Framework for Evaluating Simulations: Analysis of Student Developed Interactive Computer Tool

Kelsey Joy Rodgers, Purdue University, West Lafayette

Kelsey Rodgers is a graduate student at Purdue University in the School of Engineering Education. Her research focus is investigating how engineers' understand, develop, and use mathematical models and simulations. Her research also focuses on feedback. She is currently conducting research in first-year engineering on the Network for Nanotechnology (NCN) Educational Research team. She previously conducted research with the Model-Eliciting Activities (MEAs) Educational Research team and a few fellow STEM education graduates for an obtained Discovery, Engagement, and Learning (DEAL) grant. Prior to attending Purdue University, she graduated from Arizona State University with her B.S.E in Engineering from the College of Technology and Innovation, where she worked on a team conducting research on how students learn LabVIEW through Disassemble, Analyze, Assemble (DAA) activities.

Prof. Heidi A. Diefes-Dux, Purdue University, West Lafayette

Heidi A. Diefes-Dux is a Professor in the School of Engineering Education at Purdue University. She received her B.S. and M.S. in Food Science from Cornell University and her Ph.D. in Food Process Engineering from the Department of Agricultural and Biological Engineering at Purdue University. She is a member of Purdue's Teaching Academy. Since 1999, she has been a faculty member within the First-Year Engineering Program, teaching and guiding the design of one of the required first-year engineering courses that engages students in open-ended problem solving and design. Her research focuses on the development, implementation, and assessment of modeling and design activities with authentic engineering contexts. She is currently a member of the educational team for the Network for Computational Nanotechnology (NCN).

Yi Kong, Purdue University, West Lafayette

Yi Kong is a doctoral student in biology education and a graduate research assistant for the Network for Computational Nanotechnology (NCN) education research team at Purdue University. She received her M.S. in agriculture in Fishery Resources from Huazhong Agricultural University and B.S. in Biological Science from Shaanxi Normal University in China. Her research includes evaluating first-year engineering students' communication of nanoscience concepts through project-based-learning activities.

Krishna Madhavan, Purdue University, West Lafayette

Dr. Krishna Madhavan is an Assistant Professor in the School of Engineering Education at Purdue University. He is Co-PI and Education Director of the NSF-funded Network for Computational Nanotechnology (nanoHUB.org which serves over 330,000 global researchers and learners annually). Dr. Madhavan was the Chair of the IEEE/ACM Supercomputing Education Program 2006. In January 2008, he was awarded the US National Science Foundation (NSF) CAREER award for work on learner-centric, adaptive cyber-tools and cyber-environments. He was one of 49 faculty members selected as the nation's top engineering educators and researchers by the US National Academy of Engineering to the Frontiers in Engineering Education symposium. Dr. Madhavan leads a major NSF funded effort called Deep Insights Anytime, Anywhere (DIA2) that attempts to characterize the impact of NSF and other federal investments in the area of science, technology, engineering, and mathematics education using interactive knowledge mining and visual analytics for non-experts in data mining. DIA2 is currently deployed inside the NSF and is already starting to affect federal funding policy. Dr. Madhavan also served as Visiting Research Scientist at Microsoft Research, Internet Services Research Group. His research has been published in Nature Nanotechnology, IEEE Transactions on Computer Graphics and Applications, IEEE Transactions on Learning Technologies, and several other top peer-reviewed venues. Dr. Madhavan currently serves as PI or Co-PI on federal and industry funded projects totaling over $20M.

Framework for Evaluating Simulations: Analysis of Student Developed Interactive Computer Tools

Abstract

Computer simulations are used in educational setting either to teach students targeted concepts or to teach students how to build them. This study focuses on the later use and investigates the nature of student teams' simulations that were developed through a design challenge that required them to create their own nanotechnology-based simulation tools. In this study, 30 teams' final simulation tools were analyzed through a grounded theory approach to categorize the nature of the simulations. The resulting framework consisted of four levels. Level 1 involves only basic interactions. Level 2 consists of a basic input to output system, referred to as a black-box model. Level 3 is an animated simulation; it has the mathematical model and visualization components of a simulation, but is lacking interactivity (or user choice). Level 4 is a simulation; it consists of a mathematical model, visualization of the model, and interactivity. Based on this framework and the findings, methods for scaffolding student learning through an open-ended simulation development challenge are proposed.

Introduction

Simulations are used in education to either enable a student to investigate a concept through an expert-developed simulation or challenge a student to build a simulation.[1-4] Activities that involve building simulations typically consist of prescriptive instruction on how to develop a given simulation; such instruction fosters passive learning.[1-4] In the literature there is a lack of open-ended simulation development activities reported. This means that little is known about how students progress from concept generation to a fully developed simulation or how to design simulation development activities that promote active learning.

Since 2010, students in a first-year engineering (FYE) course have engaged in a MATLAB-based graphical user-interface design project with a variety of contexts (e.g. games, K-12 engineering education tools, course performance monitoring systems). More recent projects have evolved from industry and research center partnerships; these partners have required the development of simulations backed by mathematical models. Anecdotal evidence from instructors implementing simulation design projects suggests that students confuse general user-interactivity (e.g. button pushing at an interface), mathematical models, and simulations. A project conducted in Spring 2013 provided an opportunity to better understand this confusion. This study is driven by the following research question: What is the nature of simulations developed by FYE student teams in their design work?

Literature Review

Simulations can be classified by the level of interactivity with and visibility of the model, the types of variables incorporated, and the type of visualization. Gould, Tobochnik, and Christian (2007) and Leemis and Park (2006) discussed these features in textbooks that instruct learners on how to build simulations and the purpose for building various types of simulations.[1,2] Alessi (2000) discussed similar features in a paper that targets the development of effective simulation tools for educational purposes.[4]

Gould, Tobochnik, and Christian (2007) explained the development of a computer simulation: "The starting point of a computer simulation is the development of an idealized model of a physical system of interest. We then need to specify a procedure or algorithm for implementing the model on a computer and decide what quantities to measure" (p. 3).[2] The interactivity of simulations is discussed in terms of the number of variables and degrees of freedom in a system. They also discussed four types of interfaces: 1) function interfaces, 2) simple graphics, 3) ode and ode-solver interfaces, and 4) visualization with 3-D motion. They also described another mode for presenting a simulation – an animation or visualization of a simulation. They explained this is not simply a video, but a type of animation that presents a captured time-segment of a simulation.

Leemis and Park (2006) described some different aspects that can be used to characterize simulations.[3] The number of variable inputs indicates the level of interactivity provided by a simulation. The different types of variables involved in a simulation can be either discrete or continuous. The models that back simulations can be deterministic (not including random variables) or stochastic (including random variables). Finally, simulations can be either static or dynamic; dynamic models are time dependent.

Alessi (2000) described five different aspects of simulations that can be used to differentiate and categorize them.[4] First, simulations can be used in educational settings for using or building. Second, simulations can have a black-box or glass-box approach. The black-box approach simply converts an input to an output through a model that is hidden from the user. The glass-box approach enables the user to see how the model works – it visualizes the process as the input changes to the output or details of the outputs that allow the user to infer details about the model. Third, simulations are either procedural or conceptual – focus on a process or a concept. Fourth, the simulations are discovery or expository – used to learn new things or to communicate known ideas. Lastly, the degree of model visibility is a way to categorize simulations; this last part also aligns with the types of interfaces discussed by Gould, Tobochnick, and Christian (2007).[2]

Although these books show various types of simulations for the purpose of helping students build effective simulations, there is a lack of research on the nature of simulations that students actually develop when they are first learning about simulations. Having students build simulations to solve open-ended problems presents a unique challenge for instructors. The pedagogical approach is not about giving clear directions of what to do, it is about scaffolding student learning through effective feedback. To give effective feedback, we need to better understand students' confusion regarding interactivity, mathematical models, and simulations.

Methods

Student teams' work on a simulation design project was analyzed to categorize the nature of their simulations. The purpose was to establish a framework for assessing and scaffolding students' learning of simulations as they develop simulations. Grounded theory was selected as theoretical framework to guide this study.[5]

Participants and Setting

Two sequential required first-year engineering courses at a large Midwestern university utilize open-ended mathematical modeling problems and design challenges and scaffolding through feedback to encourage student learning of modeling and design. This study is set in the second course. Spring enrollment in this course is typically 1300-1650 students in 12-15 sections; each section consists of up to 120 students.

This course has two team projects that each span about half of the semester. The teams of three to four students were assigned at the beginning of the semester using the team formation feature of CATME;[6] this ensures, at a minimum, that underrepresented students are not isolated and students' schedules are compatible. The first project challenges the student teams to develop a mathematical model with the fundamental purpose of increasing students' understandings of models and modeling in context. The second project challenges the students to design MATLAB-based tools to meet the needs of a project partner. This study focuses on a project that was the result of a partnership with an NSF-funded nanotechnology research center.

In Spring 2013, all students first completed the modeling project that required them to develop a mathematical model to determine the roughness of nanoscale coating samples from atomic-force microscopy (AFM) data.[7] Three sections went on to complete a simulation design project that was first piloted in Spring 2012 and revised.[8] For this project, student teams were required to develop an interactive and educational MATLAB-based program that: "that engages peers (first-year and sophomore engineering students) in learning how Size & Scale and a least one other big idea of nanoscience apply to one or more engineering disciplines via model(s) or simulation(s)." Each team member was to contribute a MATLAB-based simulation tool backed by a mathematical model to their team's cohesive package. The teams iteratively developed their simulation tools through nine milestones and with continuous feedback from instructors and the project partner.[9] Further explanation of the design projects implemented in Spring 2012 and Spring 2013, along with student project samples, can be found online at: https://nanohub.org/groups/nanodesignprojects.

Data Collection

The final submissions (Milestone 9) of all 30 teams from one section were selected as the data source for this study. The teams' final deliverable consisted of their final MATLAB-based cohesive package of simulations tools (graphical user interfaces, or GUIs, with the supporting MATLAB codes) and an executive summary describing their work. Only the GUIs (as representing the simulation tools) were analyzed.

Data Analysis

Open coding and axial coding[5] were applied as the strategies of inquiry to analyze student teams' data. Coding categories were developed based on multiple sessions to categorize the nature of the simulations seen in the data and applying the categories to the data to test and modify them. When the final categories were established, the descriptions were slightly modified based on existing literature to establish more meaningful language to describe different components of a simulation. The resulting categories were four levels that are described in the results. Each of the GUIs within a teams' project was then coded based on the established criteria to determine how many teams had content within each of the four categories.

To better understand the nature of GUIs that teams developed that did not have models and simulations, some additional information was coded for *Level 1* projects. There are three different categories for types of interactions that were coded: 1) click to pull up more information, 2) select an answer to a multiple choice question, and 3) fill in numbers or text within a given blank.

To capture greater detail of the nature of the types of models and simulations students developed there was additional coding to understand the types of inputs and outputs of the *Levels 2* through *4.* For *Level 2* and *Level 4*, the coding scheme focused on the nature of the input variable/s that users can manipulate and the nature of the outputs of the models. The input variables were coded for the number of variables and type: discrete or continuous. For *Level 3*, the coding scheme focused on the nature of the outputs of the model. Table 1 shows the different outputs coded for *Levels 2 through 4*: alphanumeric text, numeric values, pictures, and graphs.

Table 1. Types of Outputs Coded for Levels 2 – 4

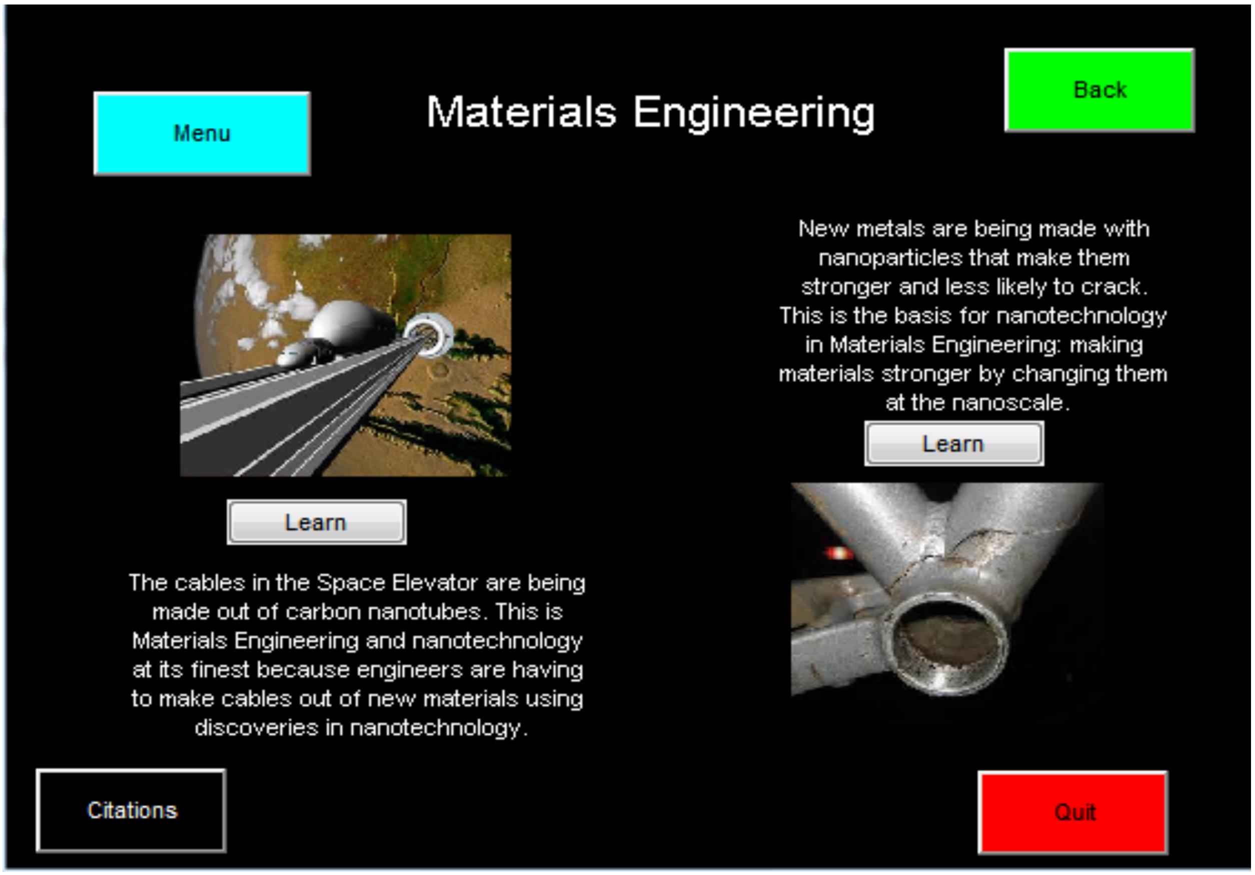
Types of Outputs	Outputs Coded		
	Level 2	Level 3	Level 4
Alphanumeric Text	X	X	X
Numeric Values	X	X	X
Picture	X	X	X
Graph		X	X

Results

Categories of Student-Developed "Simulations"

Level 1: Basic interaction. Generally GUIs at this level contain text content and clickable buttons that lead to more text or quiz-like content, both without meaningful interaction with a mathematical model. Every team had buttons on the GUIs that were used for navigation purposes to click through a collection of interfaces (e.g. next, back, home, exit), but these were not coded.

Figure 1 presents an example of a *Level 1* GUI. This GUI shows two "Learn" buttons that the user can click to bring up text. Both of the buttons have already been clicked so the text is already showing in the figure (i.e. the white descriptive text). This particular GUI was coded as two instances of *Level 1* content. The "menu", "back", "citations", and "quit" buttons were not coded as part of the *Level 1* content because these were required for navigation purposes.



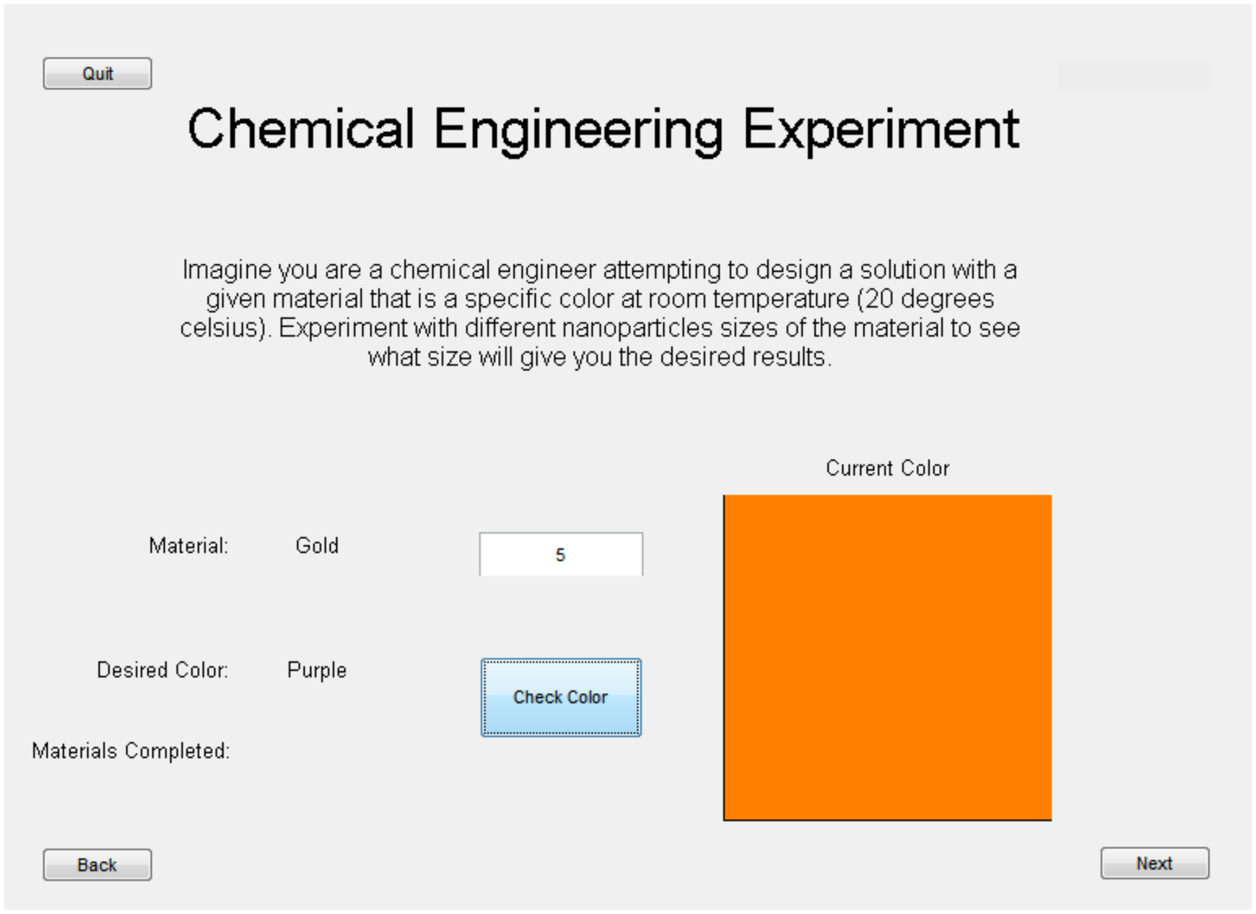
The cables in the Space Elevator are being made out of carbon nanotubes. This is Materials Engineering and nanotechnology at its finest because engineers are having to make cables out of new materials using discoveries in nanotechnology.

New metals are being made with nanoparticles that make them stronger and less likely to crack. This is the basis for nanotechnology in Materials Engineering: making materials stronger by changing them at the nanoscale.

Figure 1. Example of level 1: basic interaction

Level 2: Black-box model. This level requires some underlying mathematical model, but there is no visual representation of the nature of the model or relationship/s between the input/s and output/s. The lack of visibility of the model fulfills the definition of black-box.

Figure 2 presents an example of a *Level 2* GUI. This GUI has one input box where a user can input a numeric value that represents the size of gold nanoparticles in nanometers. The output is the colored block (currently orange). Changing the input (size) will give various outputs (colors) based on the underlying model of size dependency of optical properties. This model is not visible to users beyond this simple input-output interaction, so it is a black-box model.



Chemical Engineering Experiment

Imagine you are a chemical engineer attempting to design a solution with a given material that is a specific color at room temperature (20 degrees celsius). Experiment with different nanoparticles sizes of the material to see what size will give you the desired results.

Material: Gold

Desired Color: Purple

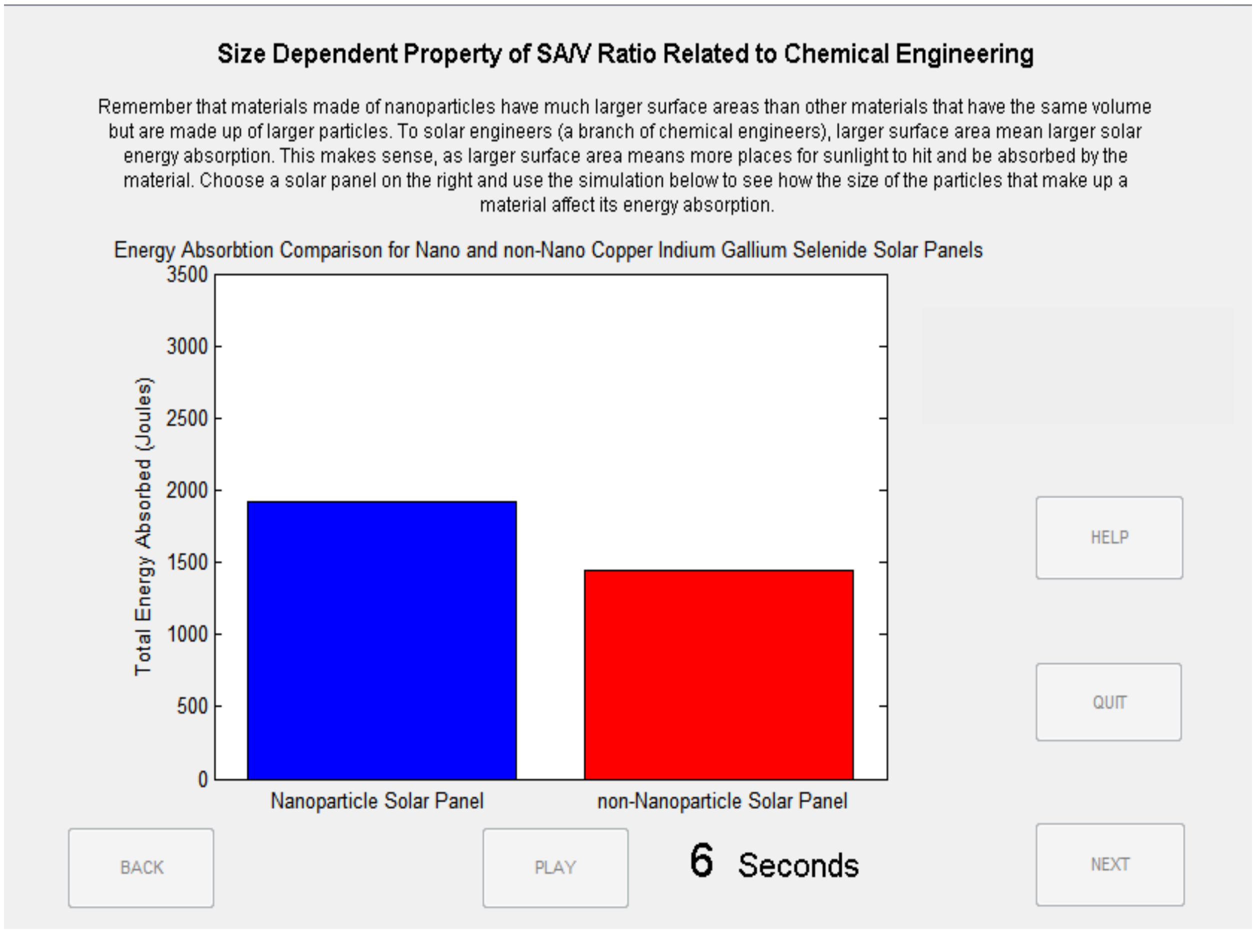
Materials Completed:

5

Figure 2. Example of level 2: black-box model

Level 3: Animated simulation. This level requires a visual presentation of a model, but users can only play the simulation with default variables; there are no input variables that the user can set. This level has a higher level of model visibility than *Level 2* and fulfills the definition of glass-box, but does not present user choice.

Figure 3 presents an example of a *Level 3* GUI. The figure is a screenshot of the GUI part way through the animation, which can be seen by the "6 seconds" note at the bottom right of the graph. The user only has the option to click the "play" button and then watch the animation of a simulation. The animation presents the amount of energy absorbed by two solar panels over time, one that uses nanoparticles in its design and one that does not. There is a model and a visual representation of it, but there is no user choice of inputs to the model.


Size Dependent Property of SA/V Ratio Related to Chemical Engineering

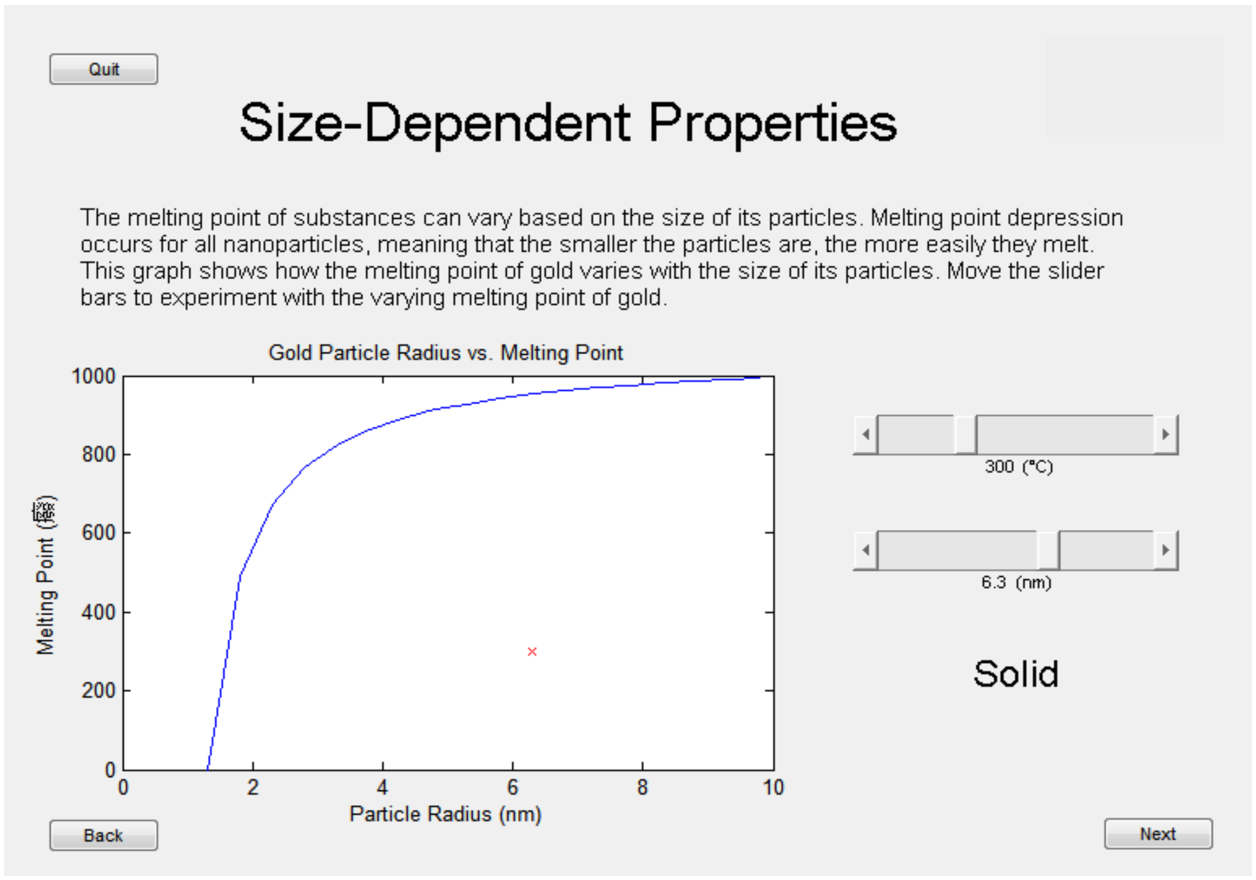
Remember that materials made of nanoparticles have much larger surface areas than other materials that have the same volume but are made up of larger particles. To solar engineers (a branch of chemical engineers), larger surface area mean larger solar energy absorption. This makes sense, as larger surface area means more places for sunlight to hit and be absorbed by the material. Choose a solar panel on the right and use the simulation below to see how the size of the particles that make up a material affect its energy absorption.

Energy Absorbtion Comparison for Nano and non-Nano Copper Indium Gallium Selenide Solar Panels
Total Energy Absorbed (Joules)
3500
3000
2500
2000
1500
1000
500
0
Nanoparticle Solar Panel
non-Nanoparticle Solar Panel
HELP
QUIT
BACK
PLAY
6 Seconds
NEXT

Figure 3. Example of level 3: animated simulation

Level 4: Simulation. At this level, the user can change input variables to explore the nature of the mathematical model behind the simulation. This level fulfills the definition of glass-box.

Figure 4 presents an example of a *Level 4* GUI. The graph shows an example of a size-dependent property; it shows that the melting point of gold increases with particle size. The two slide bars on the right are the input variables that the user can change. The user can change the temperature and the size of the gold particle to discover its current phase (i.e. solid or liquid). The red point (x) on the graph and the word "Solid" on the right are the two outputs.



Size-Dependent Properties

The melting point of substances can vary based on the size of its particles. Melting point depression occurs for all nanoparticles, meaning that the smaller the particles are, the more easily they melt. This graph shows how the melting point of gold varies with the size of its particles. Move the slider bars to experiment with the varying melting point of gold.

Figure 4. Example of level 4: simulation

The relationships among these four simulation levels are depicted in Figure 5 in terms of the presence of interactivity, a mathematical model, and visualization. *Level 1* has interaction only. *Level 2* contains a model and is interactive, but does not fulfill the visual representation requirement for true simulations. *Level 3* has a model and a visual representation of the model, but does not have user-choice interactivity. *Level 4* has all three components. True simulations are based on models, they cannot exist in isolation of models or user-interaction or visualization. Some simulations are created as animated simulations (as in *Level 3*); this removes the user-choice aspect, but these are still accepted as a form of a simulation. These animated simulations are not within the scope of this project and do not fulfill the requirement of user-choice.

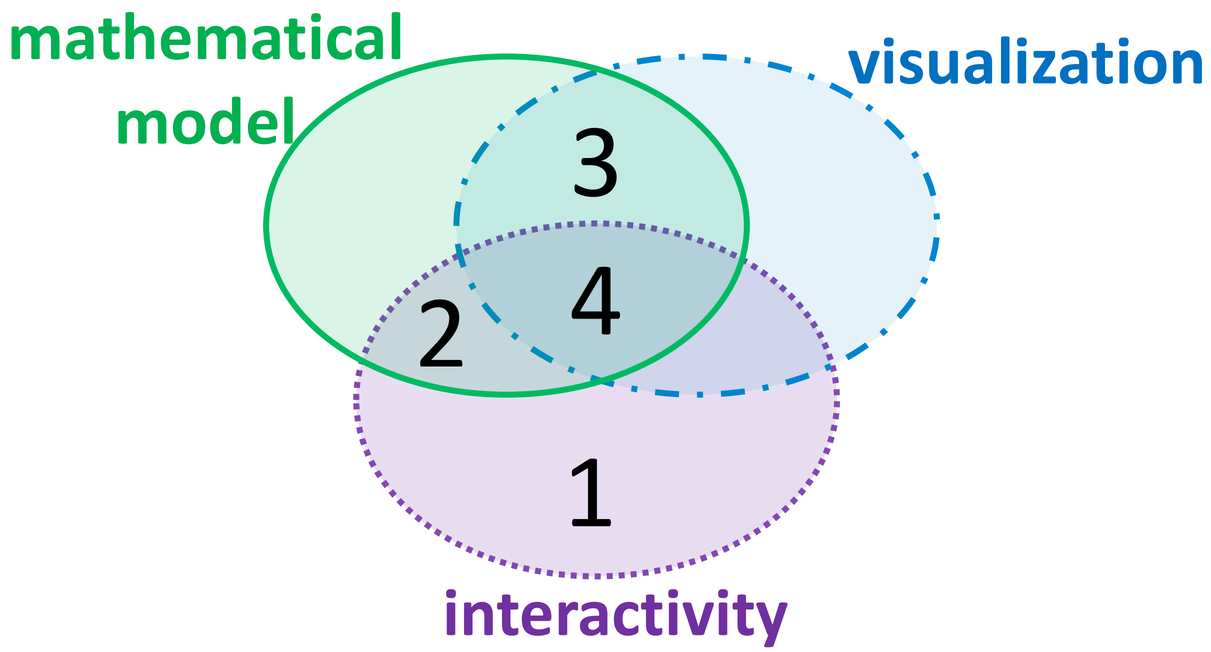


Figure 5. Components of Simulation

Number of Teams' Projects within Established Categories

All teams' projects included some basic interaction (*Level 1*), most teams' projects contained at least one black box model (*Level 2*), only one team had an animated simulation (*Level 3*), and about a third of the teams had simulations (*Level 4*) in their projects. Table 2 presents the exact number of teams that incorporated some content that fulfilled these four levels.

Table 2. Teams' Levels of Interaction, Models, Simulations

Level in Framework	**No. Teams**
Level 1: Basic Interaction	30 (100%)
Level 2: Black-box Model	24 (80%)
Level 3: Animated Simulation	1 (3%)
Level 4: Simulations	11 (37%)

The frequency of content that fulfilled the levels varied amongst the teams. Out of the 30 teams, every team had a range of 2 to 47 instances of basic interaction in their project (with a mean of 17.9, median of 16, and standard deviation of 9.6). Out of the 24 teams that had black-box models, 10 teams had one, 7 teams had two, 5 teams had three, 1 team had four, and 1 team had eight *Level 2* models. The one team that had a *Level 3* animation of a simulation had only the one example in their project. Out of the 11 teams that had *Level 4* simulations, 10 teams had one simulation and 1 team had two simulations.

The correlation between the frequency (number of instances) of *Level 1* content and the calculated value to represent the frequency of *Levels 2-4* content is -0.48. This means the more *Level 1* content; the less likely there is to be models and simulations present. The *Level 2-4* calculated values were obtained by summing (for each team) the number of *Level 2* models, number of *Level 3* animations of simulations multiplied by 2, and number of *Level 4* simulations multiplied by 3. Each of the 30 teams is represented in the Figure 6 graph. This graph shows the mapping of the number of instances of *Level 1* interaction (x-axis) to the calculated value representing the number of *Level 2-4* models (y-axis).

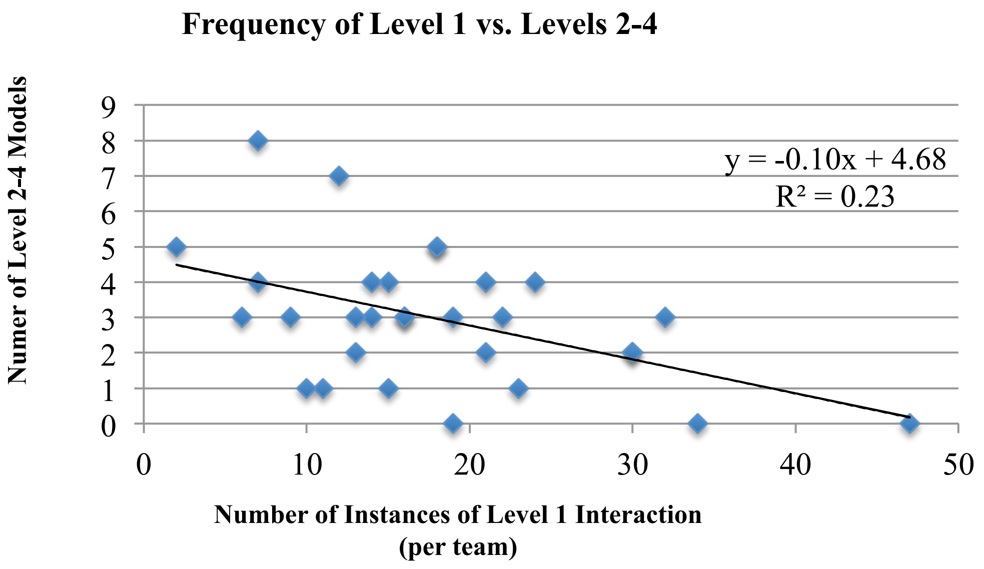


Figure 6. Graph comparing the frequencies of the levels

Details of Level 1 Projects

Level 1 interaction tools were often quizzes and information pages. Twenty-eight teams incorporated some type of multiple-choice quiz. Nine 9 teams had blank boxes in which the user to fill in a number or text. The blank boxes were typically for short answer quiz question responses. Twenty-seven teams had at least one button for pulling up additional information.

Details of Levels 2-4 Projects

There were a total of 51 *Level 2* black-box models, 1 *Level 3* animated simulation, and 12 *Level 4* simulations. There are two aspects of these models and simulations that are of interest, their inputs and outputs.

There were a total of 77 input variables for the 51 *Level 2* models. Out of these input variables, 60.0% were continuous. There were a total of 29 input variables for the 12 *Level 4* simulations. Out of these input variables, 58.6% are continuous. Table 3 presents the number of models with the corresponding number of inputs (ranging from 1 input to 5 inputs). For the four simulations that had 1 input variable, all of these were continuous variables. The *Level 3* animation of a simulation is not presented here because the user could not manipulate any input variables.

Table 3. Inputs for Level 2 Models and Level 4 Simulations

GUI Level	**Number of GUIs with respective number of input variables that users can manipulate**				
	1	**2**	**3**	**4**	**5**
Level 2	35 (69%)	8 (16%)	6 (12%)	2 (4%)	0 (0%)
Level 4	4 (33%)	4 (33%)	1 (8.3%)	1 (8.3%)	2 (17%)

There were a total of 76 outputs for the 51 *Level 2* models. The combination of types of outputs for each of the 51 models is shown in Figure 7. The one *Level* 3 animation of a simulation had two outputs (a graph and numeric values). There are a total of 15 outputs for the 12 *Level 4* simulations. All 12 simulations had a graph output. One simulation also had a numeric value. Another simulation had both a numeric and alphanumeric text output.

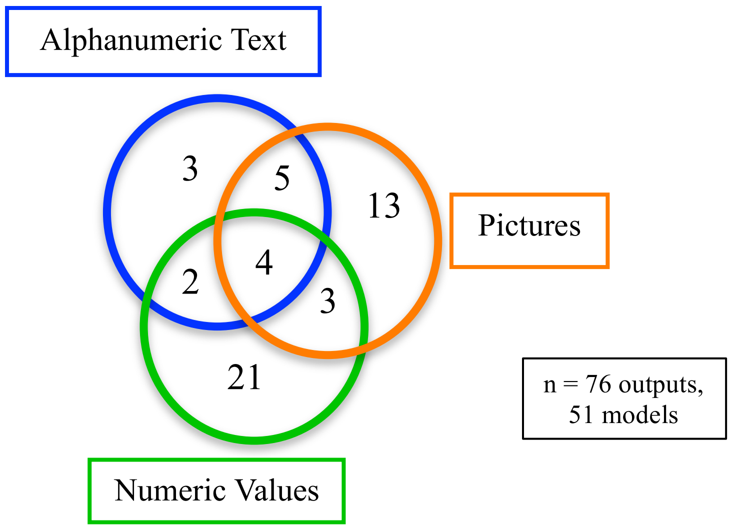


Figure 7. Venn diagram of Level 2 model outputs

Discussion

The initial finding from the analysis of student work on the simulation design project was the simple realization that the majority of first-year engineering students do not know what simulations are. Carberry and McKenna (2014) found that students typically do not know what mathematical models are either.[10] During the weeks of this project, there were minor lecture interventions to get students on track, but only minimal scaffolding focused on helping students learn about simulations. Scaffolding and assessment should focus on three key elements of a simulation: interactivity (user choice) for exploring the model, a mathematical model is the basis for a simulation, and visualization of the model is necessary. All three of these elements need to be present for a successful, rudimentary simulation.

Based on these results, we believe that students' progress through three fundamental levels of understanding to develop a *Level 4* simulation. A student must understand how to implement user interaction into a GUI to reach *Level 1*. A student must also understand what a mathematical model is and how to incorporate it into a GUI. Once a student has developed a mathematical model with user interaction in a GUI, they have reached *Level 2*. The next step is to help the students understand what visualization is, the importance of incorporating visualization to create a simulation, and how to develop visualizations. Effectively developed visualization enables more meaningful exploration of the model. Creating a simulation requires user interaction, a mathematical model, and a visual representation.

This progression of understanding how to develop a simulation does not include a *Level 3* animated simulation. We believe that this only occurred in our data set once because it is an overshot in trying to develop a simulation. There should be further investigation into a potential misunderstanding that a simulation requires animation. If a student team reaches a *Level 3* animation of a simulation, it is important to guide them towards a *Level 4* simulation by explaining the need for user choice and interaction to make their current project into a simulation.

The negative correlation between *Level 1* and *Levels 2-4* content shows that students with more basic interactions are less likely to have a model, animation of a simulation, or a simulation. One potential explanation for this is that students realized the need for user interaction in the call for a simulation. They may have received poor scores on the simulation component of their project grade and responded by continuously trying to increase the amount of interaction without realizing they were not including a model. This may further point to the need for students stuck at *Level 1* to have more direct feedback about the inclusion of a model to progress to the next level.

One notable finding about the input variables is that there appears to be a relationship between more input variables and higher-level models. Students with *Level 2* models or *Level 4* simulations need help understanding that increasing the number of inputs increases the user's ability to explore the mathematical model. Such feedback would help students better connect simulations with mathematical models.

Conclusion

This study was followed-up with the development of a guided instructional tool.[11] This tool presents students with the opportunity to evaluate a selection of previous student work that span Levels 1 to 4 before beginning to develop their own simulation projects. The tool inquires about the level of interactivity and presence of models and/or simulations. The final questions ask students to reflect on the sample pieces of work and how seeing them influences their design choices. This intervention will be assessed in future studies by comparing students' responses to the guided instructional tool and the simulation level achieved by the teams.

A limitation of this study is that it does not address the various types of simulations that occur at *Level 4* simulation. A future study should be conducted to understand how to scaffold student development of better simulations once they have reached this level.

Acknowledgment
This work was made possible by a grant from the National Science Foundation (NSF EEC 1227110; NSF EEC 0956819). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

Bibliography

1. Bell, Randy L., and Lara K. Smetana. 2008. *Using Computer Simulations to Enhance Science Teaching and Learning*. National Science Teachers Association.
2. Gould, Harvey, Jan Tobochnik, and Woflgang Christian. 2007. *An Introduction to Computer Simulation Methods*. Addison-Wesley.
3. Leemis, Lawrence M., and Stephen Keith Park. 2006. *Discrete-Event Simulation: A First Course.* Pearson Prentice Hall.
4. Alessi, Stephen. June 2000. "Designing Educational Support in System-Dynamics-Based Interactive Learning Environments." *Simulation and Gaming*. Vol. 31, pp. 178-196.
5. Strauss, Juliet and Anselm Corbin. 1990. *Basics of Qualitative Research: Grounded Theory Procedures and Techniques.* Sage Publications.
6. Ohland, Matthew W., et al. 2006. "The comprehensive assessment of team member effectiveness (catme): A new peer evaluation instrument." *Proceedings of the 2006 ASEE Annual Conference*.
7. Diefes-Dux, Heidi A., Margret Hjalmarson, Travis K. Miller, and Richard Lesh. "Chapter 2: Model-Eliciting Activities for Engineering Education." In Judith S. Zawojeski, Heidi A. Diefes-Dux, and Keith J. Bowman (eds.) 2008. *Models and Modeling in Engineering Education: Designing Experiences for all Students*. The Netherlands: Sense Publishers, pp. 17-35.
8. Rodgers, Kelsey J., Heidi A. Diefes-Dux, Krishna Madhavan, and Bill Oakes. "First-Year Engineering Students' Learning of Nanotechnology through an Open-Ended Project." In American Society for Engineering Education Conference. 23-26 June 2013. *Frankly, We Do Give a D*mn. Proceedings of the 120th Annual ASEE Conference and Exposition.* Atlanta, Georgia.

9. Rodgers, Kelsey J., Yi Kong, Heidi A. Diefes-Dux, and Krishna Madhavan. "First-year engineering students' communication of nanotechnology size and scale in a design challenge." In American Society for Engineering Education Conference. 15-18 June 2014. *360 Degrees of Engineering Education. Proceedings of the 121st Annual ASEE Conference and Exposition.* Indianapolis, Indiana.

10. Carberry, Adam and McKenna, Anne. (2014). Exploring student conceptions of modeling and modeling uses in engineering design. *Journal of Engineering Education, 103(1)*, 77-99.

11. Rodgers, Kelsey J., Heidi A. Diefes-Dux, and Krishna Madhavan. "Investigating first-year engineering students' understandings of computer simulations and interactivity." In European Society for Engineering Education Conference. 15-19 September 2014. *Educating Engineers for Global Competitiveness. Proceedings of the 41st SEFI Annual Conference.* Birmingham, UK: University of Birmingham.