing the current activity log
- Viewing the current simulation trajectories

### Loading a Previously Saved Simulation

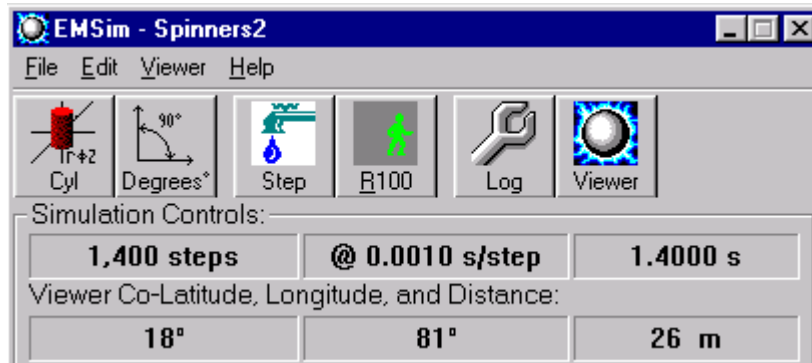
You can load a previously saved simulation via the "Open Simulation" option of the File menu (press ^O). This presents a file dialog, and shows the names of available simulations; their file extension is .ems.

Loading a simulation causes the current simulation to be lost. The loaded simulation always starts at step zero.

### Saving the Current Simulation

You can save the current simulation via the "Save Simulation" option of the File menu (press ^S). This presents a file dialog, and shows the names of available simulations; their file extension is .ems. The name of the current simulation is used as the default file name.

## Running the Current Simulation



The control panel always shows the number of steps taken, the current simulation step size (seconds per step), and elapsed time in separate panes of a status bar labeled "Simulation Controls". You can change the step size by clicking on the corresponding pane, or via the "Time per Step" option of the Edit menu (press ^T).

The current integration method is indicated by a check mark on the Edit menu: 1<sup>st</sup> Order Integrator or 4<sup>th</sup> Order Integrator. You can select either via the short cut keys (press ^F1 or ^F4).

To step once through the simulation, press the "Step" tool bar button. Each of the currently active test charges will "move" the current time step. Their new position will show in the activity log, and on the trajectory viewer.

To run multiple steps, press the "Run" tool bar button. The caption tells you how many steps it will run. You can change the number of steps to run by clicking on the "Steps Run" pane of "Simulation Controls", or via the "Steps to Run" option of the Edit menu (press ^N). A progress bar temporarily replaces the status panes. After the requested simulation steps have completed, the new positions of each active test charge will show in the activity log, and on the trajectory viewer.

During the execution of simulation steps, the "Run" button is changed to a "Stop" button. You can press the "Stop" button to halt the simulation after the completion of the current step. All simulation displays are updated after that step is completed.

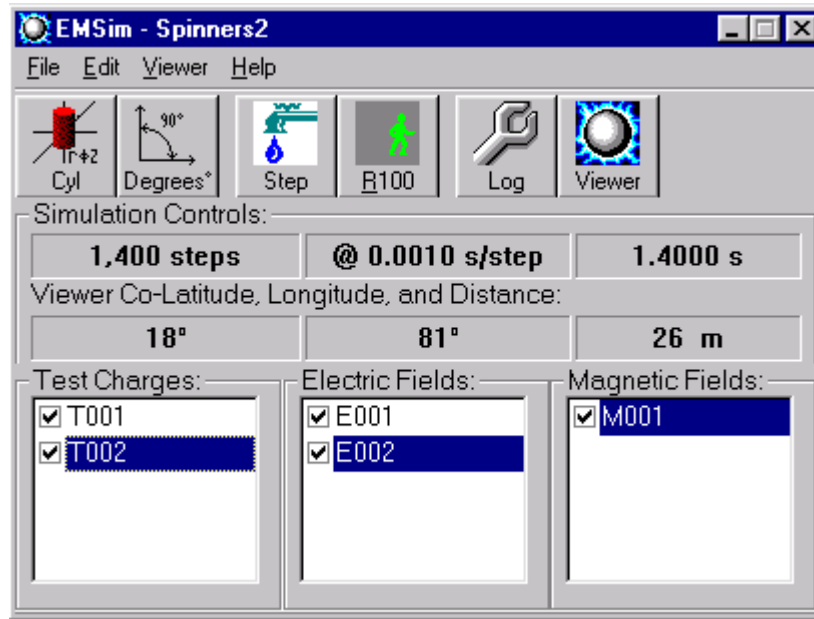
While you may alter any of the simulation parameters whenever the simulation is not running, this may give non-physical results. However, when used in conjunction with "Restart Current Simulation", this is very useful in developing a new electrodynamic simulation.

## Restarting the Current Simulation

You can restart the current simulation via the "Restart Current Simulation" option of the Edit menu (press ^R). The restart option resets the step counter and elapsed time to zero, and restores each extant test charge to the location and velocity it had as of the last time the step counter was zero. All other simulation parameters remain as they are; this includes the current time step size and integration method, and the definitions of the electric and magnetic fields.

The "Restart Current Simulation" option allows you to explore systematic changes to parameters while preserving initial location and velocity. If you wish to restore all parameters, you can reload the simulation from its .ems file.

## Modifying the Current Simulation



The objects that make up the current simulation are listed by field type. The simulation ignores them unless the "Status" box is checked; if checked, they are considered to be active.

Right click within the list box to pop up the corresponding option menu: Active/Create/Delete/Update; the choice (and its corresponding shortcut key sequence) works for the currently selected item within that list box.

Select "Active" (left click, or press space) to toggle the active state; the simulation ignores inactive objects.

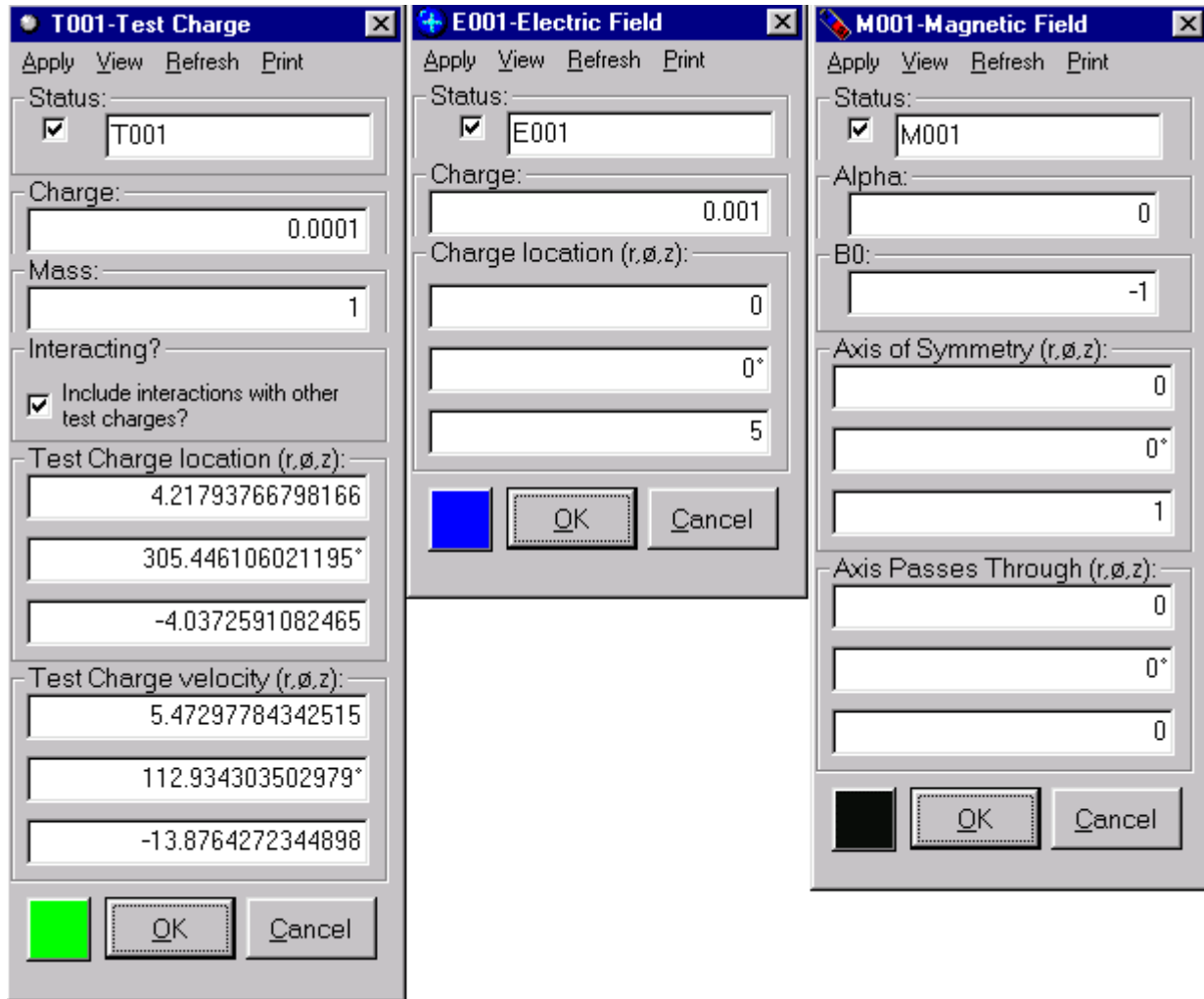
Select "Create" (press Insert) to create a new object; a name will be assigned automatically, and you will be transferred to its update form.

Select "Remove" (press Delete) to remove the selected object permanently. If the update form is being shown, this menu choice is omitted; you must first exit that form.

Select "Update" (press Enter) to transfer you to the update form for the selected object. Multiple update forms may appear on the screen at the same time; the screen location is considered part of that field's object description, and is remembered if you save and later restore that simulation. The update forms are not displayed when a simulation is first loaded; you must do this manually.

Selecting the "Print" menu option (press ^P) causes this window to be printed, sans caption.

## Updating the Field Objects



Information is typed directly onto the corresponding update form. SI base units are used throughout; reminders appear in the tooltips.

Selecting the "Apply" menu option causes the corresponding object to be updated immediately, even if the simulation is running. Selecting the "Refresh" menu option causes the current values of the object to be displayed. This is convenient way to interact with the simulation during its development.

Selecting the "View" menu option allows you to set the line style (including invisible if you need to get a better view of other trajectories), and line width (press  $\wedge F1$  through  $\wedge F9$ ), as well as color (press  $\wedge C$ ). Color can also be modified by pressing the colored button at the bottom of the form to display a color palette; the selected color is used in the trajectory history viewer.

Selecting the "Print" menu option causes the currently selected window to be printed, sans caption.

Use the toolbar buttons on the control panel to instantly change the coordinate system used for entry and display; you may choose rectangular, cylindrical, or spherical coordinates, and angles may be presented in radians or in degrees. Internal storage and calculations are always in rectangular coordinates. You may notice rounding errors (in about the 14<sup>th</sup> significant digit) due to the limitations of double precision floating point, which is used for the storage of coordinates.

## Saving the Trajectory History for the Current Simulation

You can save the current simulation trajectory information via the "Save Trajectory History" option of the File menu (press ^H). This presents a file dialog, and defaults to the name of the current simulation with file extension .csv.

The current simulation includes a trajectory history for each test charge. This consists of the location and velocity of that test charge for each time step of the simulation. This can be saved to a .csv file, which is a text file with quoted strings and comma separated values. The format is suitable for importation into Microsoft Excel.

After importing the data into Excel, you will see a few rows describing the simulation, followed by a row of column headings like:

Trajectory History for Simple Cyclotron saved in C:\EMSim\Simple Cyclotro								
Cylindrical coordinates (degrees)								
T001	Time	r	$\phi^\circ$	z	v r	v $\phi^\circ$	v z	Energy
0	0	1	0	0	1	90	0	0.5
1	0.02	1	1.145992	0	1.00005	90.5729	0	0.50005

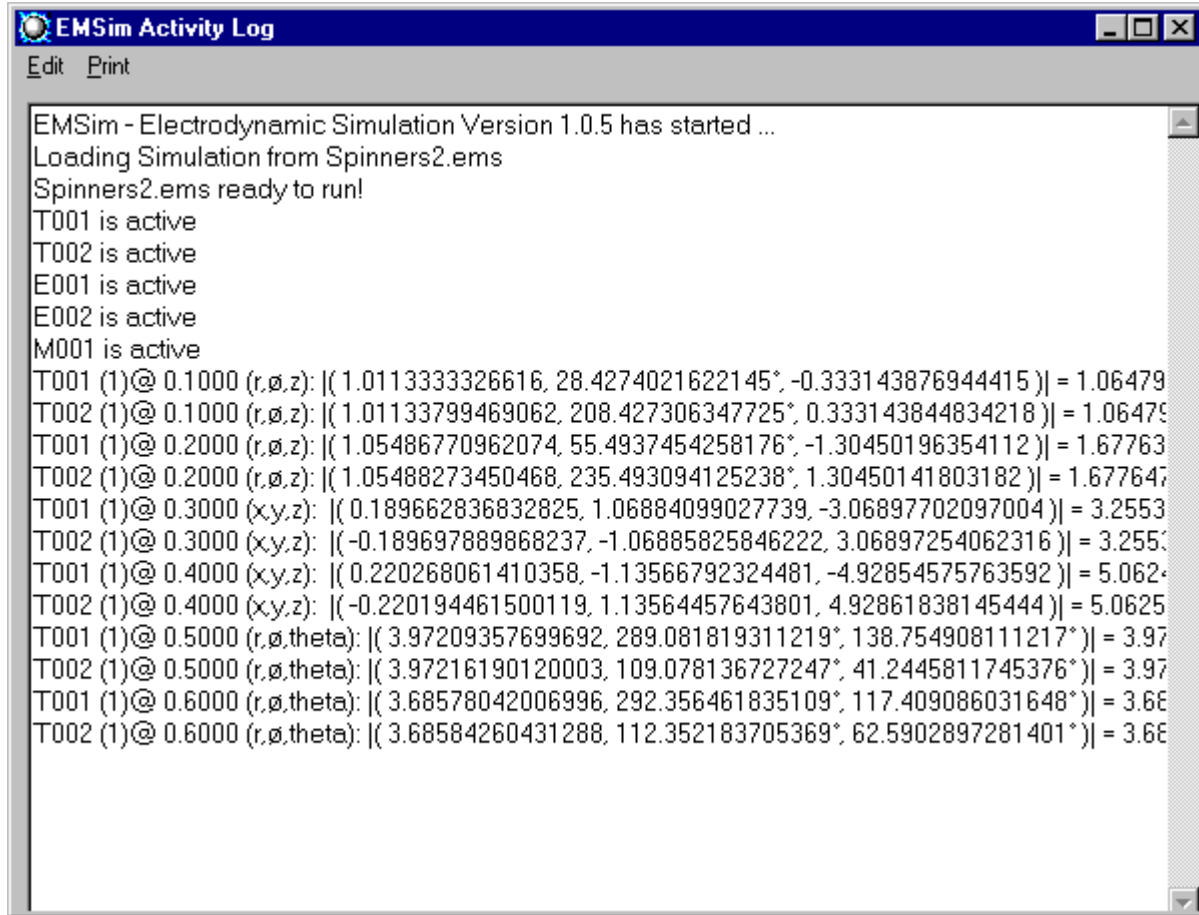
The first column heading contains the test charge identifier; the simulation step counter is listed in this column. Missing steps indicate that that test charge was disabled at that time. The next column tells the elapsed time corresponding to that step.

The next three columns give the position coordinate; the headings indicate the coordinate system in which the trajectory history is displayed. The following three columns show the velocity in the same coordinate system. The final column is the kinetic energy,  $1/2 mv^2$ , for that test charge.

If there are multiple test charges, they will be found further down. Test charges are listed sequentially, and not side-by-side. Inactive test charges will list only step zero, which gives their initial position and velocity.

The coordinate system used is the one in effect when the save was done.

## Viewing the Current Activity Log



The activity log can be selected by pressing the "Log" tool bar button, or via the "Display Activity Log" option of the Edit menu (press ^L). This action will cause the activity log window to conform to the width and location of the simulation control panel. Select **Clear Activity Log** from the Edit menu (press ^C) to clear the activity log.

Besides some general status messages concerning the status of operations, the location of each test charge is reported after the completion of each simulation run. If you want to examine the results after each step, view the trajectory history, or press "Step" repeatedly.

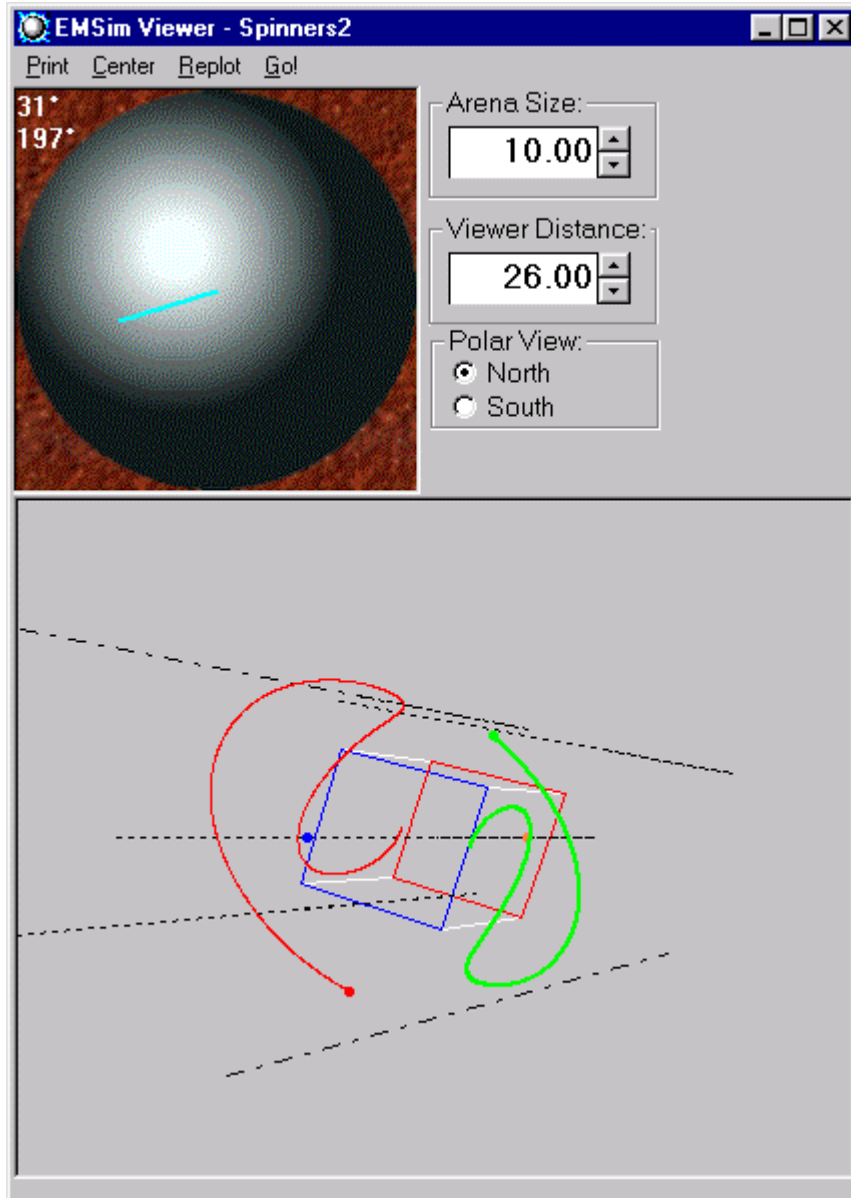
The simulation step information includes the test charge identifier, the integration mode in parentheses (1 or 4), then an '@' followed by the elapsed time. The current coordinate style is then given, followed by the location coordinates, and finally the distance from the origin is given. When the coordinate system includes angles, the ° symbol indicates degrees; otherwise radians are being used. Several of these features are shown in the above screen capture.

The information in the activity log is useful during the development of a simulation. If you wish to save the contents of the activity log, choose the "Copy All" option of the Edit menu (press ^A); you can then insert the text into any open document. This includes all items in the log, even those which have scrolled off of the screen. The Print menu option only prints the portion of the log currently visible on the screen (press ^P).

Analytical work is best done from the trajectory history file.

### Viewing the Current Simulation Trajectories:

The trajectory viewer window can be selected by pressing the "Viewer" tool bar button, or via the "View Simulation" option of the Viewer menu (press ^V). Other options include Grids (press ^G) which indicates the XY, YZ, and ZX planes, and Box (press ^B) which draws a reference cube, with the positive Z end colored blue (North Pole), and the negative Z end colored red (South Pole).



The view ball is used to select viewing angles; its center is also the center of the simulation coordinate system. Perspective projection is used, based upon the viewer's location on the view ball. Viewer distance is the effective radius of the view ball. If you save the simulation, all current viewing options are also saved.

You can make the center of the view ball image either the North Pole (positive Z), or the South Pole (negative Z). Any angular position can be obtained; your longitude and co-latitude are shown in the upper left corner of the view ball image.

## EMSim: Electrodynamics Simulation

Arena size establishes the effective radius of the simulation, and the scale factors for the viewing area; if you set it too small, your particles are likely to leave the simulation, causing those particles to be de-activated, and the simulation will stop. But if you make it too large, you won't be able to see any of the details.

The Print menu option (press Alt-P) causes the entire viewer screen to be printed.

The Center menu option (press Alt-C) reestablishes viewing from directly above the North Pole.

The Replot (press Alt-R) menu option causes the graphics to be completely redrawn; you can also click on the viewing area. As all graphical updating is automatic, you shouldn't need to use this.

The Go! menu option (press Alt-G) causes the simulation to run the established number of steps, and the trajectory view to be automatically updated.

### **Note:**

The simulation shown here is Spinners2.ems, which consists of two interacting charged particles, one assigned the color red, the other green. A widening of the trajectory lines marks their current locations. A magnetic field is active, and is shown in black: the central axis and four parallel lines. Additional fixed charges are blue and orange; they show up as large dots because their pixel width has been set to 5.

The Box option has been turned on; this can help orient you to the image.

Note that you can temporarily remove any particle by setting its line style to invisible.

## 3D Graphics Theory

### Perspective Projection onto Viewer Plane

The 3D simulation trajectory coordinates are projected onto the screen viewing area (viewport) by means of a perspective transformation, which is derived from the properties of similar triangles. If the viewport has an XY coordinate system, with origin in the center, then a given simulation point (X, Y, Z) is projected to a viewport point (X<sub>S</sub>, Y<sub>S</sub>) by means of the following transformation:

$$\begin{aligned} X_S &= V_{SX} * (\text{ViewerDistance}/\text{ArenaSize}) * (X/Z) + V_{SX} \\ Y_S &= V_{SY} * (\text{ViewerDistance}/\text{ArenaSize}) * (Y/Z) + V_{SY} \end{aligned}$$

Where ViewerDistance and ArenaSize can be set by the user; V<sub>SX</sub>, V<sub>SY</sub> are the viewport width and height in pixels.

Since 3D lines project to 2D lines, it is sufficient to project the endpoints of any line segments, and draw 2D lines between them.

### Rotation of Viewer Plane

If a line extends behind the viewer, it is clipped at the viewing plane, and the exit and entry points through the clipping plane are connected. This shows as lines running along the edge of the viewport. The most complete views are those where the viewer distance exceeds the used portion of the simulation arena. Perspective distortion is apparent whenever the viewer distance is too small.

The above description assumed that the simulation (X, Y) coordinates are horizontal and vertical with respect to the viewport, and the Z coordinate represents depth. The user is allowed to select alternative views in terms of position upon a view finder ball, specified as co-latitude  $\mathbf{q}$ , (polar angle) and longitude  $\mathbf{j}$ , (azimuth). The simulation coordinates are then transformed so that the viewer's specified location is brought to the viewport (or equivalently, that the viewport is carried to the viewer's specified location).

The transformation is built up as  $T = R_Z(\mathbf{y}) R_Y(\mathbf{q}) R_Z(\mathbf{j})$ , where  $\mathbf{y} = -\mathbf{q}$  is the third Euler angle. The actions to take are as follows: rotate about the Z-axis by our longitude,  $\mathbf{j}$ ; then rotate about the new Y-axis by our co-latitude,  $\mathbf{q}$ . The third angle,  $\mathbf{y}$ , is required to remove tilt from the object.

### View Ball Operation

A click on the view ball is read as an (X, Y) coordinate within that viewport, which has a half-width of R. If  $X^2 + Y^2 \leq R^2$ , we can assume that this (X, Y) is a projection of the view ball onto a plane, and using the full equation for a sphere of radius R, we solve for Z.

Then if we have normalized the coordinates with the radius of the sphere, the co-latitude is

$\mathbf{q} = \text{ArcCos}(Z)$ , ranging from 0 to 180°, and longitude is  $\mathbf{j} = \text{ArcTan}(Y / X)$ , ranging from 0 to 360°.