The Molecular Workbench Software: An Innovative Dynamic Modeling Tool for Nanoscience Education

Charles Xie and Amy Pallant
The Advanced Educational Modeling Laboratory
The Concord Consortium, Concord, Massachusetts, USA

Introduction

Nanoscience and nanotechnology are critically important in the 21st century (National Research Council, 2006; National Science and Technology Council, 2007). This is the field in which major sciences are joining, blending, and integrating (Battelle Memorial Institute & Foresight Nanotech Institute, 2007; Goodsell, 2004). The prospect of nanoscience and nanotechnology in tomorrow's science and technology has called for transformative changes in science curricula in today's secondary education (Chang, 2006; Sweeney & Seal, 2008).

Nanoscience and nanotechnology are built on top of many fundamental concepts that have already been covered by the current K-12 educational standards of physical sciences in the US (National Research Council, 1996). In theory, nano content can be naturally integrated into current curricular frameworks without compromising the time for traditional content.

In practice, however, teaching nanoscience and nanotechnology at the secondary level can turn out to be challenging (Greenberg, 2009). Although nanoscience takes root in basic physical sciences, it requires a higher level of thinking based on a greater knowledge base. In many cases, this level is not limited to knowing facts such as how small a nanometer is or what the structure of a buckyball molecule looks like. Most importantly, it centers on an understanding of how things work in the nanoscale world and—for the nanotechnology part—a sense of how to engineer nanoscale systems (Drexler, 1992). The mission of nanoscience education cannot be declared fully accomplished if students do not start to develop these abilities towards the end of a course or a program.

A major cognitive barrier for learning nanoscience is the lack of intuition. The nanoscale world is alien to students: electrons, atoms, and molecules are too small to be seen, their interactions resemble nothing in everyday life, and the phenomena are often counterintuitive. This is the world where electromagnetic forces, thermodynamics, and quantum mechanics govern (Drexler, 1988). There is nothing that students can assemble or tear apart with their bare hands in order to learn how these rules work. As a result, the ability to think abstractly is considered as a prerequisite. For this reason, teaching these topics is typically deferred to college level. Even when they are taught at colleges, instructors traditionally rely on some kind of formalism that is heavily based upon theoretical analysis.

Clearly, a less steep learning curve is needed for nanoscience education at the secondary level.

Wherever it is unrealistic to engage students with real experiments in the classroom, computer simulations stand out to be an attractive alternative (Feurzeig & Roberts, 1999; Panoff, 2009; Wieman, Adams, & Perkins, 2008). Unlike formal treatments that express ideas through mathematics, simulations express ideas through visualization on display devices and therefore are more likely to be comprehensible and instructive. This simulation-aided teaching is an increasingly important instructional technology as it adapts to today's students who grew up in an increasingly digital world and are more accustomed to visual learning. Good simulations can not only complement formalism to provide an additional, more accessible learning path to difficult subjects such as quantum mechanics (Zollman, Rebello, & Hogg, 2002), but in some cases, replace traditional treatments as a more effective teaching strategy (Finkelstein, et al., 2005). In addition, simulations are also cost-effective and scalable. They can be deployed online and run by hundreds of users at the same time.

This chapter presents lessons we have learned through the research, development, and classroom implementation of educational nanoscience simulations using the *Molecular Workbench* (MW) modeling software (http://mw.concord.org) developed by the Concord Consortium (Tinker & Xie, 2008). We hope these lessons will be helpful for science educators worldwide who are interested in adopting and developing interactive science simulations for better education.

Before going into details, we would like to clarify some terminology. The terms we are using are somehow overloaded with a number of subtly distinct meanings. Throughout this chapter, the word *animation* means a planned or scripted display of a sequence of images, the simplest case of which is a video. An animation cannot be changed by the viewer. As a result, all learners will see the same animation. Hence learning cannot be personalized. The word *computational engine*, or *engine* for short, stands for a computational system that does some calculations to create certain effects or solve certain problems. A computational engine is coded according to some generic scientific laws and therefore is capable of modeling a broad scope of phenomena. The words *model* and *simulation* will be used interchangeably in this chapter to represent an input to an engine that is configured to emulate a real world scenario. To inform the user, the results of a simulation are rendered as images on a computer screen. These images are often called *visualizations*.

A non-interactive simulation has no fundamental difference with an animation. But an *interactive simulation* has more illustrative power than an animation. Compared with an animation that can only illustrate situations recorded or preprogrammed, an interactive simulation can respond to students' inquiries in all possible ways permitted by the engine. If a picture is worth 1,000 words, you can imagine the information density and intelligence level of an interactive simulation. Furthermore, learning with an interactive simulation delivers a personal experience: each learner controls her/his own pace and manipulates the simulation in a unique manner. Learning through creating a simulation is even

more so (Papert, 1991). Each artifact a learner creates records her/his own learning process and contains her/his own thoughts. Empowering an unlimited number of simulations to be created, a good computational engine reflects a similar degree of variety, diversity, and complexity as observed in the real world. Because of its resemblance to a real experiment, a simulation is sometimes referred to as a *computational experiment*.

The word *activity* is used to describe a lesson based on one or more simulations. Besides simulations at the core and their inputs and outputs, an activity may consist of introductory stories, motivating questions, instructions, user interfaces, interactive tutoring, challenges, embedded assessments, and so on. These pedagogical elements scaffold a guided learning space in which the power of simulations can be maximally realized and controlled. Ideally, these elements should be seamlessly integrated with the simulations in the same environment so as to avoid the penalty of context and tool switching.

The importance of dynamic modeling

There are two major types of computer models in most science and engineering disciplines: *data model* and *process model*.

Data models are very common. For instance, Google Earth is a geographic information system driven by a data model consisting of data collected through satellite and land surveys. Those data do not change until the next survey. Another example is the Protein Data Bank that contains tens of thousands of structure data of macromolecules solved from crystallography. The structure data are coordinates (x, y, z) that define the positions of atoms in macromolecules. These data are static, characterizing stable conformations of the macromolecules when they were crystallized to be imaged.

Nature is not static, however. Our world is fundamentally dynamic. It is full of many different kinds of processes—diffusion, creep, flow, growth, propagation, reaction, explosion, and so on. We live in a four-dimensional world (x, y, z, t), not a three-dimensional one. Modeling the four-dimensional world is the purpose of process models. Dynamic modeling is the computational mean of studying process models. From a cognitive point of view, dynamic modeling converts an abstract concept to a salient show on the computer screen. This is already tremendously valuable since it is more likely to convey the knowledge to young learners. Yet the greatest strength of dynamic modeling lies in its capability of reconstructing what has happened and predicting what will happen. A manifest of this capability in the classroom is one of the most exciting and inspiring teaching moments in science education.

In comparison, static data models lack this prediction capability. Molecular visualization (José & Williamson, 2005) is an example that shows the limitation of using only static data model in education.

Since the invention of molecular graphics, a subject that focuses on visualizing molecules using 3D computer graphics, chemists have embraced molecular visualization tools capa-

ble of rapidly displaying molecular structures and viewing them from different perspectives. Several free tools have been developed for showing molecules on Web pages (Cass, Rezepa, & Rezepa, 2005). These tools are now widely used by educators to teach molecules.

Most molecular visualization tools, however, are mainly designed to show static structures. The user can rotate and translate the entire structure or change the view angle dynamically to create a motion effect, but the atoms do not move relatively to each other (which is what they are constantly doing in reality). Viewing static structures helps students learn the structures that are represented by data models, but it is often more important to learn the functions that are represented by process models. After all, we study molecules because we hope to exploit what they can do. This is especially true for nanoscience and nanotechnology that have an ultimate goal of creating nanostructures with functions we need. Although one can argue that in many cases there exists a strong structure-function relationship that can help people derive functions from structures, it is inappropriate to expect inexperienced students to be able to reason using the relationship that may be evident only to experts. Too often have we seen an excited chemist trying in vain to explain to non-experts what he or she sees in a 3D model of a molecule that is beyond a cool picture of some 3D structure to the non-experts. For example, a buckyball molecule is often used as an icon for nanoscience and there have been a lot of research and applications about it, but few educated people know what on earth this nano-sized particle can do except it looks somewhat like a soccer ball. (And by the way, what characteristic feature of a soccer ball gives rise to the utter importance of this beautiful molecule?)

Many basic concepts such as temperature, pressure, interaction, transition, and equilibrium can only be understood in terms of dynamic processes. It is desirable that a molecular process can "speak for itself" through a dynamic model to fill the cognitive gap. Some molecular visualization tools can sequentially display a series of frames, which can be different states of the same molecule observed experimentally or calculated numerically, to create an animation of a conformational change. This is a step forward to help students learn about molecular mechanisms, but it only shows what was set up to happen. For a tool to be more educational, students should be allowed to "mess around" with the models, try many hypothetical experiments, and see what happens. It is during iterations of this type of experimentation that students learn progressively and become inspired. This requires educators to develop computational engines that support interactive simulations of various molecular processes. The more powerful and interactive these engines are, the more students can learn from them. Ideally, they should be just as capable as the tools used by scientists and engineers (Israelewitz, Gao, & Schulten, 2001) and yet as easy to use as a typical computer application designed for average people. Such a capacity will provide unlimited learning opportunities to students and allow them to think and play like professional scientists and engineers. For example, when learning about the buckyball molecule, students can conduct a computational experiment to investigate if the molecule is toxic to human—if it can be easily translocated through a lipid membrane and absorbed into an animal cell (Wong-Ekkabut, et al., 2008). The inquiry at this level is much more profound than any possible inquiry design based on having students look at the static

structures of the Buckminster fullerene and a lipid bilayer and then try to reason about their interactions.

Why use first principles to build educational simulations?

A first principle is a foundational scientific law from which many phenomena can be explained and many propositions can be derived. For example, Newton's equation of motion is the first principle in classic mechanics—everything in the domain of classic mechanics can be explained by solving it using numerical simulation. This of course is the holy grail of science. But it is also important to emphasize the educational significance of simulations built from first principles.

The educational software market is largely dominated by cartoon movies, animations, and games. Many of these media were usually produced with visual effects as the paramount design goal in the developers' minds. There is seldom a need to exploit advanced mathematics and computation based on first principles. If the effective use of this power requires substantial training and investment, few commercial producers of educational media would be willing to take the financial risks. But this may change soon in science education, enlightened by the success of recent "killer applications" to be discussed below.

A strand of educational simulation programs for teaching mechanics, started with *Interactive Physics*[1] back in the 90s and significantly advanced by the recently released *Algodoo*[2] and *Crayon Physics*[3], have demonstrated great educational potential. These impressive programs allow users to draw a variety of 2D shapes, which then move realistically on the screen: they fall, slide, roll, and bump into each other—just like objects in the real world they model after do. These programs have a user interface that is very friendly to novices, especially with a freehand drawing tool connected to a digital pen on a tablet PC. With only a handful of tools, users can sketch up many interesting 2D simulations. Experienced users can build high-grade simulations as sophisticated as a vehicle impact test and a hovercraft takeoff. It is probably not an exaggeration to say that what users can create is limited only by their imagination.

There is no doubt that these entertaining tools truly motivate students, unleash their creativity, and make learning mechanics unprecedentedly enjoyable. But the important thing is that all these would not have been possible without using some high-end computational mechanics. The reason that these tools model the real world so well is because the motions of objects are calculated by solving Newton's equation of motion—to be more precise, using a computational method commonly known as multibody dynamics (Amirouche, 2006). In fact, *Algodoo* uses a computational engine called *SPOOK* devel-

[1] http://www.design-simulation.com/ip/index.php
[2] http://www.algodoo.com
[3] http://www.crayonphysics.com

oped by Dr. Claude Lacoursière (Lacoursière, 2007), and *Crayon Physics* uses a similar one called *Box2D* developed by Dr. Erin Catto (Catto, 2007). These multibody dynamics engines simulate interconnected bodies with contacts, joints, constraints, dry friction, and power input/output. *Algodoo* even has multiphysics capability by integrating multibody dynamics with the smoothed particle hydrodynamics for modeling fluids (Liu & Liu, 2003).

The multibody dynamics method was, however, not intended to be used in education. It was developed to design robots, vehicles, aircraft, and so on. The generations of computational scientists who contributed to the method presumably did not anticipate that one day the method would find its place in hundreds of thousands of middle and high schools all over the world. By the time we were writing about this, the total count of the floating point operations run in classrooms commanded by the educational tools mentioned above might have far exceeded that of those carried out for research in labs from all over the world added up together.

What does this teach us?

The first lesson we learned is that computational science is not a privilege of some scientists in ivory towers or engineers in the defense industry any more. In fact, science education and scientific research share a common goal: to understand how things work. It is, therefore, not surprising that a research tool like multibody dynamics can be so successfully converted into an effective learning tool. We would further contend that the only correct way to develop an educational tool would be to use the first principles in the corresponding domain of science as much as possible. The initial investment on such a tool may be high and risky (e.g., it needs dedicated computational scientists and programmers like those unsound heroes behind *Algodoo* and *Crayon Physics* to take their own risks for their careers), but the payback will be more powerful, useful software that can last for a long time and extend their outreach to millions of students worldwide.

The single most important reason for using first principles to build educational tools is that the power of creation and prediction embodied in these scientific principles will be given to *every* student. Such a tool can help students appreciate the unity of science—that everything can be derived from some commonality however their appearances and representations may differ. This is the most profound nature of science. What else is more important in education than passing students the greatest power and deepest wisdom brought to us by the most brilliant figures of the entire human race in hundreds of years? Now that the information technology has empowered us to deliver these intelligences through computing, an unprecedented opportunity to revitalize science education using this enabling technology is right upon us.

Unfortunately, this opportunity is often underappreciated in the educational world. The vision that the much-advocated cyberlearning infrastructure (Borgman, et al., 2008) should include smart media powered by first principles is not widely shared. Using computational science to build interactive media is not part of the design guidelines for the mainstream. Applications such as *Algodoo* and *Crayon Physics* are still scarce. There are

many more domains of science and engineering that need to be covered. Enormous volumes of literature have existed for how to simulate real world problems by numerically solving fundamental equations such as the Navier-Stokes equation for fluid dynamics and the Maxwell equations for electromagnetism and photonics. Sadly, there has been little investment and interest in making those powerful methods usable by students and the public at large, even in the face that the foundation of simulation-based engineering and science that gave birth to those methods is rapidly eroding due to the lack of student interest at the secondary level (NSF Blue Ribbon Panel on SBES, 2006).

Outside education, game developers have adopted first principles far more quickly and aptly. Games need to have realistic look-and-feels in order to be competitive in the market that always demands better realism. Major graphics libraries already provide excellent lighting functions. Realistic motions and flows powered by real-time physics engines (Bourg, 2001) and fluid solvers (Stam, 2003) are now not uncommon in games. *Algodoo* and *Crayon Physics*, despite their great educational power, are often billed as games. Hopefully, foundational open-source code libraries will be developed by the game industry and proliferate to eventually benefit educational software developers and changes will then occur.

The Molecular Workbench software

Nanoscience education aims at cultivating students' ability to reason about complex phenomena based on first principles governing the structures, interactions, and dynamics of electrons, atoms, and molecules. Dynamic modeling of nanoscale phenomena based on first principles provides a direct approach to making nanoscience more accessible and teachable in the classroom. Dynamic models render rich, salient views of the behaviors and interactions of electrons, atoms, and molecules that no microscope or ultrafast spectroscopy today can easily capture (see Fig. 1 for an example of water movement in a carbon nanotube). Through a carefully designed graphical user interface, a simulation tool allows students to manipulate a computational model, conduct various "what-if" computational experiments, and even design new simulations to test their own hypotheses.

Computational nanoscience (Rieth & Schommers, 2006; Wilson, 2003) involves sophisticated calculations using numeric methods such as molecular dynamics and quantum chemistry (Leach, 2001; Rappaport, 1997). These methods used to run on high-end computers that were not accessible to most students. But the computational power of ordinary computers today has increased to the point that these intensive computations can now be done on them to produce comfortable visualizations in real time. As computer power continues to be multiplied by multicore computing and supplemented by graphical processing units, it looks more and more feasible to create computational labs that complement wet labs in teaching and learning nanoscience.

The *Molecular Workbench* modeling software represents a decade-long effort towards the realization of this goal. Created from scratch using the Java programming language, *MW*

is a sophisticated software system that includes the modules discussed in the following sections.

The computational engines

MW's modeling power comes from its computational engines, two of which simulate phenomena at the nanoscale.

The molecular dynamics engine

The classic molecular dynamics engine in *MW* calculates the forces on atoms based on the interatomic interaction potentials such as van der Waals potential, electrostatic potential, and covalent bonding potentials (Xie & Tinker, 2006). The trajectory of each atom is then predicted by solving Newton's equation of motion based on the calculated forces on it. The results of the atomic motions are immediately rendered by using Java's graphics library and shown on the screen. The calculation and visualization are done in different threads so that a simulation utilizes both CPUs on a dual-core computer, which most computers today are. As a simulation runs in real time, the user can interact with it any time and see its response right away, making inquiry a straightforward process. This is a fundamental difference between the molecular dynamics engine in *MW* and other molecular dynamics programs that were not developed for education[4].

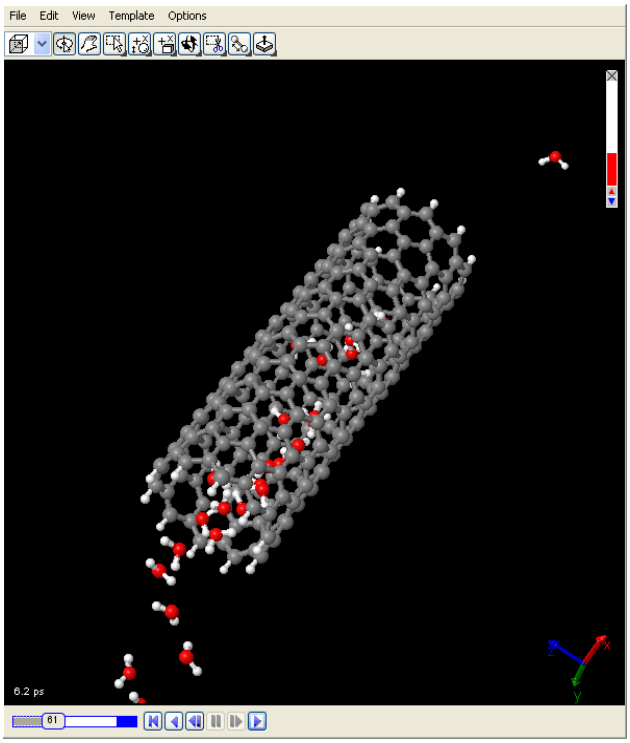

Figure 1: A screenshot of a molecular dynamics simulation of water molecules moving in a carbon nanotube using the 3D Molecular Dynamics Simulator of MW. This simulation can be used to illustrate the purification of water using nanotube filters.

Based on Newton's equation of motion, the molecular dynamics method guarantees a great deal of scientific integrity:

[4] For instance, see Gromacs: http://www.gromacs.org

- **The First Law of Thermodynamics**. This law, also known as the Law of Conservation of Energy, is automatically satisfied in a molecular dynamics simulation for an isolated system. If there is no energy input/output through external forces or dissipation through friction, the total energy, which is the summation of the potential energy and the kinetic energy for all the atoms in the system, remains constant within the tolerance of numerical errors. This can be used to check if a simulation runs properly.

- **The Second Law of Thermodynamics**. Molecular dynamics simulations of basic processes such as diffusion, heat transfer, and phase transition clearly show that the entropy of an isolated system always tends to maximize. In spite of the fact that it is possible to deliberately create special initial conditions that lead to a process of entropy reduction in an isolated system, in practice we have never found that such a condition can spontaneously arise during a simulation.

- **The Law of Momentum Conservation**. As the Law of Conservation of Energy, this law is also automatically satisfied in a molecular dynamics simulation. Together these two laws are responsible for making each collision among atoms look right on the screen.

- **Other laws**. A number of laws and conjectures in physics and chemistry that summarize insightful observations by generations of scientists, such as the Ergodic Hypothesis, the Theorem of Energy Equipartition, the Reversibility Paradox, Maxwell's Theorem of Speed Distribution, the Boltzmann Distribution, Fourier's Law of Heat Conduction, Raoult's Law, Van't Hoff's Law of Osmosis, Pascal's Principle, Archimedes's Principle of Buoyancy, and all the gas laws, can all be tested or proven using molecular dynamics simulations.

These laws and conjectures have been used as test cases for the molecular dynamics engine in *MW* to ensure its correctness. If knowledge is power, you can imagine how much power is made accessible to students when they have the molecular dynamics tool to play! Once again, this educational potential would not have been possible without the application of first principles.

The quantum dynamics engine

Quantum effects are fundamentally important in the nanoscale world[5]. Although quantum phenomena are very hard to understand, they are undeniably real and ubiquitous. Teaching quantum mechanics is so challenging that many nanoscience education programs choose to just scratch its surface or simply avoid it all together. How can we teach quantum reasoning to students without getting them bogged down in the complex mathematics of quantum mechanics and, quite possibly, the philosophical questions associated with its weird interpretations that are still at issues among scientists and philosophers? Recognizing all these learning difficulties, we set out to explore if computer simulation can deliver

[5] "We have become quantum mechanics – engineering and exploring the properties of quantum states. We're paving the way for the future nanotechnicians." – Donald M. Eigler, IBM Fellow

a pragmatic solution for *all* students to develop some quantum sense without having to learn through difficult mathematics.

Again, our strategy is to use first principles in quantum physics to build a simulation system that can teach. The quantum dynamics engine in *MW* solves the *time-dependent* Schrödinger equation using a fast and stable finite-difference time-domain algorithm (Watanabe & Tsukada, 2000). As with the molecular dynamics engine, a quantum simulation is performed in real time. The propagation of the wave function is calculated based on the Hamiltonian operator obtained from the potential fields contributed by objects that represent atomic and molecular structures. The calculated wave function at each time step is visualized on the screen immediately, rendering a continuous motion of the electron wave. Students can intervene at any time to adjust the potential fields or apply an external electromagnetic field and observe the change of the electron wave right away.

We would like to point out the value of dynamic modeling afforded by our quantum dynamics engine (see Fig. 2 for a dynamic model of the scanning tunneling microscopy). Most quantum mechanics simulations for education are based on solving the *time-independent* Schrödinger equation (Zollman, et al., 2002). Often, the visualization shows some stationary wave functions such as an atomic orbital. Visualization of stationary wave functions suffers from the same problem as with visualization of static molecular structures. The shape of an atomic orbital contains the information about the spatial probability distribution of electrons at certain energy level. Very little can be inferred from that piece of information. The result is that in many chemistry classes students are asked to memorize the shapes of the *s*, *p*, and *d* orbitals without necessarily understanding the implications of them.

The quantum dynamics engine is capable of simulating a variety of dynamic quantum processes: bound state, excited state, quantum transition, the formation of a covalent bond, chemical polarity, field-induced polarization, ionization, diffraction, interference, tunneling, quantum transport, and more. It is fascinating to see that these seemingly disparate concepts in physics and chemistry just emerge from quantum dynamics simulations that are based on a few basic assumptions—that electrons are represented by moving waves and the waves interact with each other and nuclei through electrostatic interactions! In fact, the quantum explanation of chemistry is among the greatest scientific discoveries in the 20th century, which was witnessed by several Nobel Prizes awarded to computational chemists.

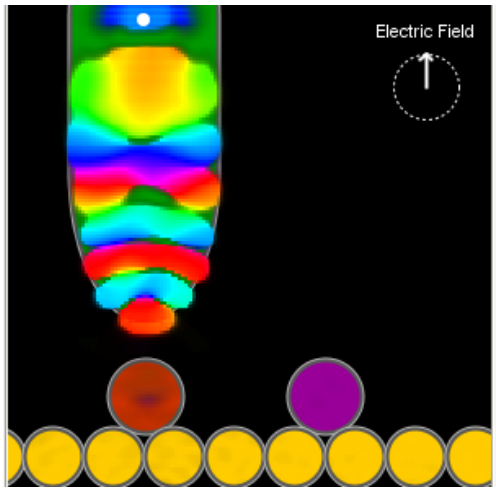

Figure 2: A screenshot of a quantum dynamics simulation of a scanning tunneling microscope that is an important tool in nanoscience. The wave function (colored by the phase) propagates in the tip and tunnels through the vacuum between it and an atom on the surface of a substrate, creating a weak electric current.

The modeling and authoring system

A unique strength of *MW* lies in its deep root in the software architecture that was designed to support model construction and activity authoring. Unlike tools that just offer existing simulations, *MW* gives the entire power of creation to users of different levels, in addition to a myriad of existing simulations available through it that anyone can pick up to use in the classroom. The constructionism strategy of engaging students to design simulations and teachers to create activities has been proven effective by a series of important work originated from MIT's Media Lab through a product line starting from *Logo* to *Scratch* (Colella, Klopfer, & Resnick, 2001; Kafai & Resnick, 1996; Monroy-Hernández & Resnick, 2008; Wilensky & Reisman, 2006). This capacity extends the application of *MW* beyond a provider of interactive simulations. Given the successful example of *Algodoo* and *Scratch*, further development of *MW* in this direction seems very promising. If at the macroscopic level multibody dynamics and fluid dynamics govern, then at the nanometer level the rulers become molecular dynamics and quantum mechanics. This perspective places *MW* at a strategically important position to become a universally useful tool in nanoscience education.

The designing of a simulation, which we call modeling, and the designing of an activity, which we call authoring, are the two distinct levels of creation in *MW*. At the modeling level, each computational engine in *MW* has its own set of user interfaces uniquely designed for building simulations of the corresponding type (see Fig. 3 for the user interfaces for creating 3D molecular dynamics simulations). At the authoring level, the user can create an activity by inserting a simulation and then setting up the pedagogical elements and assessment items around it.

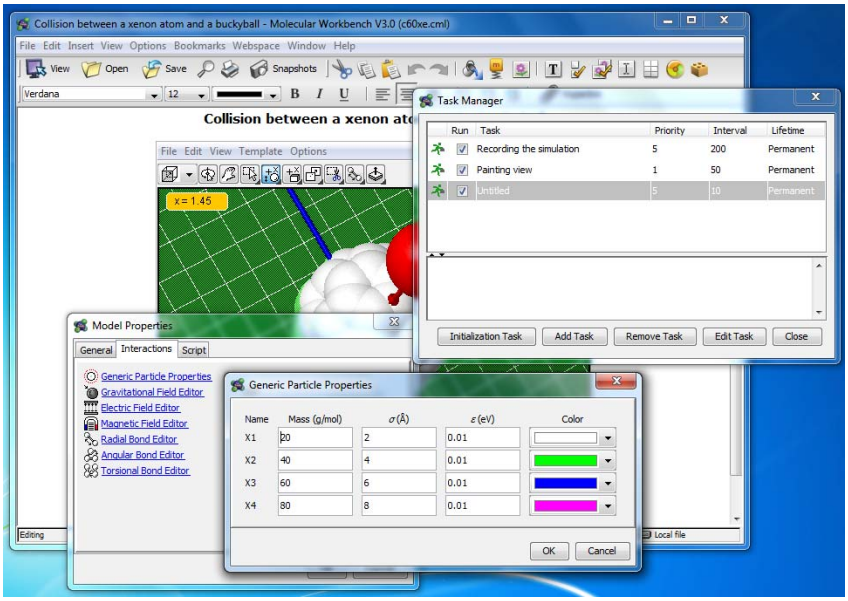

Figure 3: A screenshot of MW in the editing mode for creating a simulation-based activity, taken on Windows 7.

For both levels, graphical and drag-and-drop user interfaces were developed to allow entry-level users to get started with simple designs. For instance, with the interfaces for cre-

ating a molecular dynamics model, users can insert atoms and molecules, create bonds, apply external fields, set up boundary conditions, change temperature distribution, assign charges, add other objects such as images, lines, rectangles, ellipses, and triangles, copy/cut/paste/translate/rotate an object, edit properties of each object, and so on. With the interfaces for creating an activity, users can insert all types of standard components—buttons, check boxes, radio buttons, combo boxes, sliders, and spinner buttons—that can be used to interact with a simulation. Common data visualization components such as line graphs, bar graphs, and gauge graphs are provided to display outputs from a simulation. Widgets such as multiple choice questions and open-ended questions are available to design embedded assessment. An embedded assessment question can be placed right next to a simulation to form a compact user interface for interacting and learning.

Acknowledging the limitation of graphical user interfaces and the time constraint in perfecting them, a scripting environment was engineered to augment them. Based on an interpretive scripting language, this environment allows advanced-level users to design customized simulations and activities. For a simulation to have a customized emerging behavior, a scripted custom task can be added to the task pool of an engine. For an activity to have a customized user interface, scripts can be set for its widgets to customize the interactions they invoke. This feature of the *MW* system is particularly useful as it gives advanced-level users the ability to do anything allowed by the scripting language.

In addition, *MW* is an extensible platform that supports plug-in's written in Java such as an applet. This makes it possible for any author to easily incorporate as many types of computational engines as he or she needs without comprising the stability of the main system and other modules or engines. As a matter of fact, the entire quantum dynamics engine was designed as an applet that runs in both *MW* and a Web browser.

The delivery system

MW simulations and activities can be deployed in a number of different ways. Depending on the need of the educational developers who use them, there are three main ways to publish an *MW* creation:

- **Java Web Start**: An activity can delivered through the Java Network Launching Protocol (JNLP) supported by most servers and browsers. Once a JNLP link is clicked on a Web page, *MW* will be automatically launched to load the activity.

- **Applet**: An activity can be deployed on the Internet as an applet. Users can insert an *MW* applet in a Web page, a forum thread, a blog entry, or a Wiki page, and use JavaScript to integrate it with other Web applications written in Ajax and HTML5[6]. This capacity allows an educator to use any *MW* simulation on his/her own educational Web sites.

[6] http://molecularworkbench.blogspot.com/2010/02/mwscript-javascript-interaction.html

- **Embedded software**: *MW* is a Java system that can be conveniently used as a piece of embedded software in any open-source Java project.

The flexibility of deployment addresses different needs of educators and therefore opens up opportunities for wider adoption and dissemination of *MW* simulations and activities.

The assessment system

MW has a unique assessment system that is an integral part of the software. It supports standard instruments such as multiple-choice questions and open-ended questions. The most important innovation of the assessment system is the image question, which is our original contribution and will be introduced in the following.

The proverb "a picture is worth a thousand words" underpins the importance of visualizations in science education. Visualizations of large amounts of complex data can effectively convert the information into recognizable patterns that can be absorbed by students quickly. A picture is worth a thousand words for assessment, too. As much as the visualization of a science concept helps students gain a deep insight about it, the visualization of a student's work can help researchers gain a deep insight about how students learn. The visualizations created by students reveal what they observe, how they interpret the data being visualized, and which levels of their understandings about the concepts are. Analyzing these results, important feedback can be provided to curriculum developers about how well the learning goals have been achieved, to teachers about what their students have learned, and to researchers about how effective the pedagogy is.

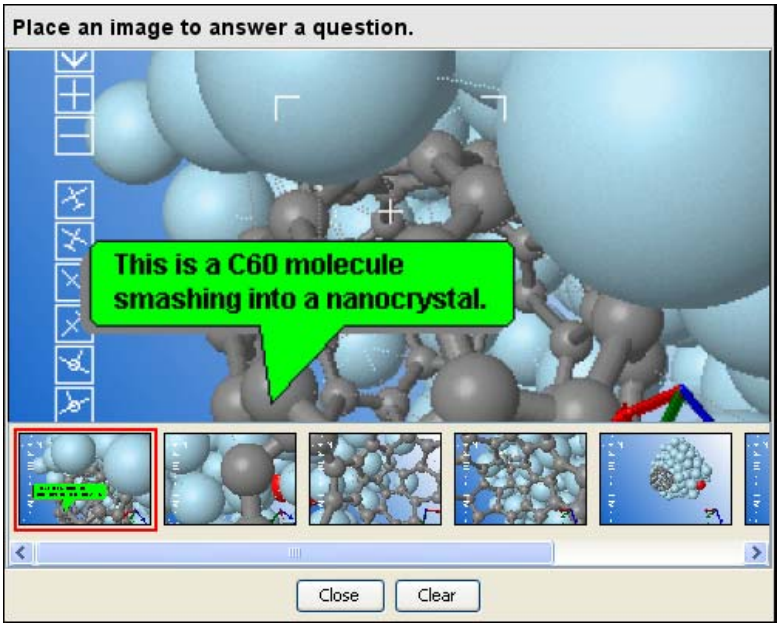

Figure 4: A screenshot of an image question. The student has to take a number of snapshot images that record his/her observation with a simulation and pick one that best answers the question. In the above image, the lower panel shows the thumbnails of some snapshot images taken in a hypothetical learning situation, from which the student will drag one and drop into the middle panel to make a selection. The green text box is an annotation that was added to the snapshot image to mark an observation.

An image question has a question, which is set by a curriculum developer, and a container,

which a student can fill with a snapshot image taken from a simulation to answer the question (see Fig. 4). A snapshot image can be annotated using a simple editing tool, which allows the student to write text and draw shapes. Answering an image question is similar to answering an open-ended question, except that the student generates an image instead of writing some text. A snapshot image the student chooses carries a lot of information that can be used to analyze the student's learning, especially when it is annotated.

Compared with other research instruments, the image question has several advantages. First, the best way to study the effectiveness of visualizations is to use the visualizations themselves. The image question provides a tool for students to describe, explain, and show what they learn from visualizations, using a snapshot tool that captures a scene of interest and an annotation tool that highlights the details on top of a snapshot image. Students thinking can therefore be made visible, assessable, and measurable. Second, the image question provides more reliable data for assessment. Unlike a multiple-choice question that could be randomly guessed and an open-ended question that could be "answered" through copy-and-paste, an image question can only be answered using an image that the student must take himself/herself during an activity. Third, the image question is more engaging to the student. Its intuitive graphical user interface allows the student to easily drag a thumbnail image from the snapshot gallery and drop it into the container.

Students' images for an image question are included, along with their answers to other types of questions, in a report that can be submitted or printed at the end of an activity. Their teachers will be notified of submissions and have access to students' work.

Results

By the end of 2009, *MW* has been downloaded over 500,000 times worldwide. Educators from all over the world have translated existing *MW* activities or developed their own in several languages including Chinese, Russian, Portuguese, Spanish, Italian, Hebrew, Norwegian, and Thai. Nearly 2,000 simulations and activities from students and teachers have been submitted to our databases. *MW* simulations have probably been run millions of times and benefited numerous students.

At the Concord Consortium, we have conducted several studies on student learning using *MW* activities, covering a broad range of content and involving students from middle school to community college. The results demonstrated that students who used well-designed activities achieved a solid understanding of atomic-scale phenomena and were able to transfer the knowledge to new contexts (Pallant & Tinker, 2004). In a retention study, volunteers participated in an interview and responded to a questionnaire about retention of core concepts two to six months after having completed activities in their classrooms. The study focused on how the modeling activities and in particular the visual representations aided student retention of concepts over time. Students were shown a screenshot of the model and prompted to describe the concepts being taught, the model parts, and the relationship to the instructional content. All students could identify the key concept, 86% could describe what all the different parts of the model represented, and 57%

could elaborate on the concept and the importance of the interactions with the model. This study clearly indicates that students retain vivid memories of their interactions with the models and of the concepts. Although these studies are not directly related to nanoscience education, the results may still shed light on how nanoscale simulations can enhance learning.

In an independent study, Moher and collaborators used *MW* to engage students to design self-assembling nanostructures (Shipley & Moher, 2008). Their results showed that students demonstrated the ability to design nanoscale models using the tool and highlighted the value of the construct-centered design methodology in learning nano-engineering (The National Center for Learning and Teaching in Nanoscale Science and Engineering, 2008).

We have also collaborated with chemistry professor Dr. Neocles Leontis at Bowling Green State University (a science teacher preparation institution in Ohio) to pilot test the constructionist strategy in his general and physical chemistry course. Professor Leontis challenged his students to answer questions by designing *MW* simulations. These questions would not have been able to be answered before, because high-level analytical skills involving advanced mathematics were needed. But because *MW* is a modeling tool for solving complex problems without using complicated mathematics, he did not have to worry about his students' mathematical backgrounds.

The results of this case study have been phenomenal (Xie & Pallant, 2009). Two classes of students submitted nearly 150 simulations, some of which have such brilliant designs that even we had never thought of before. We found no evidence that students would just duplicate each other's simulations and converge to the same ideas. In a challenge that asked them to come up with models that prove or disprove the Ideal Gas Law, students were capable of creating various simulations that attack the problem from different directions. Other design challenges included inventing nano purification devices and creating insulation blocks and thermal bridges to control heat flow at the nanoscale. It is evident that these design challenges succeeded in engaging students and greatly enhanced their analytical and problem-solving skills. The following is a testimonial from a student who created a simulation of fuel cell:

> "*Molecular Workbench* is something that I think all physics and chemistry classes should have because it gives an alternative way to grasp concepts outside of just lecturing in the classroom. It allowed me to explore in a way unimaginable before when I built a fuel cell simulation step by step myself. In essence, I could let my curiosity flow by exploring how each editing tool affected my creation. Sometimes I could not figure out how to build it myself, but the program was designed in a way that would not stop my acquisition of learning. Since it is set up like a learning community, I could view someone else's idea of how it should look. In this, I learned in two ways, by attempting my own simulation and by analyzing others." — *Britiany Sheard, student, Bowling Green State University*

This level of success would have been quite improbable using traditional teaching methods in a physical chemistry course.

Future work

Despite of the fact that dynamic modeling can foster student learning, there are some caveats. No matter how good a computer simulation is, it is in general not appropriate to replace real hands-on experiments with it. The recent virtualization trend in science education has prompted the American Chemical Society to issue a statement that suggested that in academic transcripts simulations may not substitute lab work (American Chemical Society, 2008).

While most educators agree that simulations should be helpful, extensive research needs to be carried out to substantiate the effectiveness of learning through simulations, to analyze the interplay between learning in the virtual world and learning in the physical world, and to learn how to take advantage of the strengths of both (Steinberg, 2000). Results from research in this direction will hopefully provide valuable feedback to developers and make technology work even better in the classroom.

Acknowledgment

This work is supported by the National Science Foundation under grant number DUE-0802532.

References

American Chemical Society (2008). *Statement on Computer Simulations in Academic Laboratories*. Washtington, DC.

Amirouche, F. M. L. (2006). *Fundamentals of multibody dynamics: theory and applications*: Springer.

Battelle Memorial Institute, & Foresight Nanotech Institute (2007). Productive Nanosystems: A Technology Roadmap.

Borgman, C. L., Abelson, H., Dirks, L., Johnson, R., Koedinger, K. R., Linn, M. C., et al. (2008). *Fostering Learning in the Networked World: The Cyberlearning Opportunity and Challenge*: National Science Foundation.

Bourg, D. M. (2001). *Physics for Game Developers*: O'Reilly Media.

Cass, M. E., Rezepa, H. S., & Rezepa, D. R. (2005). The use of the free, open-source program Jmol to generate an interactive web site to teach molecular symmetry. *Journal of Chemical Education, 82*(11), 1736-1740.

Catto, E. (2007). Box2D.

Chang, R. P. H. (2006). A call for nanoscience education *Nano Today, 1*(2), 2.

Colella, V. S., Klopfer, E., & Resnick, M. (2001). *Adventures in modeling; Exploring complex, dynamic systems with StarLogo*. New York: Teachers College Press.

Drexler, K. E. (1988). Studying Nanotechnology Foresight Briefing #1. from http://www.foresight.org/Updates/Briefing1.html

Drexler, K. E. (1992). *Nanosystems: Molecular Machinery, Manufacturing, and Computation* (1st ed.): Wiley.

Feurzeig, W., & Roberts, N. (1999). *Modeling and simulation in science and mathematics education*. New York: Springer.

Finkelstein, N. D., Adams, W. K., Keller, C. J., Kohl, P. B., Perkins, K. K., Podolefsky, N. S., et al. (2005). When learning about the real world is better done virtually: A study of substituting computer simulations for laboratory equipment. *Physical Review Special Topics - Physics Education Research, 1*(1), 010103-010110.

Goodsell, D. S. (2004). *Bionanotechnology: lessons from nature*: John Wiley & Sons.

Greenberg, A. (2009). Integrating nanoscience into the classroom: perspectives on nanoscience education projects. *ACS Nano, 3*(4), 762-769.

Israelewitz, B., Gao, M., & Schulten, K. (2001). Steered molecular dynamics and mechanical functions of proteins. *Current Opinion in Structural Biology, 11*, 224-230.

José, T. J., & Williamson, V. M. (2005). Molecular visualization in science education: An evaluation of the NSF-sponsored workshop. *Journal of Chemical Education, 82*(6), 937-942.

Kafai, Y., & Resnick, M. (1996). *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*. New Jersey: Lawrence Erlbaum Associates, Publishers.

Lacoursière, C. (2007). *Ghosts and machines: regularized variational methods for interactive simulations of multibodies with dry frictional contacts.* Umeå University, Umeå.

Leach, A. R. (2001). *Molecular Modeling: Principles and Applications* (2nd ed.): Pearson Education.

Liu, G. R., & Liu, M. B. (2003). *Smoothed Particle Hydrodynamics: A Meshfree Particle Method*. Singapore: World Scientific Publishing Company.

Monroy-Hernández, A., & Resnick, M. (2008). Empowering kids to create and share programmable media. *Interactions, March-April*, 50-51.

National Research Council (1996). *National Science Education Standards*. Washington, DC: National Academy Press.

National Research Council (2006). *A Matter of Size:Triennial Review of the National Nanotechnology Initiative*. Washington D.C.: National Academies Press.

National Science and Technology Council (2007). *The National Nanotechnology Initiative Strategic Plan.* Arlington, VA: National Nanotechnology Coordination Office.

NSF Blue Ribbon Panel on SBES (2006). *Simulation-based engineering science: Revolutionizing engineering science through simulation*. Washington, DC: NSF.

Pallant, A., & Tinker, R. (2004). Reasoning with atomic-scale molecular dynamic models. *Journal of Science Education and Technology, 13*(1), 51-66.

Panoff, R. (2009). Simulations Deepen Scientific Learning. *ASCD Express, 4*(19). Retrieved from http://www.ascd.org/ascd_express/vol4/419_panoff.aspx

Papert, S. (1991). Situating Constructionism. In I. Harel & S. Papert (Eds.), *Constructionism*. Norwood, NJ: Ablex Publishing Corporation.

Rappaport, D. C. (1997). *The Art of Molecular Dynamics Simulation*. Cambridge, UK: Cambridge University Press.

Rieth, M., & Schommers, W. (2006). *Handbook of Theoretical and Computational Nanotechnology* (1st ed.): American Scientific Publishers.

Shipley, E., & Moher, T. (2008). *Instructional Framing for Nanoscale Self-Assembly Design in Middle School: A Pilot Study*. Paper presented at the Annual Meeting of the American Educational Research Association, March 24-28, 2008.

Stam, J. (2003). *Real-Time Fluid Dynamics for Games*. Paper presented at the Proceedings of the Game Developer Conference.

Steinberg, R. N. (2000). Computers in Teaching Science: To Simulate or Not To Simulate? *American Journal of Physics, 68*(7), S37-S41.

Sweeney, A. E., & Seal, S. (Eds.). (2008). *Nanoscale Science and Engineering Education*: American Scientific Publishers.

The National Center for Learning and Teaching in Nanoscale Science and Engineering (2008). *Using Construct-Centered Design to Align Curriculum, Instruction, and Assessment Development in Emerging Science: http://hice.org/presentations/documents/NCLT-ICLS-03mar.pdf*.

Tinker, R., & Xie, Q. (2008). Applying Computational Science to Education: The Molecular Workbench Paradigm, Computing in Science and Engineering. *Computing in Science and Engineering, 10*(5), 24-27.

Watanabe, N., & Tsukada, M. (2000). Fast and stable method for simulating quantum electron dynamics. *Physical Review E, 62*(2), 2914–2923.

Wieman, C. E., Adams, W. K., & Perkins, K. K. (2008). PhET: Simulations That Enhance Learning. *Science, 322*, 682-683.

Wilensky, U., & Reisman, K. (2006). Thinking like a wolf, a sheep or a firefly: Learning biology through constructing and testing computational theories – An embodied modeling approach. *Cognition & Instruction, 24*(2), 171-209.

Wilson, E. K. (2003). Computational nanotechnology: modeling and theory are becoming vital to designing and improving nanodevices. *Chemical & Engineering News, 81*(17), 27-29.

Wong-Ekkabut, J., Baoukina, S., Triampo, W., Tang, I.-M., Tieleman, D. P., & Monticelli, L. (2008). Computer simulation study of fullerene translocation through lipid membranes. *Nature Nanotechnology, 3*, 363-368.

Xie, C., & Pallant, A. (2009). Constructive Chemistry: A Case Study of Gas Laws. *@Concord, 13*(2), 12-13.

Xie, Q., & Tinker, R. (2006). Molecular dynamics simulations of chemical reactions for use in education. *Journal of Chemical Education, 83*(1), 77-83.

Zollman, D., Rebello, N. S., & Hogg, K. (2002). Quantum Mechanics for Everyone: Hands-On Activities Integrated with Technology. *American Journal of Physics, 70*(3), 252-259.