# Case Study of a Video Game Design & Development Course for Mechanical Engineers

**Dr. Joseph Michael Mahoney, Pennsylvania State University, Berks Campus**

Dr. Joseph Mahoney is an Assistant Professor of Mechanical Engineering at Penn State Berks. He received both his BS (with Honors) and MS in Mechanical Engineering from Penn State. He received his Ph.D. in Engineering Science and Mechanics, also from Penn State. His research is broadly in the area of Biomechanics. His teaching is in Statics, System Dynamics, Vibrations and Video Game Design. He is a member of ASB and has reviewed for BMC Musculoskeletal Disorders, International Journal of Industrial Ergonomics, Ergonomics, and Safety and Health at Work.

# Case Study of a Video Game Design & Development Course for Mechanical Engineers

**Joseph M. Mahoney**[1]

[1] *Penn State Berks, Division of Engineering*

## Abstract

Anecdotally, it has been observed that many engineering students are not motivated to learn or apply computer programming in their courses. Possibly, this is due to computer science topics being pushed upon them rather than students learning them as needed. A senior-level video game design class was offered as a technical elective. This class combined a "humanities" viewpoint of video game design (e.g. gaming psychology, theory of fun) with the "technical" side of computer programming and physics engines. Students compared and contrasted two games and wrote a critical analysis of a game. The majority of the class was spent conceptualizing, planning and creating a video game. Groups learned the required programming skills as needed to implement their vision. Students completed a survey at the conclusion of the course. Generally, students found the course exercised their creative skills, motivated them to learn more programming and provided them experience in project management.

## Keywords

video games, computer programming, elective, creativity, group work

## Introduction & Background

Extensive research has been done showing the efficacy of using games, including video games, as instructional resources in the classroom at the K-12 and university levels to improve engagement, motivation, and outcomes[1,2]. Games have been shown to be valuable for teaching technical and non-technical subjects. A very limited selection of examples includes Mathematics, Reading and Spelling[3], Biology[4] and Social Studies[5].

Some research has been performed in utilizing game *creation* as a pedagogical method for computer programming. Games have been employed as a motivational tool for students to learn the material[6-9]. For first-year computer science (CS) students, some implementations include creating Minesweeper, Asteroids[6] and dice[7] games. According to survey data, students responded positively to having a game-based instruction of programming. These programs have been run as first courses for CS majors that presumably already have an interest in programming.

Game-based curricula have also been implemented for non-CS majors. For non-technical and younger students, Scratch has been used as an introductory programming system. Scratch has attractive features, compared to more "traditional" languages in that it is web-based, graphical-based and has a vibrant social component of sharing and remixing other's code[10]. This simplified environment can impart the basic ideas of algorithms, loops and conditional statements to students that have never programmed[11-12].

The use of video game creation as an instructional tool for non-CS engineers is less explored. These students need to use programming as *part* of their work. They, more than CS majors, often

suffer a lack of motivation for learning programming. Students are traditionally presented with a programming method they do not understand to solve a problem that they do not care about in their introduction to programming course[8,13]. One introductory engineering course gave students a video game shell for a racing game that they had to complete[14]. Here, they applied different numerical methods as needed to get the game to function.

Inspired by previous work in video game creation as a motivating tool for students to learn to program, a new senior-elective course was offered (in spring 2017) to Mechanical Engineering (ME) and Information Systems Technology (IST) majors on our campus, entitled, "Video Game Design and Development." This course had a prerequisite of any computer programming course.

**Course Philosophy, Format & Assignments**

The course was designed to include both rigorous technical components and "social science" components. The first half of the class took an academic view of video games from a psychology perspective. Starting with theory and viewpoints such as Koster's Theory of Fun and Lazzaro's Four Keys, students engaged in discussion and cited examples. In other upper-level engineering courses, discussion and difference of opinion do not occur. This course exposed students to ideas that did not come from first principals of physics but from observations.

The instructor presented interactive lectures during the first three weeks of the course. Then, the presentation of course content was handed over to the students. Groups of four students were responsible for delivering a lecture, moderating a discussion and running activities for a week (2.5 hours) of class. They could utilize textbook and online resources for their presentations.

Students wrote two compositions for the course blog. During the first third of the course, they compared and contrasted a game made before 2002 with its sequel or spiritual successor made after 2012. During the middle third of the semester, they critically analyzed a recent AAA game. Students chose the games for analysis with the instructor's approval. These assignments were focused on the theory portion of the course and applying it to real-world games. Furthermore, they required the students to think about and critique video games in an academic way.

During the second half of the course, students worked primarily on the overarching course project. Here, they were to conceive, design, and implement a video game. This project had milestones throughout the semester of (1) an initial pitch, (2) midterm update, (3) bug tracking report, (4) final demonstration and (5) final report. Groups were free to choose to make anything from a graphics-free text adventure to a graphics-intensive 3D first-person shooter. No restrictions were put on the language, engine or platform they could use for their game. The desire for the course was to not focus on the graphics components but rather the gameplay and the underlying programming to support it. This, unfortunately, is not the direction the teams chose.

One goal for the course was for student work to not be submitted to the instructor for a grade and then forgotten about. Rather, students contributed their work to the broader community for reference, criticism and reuse. Students agreed at the beginning of the course that all work would be submitted under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License. To that end, all work is publicly available. All work was peer reviewed before the final version was graded.

## Outcomes

*Student Work*

Students were challenged to make their blog responses not only thorough but *interesting* for the reader. In their upper-level courses, assignments are written and formatted as a technical or lab report. They had to shift their tone and presentation to appeal to a different audience.

For the game creation project, all four groups initially choose to create their games in a modern physics engine (one used Unreal Engine 4 (UE4), two used Unity, and one used GSC, the Call of Duty Engine). One group switched from Unity to Visual Basic. All groups produced a working interactive demonstration of their game by the end of the course.

All student blog posts, project updates and final presentations for the course can be viewed at https://sites.psu.edu/ist446bk/.

*Student Survey Data*

To determine the students' perceptions about the course, an online survey was administered during finals week via *Qualtrics*. A total of $n = 11$ ME students responded to the survey. Two students in IST responded but their submissions are not included in this analysis. Participants were asked the Likert-scale questions listed in Table 1 with the scale values listed in Table 2. Additionally, they were given the option to include open-ended commentary with their answers.

*Table 1: Survey questions given to students. Respondents selected their Likert-scale response using the choices in Table 2. Prompts marked with * (C2 and M2), used the second column of responses. All others used the first.*

| Code | Prompt |
|---|---|
| | *In my first programming course:* |
| Q1 | The way the material was presented in the class (e.g. lecture, video or textbook) helped me learn to program |
| Q2 | The homework and project problems were interesting to solve using programming |
| Q3 | At the end of the course, I was motivated to learn more programming skills |
| Q4 | I remembered what I learned in the course for use in upper-level courses |
| G1 | Having an end goal (i.e. making a video game) motivated me to learn the necessary programming |
| G2 | Learning a new programming language or programming skills in a group was helpful |
| G3 | Already knowing at least one programming language made it easier to learn a new one |
| C1 | This course required me to be creative in order to be successful |
| C2 | Compared to my other **in-major** courses, the creativity required in this course was * |
| C3 | I would prefer that my other **in-major** courses allowed me to be creative or inventive |
| M1 | To complete the project in this course, good organization and project management skills were required |
| M2 | Compared to my other **in-major** courses, the organization and project management required in this course was * |

The seven-point Likert responses to the questions in Table 1 were coded to a numerical value shown in Table 2. These values were used in the statistical analysis and act as the shorthand reference.

*Table 2: Likert-Scale choices for survey questions in Table 1. The numeric code was used for statistical analysis and shorthand notation.*

| Code | Choice | Choice* |
|---|---|---|
| 1 | Strongly Disagree | Much Lower |
| 2 | Disagree | Moderately Lower |
| 3 | Slightly Disagree | Slightly Lower |
| 4 | Neither Agree nor Disagree | About the Same |
| 5 | Slightly Agree | Slightly Higher |
| 6 | Agree | Moderately Higher |
| 7 | Strongly Agree | Much Higher |

The median value for each response was individually compared with "4" (the neutral response). All survey analysis was performed using a one-tail Sign test. The response distributions were generally not symmetric about the median and not normally distributed. The significance threshold was set *a priori* to $\alpha = 0.05$.

A boxplot of student responses to questions regarding first programming course (Q1-Q4) is shown in Figure 1. Here, a left-tail test was used to test if the median response was significantly *lower* than "4" (thus, students generally *disagreeing* with the prompt). No significance ($p \gg 0.05$) was found for all prompts.

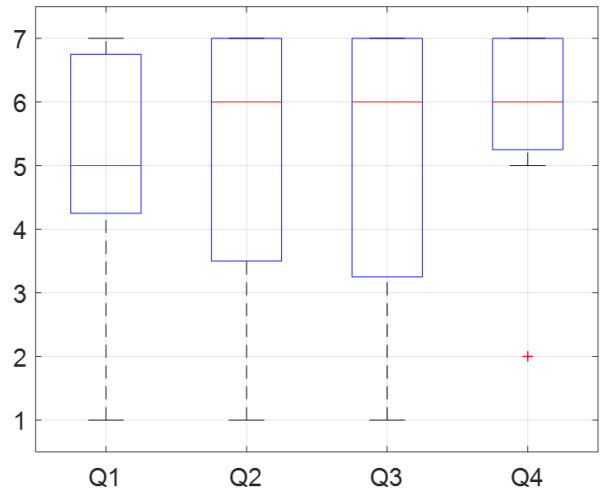


*Figure 1: Boxplot of responses concerning the first programming course from Table 1. No prompt was found to be generally disagreeable ($p \gg 0.05$ for $x < 4$).*

A boxplot of student responses to the outcome questions (G1-M2) is shown in Figure 2. Here, a right-tail test was used to test if the median response was significantly *greater* than "4" (thus, students generally *agreeing* with the prompt). Significance ($p < 0.05$) was found for all responses except for G2 (group learning).

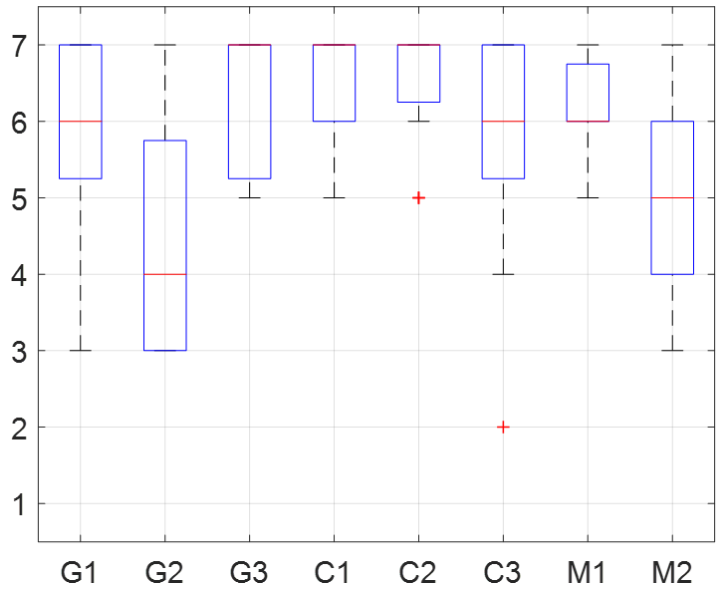


*Figure 2: Boxplot of Responses from Table 1. All questions but G2 (group learning) had $p < 0.05$ for $x > 4$.*

## Discussion

Surprisingly, the median student opinion was not in disagreement with the four prompts (Q1-Q4) concerning their introductory computer programming course. Furthermore, the median ratings were all greater than "4," though only significantly for Q4. This sentiment does not echo previous work investigating student motivation in introductory programming courses[8]. However, it should be noted that this course was an elective so there may be a selection bias of students more interested and adept in computer programming than the average student.

Based on question G1, students generally felt motivated to learn programming methods and techniques to accomplish their end goal of creating a video game. This agrees with previous work finding game creation to be a motivating factor[6-9]. Unlike these previous studies, all students in this course already had a foundation in programming prior to the course. There were no specific programming objectives set and each student chose their own path. This experience helps guide students to be lifelong learners and seek out skills and information as they need it.

G2 (team learning) was found not to be significant. Some of the groups delegated programming tasks to different members so they may not have been learning the same skills at the same time.

An unanticipated outcome was students being drawn to the "creative" aspects of the course (C3). Student groups had complete freedom in the design and implementation of their games. Students reported that the class required more creativity than their other core classes. The concept of "creativity" is, admittedly, nebulous. Furthermore, students generally wanted more creative outlets in their other core courses (C1-C2). Again, there is a sampling bias of students that selected to take this course and this recommendation may not apply to all engineering students. However, these students sought an outlet for their creativity that they felt was missing from the general program.

The need for engineering students to be able to form creative solutions or "think outside the box" has been a subject of recent consideration[15,16]. Teaching and promoting creativity in technical

majors is important but overlooked (at least in our program). Most courses have students learn equations and principals and then apply these new tools to similar problems that they saw in class. It is worth investigating how to integrate concepts from a broader range of courses (technical and non-technical) to create unique solutions to homework and project problems.

Students cited that the course required good project management and planning skills (M1-M2). Having more experience with large-scale projects is beneficial to upper-level engineers as they prepare to apply for industry jobs or graduate school.

As a final caveat on the analyses, all outcome measurements were based on self-reported views at the end of the course. Though most students participated in the survey, the statistical power is limited by having only eleven respondents in the sample. However, statistical significance was still found for most prompts.

**Lessons Learned & Future Course Changes**

The median student rating for their experience in the course was "extremely satisfied" (seven gradations from "extremely dissatisfied" to "extremely satisfied"). Students felt they received value from taking the course.

All groups were drawn to the modern game engines for their project. Unfortunately, the learning curve on these software packages is shallow and groups spent most of the semester learning the basics of the software rather than designing and refining their games. Though students did learn programming and software skills through the project, they were not using the fundamental programming skills that were directly transferable to engineering problems.

The next time the course runs, students will be given online resources (e.g. Lynda, Couresera) over winter break if they want to use a modern engine (Unity, UE4). Otherwise, they will be restricted to more functional programming (C++, Python, etc.). The programming project will begin the first week of class rather than waiting until some theory is presented.

No good examples of critical analyses of video games were found prior to the course. Now with several complete from this course, future students will have a better idea of the format and expectations for the assignment.

This implementation of the course and the survey represents only a pilot and initial data collection. The next instance of the course will have students complete a pre- and post-survey with objective and subjective questions. Questions will be more closely tied to the previous literature and more specific.

**Acknowledgments & Notes**

The author would like to thank the spring 2017 class for their participation in the first offering of this course. All survey data were collected with approval of the Penn State IRB. 

## References


1 Becker, Katrin. "Commercial Off-the-Shelf Games (COTS)." Choosing and Using Digital Games in the Classroom. Springer International Publishing, 2017. 101-118.

2 Mitchell, A. & Savill-Smith, C. The use of computer and video games for learning: A review of the literature Learning and Skills Development Agency, 2004

3 Rosas, Ricardo, et al. "Beyond Nintendo: design and assessment of educational video games for first and second grade students." Computers & Education 40.1 (2003): 71-94.

4 Annetta, Leonard A., et al. "Investigating the impact of video games on high school students' engagement and learning about genetics." Computers & Education 53.1 (2009): 74-85.

5 Maguth, B. M.; List, J. S. & Wunderle, M. Teaching social studies with video games The Social Studies, Taylor & Francis, 2015, 106, 32-36

6 Becker, Katrin. "Teaching with games: the minesweeper and asteroids experience." Journal of Computing Sciences in Colleges 17.2 (2001): 23-33.

7 Adams, Joel C. "Chance-It: an object-oriented capstone project for CS-1." ACM SIGCSE Bulletin 30.1 (1998): 10-14.

8 Moser, Robert. "A fantasy adventure game as a learning environment: why learning to program is so difficult and what can be done about it." ACM SIGCSE Bulletin. Vol. 29. No. 3. ACM, 1997.

9 Leutenegger, S. & Edgington, J. A games first approach to teaching introductory programming ACM SIGCSE Bulletin, ACM, 2007, 39, 115-118

10 Resnick, M.; Maloney, J.; Monroy-Hernández, A.; Rusk, N.; Eastmond, E.; Brennan, K.; Millner, A.; Rosenbaum, E.; Silver, J.; Silverman, B. & others Scratch: programming for all Communications of the ACM, ACM, 2009, 52, 60-67

11 Ouahbi, I.; Kaddari, F.; Darhmaoui, H.; Elachqar, A. & Lahmine, S. Learning basic programming concepts by creating games with scratch programming environment Procedia-Social and Behavioral Sciences, Elsevier, 2015, 191, 1479-1482

12 Ke, F. An implementation of design-based learning through creating educational computer games: A case study on mathematics learning during design and computing Computers & Education, Elsevier, 2014, 73, 26-39

13 Coller, Brianno. "Implementing a video game to teach principles of mechanical engineering." 114th Annual ASEE Conference and Exposition, 2007. 2007.

14 Coller, B. D. & Scott, M. J. Effectiveness of using a video game to teach a course in mechanical engineering Computers & Education, Elsevier, 2009, 53, 900-912

15 Cropley, David H. "Promoting creativity and innovation in engineering education." Psychology of Aesthetics, Creativity, and the Arts 9.2 (2015): 161.

16 Daly, Shanna R., Erika A. Mosyjowski, and Colleen M. Seifert. "Teaching creativity in engineering courses." Journal of Engineering Education 103.3 (2014): 417-449.

**Joseph M. Mahoney**

Dr. Joseph Mahoney is an Assistant Professor of Mechanical Engineering at Penn State Berks. He received both his BS (with Honors) and MS in Mechanical Engineering from Penn State. He received his Ph.D. in Engineering Science and Mechanics, also from Penn State. His research is broadly in the area of Biomechanics. His teaching is in Statics, System Dynamics, Vibrations and Video Game Design. He is a member of ASB and has reviewed for BMC Musculoskeletal Disorders, International Journal of Industrial Ergonomics, Ergonomics, and Safety and Health at Work.