https://doi.org/10.2352/ISSN.2470-1173.2017.3.ERVR-090

Virtual Reality Instructional Modules in Education Based on Gaming Metaphor

Sharad Sharma, Emmanuel Ossuetta, Department of Computer Science, Bowie State University, Bowie, MD 20715, USA
ssharma@bowiestate.edu, ossuettae0319@students.bowiestate.edu

Abstract

Virtual reality instructional modules are widely recognized in academia because they engage students and motivate them to learn by hands-on experience. For this reason, we have developed virtual reality instructional (VRI) modules for the teaching loops and Arrays that can provide a better understanding of the concept than with a traditional instruction approach. This paper focuses on the development and evaluation of the VRI gaming module, which adds more inquiry-based problem-solving activities and hands-on experiences based on gaming and virtual reality. VRI modules are designed to encourage faculty to teach and motivate students to learn the concepts of loops and Arrays using interactive, graphical, game-like examples. The VRI modules act as a supplement to an existing course and enables faculty to explore teaching with a game-theme metaphor. We have evaluated the VRI modules in introductory programming courses during the semesters for computer science students. The student survey baseline results demonstrate positive student perceptions about the use of gaming instructional modules to advance student learning and understanding of the concepts. The results of the evaluation of VRI modules also demonstrate the effectiveness of instructional modules and the possibility to include them in the existing curriculum with minimum alterations to the existing established course material.

1. Introduction

Learning algorithms during early programming classes in a university is not a simple task, as it involves problem-solving techniques. It requires logical thinking and more inquiry-based problem-solving activities. Students need more concentration and attention to learn algorithms and data structures [1]. The traditional way of teaching is text-based programming, which is not so inspiring to most of the students. Today computers are a part of everyone's life regardless of their educational level. Teaching algorithms using gaming modules can help both students and instructors [2, 3]. In traditional teaching, the instructor clarifies a topic and students take notes. Education now involves encouraging students to stir their interest and yearning to learn. Due to the change in education system, a number of different teaching modules are being developed, including the following:

- Flipped classroom: Urging students to get ready for the lesson before class.
- Case method: It involves solving real-life cases by group analysis and innovative ideas. Stimulates student's interest, explanatory abilities, and imagination.
- Self-learning method: Learning exercises intended for students to do freely when they cannot go to individual or gathering instruction sessions (e.g., mind map, a visual representation of a lecture that improves students learning in different ways.
- Online tools for learning: There is a variety of free web-learning devices accessible that educators can use to energize engagement, investment, and a feeling of fun into the classroom.
- Gamification: It involves teaching students with the help of games.

Among these teaching modules games, play an important role. They create interest and act as a medium to learn by providing information in an interactive way [4]. Games also provides motivation, as it improves thinking capacity for a particular task. Gaming modules are superior to any other teaching method [5]. Game-based teaching modules are developed to encourage students and instructors [6]. Game in education easily grabs the student's attention due to an emotional link between game and player [7]. Researchers have proved that using games as teaching modules enhances the learning and understanding capacities [8, 9]. Many gaming instruction modules have been used successfully to help students in understanding the difficult programming concepts [10-14].

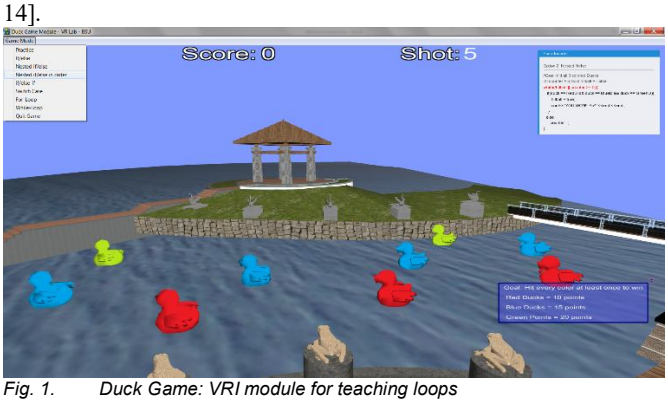

Fig. 1. Duck Game: VRI module for teaching loops

Education and the retention of training principles via a hands-on approach has been employed in various disciplines. The constructivist approach is used frequently in education and training to provide an in-depth grasp of the subject matter taught to students. In constructivism it is believed that people learn through understanding past experiences, reflecting on past experiences, and adding detail to knowledge already known [15-16]. Constructivism is based on the theory that hands-on activities and personal experiences generate knowledge and knowledge is not transmitted. In other words, training is not maximally effective if the subject matter is presented passively. The trainee must actively take the information given and fit it into their belief system. Learning is achieved through student-centered activities and not instructor-led activities. The training is enhanced through social interaction with others;. In doing so, the trainees can compare their ideas and learn from others [17]. In addition, to gain full potential from training, students must display their knowledge in a variety of ways, instead of just examinations. The training must simulate situations that could occur in real life [18]. Learning is best achieved when the student is fully immersed within the 3-D learning environment [19]. The virtual reality classroom, which employs a 3-D model of the

classroom, has been used for teaching using a constructivist approach [20]. It is based on powerful scientific visualization techniques and can be used as an effective aide in online virtual teaching.

This paper discusses the design, implementation, and evaluation of the two virtual reality instructional (VRI) modules for the teaching loops and Arrays to computer science students in an introductory programming course. The Duck Game was developed to teach loops, as shown in Fig. 1. The VRI modules are developed using virtual reality software and provide better understanding of the concept of loops and Arrays. The gaming modules demonstrated in this paper were developed using vizard, which is a virtual reality toolkit used for developing virtual worlds and immersive applications for visualization and simulation using Python as its scripting language. Vizard supports high-quality 3-D sound and multi-user networking and can import other gaming environments by adding them in the script. The main objective of the VRI module is to provide better understanding of the concept of loops and Arrays by using a constructivist approach of learning by doing. During the semesters, the modules were evaluated with undergraduate and graduate students for introductory programming courses. The students experimented with the VRI module and took a survey afterward. The evaluation studies have shown a remarkable improvement in conceptual understanding of the subject. Additionally, these studies include the results of impact on enhanced student learning.

In this paper, we show two VRI modules implemented for introductory program courses in Computer Science majors: the Array game, which teaches 2-D Arrays, and the Duck Game, which teaches loops. Section II briefly describes the work done previously. Section III discusses the modeling of VRI modules. Section IV discusses the implementation of VRI modules. In Section V, we evaluate the effectiveness of the two modules with a class of undergraduate and graduate computer science majors. It also describes the results. Section VI includes conclusions and future work.

2. Related Work

Educational software helps students learn concepts by applying classroom theory to real-world events. Sharma et al. [21] have demonstrated the use of the Game–Theme Based Instructional (GTI) module for teaching object-oriented programming to computer science students. GTI modules were designed to encourage faculty to teach and motivate students to learn the concepts of object-oriented programming by using interactive, graphical, game-like examples. On the other hand, M. Tretinjak et al. [22] have focused on teaching through the flipped teaching technique and gamification. In flipped teaching, students study the materials from pre-existing videos, research papers, and assignments. During class time, students perform interactive activities, exercises, and discussions. On the other hand, gamification is based on teaching by the use of games as teaching modules. They can be used alone or along with other teaching modules [22].

Doerschuk [23] has presented a java-based gaming platform called greenfoot. It consist of five modules, each module focusing on computing fundamentals. Each module consists shell of game, lessons on programming concepts, exercises, and questionnaires. By analyzing the performance of students in quizzes, it was concluded that students showed significant growth on programming concepts with that gaming module. T. Singh et al. [24] have used World Wide Web applications for their research into learning application. Complete software modules were provided to students for improving the students learning capacity. They have used virtual environments such as CAVE, immerse desk, and infinity wall for their study. They concluded that virtual reality environments provide high degree of realism. Carol C. et al. [25] have proposed low-cost online modules to enhance the introductory programming course taught in C++. These modules complemented the existing course module. A set of online tools was developed that can reduce the repetition of the classes. Those modules offer tutoring, which was helpful for the students. Apart from tutoring, the module also had quiz sections, which were helpful for self-check. Ekong [26] has also described using an instructional module to teach ethics. This module contains case studies and classroom exercises. Yohannis [27] divided his research into three phases to develop an instruction module. In the first phase, a literature survey was conducted to provide a concept-to-design model based on existing theories. The second phase involved the designing of the instruction module. In third phase, testing examined the performance of the module in how it influenced the learner outcomes and learner engagement. Yohannis [27] conducted a user study on 59 participants. Participants were divided into two groups. The first group was book-first group and the second was a game-first group. A textbook was provided to the first group for reading algorithms while a group gaming module was given to the second group. The results showed that the participants had better scores with the use of gaming module than with the textbook.

Byrne et al. [28] have proposed a learning algorithm using visualization techniques. Visualization transforms algorithm concepts into organized forms to show process steps while–a visual illustration of the algorithm. By this approach learners can easily grasp the concepts of the algorithm. Clark et al. [29] have proposed the use of modular education in games by providing a framework that would be capable of entertainig students as well as offering a natural, effectively adaptable path for instructors. This module was based on a client-server model and consisted of three layers that include database, interface, database manager. In the database layer teachers create new lessons and stores old lessons. The interface layer is between the database and the manager, reducing querying and storing lesson information. The database manager layer was so that new games can be added to the system. Cetin [30] has also presented a game-based application for engineering education using a 3-D gaming environment.

3. Modeling the Virtual Reality Instructional Module

This section discusses the modeling of the two VRI modules: Duck Game and the Array game. It gives a brief overview of the Vizard platform used to develop the games as well as the concepts covered in the duck and Array game.

3.1. Design Considerations

The modules were built with the constructivist theory in mind, which states that the students build knowledge by experiencing it for themselves in the real world. The modules were also built using functional and nonfunctional requirements. The functional requirements were taken from the student's perspective while the nonfunctional requirements were taken from the instructor's perspective. The functional requirements include the following:

1. *User interface must be intuitive* – students using any of the modules should not have any difficulty interacting with them.
2. *The student should be able to restart either module at any time* – if a student experiences a technical difficulty (e.g., a

module crashes, not getting correct responses), then the module should be restarted.
3. *The student should be able to understand the instructions* – user instructions detailed in each module should be easy to understand so that the student can navigate through the module without confusion.
4. *Pseudocode displayed should be comprehensible* – as with the instructions, pseudocode should also be easy to understand. The purpose of it is to give the student a high-level view of how a loop or Array operates.
5. *Within each module, the student should be able to switch between one game and another* – if a student grows tired of a game within any module, the student should switch from that game to another without difficulty.

The nonfunctional requirements include the following:
1. *Each module should motivate the student to learn* – the objects (pseudocode, graphics, etc.) featured in each module should spawn interest in the student for programming. Each module should motive the student to further his/her knowledge of loops, Arrays, and other data structures.
2. *Each module should teach the student about the subject* – although each module features examples from disciplines outside of Computer Science, the module should ultimately teach the student about loops and Arrays. The examples featured in each module should not deviate from the subject matter.
3. *Reaction to user input should be immediately rendered* – the student should not wait for any reaction to his/her input (i.e., button clicks). Waiting for a reaction could impair user experience.
4. *Graphics should be appealing to the student* – each module should feature graphics (3-D models, buttons, text, etc.) that visually appeal to the user. This requirement enhances user experience.
5. *The modules should be portable* – the student should be able to play each module on any platform. That is, it should work whether the student is using a Windows or Macintosh system.
6. *An award system should be featured* – having an award system would give the student some idea of his/her learning progress for loops and Arrays.

3.2. Vizard Framework

Vizard is a virtual reality toolkit is a Python-based integrated development environment (IDE) used to develop virtual reality applications. Three-dimensional models can be built in 3-D MAX and then imported into the Vizard environment using its built-in exporter. Models and images imported into Vizard can be positioned into the environment and scaled to fit. Through its libraries, Vizard provides built-in functions that govern interactions between objects and their environments. One can also add shapes, text, buttons, and sliders through those functions. Vizard consists of a Python script editor and a debugger. After a script has been created, it can be run with or without debugging. It can also be published as an executable (.exe) file for use by the general public.

3.3. Loop VRI Module Concept

The Duck Game covers seven different types of loops. They are listed as follows:
1. *If/Else* – In this type of loop, if a condition is true then an action is carried out. If a condition is false, a different action is carried out.
2. *Nested If/Else* – This is type of loop is executes the same way as an if/else loop executes; the only difference is that this loop is an if/else loop located within another type of loop.
3. *Nested If/Else (in order)* – With this type of loop, a certain order of conditions must be true before the inner-most action can be executed. The loop breaks when either one of the necessary conditions is false.
4. *If/Else/If* – In this type of loop, if a condition is true, then an action is carried out, as in the if/else loop. If another condition is true, then a different action is executed.
5. *For* – This type of loop repeats until a certain number of iterations has been reached.
6. *Switch Statements* – In this type of loop, if a variable being evaluated meets a certain condition, then an action is executed for that condition. Otherwise, a default action is executed.
7. *While* – This loop repeats as long as a certain condition is met. The loop breaks when the condition is no longer met.

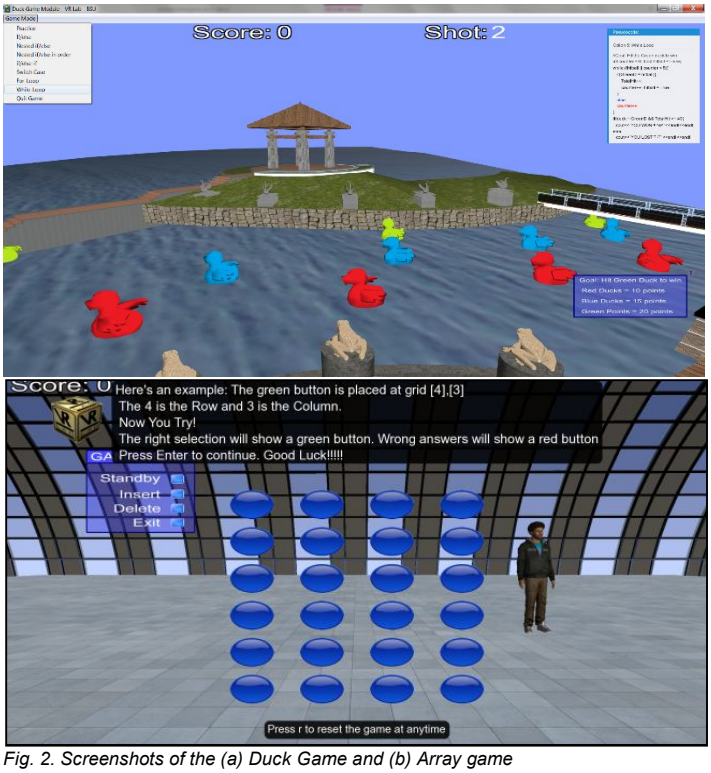

Fig. 2. Screenshots of the (a) Duck Game and (b) Array game

3.4. Array VRI Module Concepts

The Array game covers indexing an array and deleting from an array. The two concepts are detailed as follows:
1. *Indexing an array* – 2-D arrays are referenced using the format [*x*][*y*], where *x* is the row and *y* is the column.
2. *Deleting from an array* – an element is removed from an array by first specifying a location in the array using the format [*x*][*y*]; then the element in that location is removed by being replaced with a "null" value.

4. Virtual Reality Instructional (VRI) Module Implementation

This section details the implementation of the two modules and the user interaction featured within them.

4.1. Game Engine

The Duck and Array games were implemented using the Python programming language within the Vizard Virtual Reality Toolkit, a Python-based integrated development environment (IDE) used to develop virtual reality applications. A screenshot of the Duck Game is shown in Fig. 2(a) and a screenshot of the Array game is shown in Fig. 2(b). A deployment diagram for the two games is given in Fig. 3. On the developer's side, DeveloperPC in the diagram, the Vizard IDE provided the libraries needed to develop the source code needed for the two games, represented as arrayGame.py and duckGame.py in Figure 3. After development, the Python files were each published as executable files, arrayGame.exe and duckGame.exe. Once the executable files were made, they could be used on a standalone computer, represented as UserPC in Figure 2, without having Vizard installed on it beforehand.

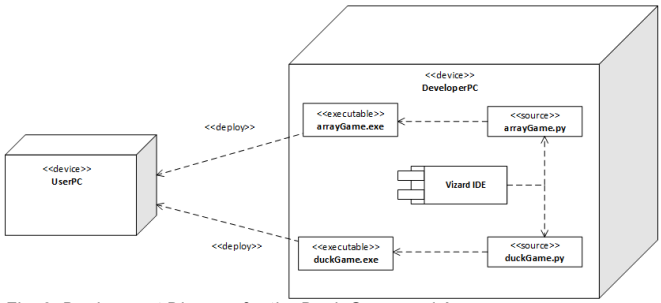

Fig. 3. Deployment Diagram for the Duck Game and Array game

4.1.1 The Duck Game

In the Duck Game, the user is greeted with a prompt instructing them how to navigate through the module. The game starts in Practice mode where the user gets to take practice shots to get a feel for the game. The game offers seven different options (shown in Figure 3) with each option representing a different kind of loop (if/else, nested if/else, while, for, etc.). The ducks featured in the module swim in rows. Red ducks are in the closest row to the user's perspective. Green ducks are located in the farthest row from the user's perspective. Blue ducks are located between the red and green ducks. The Duck Game offers seven choices corresponding to the types of loops listed in Section III. Each choice consists of a goal and the user has five chances to reach it. In each choice, if the user hits a red duck, 10 points are awarded. If the user hits a blue duck, 15 points are awarded. If the user hits a green duck, 20 points are awarded. The choices are explained as follows:

1. *If/Else* – the user must hit a duck, regardless of color, to end the game.
2. *Nested If/Else* – the user must hit a duck of every color (red, blue, and green) at least once.
3. *Nested If/Else (in order)* – the user must hit a blue duck, green duck, and red duck, in that order.
4. *If/Else/If* – in this option, the user realizes his or her "hunt level". Points accumulate using the same scoring system as the other options. If the user earns 90 points or more, the user is deemed an "Expert duck hunter"; if the user earns 70-89 points, the user is an "Experienced duck hunter"; if the user earns 50-69 points, the user is a "Novice duck hunter"; otherwise, the user "needs practice".
5. *Switch* – this option also determines the user's "hunt level" using same award system as the if/else/if option. Here, the points are determined using a switch statement rather than an if-else-if loop.
6. *For Loop* – the user must score at least 60 points to win. After the user's five chances are up, a prompt telling the user whether or not he or she won is shown.
7. *While Loop* – in this option, the user must hit the green duck and earn 40 points to win.

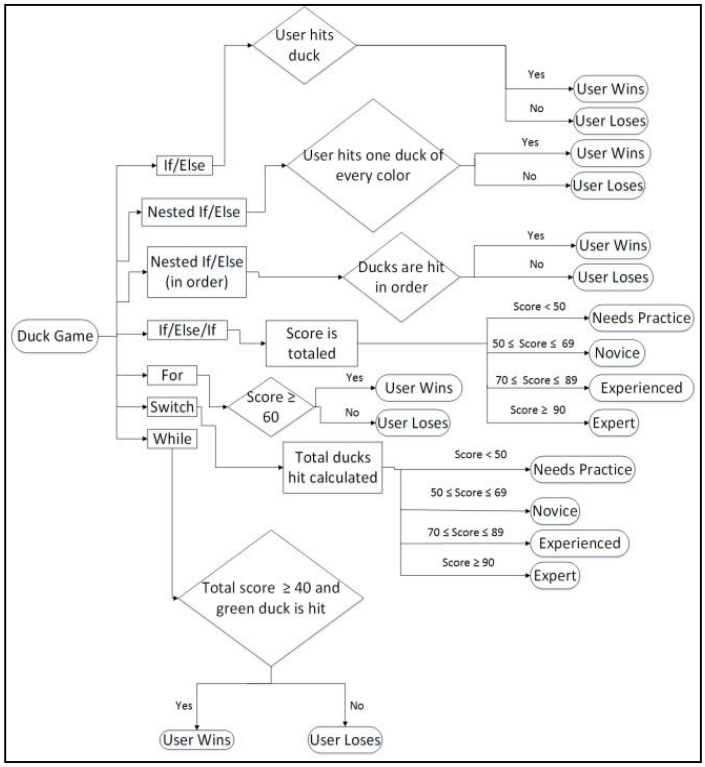

Fig. 4. Flowchart representing the different choices offered in the Duck Game along with the goals for each choice and possible outcomes for user's actions

4.1.2 The Array game

In the Array game, the user is greeted with an avatar giving them an overview of the game and instructions on how to navigate through each game within the module. They are also presented with a menu of gaming options, two of which are illustrated in Fig. 5 and explained as follows:

1. *Insert* – the user is given a specific location in the matrix of buttons to click on. If the user clicks on the correct button, the button in that location is displayed in green; otherwise, it is displayed in red.
2. *Delete* – as in the Insert option, the user is given a specific location in the matrix to click. If the user clicks on the correct location, the button in that location is removed; otherwise, it is displayed in red.

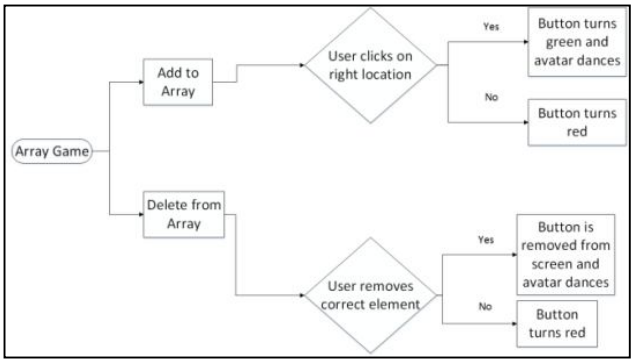

Fig. 5. Flowchart representing the different choices offered in the Array game

4.2. Experimental Design Setup

The overall architecture of both VRI modules is shown in Fig. 6. The architecture for the two modules is based on the MVC (Model, View, Controller) pattern. The *model* refers to the data, the *view* refers to how the data is presented to the user, and the *controller* refers to how the user is able to manipulate and view the data. The data is realized as the Python program running each module. The view is represented by the 3-D models placed into each program, the renderer generating each scene, and the buttons through which the user can interact with each module. The controller in this setup is the mouse clicks and the keyboard commands made by the user. Models can be added into each module through the Python scripts. Both scripts specify the keyboard and mouse commands. Through those commands, the user can manipulate data in the scripts, namely the score. Some of the user interactions trigger animation (avatar talking or ducks quacking). The buttons placed on the screen allows the user to interact with the scenes.

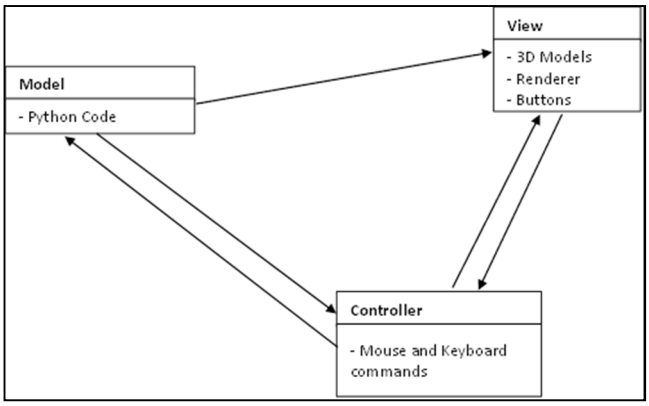

Fig. 6. Software architecture for both modules

5 Results and Analysis

We developed online surveys to collect student data in computer Science I and II in which the modules were being implemented for the first time. A total of 36 students completed surveys across three courses. Table 1 shows the classes and terms of survey administration.

Table 1. Student surveys administration: spring and fall 2013 computer science courses

Surveys	Courses	Term	No. Students
Virtual Reality Instructional Modules (Loops and Arrays)	COSC 113 (Computer Science II) COSC 503 (Software Design & Develop)	Fall 2013	12 10
	COSC 112 (Computer Science I)	Fall 2014	14
Total N		**36**	

5.1. Gamed-Themed Instructional Modules: Duck and Array

Students were asked to report on the extent to which the Duck and Array instructional modules enhanced their learning. Generally, the majority of students felt the two modules positively impacted their learning. As shown in Fig. 7, students responded more positively to the Duck module than the Array. Eighty-six percent of students reported a good to excellent level of impact associated with the Duck instructional module compared to 79% for the Array module. In terms of extent of impact, most students reported a satisfactory level (i.e., "good) at 62% and 58% for the duck and Array instructional modules respectively. The interpretation of the Likert scale used on these surveys did not include the rating of "Very Good", typically included in a Poor to Excellent continuum. This analysis therefore defines the rating scale in relationship to impact as follows: Excellent=Significant, [Very Good=Substantial], Good=Satisfactory, Fair=Moderate, Poor=Minimal.

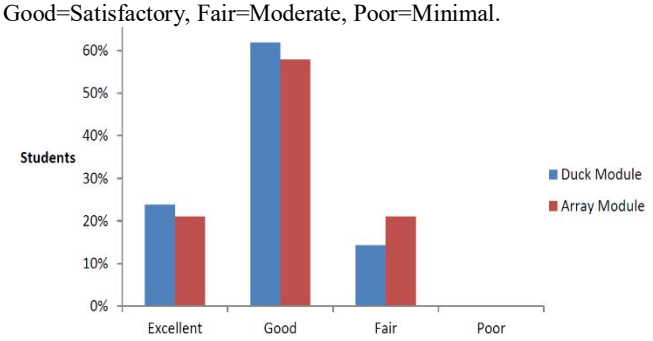

Fig. 7: Duck and Array Instructional Modules Impact on Enhanced Student Learning

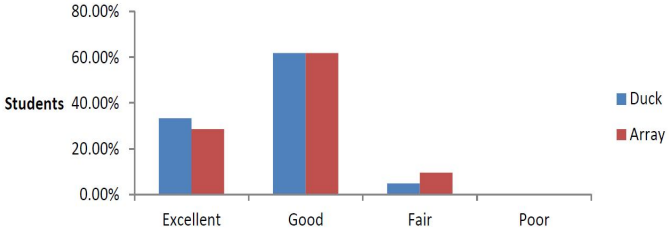

Fig. 8: Duck and Array Instructional Modules Impact on Understanding Concepts

Less than a third of students reported a significant impact of the instructional modules on their learning with 24% and 21% reporting a level of excellent for the Duck and Array modules respectively. While students were positive about the modules' impact on learning, most students reported satisfactory effects (i.e., good). The contribution of the instructional modules on students' understanding of key concepts was more pronounced than the impact associated with enhancement of students' overall learning. As shown in Fig. 8, approximately a third of students felt the instructional modules impacted their understanding of concepts at a significant level. Approximately 60% reported a satisfactory level of impact on conceptual understanding. Based on the perceptions of students, the modules were more beneficial in contributing to their understanding of specific content topics than enhancement of learning more generally in the course.

Fig. 9 and Fig. 10 show the specific content areas associated with students' perceptions of positive impact of the instructional modules on their conceptual understanding. The majority of students were positive about the impact of the Duck and Array instructional modules in improving their conceptual understanding in key content areas. In relationship to the Duck instructional modules, the majority of students reported significant impact (i.e., excellent) in understanding for the following topics: "For Loop" (57%), "If/Else", "Multiple If/Else-if", and "While Loop", (52% respectively). Most students felt the Duck module was satisfactory for learning two content areas – "Nested If/Else" and "Nested If/Else in Order". Students were equally divided between a significant or satisfactory impact of the module in learning "Switch Case".

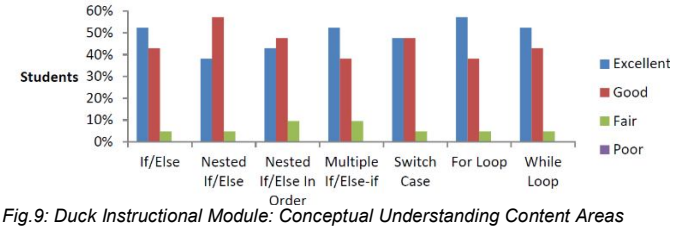

Fig.9: Duck Instructional Module: Conceptual Understanding Content Areas

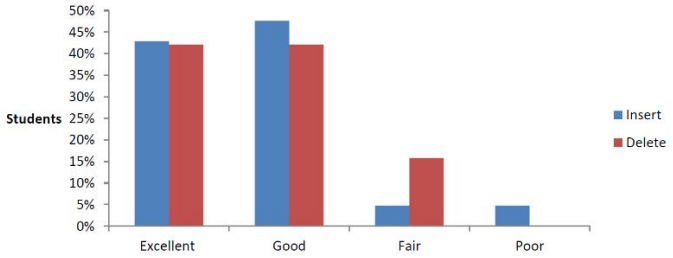

Fig. 10. Array Instructional Module - Conceptual Understanding Content Areas

In response to the Array instructional module, more students reported a positive impact on conceptual understanding for the "Insert" topic area (91%) than the "Delete" concept (85%). As shown in Fig. 10, a substantial percentage of students reported that the Array instructional module had a significant impact on their conceptual understanding, 43% and 42% for understanding "Insert" and "Delete" respectively. A small proportion of students reported moderate to minimal impact of the Array instructional module in contributing to their conceptual understanding. Particularly in relationship to understanding "Delete" almost a fifth of students reported a moderate level of impact.

5.2. Course Differences

Between course results revealed differences in student responses by course on the impact of the instructional modules to conceptual understanding. For the Duck module, more students in the COSC503 – Software Design and Development course expressed significant impact in their conceptual understanding. Student expressed significant impact for understanding If/Else (67%), and Loop and While Loop (56%). In contrast, we did not find a majority expressing significant impact of the instructional module on understanding for students in COSC 113 – Computer Science II. However, half of the Computer Science II students did report significant impact for the content areas associated with the Duck module with the exception of understanding If/Else (42%).

Similarly, more students in COSC503 – Software Design and Development were positive toward the impact of the Array module on conceptual understanding students in COSC113 – Computer Science II. Especially in relationship to understanding the concept of Insert, 56% of COSC503 students felt the Array module had significant impact compared to only a third of students in COSC113. The between course result was similar for the Delete with 44% and 40% reporting significant impact in COSC503 and COSC113, respectively. The between course differences may be related to the nature of the courses, prior coursework, student factors, or curricular enactment. Additional data collection activities would be needed to investigate these other factors.

5.2. Gaming Resources and Student Learning

Gaming resources, including the use of videos, web links, and YouTube, among others, were also used in both courses to support student learning. All students in both classes responded positively about the use of gaming resources to advance learning. Particularly in Computer Science II, slightly more than a third of the students reported a significant and positive impact on their learning. All students in both classes felt the gaming resources had an at least a satisfactory level of impact. About a third of students felt the impact was only moderate. While the impact of the gaming resources are similar to the use of the gaming assignments, student responses reflect slightly better results. No students viewed the gaming resources as having minimal impact, which was not the case for the gaming resources. In relationship to significant to moderate levels of impact, more students responded positively to the gaming resources than to the assignments.

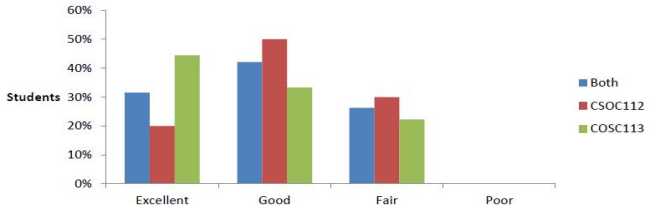

Fig. 11: Gaming resources impact on student learning

The student survey baseline results demonstrate positive student perceptions about the use of gaming assignments and instructional modules to advance student learning and understanding in Computer Science I and II. These baseline positive survey results provide evidence that the materials have some potential for improving the teaching and learning of computer science introductory courses. The continuation of the materials development process along with the inclusion of triangulation of data points in subsequent analyses will be needed to increase the likelihood of realizing this potential.

6 Conclusions

We designed and implemented two modules teaching arrays and loops aiming to provide an alternative way of teaching the two data structures to students in introductory Computer Science courses. To accomplish this we developed the two VRI modules

within the Vizard development platform using Python. We also included seven different types of loops for the Duck Game and two functions for the Array game. The student survey baseline results demonstrate positive student perceptions about the use of gaming instructional modules to advance student learning and understanding in Computer Science I and II. These baseline positive survey results provide evidence that the materials have some potential for improving the teaching and learning of computer science introductory courses.

Acknowledgments

This work is funded by the National Science Foundation under grant number HRD-1238784. The author would like to acknowledge the contribution of Monica B. Mitchell, E.D., MERAssociates, LLC for conducting the evaluation for the instructional modules as part of the NSF grant.

References

[1] S. Shabanah, J. X. Chen, H. Wechsler, D. Carr, and E. Wegman "Designing computer games to teach algorithms", pp.1119-1126, (2010) .

[2] P. D. Eades and K. Zhang, Eds., Software Visualization, World Scientific, vol 7, (1996).

[3] R. Baecker and D. Sherman, "Sorting out sorting," 30 minute color sound film, Dynamic Graphics Project, University of Toronto, 1981, excerpted and reprinted in SIGGRAPH Video Review 7, (1983).

[4] M. Wolf, "The Medium of the video game", "1st" ed. University of Texas Press, (2002).

[5] C. Hundhausen, S. Douglas, and J. Stasko, "A metastudy of algorithm visualization effectiveness," Visual Languages and Computing, vol. 13, no. 3, pp. 259–290, (2002).

[6] S. Christina and K. Dimitrios, "Applying constructivism for interactive educational software: a research based design, implementation and evaluation method" in *Proc. IEEE International Conference on Advanced Learning Technologies (ICALT '04)*, (2004).

[7] S. Noordin and W. F. W. Ahmad, "Using game as part of the knowledge transfer module in a multimedia courseware", vol 16, (2011).

[8] R. Ploetzner and S. Schlag, "Strategic learning from expository animations: Short- and mid-term effects" Computer & Education, Vol.69, pp.159-168 (2013).

[9] R. K. Lowe, "Animation and learning: selective processing of information in dynamic graphics", Learning and Instruction, Vol.13, pp.157–176 (2003).

[10] J. D. Bayliss, "Using games in introductory courses: tips from the trenches", Proceedings of the 40th ACM technical symposium on Computer science education (SIGCSE'09), March 04-07, Chattanooga, TN, (2009).

[11] S. Leutenegger, J. Edgington, "A games first approach to teaching introductory programming", Proceedings of the 38th SIGCSE technical symposium on Computer Science Education, Mar. 7-10, Covington, KY, (2007).

[12] M. Eagle, T. Barnes, "Wu's Castle: Teaching Arrays and loops in a game", Proceedings of ITICSE '08, June 30-July 2, Madrid, Spain, (2008).

[13] K. Sung, R. Rosenberg, M. Panitz, R. Anderson, "Assessing game-themed programming assignments for CS1/2 courses", Proceedings of the Third International Conference on Game Development in Computer Science Education (GDCSE'08), Miami, FL, Feb (2008).

[14] J. Zhang, M. Atay, E. Caldwell, E. Jones, "Visualizing loops using a game-like instructional module", Proceedings of the 13th International Conference on Advanced Learning Technologies (ICALT), Beijing, China, July, (2013).

[15] Burger, J. Personality (6th ed.). Belmont, CA: Wadsworth. (2004).

[16] F. Christy, "Constructivist approach to teaching and learning", April 8, (2012).

[17] Sjoberg, Svein. Baker, E. McGraw, B. Peterson, P., "Constructivism and Learning", International Encyclopedia of Education 3rd Edition. University of Oslo, Norway. Oxford. (2007).

[18] Cooperstein, Susan E. Kocevar-Weidinger, Elizabeth, "Beyond active learning: a constructivist approach to learning", Reference Services Review. Vol. 32. (2004).

[19] Dalgarno, Barney, "The potential of 3D virtual learning environments: A Constructivist Analysis. E-Journal of Instructional Science and Technology", Center for Research in Complex Systems. University of Southern Queensland. Australia, (2002).

[20] S. Sharma, R. Agada, J. Ruffin, "Virtual reality classrooms as a constructivist approach", Proceedings of IEEE Southeastcon. Jacksonville, FL, pp. 1-5, (2013).

[21] S. Sharma, J. Stigall, S. Rajeev, "Game–Theme based instructional module for teaching object oriented programming", proceedings of the IEEE International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, USA, pp 252-257, DOI 10.1109/CSCI.2015.3, December 7-9, (2015).

[22] M. Filipović Tretinjak, A. Bednjanec and M. Tretinjak, "Application of modern teaching Techniques in education process" 37th International Convention on Information and Communication Technology, p.p.628-632, May, (2014).

[23] P. Doerschuk, V. Juarez, J. Liu, D. Vincent, K. Doss, Judith Mann, "Introducing programming concepts through video game creation" 2013 frontiers in education conference, p.p. 523-529, (2013).

[24] T. Singh, M. Zhu, U. Thakkar, and U. Ravaioli, "Impact of World Wide Web, Java, and virtual environments on education in computational science and engineering" 1996 frontiers in education conference, p.p.1007-1013, Vol 3,(1996).

[25] Carol C. W. Hulls, Member, IEEE, Adam J. Neale,Benyamin N. Komalo, Val Petrov, and David J. Brush, " Interactive online tutorial assistance for a first programming course", p.p.719-728,vol 48, (2005).

[26] D.U Ekong, "An Instructional module for teaching ethics with in an ECE curriculum", p.p.1-4, April, (2015).

[27] A. Yohannis and Y. Prabowo, "Sort Attack: Visualization and gamification of sorting algorithm learning," Games and Virtual Worlds for Serious Applications (VS-Games), 7th International Conference on, Skovde, 2015, pp. 1-8. (2015).

[28] M. D. Byrne, R. Catrambone and J. T. Statsko, "Do algorithm animation aid learning?" Tech. Rep. GIT-GVU-96-18, (1996).

[29] Clark. J,"Modular Educational game system a customizable framework for learning", 16th international conference, p.p.248-258, (2011).

[30] Cetin.A," A 3D game based learning application in engineering education: powering a recreational boat with renewable energy sources", 15th International Conference on Interactive Collaborative Learning (ICL),pp.1-4, (2012).

Author Biography

Dr. Sharad Sharma is an Associate Professor in Department of Computer Science, Bowie State University, Bowie, MD 20715 USA. He has received Ph.D. in Computer Engineering from Wayne State University, Detroit, MI, USA in 2006 and M.S. from University of Michigan, Ann Arbor, MI, USA in 2003. Dr. Sharma is the Director of the Virtual Reality Laboratory at the Bowie State University. His research focuses on using virtual reality and augmented reality as tool for learning, training, and education. He specializes in modeling and simulation of multi-agent systems and multi-user virtual reality environments for emergency response and decision making strategies.

Mr. Emmanuel Ossuetta is an undergraduate student in Computer Technology at Department of Computer Science, Bowie State University, Bowie, MD 20715 USA. He works as a research assistant in Virtual Reality Laboratory at the Bowie State University. His research interest include virtual reality, augmented reality, and collaborative virtual environment.