# Using Video Tutorials as a Carrot-and-Stick Approach to Learning

Jason Wells, Robert Mathie Barry, and Aaron Spence

***Abstract*—Traditional teaching styles practiced at universities do not generally suit all students' learning styles. For a variety of reasons, students do not always engage in learning in the courses in which they are enrolled. New methods to create and deliver educational material are available, but these do not always improve learning outcomes. Acknowledging these truths and developing and delivering educational material that provides diverse ways for students to learn is a constant challenge. This study examines the use of video tutorials within a university environment in an attempt to provide a teaching model that is valuable to all students, and in particular to those students who are not engaging in learning. The results of a three-year study have demonstrated that the use of well-designed, assessment-focused, and readily available video tutorials have the potential to improve student satisfaction and grades by enabling and encouraging students to learn how they want, when they want, and at a pace that suits their needs.**

***Index Terms*—Curriculum development, education, learning, programming, tutorials, videos.**

## I. Introduction

HIGH failure rates are of a particular concern to universities [1] both for the implications to students and the reflection on the institutions. While new ways of teaching, such as the introduction of multimedia content into a course, can result in improvement to student performance, sometimes this increase is only seen in the average to above average students, while the number of students who fail remains constant [1]–[3]. This study examines the design and use of multimedia resources within a first-year programming unit in an attempt to address high failure rates.

## II. Difficulties in Teaching Programming Courses

Programming courses at universities have experienced poor outcomes over the last decade [1], [4], [5]. There is no single reason for this; the university, the teaching, and the student each have their role to play in ensuring positive outcomes. Although it is not unusual to encounter poor outcomes in university-level courses, it is clear that teaching programming presents many challenges.

Manuscript received November 03, 2011; accepted January 09, 2012. Date of publication February 29, 2012; date of current version October 26, 2012.

J. Wells is with the School of Information Technology, Deakin University, Waurn Ponds, Vic. 3216, Australia (e-mail: wells@deakin.edu.au).

R. M. Barry and A. Spence are with the School of Information Technology, Deakin University, Geelong, Vic. 3220, Australia.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TE.2012.2187451

Students do not fail on purpose [6]. A study by Sheard *et al.* [1] surveyed 84 students to discover why computer science students were failing. Initial factors identified were that the failing students were working many hours in outside employment, had a more sporadic lecture and tutorial attendance rate, and had a generally low motivation to learn and to seek out any extra resources on their own initiative.

Programming is a difficult skill to learn; experienced programmers draw upon many diverse skills to produce computer code [6]. Jenkins [6] highlights some difficulties encountered by the students. Among these is the complexity of the subject-matter content, which requires students to acquire many skill sets, such as problem solving and basic mathematics. In addition, the teaching pace of the course material is also outside the control of the students, which can pose a problem to any student falling behind due to slow comprehension, confusion over course content, or simply missing a class. Difficulty may result from an instructor who, although skilled in programming, may be poor at teaching or at engaging the students. The opposite case may occur of students becoming bored if they perceive their teacher as being unskilled in programming [7].

In addition, there is the possibility that the content of a programming course may not be appropriate to students' skill levels [5], which can vary substantially due to factors such as personal aptitude, previous experience gained from employment, or participation in computing units during high school [7]. Having previous programming knowledge and experience has shown to be of marked benefit to students engaging in Computer Science degrees [8].

Research has suggested students do not undertake programming units before their second year of studies, as they settle into their degree programs after the transitions of their first year at university [6]. Before changing elements of course structure or content delivery, adequate consideration must be given to student-specific issues, such as the learning environment and assessment of student proficiency in technology [9].

Students in all subjects are becoming significantly more computer literate [9]–[11]. Among the current generation are the "Digital Natives" [11], [12], consisting of the approximately 85% of students who use online resources for their education [10] as well as for socializing [9], e-mail, and general information gathering [11]. Computer technology is a part of the students' lifestyle, creating a preference for multitasking, fast information presentation, and "nonlinear" access to learning material [11], [13].

The key issue that universities can control is the way courses are taught, and in particular understanding the various ways students learn and making efforts in their teaching to accommodate

0018-9359/$31.00 © 2012 IEEE

Authorized licensed use limited to: University of Southern Denmark. Downloaded on August 19,2022 at 07:13:55 UTC from IEEE Xplore. Restrictions apply.

these differences. Understanding these two parts of the equation is the first step to finding potential avenues to improve the grades of the failing students.

The essential for reaching a level of competence in programming is practice, practice, practice [14]. Most students accept this as necessary, but there is an increasing trend for some to avoid this type of applied learning, and as a consequence, a higher failure rate is becoming common. To address these issues, a new method of providing tuition has been developed to increase student engagement in learning.

## III. Study Background

Within the School of Engineering at Deakin University, Australia, there is a requirement to teach undergraduate students computer programming as part of their first-year engineering studies. Many engineering students either find computer programming difficult to learn or simply do not feel they should have to know programming concepts as part of an engineering degree; consequently, they approach the subject with a negative attitude. As a consequence, the failure rate for the subject has been high, ranging from 30% to 38%, and the student perception of the unit has been poor. In an attempt to address the failure rate and these negative perceptions, new teaching techniques have been developed and studied, resulting in a dramatic turnaround.

The School of Engineering uses traditional teaching methodologies consisting of lectures where students are presented with slides, examples, and demonstrations and have an opportunity to ask questions. Students are expected to have read any prescribed readings before attending the lecture and, as a consequence, have some knowledge of the subject being presented by the lecturer. In addition to lectures, there are practical sessions where subsets of students attempt and practice programming problems on computers and have the opportunity to ask questions and review solutions. Assessment tasks are provided in which students solve programming problems, and an exam is given at the end of the semester to determine the level of programming knowledge the students have obtained. The methodology assumes that the lectures present the concepts, the practicals encourage their application, and the assessment examines how much the students have learned. In all cases, the students are expected to attend their allocated lectures and practical classes and attempt all assessment tasks.

With the introduction of the Internet and e-Learning portals, students are provided with many resources that are accessible 24 h a day via the Internet. These include lecture slides, video recordings of the lectures, practical descriptions, practical solutions, exercises and solutions, online quizzes, forums, notices, practice exams, and links to external resources. Delivering content via e-Learning portals enhances the learning experience by providing more flexible ways for students to access and complete the unit requirements. Students missing a lecture or practical can download, read, watch, and complete the weekly tasks; ask questions via the forums rather than waiting for the lecture or allocated practical class; discuss assessment tasks online; and potentially work where they want, when they want, and how they want. One would expect the natural extension of this new method of delivering and conducting courses in addition to the traditional methods to be an increase in the quality of learning taking place and consequently higher results and lower failure rates. Sadly, this has not been the case, and in some instances the failure rate is actually increasing.

A three-year study has been conducted to investigate the use of screen capture technology to deliver learning resources for the SIT172 Programming for Engineers unit within Deakin University. As part of their Engineering degree, all students must complete the SIT172 programming unit regardless of their chosen discipline. The unit focuses on the fundamentals of programming and currently teaches the application of C programming as the primary language. The unit is conducted over 12 weeks of lectures, plus one week of revision and a two-week exam period during which students sit a 3-h closed-book exam to be completed without the use of a computer. The unit consists of both on- and off-campus students, with approximately 20% of students being off-campus students. On-campus students have 3 h of lectures and a 3-h practical class weekly. Off-campus students do not attend face-to-face classes, and they work independently. All unit material is provided online via an e-Learning portal.

Assessment consists of four assignments, each worth 10%, and an end-of-year exam worth 60%. To pass the unit, students must achieve an overall mark of 50% or greater. Assignments were due in weeks 3, 6, 9, and 11. All assignments were based on a single problem that was broken into four stages, with each stage building on the previous one, culminating in a complete solution in week 12. Each stage examined the unit content from previous weeks; all concepts had been covered both in lectures and practicals. A solution was provided after each assignment was submitted and was used as the basis for the next assignment as well as to provide feedback as to how to approach the problem. A marking guide was provided that clearly outlined what was expected and how many marks were allocated to each aspect of the assignment tasks.

## IV. Multimedia Content

Programming uses a computer and software tools to prepare, compile, execute, and debug a program. Since the introduction of screen recording software such as Adobe Captivate and Camtasia Studio, it is now possible to record screen images, mouse movements, keystrokes, and menu selections, as well as to provide audio of the person using the software. This recording can be stored and delivered via the Internet in a number of different formats and can be streamed via a Web browser or downloaded and viewed on most standard computers. The recording can be replayed, stopped, rewound, and advanced as many times as the student wishes.

This capability enables the creation of tutorials that demonstrate the use of the programming environment, the use and application of the programming language based around specific concepts, feedback to students on how to approach a problem, general revision, and many more possible uses. In addition, the instructor can record a narration to accompany the video that provides the opportunity to explain general concepts, tips, tricks, suggestions, and ideas. The result is a rich, versatile resource that is proving to be the most popular resource with

Authorized licensed use limited to: University of Southern Denmark. Downloaded on August 19,2022 at 07:13:55 UTC from IEEE Xplore. Restrictions apply.

students, especially with those who choose not to engage in the traditional teaching methods provided.

Three case studies focusing on the use of video tutorials in a university setting were examined to establish the foundations of the study. Carver *et al.* [3] highlighted the need to clearly define the association between the resource and the topic being studied. Smith [15] and Nicholson and Nicholson [16] confirmed the use of the resource was both effective and well received by those students who used the resources. Nicholson and Nicholson [16] also highlighted that the creation and maintenance of the resources requires extra costs and training, but the overall benefits were said to far outweigh these few shortcomings.

In 2008, 12 video tutorials were created. Each video focused on the weekly unit content and consisted of the following skill sets.

- Presentation and explanation of the programming problem: The programming problem was described, and the steps required to complete the solution outlined. It is important that the student clearly understands the problem and each step that must be coded to solve the problem.
- Step-by-step coding of the solution: The code was developed step by step. At each step, the code was compiled and examined to ensure it was correct. Tips, tricks, and suggestions were outlined to ensure the students avoided frustrating coding errors that have the potential to bog a student down for hours. The order of the code development demonstrated the best approach, layout, style, and presentation expected.
- Debugging and program testing: At regular intervals, demonstrations were provided on how to best test the code and on the use of the debugger software to examine in more detail the data flow through the code.

No special attention was applied to allocating specific time to each skill set. Each video presented a problem that reflected the weekly topic being studied, but the content was not directly associated with the assignment requirements. Videos were generally 10–15 min in duration and, where required, broken into parts to ensure individual concepts were not mixed. This is particularly important as some students may want to focus on a single concept and can target a particular concept/video as required.

Students were instructed to watch the video, to discover the key concepts that they were required to apply, and then complete the coding requirements for the assessment tasks.

Initial feedback was very positive, especially from the off-campus students. Off-campus students have very little direct interaction with the lecturer, other than via the iLecture recordings; in general, they were required to learn and solve problems independently, which is not a simple task when learning programming, as many obstacles must be overcome to code a working solution. Many of the on-campus students also expressed very positive feedback associated with the videos, which soon became the main learning resource for the unit.

From a lecturing point of view, the videos provided many benefits. A single video has the potential to answer many questions, and with good design and execution, the need to explain a concept many times to many students diminished. The videos allow efficient transmission of the sort of experience that helps avoid many programming mistakes and constantly reinforces examples of best practice, thus reducing the amount of "stuck" periods that students experience [17]. The videos were simple to create, and many were produced to address specific problems as they arose. This enabled the lecturer to target specific issues quickly and efficiently for all students as required.

| Year | Campus | Mean | Responses |
|---|---|---|---|
| 2006 | On | 3.81 | 54 |
| 2006 | Off | 3.69 | 16 |
| 2007 | On | 3.12 | 59 |
| 2007 | Off | 2.72 | 18 |
| 2008 | On | 4.20 | 59 |
| 2008 | Off | 4.25 | 12 |
| 2009 | On | 4.48 | 52 |
| 2009 | Off | 4.42 | 12 |
| 2010 | On | 4.57 | 47 |
| 2010 | Off | 4.30 | 10 |

Fig. 1. SETU results for both on and off campus. Mean results are out of 5.

| Grade | Mark out of 100 |
|---|---|
| **N** | 0 – 49 (Fail) |
| **P** | 50 – 59 (General pass) |
| **C** | 60 – 69 (Credit pass) |
| **D** | 70 – 79 (Distinction pass) |
| **HD** | 80 – 100 (High Distinction pass) |

Fig. 2. Results table.

## V. Initial Outcomes

In 2008, student results only improved slightly, but the student satisfaction rating improved dramatically, especially from off-campus students. Student Evaluation of Teaching and Units (SETU) are conducted by the university at the end of each semester. Students complete the voluntary survey anonymously, and the results are considered by the university to be an important guide to the level of service and quality of teaching and are regularly scrutinized by management. Prior to 2008, the unit consistently scored poorly in relation to student satisfaction.

Six statements are made, to which students provide a rating from 0 to 5. Students are also invited to provide written comments associated with the unit. Fig. 1 summarizes the results from 2006 to 2010 for both on- and off-campus modes for the following statement: *"I was satisfied with the quality of teaching from this teacher in this unit."*

The introduction of the video tutorials in 2008 was the only change to the unit. The SETU results for 2008 indicate they played an important role in the improvement in the overall ratings and perceptions of the unit content, teaching approach, and service being provided, but the failure rate remained relatively unchanged. The improvement in student satisfaction was reflecting a positive perception for the videos, especially from off-campus students who rarely received an interactive teaching experience. This may also have been because the use of the technology was new and not being used in other subjects they were studying. An examination of assessment submissions found 40% of students were still not engaging in the unit content and were not using the videos as expected despite their value and accessibility.

Authorized licensed use limited to: University of Southern Denmark. Downloaded on August 19,2022 at 07:13:55 UTC from IEEE Xplore. Restrictions apply.

| On/Off | 2006 | 2007 | 2008 | 2009 | 2010 |
|---|---|---|---|---|---|
| N % | 30 | 30 | 26 | 13 | 13 |
| P % | 18 | 24 | 21 | 16 | 14 |
| C % | 20 | 18 | 22 | 17 | 23 |
| D % | 19 | 20 | 19 | 29 | 29 |
| HD % | 14 | 8 | 12 | 25 | 21 |

| ON | 2006 | 2007 | 2008 | 2009 | 2010 |
|---|---|---|---|---|---|
| N % | 28 | 28 | 27 | 11 | 14 |
| P % | 19 | 23 | 22 | 17 | 15 |
| C % | 20 | 19 | 20 | 18 | 23 |
| D % | 17 | 20 | 19 | 27 | 30 |
| HD % | 17 | 9 | 12 | 26 | 18 |

| OFF | 2006 | 2007 | 2008 | 2009 | 2010 |
|---|---|---|---|---|---|
| N % | 38 | 37 | 24 | 20 | 6 |
| P % | 14 | 26 | 16 | 10 | 6 |
| C % | 19 | 15 | 28 | 15 | 25 |
| D % | 29 | 19 | 20 | 35 | 25 |
| HD % | 0 | 4 | 12 | 20 | 38 |

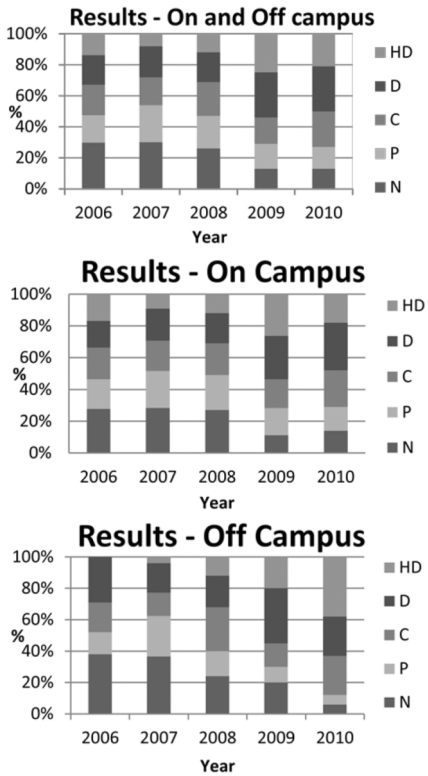


Fig. 3. Final results.

## VI. Refinements

In 2009, the design of the video tutorials was adjusted to align them more closely with the four assignments. The grade achieved in the assignments is the only measure of the student's depth of understanding of the subject and, as a consequence, is the primary focus for all students throughout the semester. This refinement of the video was thus designed to be attractive to the 40% of students who were either not completing all the assignments or were completing the assignments poorly.

Guessing a solution to a programming problem is unlikely to be successful. To code a solution, it is necessary first to understand the problem and then understand how to code the solution using the required programming language. To learn this process, students must practice the concepts, which can be demanding of time and effort. Not all engineering students can muster the motivation to learn programming and, as a consequence, either lose confidence and drop out, or simply are unable to use the resources efficiently within the time available to learn the unit material as required. As a result, a new strategy was developed to streamline the learning process and to focus all the videos on how to complete the assignments.

Each video was based around a problem similar to the assignment problem the students were expected to complete. Relating the video content more closely to the assignment problem implies that by watching the video, students will get more help in completing the assignment. This strategy was used specifically to appeal to those students who were not engaging in learning. By attempting and completing the assignments, students engage in learning, earn marks, and consequently pass the unit.

| Year | Students | Responses | On% | Off% |
|---|---|---|---|---|
| 2009 | 136 | 56 | 96 | 4 |
| 2010 | 121 | 23 | 65 | 35 |

Fig. 4. Survey response rate.

Students were instructed to watch the videos and then attempt the assignment. Each video takes the students step by step through the concepts required to complete the assignment. Students can pause the video, apply what they have seen to the assignment, test and confirm the code, and then move on to the next concept. The net result would be two to five hours of programming practice and a partly or entirely completed assignment.

The danger in this strategy is that the videos could compromise learning by giving the student too much information about the assignment solution, allowing them to bypass quality learning; as a consequence, students might not have the knowledge required to solve exam questions or other programming problems by using their acquired knowledge alone, without the aid of an accompanying video.

This method resulted in an increase of student submissions for each assignment and an increase in overall grades for the assignments.

The end-of-unit exam component of the assessment consisted of a closed book, 3-h supervised exam worth 60% of the overall assessment marks for the unit. The exam consisted of multiple-choice questions that required students to read, understand, and predict the outcome of blocks of code and short-answer questions requiring students to write code to solve a given problem.

Authorized licensed use limited to: University of Southern Denmark. Downloaded on August 19,2022 at 07:13:55 UTC from IEEE Xplore. Restrictions apply.

| How do you rate the video tutorials provided? | | |
|---|---|---|
| | 2009 % | 2010 % |
| Very unhelpful | 0 | 2 |
| Unhelpful | 0 | 0 |
| Average | 9 | 0 |
| Helpful | 26 | 11 |
| Very helpful | 65 | 88 |

| How do you rate the content of the video tutorials? | | |
|---|---|---|
| | 2009 % | 2010 % |
| Very poor | 0 | 0 |
| Poor | 0 | 0 |
| Average | 13 | 2 |
| Good | 39 | 27 |
| Very good | 48 | 71 |

| How did you use the video tutorials? | | |
|---|---|---|
| | 2009 % | 2010 % |
| Learn C programming | 70 | 54 |
| Learn the weekly material | 48 | 34 |
| How to complete the assignments | 87 | 95 |
| Study for the exam | 57 | 38 |
| I did not watch them | 4 | 0 |

| Did the video tutorials help you learn the unit material? | | |
|---|---|---|
| | 2009 % | 2010 % |
| Not at all | 0 | 0 |
| Sometimes | 13 | 9 |
| Often | 48 | 45 |
| Always | 39 | 46 |

| Did the video tutorials encourage you to complete the assignments? | | |
|---|---|---|
| | 2009 % | 2010 % |
| Never | 0 | 4 |
| Sometimes | 13 | 9 |
| Often | 48 | 29 |
| Always | 39 | 59 |

| How often did you use the video tutorials to help you complete the assignments? | | |
|---|---|---|
| | 2009 % | 2010 % |
| Not at all | 4 | 0 |
| Sometimes | 26 | 14 |
| Often | 30 | 21 |
| Always | 39 | 64 |

| Which of the following resources did you find most helpful? | | |
|---|---|---|
| | 2009 % | 2010 % |
| Lectures | 30 | 27 |
| Lecture slides | 78 | 63 |
| Video tutorials | 65 | 84 |
| Assignments | 52 | 39 |
| iLecture | 30 | 2 |
| Student News | 4 | 5 |

| Which of the following is the most valuable resource provided for this subject? | | |
|---|---|---|
| | 2009 % | 2010 % |
| Lectures | 9 | 2 |
| Lecture slides | 0 | 0 |
| Practicals | 44 | 37.5 |
| Video tutorials | 35 | 59 |
| iLecture | 13 | 0 |

Fig. 5. Survey results.

The exam was important as it provided an unassisted environment that tested students' knowledge of the unit and their ability to write logical, well-structured code.

## VII. Improved Outcomes

The 2009 exam results were excellent, resulting in the some of the best code seen in an exam situation for this subject, and overall final grades improved. Grades are defined in Fig. 2. The fail rate decreased from 30% to 13%, and the overall standard across all grades improved (Fig. 3).

The SETU results for 2009 (Fig. 1) were also very positive and again confirmed the use of the video tutorials as a major contributing factor in the improved results. Similar results were achieved in 2010.

Students studying in off-campus mode appeared to gain the most benefit from the video tutorials as the failure rate dropped from 38% to 6% and grades of 80% and above rose from 0% to 38% in 2010. Off-campus students do not have the luxury of face-to-face contact via lectures and practicals, and had been left to their own devises to learn the unit content. The video tutorials filled this void and were directly responsible for the dramatic turnaround to excellent results and unit perceptions.

## VIII. Validation

To determine in more detail what resources were being used and valued by the students, and to gain a better understanding of student perceptions of the unit, a survey was administered to all students in 2009 and 2010 (Figs. 4 and 5). The surveys were anonymous and conducted in the final week of lectures, during revision week and prior to the exam. The survey was designed to provide a better understanding of how important each resource was and to provide a clear guide as to where effort should be directed in the future to ensure the teaching methodology suited the students and the way they needed to learn.

The results reveal that more than 87% of the students used the tutorials to complete the assignments, and the perception of the video tutorials is very positive. The surveys clearly indicate the introduction of the video tutorials into the unit has provided a valuable resource that is being used primarily by students to complete the assignments and to learn the unit material, which is corroborated by the improvement in the unit results.

An unexpected outcome of introducing the video tutorials to the unit resources was the devaluing of the unit's lectures. The surveys clearly indicate video tutorials are the most valued resource, but face-to-face lectures were rated very low. This, in hindsight, was being reflected in the low attendance rates at lectures and indicates that students prefer to use independent resources to help them complete the unit expectations. Lectures require a student to attend a class at a specified time and place, and their content may not immediately relate to the assessment tasks students are required to complete. The video tutorials provide the specific information needed to complete the assessment and can be watched when, where, and as often as required.

## IX. Conclusion

This study has highlighted that using assessment-focused video tutorials as a teaching tool has many benefits for both the student and the teacher. Integrating the assessment criteria into the videos motivates students to watch them, and consequently the students learn and complete the assessment. This approach

Authorized licensed use limited to: University of Southern Denmark. Downloaded on August 19,2022 at 07:13:55 UTC from IEEE Xplore. Restrictions apply.

is not just beneficial to teaching university-level programming, but can be applied in any discipline at any level of education where students are required to learn a process to achieve a particular product or outcome [18]. Static content struggles to pass on experience, technique, process, and an application of knowledge. This is especially relevant where applying that knowledge requires the use of unfamiliar tools and environments, whose pitfalls have the potential to frustrate and distract the learner.

Traditional teaching styles, where students are expected to attend lectures and practical classes to gather an understanding of the unit material, work for those students who come to class, but they have the capacity to impact on those students who do not. The introduction of the video tutorials has given those students who do not attend classes and consequently struggle to pass the unit an opportunity to engage in the important aspects of learning. Where the assessment is the motivation to learn, video tutorials designed to focus on the concepts required to complete the assessment are valuable, much used, and have very positive outcomes. The video tutorials benefit not only those who are not attending class, but the entire student cohort. This is reflected in a general upward shift of the unit results.

Students must overcome many issues to complete coursework; equally, lecturers face many issues to ensure that students receive the service and resources required to guide them through this process. This study has confirmed the valuable role that video resources can play in addressing many of these issues, especially those that acknowledge the importance of assessment as a learning tool and focus the student on this assessment, encouraging them to watch the videos and thereby engage in learning. Simply providing video tutorials related to the course material may be valuable to some students, but will not motivate those who are not engaging in learning. Careful consideration must be applied to the design and focus of the video tutorials. Aligning the video tutorials to the assessment encourages more students to engage in learning, but this alone is not enough to assist all of the failing students. There remains a cohort of students who enroll in units but simply do not engage at all. Future research should focus on the specific qualities of these "failing" students, rather than general student failure and performance. This would provide a better understanding of their learning styles and appropriate content delivery, allowing them to be given tailored assistance where required.

## REFERENCES


[1] J. Sheard and D. Hagan, "Our failing students: A study of a repeat group," *SIGCSE Bull.*, vol. 30, no. 3, pp. 223–227, 1998.

[2] Q. H. Mahmoud, W. Dobosiewicz, and D. A. Swayne, "Making computer programming fun and accessible," *IEEE Comput.*, vol. 37, no. 2, pp. 106–108, Feb. 2004.

[3] C. A. Carver, R. A. Howard, and W. D. Lane, "Enhancing student learning through hypermedia courseware and incorporations of student learning styles," *IEEE Trans. Educ.*, vol. 42, no. 1, pp. 33–38, Feb. 1999.

[4] R. Lister and J. Edwards, "Teaching novice computer programers: Bringing the scholarly approach to Australia," Australian Learning and Teaching Council, Sydney, Austrailia, 2010.

[5] M. McCracken, V. Almstrum, D. Diaz, M. Guzdial, D. Hagan, Y. B.-D. Kolikant, C. Laxer, L. Thomas, I. Utting, and T. Wilusz, "A multinational, multi-institutional study of assessment of programming skills of first-year CS students," *SIGCSE Bull.*, vol. 33, no. 4, pp. 125–180, 2001.

[6] T. Jenkins, "On the difficulty of learning to program," in *Proc. 3rd Annu. Conf. Learning Teaching Support Netw. Cen. Inf. Comput. Sci.*, Loughborough, U.K., Aug. 2002, pp. 27–29.

[7] D. Cummings and C. Buzzard, "Technology, students, and faculty. how to make it happen!," in *Proc. ASCUE Conf.*, Myrtle Beach, SC, 2002, pp. 55–60.

[8] D. Hagan and S. Markham, "Does it help to have some programming experience before beginning a computing degree program?," *SIGCSE Bull.*, vol. 32, no. 3, pp. 25–28, 2000.

[9] S. Lohnes and C. Kinzer, "Questioning assumptions about students' expectations for technology in college classrooms," *Innovate, J. Online Educ.*, vol. 3, no. 5, 2007 [Online]. Available: http://www.innovateonline.info/pdf/vol3_issue5/Questioning_Assumptions_About_Students%27_Expectations_for_Technology_in_College_Classrooms.pdf

[10] J. B. Caruso, "ECAR study of students and information technology, 2004: Convenience, connection, and control," Educause Center for Applied Research, 2004.

[11] G. E. Kennedy, T. S. Judd, A. Churchward, K. Gray, and K. L. Krause, "First year students' experiences with technology: Are they really digital natives?," *Australian J. Educ. Technol.*, vol. 24, no. 1, pp. 108–122, 2008.

[12] A. G. Picciano, "Beyond student perceptions: Issues of interaction, presence, and performance in an online course," *J. Asynchr. Learning Netw.*, vol. 6, no. 1, pp. 21–40, 2002.

[13] M. Prensky, "Digital natives, digital immigrants part 1," *On Horizon*, vol. 9, no. 5, pp. 1–6, 2001.

[14] L. Winslow, "Programming pedagogy—A psychological overview," *SIGCSE Bull.*, vol. 28, no. 3, 1996.

[15] L. E. R. Smith, "Does digitally enhanced instruction benefit student learning," Master's of Science in Information Science thesis, University of North Carolina, Chapel Hill, NC, 2009.

[16] J. Nicholson, J. , and D. B. Nicholson, "A stream runs through IT: Using streaming video to teach information technology," *Campus-Wide Inf. Syst.*, vol. 27, no. 1, pp. 17–24, 2010.

[17] C. Hundhausen and J. Brown, "What you see is what you code: A 'live' algorithm development and visualization environment for novice learners," *J. Visual Lang. Comput.*, vol. 18, no. 1, pp. 22–47, Feb. 2007.

[18] J. B. Bennedsen and M. E. Caspersen, "Exposing the programming process," in *Reflections on the Teaching of Programming*. New York: Springer-Verlag, 2008, LNCS 4821, pp. 6–16.



**Jason Wells** received the B.A. degree in science from Deakin University, Waurn Ponds, Australia, in 1991.

From 1992 to 2004, he was a Graduate Research Assistant with the School of Information Technology, Deakin University, working in the field of machine learning. In 2005, he took a position of Associate Lecturer and was promoted to Lecturer in 2008. His research interests include machine learning, information security, and the use of information technology within education.

**Robert Mathie Barry** received the B.A. degree in psychology and B.A. (Hons.) degree in information technology, for the thesis "Understanding Player Preferences of Video Game Design: An Application of the Uses and Gratifications Paradigm to the Fundamentals of Video Game Design," from Deakin University, Geelong, Australia, in 2011 and 2012, respectively.

He has been employed as a Research Assistance and Demonstrator with the School of Information Technology, Deakin University. His interests include the application of human motivational theories from psychology and other fields to assist an understanding and improvement of video game enjoyment.

**Aaron Spence** received the B.A. degree in science and B.A. (Hons.) degree in information technology, for the thesis "Assessment of the Feasibility of Handheld Video Game Development in Tertiary Curriculum," from Deakin University, Geelong, Australia, in 2009 and 2011, respectively.

He has been employed as a Research Assistant and Demonstrator with the School of Information Technology, Deakin University. His interests include strengthening the creative ties between players and gameplay through unique and inspiring game design.


Authorized licensed use limited to: University of Southern Denmark. Downloaded on August 19,2022 at 07:13:55 UTC from IEEE Xplore. Restrictions apply.