Using OpenDoc To Create Low-Cost Physics Simulation Tools For Secondary and Higher Education

Scott Tooker and Rodney Cole
Physics Department
University of California, Davis
Davis, CA 95616

Abstract

Currently, while there is a large selection of education physics programs available, few present an inclusive approach to physics simulation. Those that do are usually hard to use and/or very expensive to purchase and keep updated. To address this our goal is to develop a low–cost set of modular physics tools using the OpenDoc Framework [1] that are easy to use and customize, allowing both students and educators to set up simulations of various force–based problems. Tool palettes will be modular so that a simulation can be customized for the specific problem, thus hiding unwanted complexity from the student. Also "cameras" can be placed in the system to provide multiple frames of reference when viewing the simulation.

The first phase, currently under way, is to develop tools that allow the student or educator to simulate point particles influenced by gravitational and electromagnetic forces. Hooks will also be in place to add further functionality through future modules (spring force, harmonic motion, extended objects).

Concept

The set of physics simulation tools that we are creating contain four main components: systems, objects, forces, and views.

The system is the part that represents the "universe" that our simulations occur in. The user can set the shape, extent (finite versus infinite), and global properties (μ_0, ε_0, c, G, e) for a given system.

Objects are those things that forces act upon. In theory, objects can be of any shape or size, but our first goal is to accurately simulate point particles. Therefore, whenever objects are mentioned, they will be assumed to be point particles unless otherwise stated. The user can set the local properties of the object (m, q, ρ, μ, ε), the position and velocity of the object, and the forces allowed to act upon it.

The crux of our tools, forces impart a given acceleration to the objects they act on. The first version of our tools will ship with gravitational and electromagnetic force modules (with other force modules to be added in future releases). A force can be attached to a specific object or to the system itself (e.g. a uniform gravitational field). The user can set the magnitude and direction of the force (in addition to placing it on an object or system).

A view is simply a frame of reference for viewing the simulation. For example, the default system view is an inertial frame at rest with respect to the system where the "camera" is far away from the origin. However, the student or teacher can add "cameras" to the system that act as phantom observers, simply recording the outcome of the simulation. These cameras can be given arbitrary position, velocity, and acceleration vectors. This flexibility in placement allows the user to view different reference frames. For example, in a simple elastic collision the student could watch the simulation in both the lab and center–of–mass frames simultaneously.

Interface

The user accesses these parts initially by clicking on a system stationery document which produces a blank document containing a view from the system camera (for more information on the OpenDoc Human Interface see [2]). The system and its contents are portrayed in three dimensions (as are all the different parts) and contains a 3D Cartesian coordinate system which is used as a reference system when placing objects, forces, and cameras. At the bottom of the view is a group of buttons that allow the user to move, rotate, and zoom the system camera. There is also a group of VCR style buttons (play, stop, rewind, fast forward, pause, record) that allow the user to start the simulation, pausing or rewinding when needed. The user can also save any view content to a QuickTime file. Elaborating on the previous example, if the user is simulating a two–body collision, they would be able to not only view the different perspectives (lab and center of mass) simultaneously but they can also modify the cameras to provide different views and/or save views for later perusal or inclusion in other electronic documentation or presentations.

There is also a customizable floating palette that contains the tools used in constructing simulations. The user can add custom objects, forces, and cameras to it, allowing for easy construction of custom simulations. The palette can then be saved for later use. As a default, only one palette would normally be shown, but the user can have as many palettes open as desired (barring memory constraints).

All objects are displayed as small spheres (an extended object would appear as the object itself). Force and velocity vectors appear as translucent arrows attached to objects that can be directly manipulated. Cameras appear as small three dimensional eyeballs. Each camera is connected to a window that corresponds to what it sees. This window has the same interface present on the system view (ability to zoom, rotate, and move, and the VCR buttons). Velocity and acceleration vectors for the camera are displayed as translucent vectors that can be directly manipulated by the user.

In addition, since we are developing these parts as OpenDoc components [2] we gain a shared file format, Bento, that allows us to easily share information with other components. One possibility is to "link" a view to a third party graphing component. Once we establish this "link", any numerical data calculated by the simulation can be plotted to provide the user with more information about the simulation. For example, the user could create a simulation concerned with radiating electric and magnetic fields and use the "links" mentioned above to map out the electric and magnetic fields over time using a graphing part. Other possibilities include adding text labels to objects, cameras, or forces through a third–party text editor, providing text, sound, or video in another section of the document to provide background and insight into the simulation, or provide hooks that allow our tools to utilize high–powered workstations or supercomputers on the Internet to perform the numerical calculations needed for more complex simulations.

However, probably the most important benefit of creating the components with OpenDoc is that they can be used in any container (OpenDoc terminology for a component that can contain other components) or application that supports OpenDoc. These tools can even be used on Windows with OLE, Microsoft's interapplication communication standard.

Implementation

Currently these tools are being written using Metrowerks' CodeWarrior 9 with the OpenDoc Framework on a Power Macintosh 7100/66AV, Power Macintosh 8500/120, and PowerBook 5300cs. In addition to OpenDoc, the tools will use Apple Computer's QuickDraw 3D technology [3] for 3D imaging and rendering including hooks to provide hardware–based 3D rendering. We are also considering using SoundSprocket, part of Apple Computer's Game Sprockets API [4], to provide the ability to simulate 3D sound sources.

The initial release of our tools will be available for any Macintosh capable of running OpenDoc and QuickDraw 3D. When available, the tools can be downloaded from http://maxwell.ucdavis.edu. We hope to support Windows and UNIX systems in future releases.

Conclusion

Although our first release has much of what is needed for basic force–based simulations, we hope to go much further than that. First, we plan to create more modules to expand the range of forces (spring force, harmonic motion) and type of objects (extended objects, objects that are system–like and can contain other objects). Then, we plan to make available a set of specifications detailing the format of these modules so that other educational programmers can contribute to this effort, leading to a rich set of powerful physics simulation tools available to students and teachers at minimal cost on a variety of computer platforms.

References

1. http://www.devtools.apple.com/odf/overview.html
2. Apple Computer. *OpenDoc Programmer's Guide.* 1995.
3. Apple Computer. *3D Graphics Programming With QuickDraw 3D.* 1995
4. http://dev.info.apple.com/evangelism/games/games.html