An Easy to Use Tool for Augmenting Multi-media Lectures with Accessible Self-assessment Exercises

Michael Litzkow and Gregory Moses
Department of Engineering Physics, University of Wisconsin-Madison, Madison, WI 53706
mlitzkow@facstaff.wisc.edu moses@engr.wisc.edu

Abstract - eTEACH is an authoring tool for creating on-line streaming-video multi-media lectures viewed in a web-browser. Using eTEACH to provide out-of-class lectures allows faculty to engage students in active learning and problem-solving activities in class; with measured benefits. We report on two significant enhancements to eTEACH that further define it as a learning tool. First, we have added a self-assessment quiz feature that focuses the student on the content and indicates the level of competence with the material that he/she is expected to attain after having viewed the presentation. By asking questions related to the presentation and providing answers to those questions as part of the presentation, the student is more actively engaged. Quizzes are constructed using a Microsoft Word template and MathType plug-in. Second, we have a strong commitment to supporting students with disabilities, including blind students. The MathType plug-in produces mathematical formulas in MathML which the authoring tool uses to produce content that is easily navigated by screen-reading software.

Index Terms – Accessibility, assessment, multi-media, on-line lecture.

Motivation

In recent years the use of traditional classroom lectures as the primary means of educating our students has been widely criticized. Detractors claim that lectures are of limited value because they are largely one-way information transfer experiences, and fail to engage students with the concepts and materials being taught. Time spent actively engaging students in problem-solving and discussion is thought to be much more valuable. Unfortunately, we find that students cannot effectively engage in these activities until they have been taught something of the basic concepts involved. We felt that providing the traditional lectures in an on-line format, which students could watch on their own, would free up classroom time for more interactive activities.

Starting in 1999 we attempted to identify the elements of a classroom lecture and to find appropriate mechanisms for transferring those elements to a web-based format. The main elements of our approach are streaming video and audio of the lecturer and a time-coordinated slide show illustrating points as the lecturer discusses them. This format mimics many modern day lecture halls, but has its roots in the time-honored tradition of lecturing while illustrating points on a chalk board. To this we added such elements as an interactive table of contents, web links, and closed-captioning for students with hearing disabilities. We found that creating such materials required a lot of technical knowledge, and was beyond the expertise and time constraints of many faculty members. We addressed this problem by creating an authoring tool called eTEACH. The use of eTEACH greatly simplifies the process, and allows for extensive use of mathematics in both questions and answers and significant support for use by students with disabilities, including blind students. We report on our experiences using this self-assessment mechanism in teaching a class in nuclear reactor theory in a separate paper [3].

Along with colleagues at the University of Wisconsin, College of Engineering we have used eTEACH to create lectures for engineering students in both very large and very small classes. So far, over 2000 students have taken these courses with no live lectures at all. Student reaction has been carefully studied, and is generally positive. We have also made eTEACH freely available to other teachers and researchers. eTEACH has been used successfully to create lectures on a wide variety of subjects ranging from nursing, to business, to food science. Much of this work has been reported elsewhere [1], [2].

As we gained further experience with eTEACH, we began to feel that students might be a little “too relaxed” while viewing the lectures. In truth, it’s easy to sit back and enjoy such a lecture in the way one might view a television news broadcast. That is, without thinking very deeply about the concepts being presented. Sometimes even well-motivated students had difficulty determining at what level they needed to understand the concepts being taught. We hypothesized that some well designed self-assessment questions integrated into the lecture format would help students determine for themselves whether they understood the material at the level intended by their professor. In this paper, we report on the implementation of a self-assessment quiz feature with support for extensive use of mathematics in both questions and answers and significant support for use by students with disabilities, including blind students. We report on our experiences using this self-assessment mechanism in teaching a class in nuclear reactor theory in a separate paper [3].
Preliminary Work

To determine whether such built-in assessment would be helpful to students, we developed a mechanism for embedding simple “quizzes” within our lectures. When students reach a point in the lecture where a quiz should appear, the video pauses, and multiple-choice and true/false questions are presented. The students can answer these questions simply by clicking on their chosen answer. They are immediately told whether their answer is correct – this is designed to encourage learning, it’s pointedly not a test. If students are unsure of the answer (regardless of whether they might have guessed correctly), they can click on a “Review” button which takes them to the part of the lecture where the relevant topic was discussed. Later they can return to the quiz and try again. In its original form, this mechanism required “hand coding” of DHTML pages which implemented the quizzes.

After providing self-assessment exercises for about a year and a half, we felt that this was indeed a useful feature which helped students to learn difficult materials. However several challenges remained. First, implementation of the exercises required hand-coding of dynamic HTML pages. Few professors have hands-on experience with this technology, and in any case they don’t have the time to create and debug materials in this way. Second, the subjects we teach involve a lot of mathematical ideas, which results in the need for complex functions and equations in both the questions and possible answers. We were able to create this math content using MathType with Microsoft Word, but the equations were difficult to transfer to our browser-based quizzes. These were delivered in the form of GIF files, which provide a “picture” of an equation, but don’t carry any semantic information. This format is not accessible to many students with disabilities, e.g. blind students who use screen-readers. Finally, we wanted more flexibility in our mechanism so that we could provide students with “hints” and “feedback” as to why the answer they chose was correct or not. Again, our goal was to provide a learning aid, not a test.

Technical Overview of Our Solution

I. Ease of Use

Our first goal was to create a tool which would be easy for professors to use, and would hopefully build on technology already familiar to them. Considering the need to support complex mathematical constructs, and depending on the background of the particular professor, we felt that some version of LaTeX [4] or MathType [5] would most likely be familiar. Unfortunately, experience has shown that professors who are familiar with either of these tools will not usually want to use the other, making this a very difficult decision. Eventually, we decided that Microsoft Word with the MathType add-on would be most familiar to the largest number of professors in many fields.

To make the process as easy as possible for professors, we developed a powerful Word “template” to support the creation process. When a new document is created using this template, a menu and a corresponding shortcut toolbar are added to Word which provide an easy mechanism for creating and editing the quiz. Figure 1 shows a self-assessment exercise under construction using the Word template. Notice the rightmost toolbar in the third toolbar row. This “self-assessment” toolbar provides quick access to common functions such as adding a multiple-choice or true-false question. For example, clicking on the “M/C+” toolbar button, adds spaces for typing in a question, and four choices. Clicking on the “Eqn+” toolbar button opens the MathML application so that a mathematical expression can be entered. In the example shown, one complete multiple-choice question has been entered, and a second question is in progress. The template provides spaces for the questions and corresponding answer choices as well as buttons which allow the professor to specify which choice is the correct one. In the figure, the second option has been designated as the correct answer, and two answer choices remain to be filled in. Mathematical expressions can be placed in both the questions and answers as needed.

When documents created with the special template are saved, they do not just get saved as Word documents. Instead data is extracted from the Word template and stored in a well-documented XML format. Next, the XML data is transformed into appropriate web-based forms for both sighted and blind users. Since the intermediate format is well-documented XML, the data could be created with tools other than Microsoft Word, if desired. For example, we could add a LaTeX front-end at a future time. This would not require any major re-structuring of our application. Figure 2 shows the same self-assessment exercise as it is presented in a student’s browser, after it has been transformed into DHTML.
Quiz 1

Which is a correct statement of Watt’s law?

- $e = ir$
- Both $e = ir$ and $i = \frac{e}{r}$
- Either $r = \frac{e}{i}$ or $i = \frac{e}{r}$
- $p = i^2r$

How many real roots does the function $f(x) = 9x^2 + 6x + 1$ have?

- 0
- 1
- [Click here and type a possible answer]
- [Click here and type a possible answer]

1. Which is a correct statement of Watt’s law? Review Hint
   - $e = ir$
   - Both $e = ir$ and $i = \frac{e}{r}$
   - Either $r = \frac{e}{i}$ or $i = \frac{e}{r}$
   - $p = i^2r$

2. How many real roots does the function $f(x) = 9x^2 + 6x + 1$ have? Review Hint
   - 0
   - 1
   - 2
   - 3
   - 6

3. Which is a discriminant? Review
   - $b^2 - 4ac$
   - $\sqrt{b^2 - 4ac}$
   - $\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$
   - A person who is discriminated against

4. Which formula gives the hull speed of a displacement vessel? Review
   - $\sqrt{\frac{9.8 \cdot 9}{2}}$
   - $1.44 \cdot \sqrt{\frac{9}{2}}$
   - $\frac{9}{2} \sqrt{\frac{1}{2}}$
   - None of the above

5. What is Ohm’s law? Review
   - [View Instructor’s Answer]

FIGURE 1
SELF-ASSESSMENT EXERCISE UNDER CONSTRUCTION USING SPECIAL WORD TEMPLATE

FIGURE 2
STUDENT’S VIEW OF SELF-ASSESSMENT EXERCISE
II. Accessibility

Our second goal was to make this material accessible to students with disabilities – even blind students. Some understanding of how blind users access the highly visual medium of a computer screen is required to realize why this is a difficult problem, and to understand the techniques we used to solve it. Blind users employ special software programs called “screen readers” which provide an interface to common application software which may not be designed for access by non-visual users. Screen readers are sophisticated software which operate at a level between the operating system and normal applications and can read every word that is printed on the computer’s monitor. In addition to reading all the text, screen readers provide an interface allowing users to control menus, buttons, drop-down lists, fill out forms, start, stop, and switch between applications, and perform all other functions necessary to interact with a modern personal computer. This technology is truly amazing, and the ability of experienced blind users to navigate the complex functions of a modern computer must be seen to be believed. However, screen readers have a crucial limitation, which is that they can only read material that appears on the screen as character data. Icons and pictures that appear on the screen as bitmaps, JPEGs, GIFs, and other “compressed bitmap” formats cannot be interpreted by screen readers, unless a textual description of the image is also provided. Complex mathematical expressions which include superscripts, subscripts, square root signs, summations, integrals, and other symbols comprise a 2-dimensional graphical language which is not directly readable by screen readers. To overcome this problem, the World Wide Web Consortium has developed an XML-based language called MathML [6]. MathML allows for the description of very complex mathematical expressions in a purely textual format. Since MathML is the standard for providing accessible math content on the web, it was clear that we would need to provide our math content in MathML, if we were going to make it accessible to blind students. Fortunately, we were able to utilize off-the-shelf technology for converting MathType into MathML as a part of our solution.

Once we had the MathML in hand, we found a few additional steps were needed to make this content accessible and convenient for both blind and sighted users. While creating a quiz tool that would be accessible to blind students was an important goal, it was not our intention to build a tool for blind users only. Blind students are an important, but nevertheless very small portion of our intended audience. Naturally, sighted students want to see the usual graphical expressions, not the text and angle-brackets which comprise MathML. While some browsers can read MathML and create the implied graphical format, most require a plug-in for this purpose. Using the plug-in, the math content is both visible to sighted users and readable by blind users. However, many sighted users aren’t aware of MathML, and do not have the plug-in installed on their machines. To avoid inconveniencing sighted users, we decided to create two versions of each quiz page – one using MathML, and the other using GIF images of the mathematical expressions.

Since we had chosen to use MathType for the initial implementation of our tool, we obtained a software development kit (SDK) from Design Science (the makers of MathType). The MathType software consists of a native Windows application, and a set of Visual Basic functions that interoperate with Microsoft Word. These functions make it very easy to add equations to Word documents. The SDK exposes the code in the Visual Basic functions, allowing developers to modify and enhance the way Word interacts with the MathType application. Using the SDK, we added functions which allow us to detect all MathML expressions embedded in a Word document, and to obtain both MathML and GIF versions of those expressions.

While making the mathematical expressions accessible to blind users was an important step, we found that some additional steps were required to make the quizzes actually usable by blind users. When we asked actual blind users to try out our assessment exercises, we found was that it’s very easy for screen reader users to get lost in the quiz document and lose track of what question they are on, or which answer choices go with which question. Also, in previous non-accessible versions of our quizzes we told students whether or not their chosen answer was correct by dynamically changing the content of the web page. We would simply add the word “correct” or “incorrect” right next to the answer choice they had clicked. This turned out to be a problem for blind users since screen readers usually begin reading a page again from the top whenever they detect a dynamic change! We then tried putting the “feedback” regarding the student’s answer in a pop-up window. We found that the screen reader would read the pop-up window correctly, but then lose its place in the original document, so that when the student closed the pop-up they could not easily continue with the next question. Eventually, we discovered a mechanism for bringing up a dialog box within the browser which gives the student the required feedback, but doesn’t cause screen readers to lose their place. The help we obtained from actual blind users was essential in ensuring that the layout and formatting of all our forms would be quick and easy for blind students to navigate. This layout is also visually appealing, and in no way detracts from the experience of visual students. A crucial part of our approach to accessibility is that professors don’t need to do any extra work to support it, and don’t even need to be aware that it is an issue. The software automatically produces both MathML and GIF versions. Students who need the MathML version already know that, and will have the required plug-in installed. Other students use the GIF-based versions, and don’t need the plug-in.
III. Advanced Features

Our third goal was to extend the level of feedback offered to our students beyond simply telling them whether their selected answer was correct or not. The template we provide is sophisticated, and comes with its own menu and quick-access toolbar. This supports adding new multiple-choice or true/false questions to the form, as well as additional “feedback” features. By clicking a button the professor can add a “Review” link which takes the student to a relevant part of the same or another eTEACH lecture, or any other web-based material. With another button the professor can add a “hint” to give the student some help in thinking about the questions. Finally, another button allows the professor to add feedback to individual answers that the students may have chosen. This allows the professor to explain why an incorrect answer is not right. Naturally, hints and feedback may need to contain complex mathematical constructs too, and these are handled with the same easy to enter and accessible mechanisms described above. Figure 3 shows a self-assessment exercise with review points, hints, and feedback as it is being edited. This meta-data is displayed using alternate colors and brackets to differentiate it from the questions and answers which will show up directly on the student’s assessment form. This meta-data was entered using buttons provided on the self-assessment toolbar. The creation of hints, review points, and feedback is at the discretion of the professor. They can choose to use these features or not, as they deem appropriate.

We made extensive use of XML technology in every phase of implementing this project. Since the XML formats used are “vendor neutral” and well documented by XML schema, they provide convenient interfaces where new or different technology can be substituted. As mentioned earlier, a LaTeX front-end could be created since the only requirement is to extract the data into an XML file that matches our schema. In a similar way, XSLT transformations are used to create HTML and XHTML files which eventually present the materials to the students in their browser. Non-compatible browsers can be supported by supplying new XSLT stylesheets, without disturbing the basic mechanism.
**RELATED WORK**

The two main goals of our project were to provide an easy to use method for professors to add mathematical content to all parts of their quizzes, and to ensure that those quizzes would be accessible and useable by blind students without requiring additional effort on the part of the professor. Numerous products and tools have been introduced which meet one or the other of those goals, but few have addressed both issues. For example, EzMath by Dave Raggett and Davy Batsalle provides a way to generate the MathML, but requires cutting and pasting that content into HTML documents as part of the process [7]. Respondus is a commercial product designed to create quizzes for various course management systems and can generate MathML as one means of including mathematical content [8]. The actual usability by blind students of such quizzes varies depending on how quizzing is implemented by the particular course management system in use. William Moss evaluates several methods for creating MathML content in quizzes for one course management system (WebCT) [9]. All of the approaches have various advantages and disadvantages. We believe that our combination of ease of use for the professor along with careful attention to usability for screen-reader users (which goes beyond using MathML to describe equations) sets our approach apart.

**CONCLUSIONS**

Our previous experience with self-assessment in a more primitive form supports our belief that this mechanism will be helpful to students. The use of a Word template with MathType for entering equations makes the process of creating these exercises very easy for a large number of professors in technical fields. The fact that professors don’t have to do anything special to support students with disabilities means that such students will find their materials ready to go, when they need them. Our flexible design and extensive use of XML technology means we can easily extend our implementation to cover different requirements both at the creation and delivery ends of the process. Our software is available at no cost for not-for-profit educational users, which will allow other educators to try out self-assessment as an enhancement to their own on-line lectures. Furthermore, we hope that by showing that this kind of mechanism is practical and useful, commercial vendors will create their own implementations so that main-stream users can benefit from easy to create, accessible, self-assessment exercises too.

**ACKNOWLEDGMENT**

This research was supported in part by the National Science Foundation as part of the Education, Outreach and Training Partnership for Advanced Computational Infrastructure (EOT-PACI), the University of Wisconsin-Madison College of Engineering and the UW Graduate School.

**REFERENCES**


