Using Graphical User Interfaces with Try-Again Feedback

Dr. M. Austin Creasy, Purdue Polytechnic Institute, West Lafayette

Assistant Professor Mechanical Engineering Technology Purdue University

Using Graphical User Interfaces with Try-Again Feedback

Abstract

Try-again feedback is a feedback type that provides students with immediate feedback on assignment submissions and allows a student to retry a problem if the submission was incorrect. This feedback type can be easily automated with certain problem types that include: multiple choice problems, matching problems, definition problems, numerical problems, and some equation solutions in engineering. Some engineering solutions require that students create diagrams/graphs that are the solution being submitted. Many learning management systems (LMS) are not able to provide automated try-again feedback on these types of solutions and therefore the effectiveness of try-again feedback is minimized for graphical solutions. Graphical user interfaces (GUIs) are interfaces where a computer allows a student to interact with graphics controlled by the user through underlying code. GUIs can be programmed in many computer languages. This work uses Matlab GUIs to present students with shear and moment diagram problems that are interactive with the graphical solutions. A student is able to draw the shear and moment diagram, using a stylus or mouse, on the computer screen and submit the drawn diagram for immediate grading. The GUI informs the student if the diagram is correct and allows the student to try again after an incorrect solution. GUIs with try-again feedback were used in a Strength of Materials course where shear and moment diagrams are a part of the learning objectives. The development of the GUIs, results from this course, and the effectiveness of using GUIs to provide try-again feedback are presented.

Introduction

Try-again feedback is a feedback method that informs a student during assessment if an answer is correct and allows the student to repeat the assessment exercise if the answer is incorrect.[1] The number of times that the assessment is repeated with try-again feedback, is determined by the instructor providing the assessment and the timing of the feedback can vary between immediate and delayed. Research has shown benefits for both immediate and delayed feedback,[2] but immediate appears to be more beneficial from the author's opinion when implementing try-again feedback. Current learning management systems (LMSs) are excellent at providing immediate feedback for certain problem types because the solutions are specific answers. A LMS can be programmed to allow students to retry assignments when incorrect answers are submitted.[3, 4] Multiple choice, matching, fill-in the blank, numerical, equation derivations, and short answers are a few examples of question types that these LMSs can support. A significant amount of research for online homework tools has been conducted for immediate and try-again feedback on these question types.[5-8] One area where LMSs lack assessment ability is with graphical solutions provided by a student.

Mechanics courses require several types of graphical results in a solution depending on the problem. Statics is the first mechanics course in many programs[9, 10] and has several graphical solutions that can be implemented in the learning topics. Examples of solutions viewed graphically include vector addition,[10] free-body diagrams,[5, 11] and shear and moment diagrams.[9, 12] Incorporating a graphical solution for assessment is problematic with current LMSs tools used by academic programs because they are unable to analyze graphical solutions. Many textbook

publishers are working on including graphical assessments with companion textbook material, but this solution increases the financial cost with the ever increasing textbook expense. Graphical User Interfaces (GUIs) is one method that instructors can use to provide try-again feedback for graphical problems. Depcik and Assanis[13] has stated: "The most important benefit of a GUI is that it can *post-process* the results of the simulation providing the user with *instant feedback*," where the italic emphasis is maintained from the original source. The question for the instructor is how do you incorporate GUIs to provide try-again feedback with submitted graphical solutions?

Numerous educators have used GUIs in varying instructional situations. GUIs can be written in numerous high-level programming languages that will produce a visually pleasing interface for an individual using the GUI.[13] The majority of GUI usage (found for this literature review specifically for Matlab GUIs) involved teaching students to programs GUIs[14, 15] or using GUIs to show graphical outputs as specific model parameters are varied.[9, 10, 12, 16, 17] These examples do not use GUIs for assessment of student performance.

This work develops GUIs as an assessment tool in mechanics courses using try-again feedback. Specifically, shear and moment diagrams are the focus of the GUI assessment tool. Shear and moment diagrams are used to show how the internal shear force and the internal bending moment of a beam vary along the length of a beam. These diagrams provide values that can be used to determine internal stresses of the loaded beam. The GUI is designed to present a student with a loaded beam and empty plots for the student to sketch both the shear and the moment diagrams. The student adjusts the axes of the diagrams as needed and uses a stylus (on touch screen computers) or a mouse to sketch the diagrams. Once a diagram is sketched, the student can check if the solution is correct. A correct solution will provide a credit code for the student to enter in a LMS for assignment credit. An incorrect solution will inform the student that the solution is incorrect and allow the student to repeat the problem. Hence, implementing try-again feedback. This paper will review the development of the GUIs and review the initial use of the GUIs in a sophomore level Strengths of Material class where the students are being exposed for the second time in the curriculum to shear and moment diagrams.

GUI Development

The idea for creating these GUIs for shear and moment diagrams was developed by the author after attending a presentation about Lon-Capa at Michigan State in 2015. Bauer[18] presented a workshop on using HTML5 to provide automated grading of free-body diagrams through submission on the LMS Lon-Capa. The author used this motivation to research different options for implementing try-again feedback with graphical solutions and settled on using Matlab GUIs. Matlab was chosen because students in the academic program where these GUIs are implemented have access to Matlab through lab computers and online using a remote internet site with university login credentials.

The developed GUI has three plot windows. The top window is a figure of the beam with the applied loads presented for analysis. The middle window is an area reserved for the student to draw the shear diagram on provided axes. The bottom window is a figure reserved for the student to draw the moment diagram on provided axes. Figure 1 shows an example of one of the

shear and moment diagram GUIs developed that was provided to the students as a homework assignment. The beam with the geometry and loading is shown at the top with the beam boundary conditions. The axes values on the middle and bottom windows are adjustable by the student. Even after starting the assignment, the student can adjust the minimum and maximum values of an axis to any desired value and the tick marks of the axis will update automatically. The units of the axes cannot be changed in this current edition of the GUI. The buttons on the right of each diagram are used to delete the current sketch and to check the solution of a current sketch.

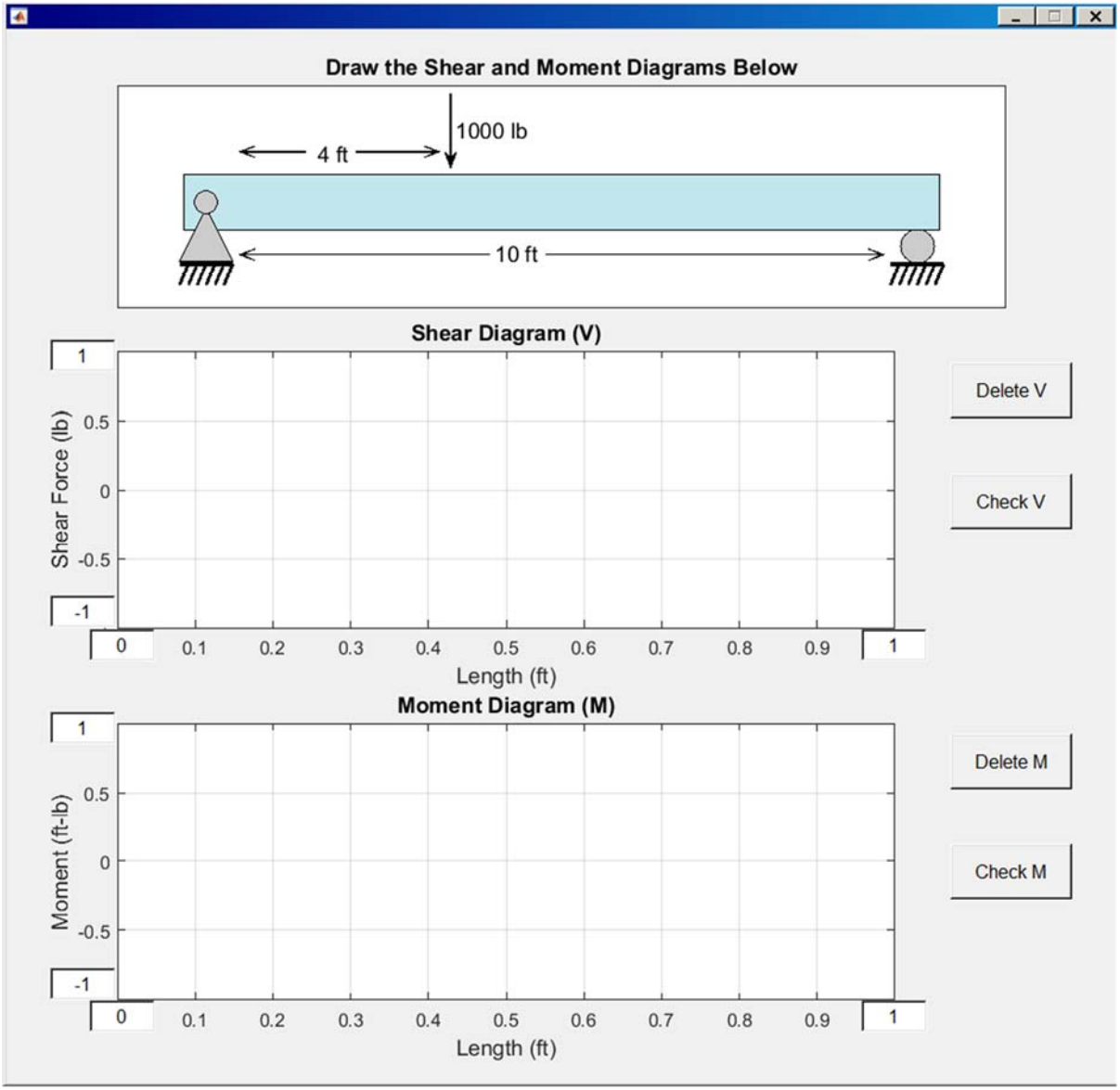

Figure 1. Matlab GUI showing an example of a simply supported beam with all the user options provided for completing a shear and moment diagram of the beam.

The GUI code for the developed shear and moment diagrams has eleven nested functions associated with the different actions of the GUI. The functions include: one figure update function, four separate functions for the push buttons, two plotting functions (one for the shear

diagram and anther for the moment diagram), two stop plotting functions (one for the shear diagram and another for the moment diagram), and two answer checking functions (one for the shear diagram and anther for the moment diagram). The plotting functions allow the student to draw on the figures with a stylus and/or mouse. As the student presses the mouse button or touches the screen in the figure with the stylus, the GUI will start recording and plotting the points on the figure of the student's sketch. The GUI is programmed so the student does not have to draw the entire plot in a single movement without moving the stylus or mouse to perform other actions on the computer. The plots can also be drawn in any order without the need to draw from the left to the right. As the student draws the figure, two Matlab vectors record the plotted data. One vector records the position while the second vector records the load.

The figure update function allows for the student to update the axes values and re-plot any drawn diagrams with an updated axis value typed on any of the axes. This function is also used when one of the "Delete" buttons is pushed. If the student makes a mistake, the "Delete" button can be pushed to implement one of the push button functions that will delete the recorded data points of the sketch and any text below the "Check" button. Once these items are deleted, the push button function will implement the figure update function. When a student has completed drawing the diagram, the student may push the "Check" button to use another push button function to test the drawn solution against the know solution. If the answer is correct, a "credit code" will appear below the button that the student can enter in a LMS for completing the problem. For incorrect solutions, the word incorrect will appear below the button. The moment diagram controls are identical to the shear diagram controls and each diagram can be drawn and checked independently.

For each GUI created, a solution GUI is also created to view the error bounds for the solution. Figure 2 shows the solution GUI of a simply supported beam with a distributed load. The error bounds are problem specific and are coded to allow students to be able to sketch the solution with a mouse when the axes are expanded to include the entire diagram. Therefore, the error bounds are larger in magnitude for the moment diagram than the shear diagram because of the maximum and minimum values of the diagrams. These bounds are used when the "Check" button function is used to compare the provided solution of the student to the bounds shown. The "Check" button function has two scenarios that are checked to ensure that the solution provided is correct. For the first scenario, the function checks to make sure that the student has sketched points in every section of the diagram. If the student only draws a partially correct diagram, the function will consider the diagram incorrect. For the second scenario, the function checks to make sure that the provided sketch from the student's drawings are in the error bounds. If the sketch passes both of these scenarios, the function will consider the diagram correct and provide the credit code.

The Matlab codes for the created GUIs were between 700 and 900 lines long. The credit code was a five-digit number defined in the "Check" push button function. To keep students from searching through the GUI code for the credit code, the GUI code was saved as a p-code to obfuscate the Matlab code. This change increased the complexity in searching through provided GUI code for the credit code without completing the assignment. The p-codes were provided to the students through the LMS and the students downloaded the GUI p-code and ran the GUI with Matlab. Figure 3 shows an example of a screen shot of a student working on the problem shown

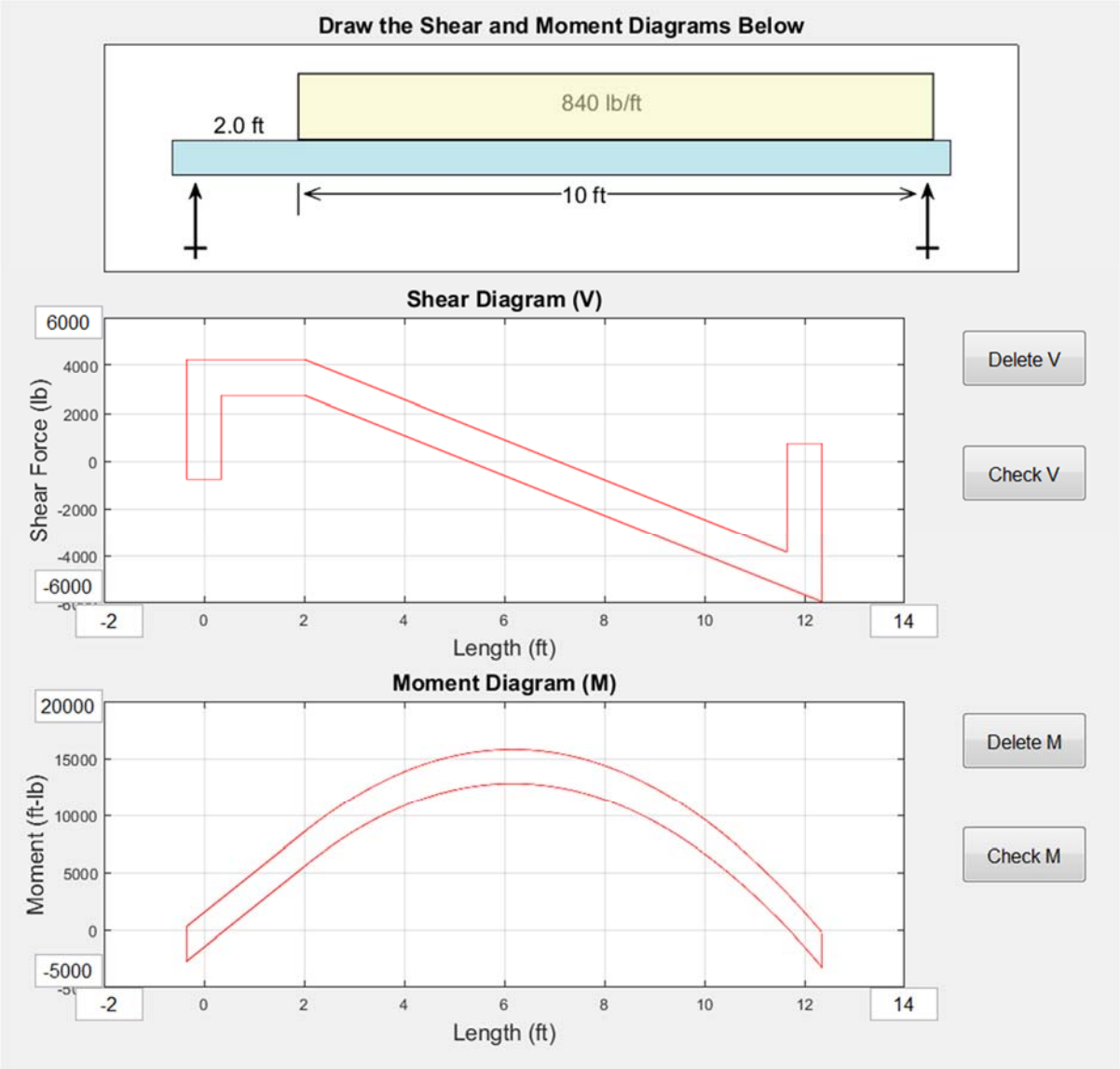

Figure 2. A solution GUI created for the simply supported beam shown. The red lines are the error bounds that show the variation allowed in the student's response.

in Figure 2. Camtasia software was used to record the student's progress with the problem.[19] The student used a stylus to draw the attempt on a Surface Pro 3 running Matlab. The student was able to use a straight edge placed on the screen to assist in sketching the graph. This example shows that the student completed the shear diagram correctly and has started to work on the moment diagram. The two circles between 10 and 12 on the shear diagram were caused by the recording software and are not part of the GUI.

These GUIs were initially used in a Strength of Materials course with 18 total students. Instruction was provided to the class about how to use Matlab and start the GUIs because some of the students had not taken a programming course before this course offering. The GUIs were used on low stakes homework assignments. Assignments for every student were identical, but varying credit codes for each student were used for two problems. For these two problems,

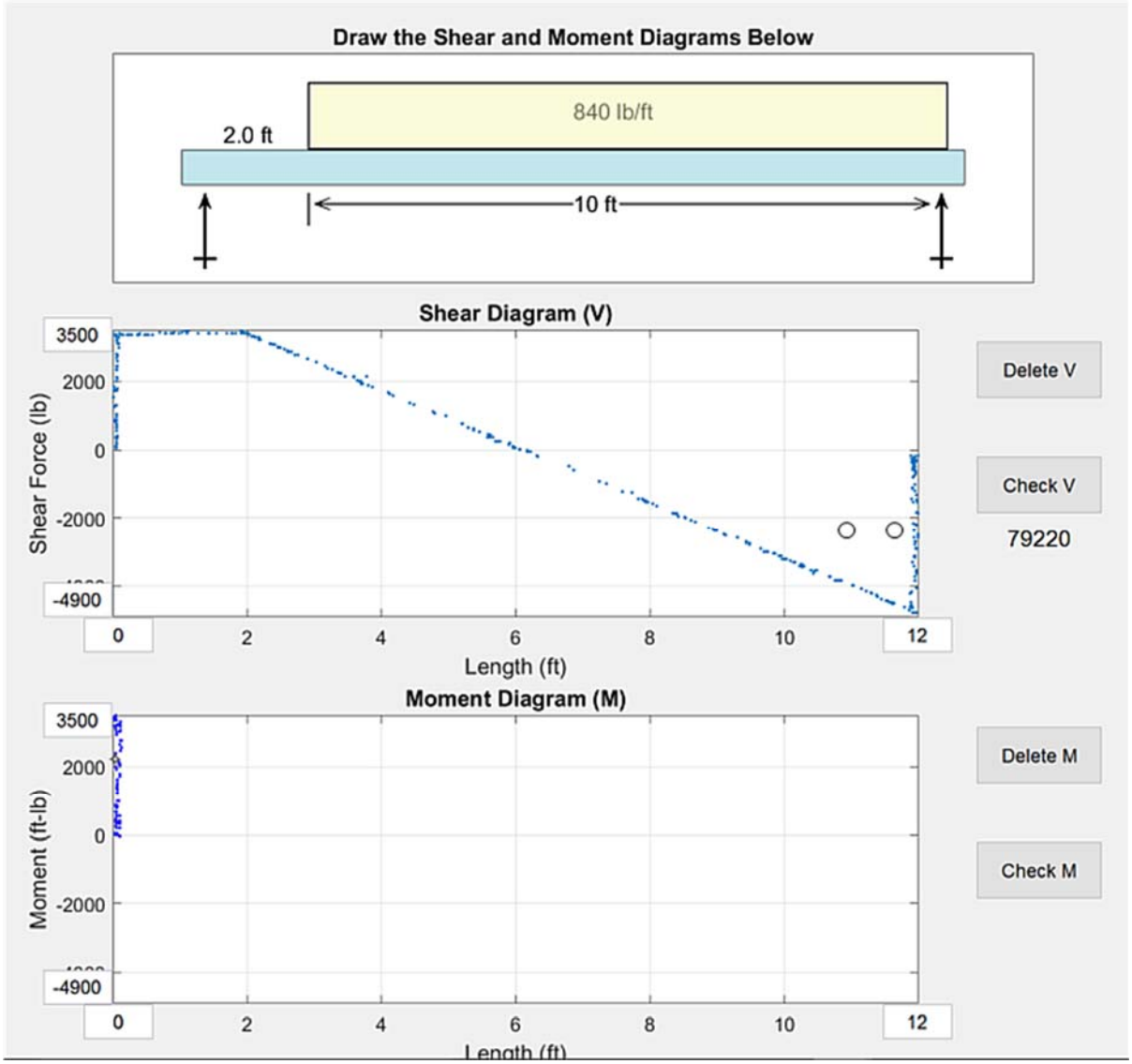

Figure 3. Screen shot of a student with a completed and assessed shear diagram and starting to attempt the moment diagram.

every student was provided a different p-code on the LMS that contained a small variation in the credit code. All of these codes were accepted for credit and the students were not informed that specific problems had varying credit codes. In one of these examples, the code credit ranged from 96481 to 96498 where each numerical value was assigned to a specific student. Figure 4 shows the results of comparing each numerical value to the associated student (each student was assigned a number 1 through 18 for this assessment). If each student used their individually assigned code, the plot would produce a straight line. Deviations from a straight line indicate a student using another student's assigned code. This chart indicates that 7 of the 18 students used an assigned code that was not assigned to them through the LMS and 2 students did not answer this question.

Students were allowed to work together on these low stakes assignments and the author expected that students working in groups would use the same credit code. For example, code 96489 was

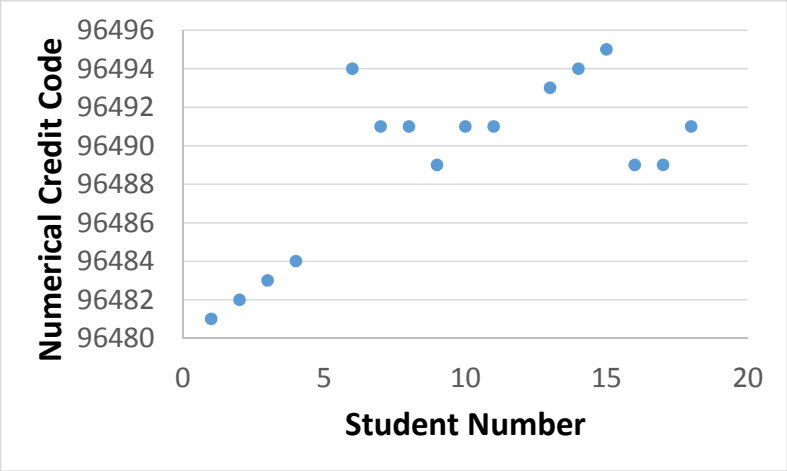

Figure 4. Comparison of a GUI problem where each student was provided an individual credit code. Repeat codes indicate students using codes not assigned to them.

used by three different students for credit, but the author witnessed these students working on homework together on numerous occasions. Five students used code 96491 for credit and the student that was assigned this code typically completed the assignments early and the author did not witness the other students working on the assignment with this student. Assessment of the final exam would indicate that some of these students were copying the credit code without learning the material.

Discussion

The GUIs worked well in providing try-again feedback to students where the GUI was able to assess sketched shear and moment diagrams. The students were able to attempt the problems, receive immediate feedback, and retry the problems if the answers were not correct. Informal interviews were performed with several students about the use of the GUIs in providing assessment of the shear and moment diagrams. The feedback from students was positive except for two issues. One was a programming issue and the other was related to sketching the moment diagram for systems with distributed loads. The only issue that arose with the programming related to older laptops with low screen resolution. The "annotation" code was used to produce some of the arrows and lines shown in the beam diagram with the forces and the geometry. This specific code caused issues with the figure because the entire figure did not fit on the screen of machines with low screen resolution. The arrows were not placed in the correct locations. The GUI code is being updated to replace the "annotation" code with lines on the figures to eliminate this problem. For systems with low screen resolution, part of the figure will still be partially off the screen, but the students can use the mouse to move the figure around the screen in order to view all parts of the interface. The issue with distributed loads was some students had difficulty in drawing the parabolic moment diagram between the error bounds. This issue was fixed by the students calculating specific points on the graph to insure that the curve followed the appropriate direction.

The GUIs were also used as a tool to verify which students were working together or copying answers for the assignments. In a couple of problems, specific credit codes were inserted into the p-code files for each individual student. Several of these individual codes were used by

multiple students that were not assigned to the specific code. In some of these instances, students were working in a group on the assignment and therefore used the same code when the group figured out the solution. These instances were verified by the instructor's observance of the group working together. The instance of copying was observed by the instructor because the student providing the credit code finished the assignment early and the other students finished the assignment much closer to the deadline. The recorded submission records in the LMS confirmed this assessment.

Future Work

Additional GUIs will be written to include more examples of shear and moment diagrams using this method of assessment with try-again feedback. The initial GUI was programmed in 2 weeks with all the required functions. Now a new GUI and solution GUI can be programmed in as little as a few hours, depending on the complexity of the loads and the complexity of the shear and moment diagram. Only certain sections of the code (the graphical representation of the beam and the error bounds) within a couple of functions need to be changed. This significantly reduces the coding time from the initial development. The GUIs could also be programmed to provide additional feedback or partial credit based on the correctness of the provided sketch. This additional feedback will be research and considered for future GUIs.

Additional GUIs will also be developed for other graphical mechanics problems that will provide students try-again feedback. Specific examples include graphical vector summation, free body diagrams, and kinetic diagrams for dynamics courses. The specific form of graphics used and the error bands will need to be defined for each GUI. The GUIs need to also be robust and allow students to solve the problem correctly without becoming frustrated because of noise in their sketching ability.

Conclusion

GUIs have been shown to be a tool that can be used with try-again feedback graphical problems. These GUIs can be programmed to allow students to sketch their graphical solutions and submit them for automatic assessment. The program can assess the student provided sketch and inform the student if the sketch is correct or incorrect. A LMS can record credit provided by the GUI for correct solutions. Instructors can also use these programs to assess if students are performing their own work or using solutions from other students. These GUIs are able to be programmed in Matlab and therefore instructors can eliminate the need to have tools provided by textbook publishers for this type of assessment that increases course costs for students.

References

1. Shute, V. J. (2008) Focus on Formative Feedback, *Review of Educational Research 78*, 153-189.
2. Hattie, J., and Timperley, H. (2007) The Power of Feedback, *Review of Educational Research 77*, 81-112.
3. Kortemeyer, G., Kashy, E., Benenson, W., and Bauer, W. (2008) Experiences using the open-source learning content management and assessment system LON-CAPA in introductory physics courses, *American Journal of Physics 76*, 438-444.
4. Bradford, P., Porciello, M., Balkon, N., and Backus, D. (2007) The Blackboard Learning System: The Be All and End All in Educational Instruction?, *Journal of Educational Technology Systems 35*, 301-314.
5. Fries, R., Cross, B., Rossow, M., and Woehl, D. (2013) Student Perceptions of Online Statics Homework Tools, *Journal of Online Engineering Education 4*.
6. Bonham, S. W., Deardorff, D. L., and Beichner, R. J. (2003) Comparison of student performance using web and paper-based homework in college-level physics, *Journal of Research in Science Teaching 40*, 1050-1071.
7. Cheng, K. K., Thacker, B. A., Cardenas, R. L., and Crouch, C. (2004) Using an online homework system enhances students' learning of physics concepts in an introductory physics course, *American Journal of Physics 72*, 1447-1453.
8. Dillard-Eggers, J., Wooten, T., Childs, B., and Coker, J. (2008) Evidence on the Effectiveness of On-Line Homework, *College Teaching Methods & Styles Journal 4*, 9-16.
9. Al-Masoud, N. (2006) Integrating Matlab Graphical User Interface in Statics Course, In *2006 ASEE Annual Conference and Exposition*, p 1, Chicago, IL.
10. Narayanan, G. (2005) Inclusion of Hands-on Interactive Programs for Teaching Statics, In *2005 ASEE Annual Conference and Exposition*, Portland, OR.
11. Steif, P. S., and Dollar, A. (2005) Reinventing the Teaching of Statics, *International Journal of Engineering Education 21*, 723.
12. Wang, S.-L. (2003) MATLAB courseware for machine design, In *2003 ASEE Annual Conference and Exposition*, Nashville, TN.
13. Depcik, C., and Assanis, D. N. (2005) Graphical user interfaces in an engineering educational environment, *Computer Applications in Engineering Education 13*, 48-59.
14. Lent, C. S., Brockman, J., Goodrich, V., and Meyers, K. Teaching MATLAB in First-year Engineering: A GUI Tool Directed Approach, In *4th First Year Engineering Experience (FYEE) Conference*, pp 9-10.
15. Koh, M.-S., Rodriguez-Marek, E., and Talarico, C. (2007) Class Projects with Graphic User Interfaces in Matlab, In *2007 ASEE Annual Conference and Exposition*, Honolulu, Hawaii.
16. Pang, C. K., Wong, W. E., Li, C., and Al Mamun, A. (2008) A Toolkit with MATLAB GUI for Learning Position Error Signals in Data Storage Systems, *International Journal of Engineering Education 24*, 1242.
17. Kim, K.-J. (2007) Computer-aided Instruction of Fundamental Mechanics Courses using Matlab, In *2007 ASEE Annual Conference and Exposition*, Honolulu, Hawaii.
18. Bauer, W. (2015) HTML5/JavaScript, In *2015 LON-CAPA Conference and Workshop*, East Lansing, MI.
19. Creasy, M. A. (2016) Measuring the Dynamic in Learning, In *123rd ASEE Annual Conference & Exposition*, New Orleans, LA.