An Open Source Virtual Lab for School Physics Education

by

Jeetinder Singh, Harini Sampath, Jayanthi Sivaswamy

in

The National Conference on Open Source Software, C-Dac, Nai Mumbai
(NCOSS-2009)

Report No: IIIT/TR/2009/239

Centre for IT in Education
International Institute of Information Technology
Hyderabad - 500 032, INDIA
May 2009
Abstract—There is a need to create quality open source educational resources to address the needs of Indian schools. In this paper, we describe a virtual physics lab which we built based on open source components. This is basically an authoring tool that lets users create their own experiments. We first discuss the need for virtual labs. We then present the architecture of our tool and the technical challenges we faced in creating it. We also present the feedback we received from school students on this tool.

I. INTRODUCTION

Education is the fundamental mechanism to achieve social equality. Providing quality education empowers people, creates opportunities for them and aids them in making informed decisions. Our success in the knowledge economy would critically depend on providing quality higher education for all. This includes providing proper infrastructure and high quality content for the teachers and students. But lack of proper infrastructure, access, quality has been a major concern area for many years and large segments of Indian population do not have access to higher education due to financial constraints [1], [2]. It is, therefore, essential to ensure that people have widespread access to high quality educational resources.

One of the most effective ways of achieving this would be to develop and disseminate quality Open Source Educational Content or Open Source tools to create educational content [3]. This would facilitate easy and widespread access to high quality educational resources. Some of the available educational resources (not all of them are open source) are

- PhET Interactive Simulations [4], [5] is an ongoing effort to provide an extensive suite of simulations to improve the way that physics, chemistry, biology, earth science and math are taught and learned. The simulations are interactive tools that enable students to make connections between real-life phenomena and the underlying science which explains such phenomena.
- Phun is an educational project intended to bring in interactive 2D physics to build virtual worlds [6]. You can use several kinds of objects to build your experiments: fixed planes, circles, squares, irregular shapes, hinges, planes, chains, springs and water can be added as well. To each object, you assign a color, density, friction and bounciness. Also one can create objects of different shapes, connect them with chain or springs, fixate them and can watch them interacting, according to the laws of physics.
- In MyPhysicsLab [7] users can try out different physical experiments online. These 21 experiments include springs, pendulums, rigid body collisions, molecules and even a sumo wrestling game for two players. For each experiment, a number of parameters can be set such as elasticity, gravity, damping, thrust, mass and stiffness.
- Open Dynamics Engine (ODE) is an open source library for simulating rigid body dynamics [8]. ODE is used in many computer games, 3D authoring tools and simulation tools. ODE is used for simulating the dynamic interactions between bodies in space. It supports several geometries, advanced joint types and integrated collision detection with friction.
- SOFA is an Open Source framework primarily targeted at real-time simulation, with an emphasis on medical simulation, but can also be used as an efficient prototyping tool or as physics engine [9]. It allows simulating the dynamics of interacting objects and modifying most parameters of the simulation by editing a description file.
- Worldwide Telescope (WWT) is a free Microsoft application [10] that lets you travel through outer space using virtual telescope. You can pan in and zoom in close to moons, planets, solar systems and galaxies.
- GeoGebra [11] is a free and multi-platform dynamic mathematics software for schools that deal with variables for numbers, vectors and points, finds derivatives and integrals of functions and offers commands like Root or Extremum. You can do constructions with points, vectors, segments, lines, conic sections as well as functions and change them dynamically afterwards.
- The Chemistry Collective [12] is a digital library of activities for introductory college and high school chemistry. It provides a flexible simulation environment so that instructors and students may use it to create scenario-based learning activities, design and carry out their own experiments, interactive tutorials that support student problem solving.

Within higher education students need to understand how data can be gathered, analyzed, interpreted and how to evaluate the findings. Moreover, built into such exercises is the oppor-
tunity for students to learn by constructing their own ideas and knowledge. There are many theories on the best way to learn and teach. One among them, the experiential learning theory says that knowledge is created through experience [13]. This means apart from the traditional class room teaching students should have an opportunity to experiment with the concepts and be creative.

Virtual Environments (VE) offer strong benefits in simulating science experiments that are impractical, expensive, impossible, or too dangerous to run. These technologies aim at developing interfaces that are closer to natural human behaviors in an attempt to create Virtual Reality (VR) by enabling underlying laws governing the objects to produce a higher level of interaction and visualization tools. These environments offer the learner a means to explore, experience and express themselves [14]. This gives students the opportunity to explore physical situations and interact with it. The virtual manipulations of the virtual objects help the students to understand the process of performing physical acts subsequently. Not only does the simulation based active learning contribute to conceptual change [15] but also provides tools for scientific inquiry [16]. In this way the simulation experience may benefit students by enabling them to develop their own understanding of abstract concepts [17] and problem solving experiences [18]. Such open-ended experience based approach to learning development is underpinned by constructivist theory of learning [19]. There is growing interest in multimedia supported, highly interactive, collaborative computer simulations tools because of their potential to supplement constructivist learning [20], [21]. It is [22] argued that, simulations are a very effective learning activity that can provide the environment, and within it the concrete experiences necessary, for the development of insight about abstract concepts.

But the success of computer simulations and its use in science education depends on how they are incorporated into curriculum and how teachers use it [23]. Also before implementing such tools, we need to evaluate (a) how effective this type of tool will be for learning the given concept, (b) can the tool be supported with the infrastructure the end user has. Past research shows that the most appropriate use of virtual simulations lab seems to be to use them as a supplementary tool for classroom instruction and laboratory [24].

In this paper, we describe the details of an open source virtual physics lab that we constructed based on the Qt [25], Open Dynamics Engine (ODE) [8] and OpenGL [26].

II. THE NEED FOR VIRTUAL LABS

As we mentioned earlier real experience aids in better understanding. There are many ways by which a student could attain this experience - through real physical activities or through computer generated interactions. These computer based multimedia environment could be collections of images, animations, simulations, authoring tools, games or virtual reality. These environments offer students a means to explore, experience and express themselves.

In a virtual environment, the students can posit hypotheses about a science concept, conduct as many experiments as they want, observe and record data and draw conclusions. This might not always be possible in real laboratory because of many reasons. The physical phenomenon could itself be complicated to observe or the experiment setup could be time consuming. For example, let us say the student wants to know how the initial velocity of launch of a projectile affects it range. Now controlling the initial velocity in a real lab is very difficult. However in a virtual environment, it would be just the click of a button. This ease of setting up an experiment can let them conduct many experiments and view the results in many perspectives [27].

There is no doubt that experiments with real materials would provide students with a more realistic and arguably more exciting appreciation of the physics. The disadvantages of using real materials are often managerial and cost-related. The materials themselves may be expensive to buy or collect, the laboratories have to be maintained and teaching staff must be provided. In addition, experiments dangerous or too expensive to conduct in an actual laboratory can be simulated with the use of virtual lab.

An appropriate way to use virtual labs in science education is to use them as a supplementary material. The study [24] of using computer simulation in a first-year general chemistry course shows that the combination of simulations and laboratory offers advantages in time spent so that the in-lab portion can be reduced in length and students using the simulations have a slightly better knowledge of the practical aspects directly related to laboratory work.

III. AUTHORING TOOLS - A CONSTRUCTIVE APPROACH TO VIRTUAL LABS

A virtual lab could be a set of simulations put together (examples are applets, flash-based demos, animations). The student could manipulate various parameters of the simulation and observe the results. In this approach there are certain advantages - it is very easy to learn how to use them, the learning objective is more clearly defined. Such tools are called instructive as they try to instruct the learners. However, the students play a very passive role in this process. There is no scope for them to create anything of their own.

Another approach to a virtual lab could be to provide a virtual work place that obeys the laws of physics. Users can arrange objects in this virtual space, connect them together, modify their properties and observe the behavior. In this case we are basically providing them an authoring tool where they can create their own experiments. Such an approach where the students create their own learning experiences are called constructive. But some effort might be needed to learn the tool. Further the students might treat it like a game and the learning objective might be lost.

Choosing between these approaches give us a few design challenges:

- Teachers and students do not have the same level of comfort with computers.
Balancing teaching/learning and playing - This is determined by the instructive or constructive nature of the design. We need a correct balance between a system that clearly sets a learning objective and the one that allows the user to be creative.

Range of concepts that need to be covered are wide involving different degree of idealizations of physics laws.

Next we propose an approach for the development of a 3D virtual physics laboratory (3D-ViPLab) using open source components. A virtual lab interface brings together a 3D model of real object and a virtual visualization of physical situations in an interactive manner. In designing the 3D-ViPLab, we have consciously avoided using resource-intensive graphics and visual effects to keep the system requirements as low as possible to accommodate the available computing infrastructure in most Indian schools. The students first need to learn the required factual information and principles and then use the virtual lab workspace to relate and apply that knowledge.

The virtual laboratory consists of three modules:

- Simulation Modelling: This is responsible for the behavior of the virtual objects and the simulation of the experiments.
- Visual Representation: Displays all the 2D/3D data/object that is required for the users view.
- User Interface: Responsible for the human-computer interaction.

IV. 3D VIRTUAL PHYSICS LAB AUTHORING TOOL

The 3D virtual physics lab (3D-ViPLab) we describe next is designed to support primary to high school physics education. It provides a virtual workplace to create new experiment. A set of experimental (domain) objects are made available to the user to construct experiments within the virtual environment space. The Physics engine controls the behaviour of the domain objects by implementing basic physics laws. The physics simulation engine helps in simulating the physical environment once a required structure has been constructed. Further control is possible by users by way of changing object attributes or environment properties. Entire experiments can be enhanced with related multimedia content and saved to serve as complete instructional material.

The system as shown in Fig 1 consists of four different modules:

- GUI: All interactions with the 3D-ViPLab are made to be mouse-based rather than keyboard based to make the ViPLab more user-friendly and accessible to users in a wide age group. The components of our GUI are described next.

- Menu: On the top of main screen we have a menu bar as shown in Fig 3 having various options for all the possible features of the virtual space and parameters of the object created. ViPLab different mode defines the possible state of the environment at any give instant of time viz. pause/play experiment, change object parameters, show/hide graph and recording measurements. This menu provides basic functionality to the user. User can create/open one experiment at a time, (Fig 2). To simulate virtual environment an input using mouse click on menu text control 'play/pause' is required. Measurements of various objects parameters during the course of an experiment are possible by selecting objects. It allows the user to create report for individual object instead of all objects. The real time graph can be plotted and viewed by selecting graph variables for the selected object.

- Tool Box: It provides primitive geometric shapes/objects which can be used to create the experiment. These objects
can also be used to create complex shapes. Toolbox provides basic and reusable objects like sphere, cube etc which can be arranged in the environment. Also a set of connectors like rope defining relationships between objects and sequencing their interactions in an appropriate order are available. This way, user can easily create the desired environment using objects and connectors (Fig 3).

- Virtual Workspace: It provides virtual space to build the experiment, play/pause the experiment (Fig 3). This is a 3-D space that simulates a real physical space where basic laws of physics hold. Properties are associated with this space and each virtual object as shown in Fig 4 to replicate a real physical space. Examples of such properties are gravity, friction, coefficient of restitution etc. Most of these properties can be varied (within bounds) by the user. These properties are a set of assumptions under which experiments are performed. To simulate the experiment, user is allowed to design the experiment. Once the experiment is set, experiments will simulate the virtual physics environment. It will provide a user friendly environment where one will be able to see the simulated (near to real experiment) behavior. The magnitude of the total force acting on an object is calculated at every point at all times.

**Application Controller:** It is the event handler of the tool which controls the users navigation and interprets his/her actions. It identifies events and associates responses to them. Events are inputs to the system. Responses can be visual/non-visual feedback to specific events.

**Rendering Engine:** The visual representation is created by the rendering engine. It displays all the three dimensional object and data that is required for the users view, according to his/her position in the virtual space, and has also the ability to create or destroy 3D models dynamically and transform them in real time. While an experiment is running, it receives values directly from the 3D Physcis Engine part, and updates the visual attributes (position, orientation, scale, material) of the objects that interact.

**3D Physics Engine:** The 3D-ViPLab uses Open Dynamics Engine (ODE) for simulating rigid bodies and its collision detection mechanism to compute the physical interaction with objects. ODE is a widely used physics engine in the open source community, for research, authoring tools, gaming etc. It consists of a high performance library for simulating rigid body dynamics using a simple C/C++ API. ODE allows simulation of the experiments and dynamic visualization of the virtual objects. It controls the virtual object's attributes (e.g. position, velocity), checks for interaction between them and applies the laws of physics.

The ODE physics engine module helps in simulating the physical environment as shown in Fig 5 once the required structure has been constructed in the virtual environment. It applies the forces, that have been activated, and physical laws on the objects in the virtual environment. Every object introduced in this space is subject to a downward gravitational force. The engine tracks the objects present in the virtual space and computes the net force acting on every object at every instant of time. Collision between objects and between an object and the virtual space boundary are also handled by this engine. ODE calculates virtual space parameters which
are either input from the users, from dynamic setup of virtual bodies in an experiment, information from shape and structure of bodies, bridging the limitation between the device and calculation artifacts. The ODE engine has been evaluated for calculating required information in a progressive manner.

Facilities to record data and plot graphs have been added to enable students conduct a full fledged experiment.

- **Data Recording:** The goal of any experiment is to gather and analyze data. So it is imperative that we provide a mechanism to gather the data from the experiment. The Data Recording module handles this. Before beginning the experiment, users can specify the parameters they would like to be gathered. These parameters are written to a file as the experiment runs. This is available to the user after the experiment completes.
- **Graph Plotter:** Visualisation of data aids in better understanding of the underlying phenomenon generating the data. The Graph plotter module plots a real time graph of a parameter as the experiment runs. This graph is presented in the same screen as the experiment. This assists in better correlation of experiment and the data it generates.

A. **Physics Engine Requirements and Integrations Issues**

The physics simulation engine needed for the virtual physics lab gives system the ability to reach unprecedented levels of realism in modeling physical laws of the real world (or arbitrary imaginary worlds). There are many open source and commercial 3D physics engines available in comparison to the 2D ones. The available 3D open source physics engines are primarily aimed at the gaming industry, [8] [28] [29].

The following functional requirements were considered during the selection of physics engine.

- **Virtual space/World Parameters:** Every object introduced in this simulation space is subject to a certain constraint like gravity, damping, deactivate/active objects, stepsize.
- **Object Physics Parameters:** Physical attributes which are allowed to set and get from a virtual object in the scene.
- **Joints/Connectors Allowed:** Support for connectors are needed to attach bodies to simulate the experiments having pendulum, spring in the scene.
- **Support for Complex Object:** Should allow object which can be created from more than one geometry (primitives/non-primitives).
- **Collision Handling:** The physics engine should be able to detect the collision between two bodies in the virtual space and handle this according to laws of physics. This should apply for both objects of simple and complex geometries. By complex, we refer to objects that could be convex, concave or any composite of other simple objects.
- **Precision:** This decides the trade-off between the accuracy and performance of the system. If the precision is too low then due to round off errors the object tends to deviate from its correct position. If it is high, it can bring down the speed of the simulation.

Available physics engines are more targeted at games where speed of simulation is more important than accuracy of simulation and little focus has been given to the educational use of such engine. ODE is an open source, high performance library for simulating rigid body dynamics. It has two main components 1) rigid body dynamics simulation engine and 2) a collision detection engine. Currently, ODE supports bounding box, sphere, capsule and capped cylinder, trimesh with various collision primitives but less-sophisticated support for arbitrary trimesh collisions. It also implements material properties (e.g. friction and elasticity) as well as a basic set of rigid joints such as hinges, ball and socket joints, and sliders. It is independent of any particular graphics package. ODE supports most of the functional requirements and is relatively simple to use. But based on our experience of the initial use of ODE we have observed several constraints in simulation of a real world with available resources. A few are listed next.

- **Joints/Connectors:** Joints are needed for many physics experiment like pendulum, spring, pulley etc. But ODE doesn’t provide direct support for joints like spring, rope etc., which are commonly used in experiments. It supports hinges, ball and socket joints etc., So one needs to write a wrapper over the existing joints to realise the joints that would be used in the experiment.
- **Constraints:** In real world all constraints apply in parallel to a body. However ODE or any physics engine for that matter can not handle all the real world constraints and in most times the constraints are satisfied sequentially rather than parallel. This gives rise to situations where the simulations deviate from the real world behavior.
  - **Case1:** In the pendulum setup like in Fig 4(a) box is a fixed point, spherical body is attached to the body through a connector. The length of this connector is fixed and is a constraint which the physics engine has to handle. During the simulation, suppose the mass of the sphere is increased to a very high value. In real world, beyond a particular mass the connector will expand and finally break. But in the simulation, the length constraint holds even with very heavy objects. But students intuitively expect that there must be a maximum weight when the constraint will fail and
the spherical object will detach from the box. Such scenarios need to be handled separately.

- Case2: In the similar setup, suppose the user sets a high initial velocity for spherical object. When the sphere is released one could observe that the connector stretches initially and then the length constraint is restored. This is because constraints are applied sequentially rather than in parallel.

- **Collision Handling:** Detection of collision and handling it for different structures (convex, non-convex, trimesh and primitives) is more difficult and common problem for most of the physics engines. The Fig 6 shows that collision between primitive objects is not satisfactory. The sphere penetrates into the ground which is modeled as a cube.

- **Precision Handling:** The ODE allows either single or double precision floating point numbers. Single precision is faster and uses less memory, but the simulation will have more numerical errors that can result in visible problems.

  - Case1: In the scenario where objects collide and bounce, we tried to find if objects gain, lose or preserve the right amount of energy on collisions. A sphere will be placed above a plane in a world with no gravity. To compare, the kinetic energy is calculated before and after the collision. ODE gives the objects a velocity in the opposite direction of the supposed collision. The energy and the resulting velocity is the same for all three tested restitution coefficients, 0.0, 0.5 and 1.0.

  - Case2: To test if ODE preserves angular momentum and energy, a cuboid is placed in a world with no gravity and a starting angular velocity. The simulation will explode after some time. The documentation for ODE (section 12.12) explains this problem and gives some suggestions on how to make a simulation more stable.

Other non functional requirements which were considered during the design and implementation are:

- **Virtual 3D environment implementations are expected to provide useful rendering rates on most modern PCs, especially those with 3D graphics accelerator cards.** The support for complex 3D object in virtual worlds needs high-priced workstations to meet real time performance expectation. Also physics precision involves the framerate, or the number of moments per second when physics is calculated, the step-size. When the number of objects in the environment is increased, the performance of the lab degrades depending on the PC configuration.

- **Since ODE uses a simple rendering engine based on OpenGL, it has limitations for the rendering of complex environments comprising many objects and bodies.** This can significantly affect the simulation speed of complex simulation experiments.

- **To simulate a wide range of concepts related to physics experiment in secondary and higher secondary classes, different sub-physics engines are needed which work in synchronisation with each other and with a common object data structure.** The available joints are not flexible enough to provide commonly needed connectors, for example, slider joint can not be used as spring since it restricts the connected objects movement to be unidirectional.

- **Finally, 3D geometric models are used to construct virtual object and rendered in virtual environment.** ODE supports various collision primitives (bounding box, sphere, capsule and capped cylinder) as well as for arbitrary trimesh collisions. The trimesh collision functions are slow, inaccurate and do not perform well if step-size is not chosen properly to synced with frame-rate. For large systems selection of step-size is important as this will use a lot of memory and can make system very slow.

Also the authoring tool needs to provide complete set of primitive objects which can be used in various laboratory experiment. Of course, these primitives can be used to create any complex structure depending on how they are grouped together. But manipulation of 3D objects with 2D user interfaces is very difficult for non experienced users. The GUI needs a simple and flexible mechanism for manipulating and controlling the object in 3D environment, making the interaction much simpler and more intuitive. Also the 3D objects are generally detailed geometric models, and many techniques exist for creating them. Many applications are readily available in the areas of 3D model construction, but in general, these products are difficult to use and are not available freely. We chose to compose virtual environment consisting of objects from the Princeton Shape Benchmark model database [30]. The system contains a database of polygon mesh models representing various types of objects. This database contains models collected from the World Wide Web have wide range of arbitrary models that are freely available.

**B. System Requirement**

3D-ViPLab is aimed to be platform independent and targeted at low-cost midrange workstations , table I. The rendering engine supports lower-level graphics rendering APIs which can be tuned without sacrificing performance. Use of Qt which is cross-platform application framework allows us to deploy authoring tool across multiple platforms without rewriting the source code.

<table>
<thead>
<tr>
<th>Group</th>
<th>Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>Languages and libraries</td>
<td>Qt 4.3.0, C++, ODE, OpenGL</td>
</tr>
<tr>
<td>Processor</td>
<td>Celeron, P III, PIV, AMD, 1GHz</td>
</tr>
<tr>
<td>HDD Space required</td>
<td>Min. 50 MB</td>
</tr>
<tr>
<td>RAM</td>
<td>128 MB</td>
</tr>
</tbody>
</table>

**TABLE I**

**SYSTEM REQUIREMENT.**
V. USER INTERACTION WITH 3D VIPLAB

The current 3D-ViPLab prototype has been used by 36 high school students in the presence of physics teacher. Our initial study was aimed at understanding the usability aspect of 3D-ViPLab and how users interact with it.

A. Methodology

All the student interacted with physics experiment basically consisted of a cube, sphere and a pendulum (cube connected with ball, with the cube being fixed or made immovable). A list of tasks were given to the user: 1) Set the parameter for an object, 2) Plot the position graph for an object 3) Record the velocity and position values of an object in experiment report. Sufficient time was given to complete the task. No initial instructions were given on how to use the tool. We then received feedback from them on the task they are able to complete, amount of time they took to complete all the task and their comfort level with the tool.

B. Observations

In spite of no initial instructions being given, most students were able to work out how to use the tool. Once they had a hang on the tool, we found that they started experimenting. They varied the initial parameter values, position of the objects and observed the behavior. Such experimenting would have taken a lot of time in a physical lab. But they were able to do this in less than 15 minutes with the virtual lab. They also tried to interpret the graph drawn by the tool. The immediate cause and effect relation they were able to see encouraged them. We got a very positive feedback on their experience with the tool.

In future, we are planning to conduct a controlled experiment to quantitatively evaluate the improvement in student’s learning with this tool.

VI. DISCUSSION AND CONCLUSION

Authoring tools especially those available under open access can speed up the creation, adaptation, and utilization of open educational resources. These constructive laboratory environments can improve the conceptual understanding of students and change their attitude towards the science education. There is need to study the effectiveness of the use of virtual labs and tools and consider them as an integral part of the science curriculum.

The gap between the requirements of the Indian higher education and the available infrastructure and content is huge. Large sections do not have access to high quality education. Developing open source tools can aid to bridge this gap. They can help in educating much larger numbers without diluting academic standards. Since they are virtual and open source, the amount of financial resources that go into them are absolutely minimal. A single computer can act as a physics, chemistry and biology lab. Apart from being cost effective alone, they can save a lot of time for both teachers and students. Finally, they can alter the student’s in the learning process from a passive viewer to active creator.