Using video tutorials as a carrot and stick approach to learning

Jason Wells
Deakin University
Pigdons Road
Waurn Ponds, VIC 3217
Australia
wells@deakin.edu.au

Robert Barry
Deakin University
Pigdons Road
Waurn Ponds, VIC 3217
Australia
rmba@deakin.edu.au

Aaron Spence
Deakin University
Pigdons Road
Waurn Ponds, VIC 3217
Australia
amspe@deakin.edu.au

Abstract

Traditional teaching styles practiced at universities generally do not always suit all students learning styles. Students enrolling in university courses are not always engaging in the learning for many reasons. New methods to create and deliver educational material are available but do not always improve the 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 model that is valuable to all students and in particular those students that are not engaging in the learning. The results of the study have demonstrated that the use of video tutorials that are well designed, assessment focused and readily available enable and encourage students to learn how they want, when they want, at a pace that suits their needs and based on a three year study have the potential to improve results, student satisfaction and attract those students not engaging in the learning back from failure.

1. Introduction

High fail rates are of a particular concern to Universities [15] both for the implications to the students and its reflection on the institutions. While new ways of teaching, such as introduction of multimedia content into a course, can result in improvement to students performance, sometimes this increase only affects the average to above average student while the number or students who fail still remains [10], [2], [15]. This study examines the design and use of multimedia resources within a first year programming unit in an attempt to address high fail rates.

2. Teaching and Course Difficulties, Specific to Programming

Programming courses at universities have experienced low performance over the last decade [7], [11], [15]. There is no single reason for the low performance as both the university and student have their role to play to ensure the outcomes are positive. Although this is not unusual at a university level it is clear that teaching programming presents many challenges.

Students do not fail on purpose [5]. A study by Sheard et al. [15] surveyed 84 students to discover the reasons why computer science students were failing. Initial factors found 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 seek out any extra resources on their own initiative.

Programming is a difficult skill to learn with experienced programmers drawing upon many diverse skills to produce computer code [5]. Jenkins [5] highlights some difficulties had by the students. Complexity of the content is apparent with the requirements of 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 potentially becoming problematic if students were to fall behind due to slow comprehension, confusion of the content, or simply missing a class. Difficulty may result from an instructor being poor at teaching and engaging the students even when they are skilled at programming. The opposite is also possible in that some students may become bored as they perceive their teacher as being not as skilled at programming [3].

In addition there is the possibility that the content within a programming course is not appropriate to students skill levels [11]. Having previous programming knowledge and experience has shown to be of marked benefit to student engaging in computer science degrees [4]. Skill levels can vary substantially due to factors such as personal aptitude to the course work and previous experience gained from employment, or participation in computing units during high school [3].

Research has suggested students do not undertake programming units before their second year of studies as students may be going through many life transitions at the time [5]. Before any consideration is given to the concept of changing elements of course structure or content delivery, adequate consideration of student issues, such as the learning environment and assessment of students proficiency in technology [9] should take place.

Students in all subjects are becoming consistently more technologically savvy [1], [6], [8]. The current generation, the 'Digital Natives' [6], [13], has been found to consist of approximately more than 85% of the students who use online resources for their education [1] as well socializing [8], email and general information gathering [6]. The frequency of use of the internet being daily, or at least weekly [6]. Technology is a part of the students' lifestyle, creating a preference for multi-tasking, fast information presentation and 'non-linear' access to learning material [6] , [14].

The other key issue that is under the control of the universities is the way the courses are taught, or more specifically understanding the way students learn and making efforts in their teaching to accommodate the differences. Understanding these two parts of the equation is the first step to finding potential avenues to assist the grades of the failing students.

The key to reaching a level of competence in programming is through practice, practice, practice [18]. For most students practicing is not an issue but for many there is an increasing trend to avoid this type of applied application and as a consequence a higher than normal fail rate is common. To address these issues a new method of providing tuition has been developed that attempts to attract the 40% of students not engaging in the learning.

3. The study background

Within the School of Engineering at Deakin University 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 and consequently approach the subject with a negative attitude. As a consequence, the fail rate for the subject has been high, ranging from 30 to 40% and the perception of the unit from a student's perspective has been low. In an attempt to address the fail rate and negative perceptions, new teaching techniques have been developed and studied, resulting in a dramatic turnaround in the fail rate and the perception of the unit.

The School of Engineering uses traditional teaching methodologies consisting of lectures where students attend and are presented lecture slides, examples, demonstrations and have an opportunity to ask questions. There is an expectation that the students 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, practical sessions are provided 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 where students solve programming problems and an exam conducted at the end of the semester to determine the level of programming knowledge the students have obtained. The methodology assumes the lectures present the concepts, the practicals encourage the application of the concepts, and assessment examines how much the students have learned. In all cases the students are expected to attend their allocated lecture and practical classes and attempt all assessment tasks.

With the introduction of the Internet and eLearning portals, many resources are provided to the students that are accessible 24 hours a day via the Internet. These

resources 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 eLearning portals enhance the learning experience by providing more flexible ways students could access and complete the unit requirements. Students that missed a lecture or practical could 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 wanted, when they wanted and how they wanted. The natural extension of this new method of delivering and conducting courses in addition to the traditional methods would be an increase in the quality of learning that was taking place and consequently high results and lower fail rates. Sadly this has not been the case and in some cases the fail rate is increasing.

A three year study has been conducted to investigate the use of screen capture technology to deliver learning resources for the SIT172 programming unit within Deakin University. As part of their Engineering degree, all students must complete the SIT172 Programming for Engineers unit regardless of their chosen discipline. The unit focuses on the fundamentals of programming and currently teaches the application of the C programming as the primarily language. The unit is conducted over 12 weeks of lectures, one week of revision and a two week exam period where they must sit a three hour closed book exam completed without the use of a computer. The unit consists of both on and off campus students where approximately 25% students are off campus students.

On campus students have 3 hours of lectures and a two hour practical class. Off campus students do not attend face to face classes. All unit material is provided online via an eLearning portal.

Assessment consists of four assignments worth 10% each and an end of year exam worth 60%. To pass the unit, students require 50% or greater overall and no hurdles exist. Assignments were due in weeks three, six, nine and eleven. All assignments were based on a single problem broken into four stages where each stage built on to the previous, culminating in a complete solution in week 12. Each stage examined the unit content from previous weeks and 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.

4. Multimedia content

Programming uses a computer and generally 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 the screen images, mouse movements, keystrokes, menu selections as well as provide audio of the person using the software to store and deliver the recording via the Internet in a number of different formats. The recordings 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 enables the creation of tutorials that demonstrate the use of the programming environment, the use and application of the programming language based around a 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 sorts after resource by all students, especially those students that are not choosing 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, Howard, and Lane [2] highlighted the need to clearly define the association between the resource and the topic being studied. Smith [16] and Nicholson and Nicholson [12] confirmed the use of the resource was both effective and well received by those students who used the resources. Nicholson and Nicholson [12] 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:

- 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 is required to 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 best to test the code and in the use of the debugger software to examine in more detail the data flow through the code.

Each video presented a problem that reflected the weekly topic being studied but the content was not directly associated with the assignment requirements. Video were generally 10-15 minutes in duration and where required, broken into parts to ensure individual concepts were not mixed. This is particularly important as some student may want to focus of a single concept and can target a particular concept/video as required.

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

Initial feedback was very positive, especially from the off campus students as they had very little direct interaction with the lecturer other than via the iLecture recordings and in general 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 and they 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 requirement to explain a concept many times to many students diminished. Experience that helps avoid many programming mistakes can be passed on efficiently and an example of best practiced reinforced constantly. The videos were simple to create and many videos were produced to address specific problems as they arose. This enabled the lecturer to target specific issues as required, quickly and efficiently to all students.

5. Initial outcomes

In 2008 the results achieved by the students 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 voluntarily complete the survey anonymously and results are considered by the University as an important guide to the level of service, quality of the teaching and regularly scrutinized by management. Prior to 2008 the unit consistently scored poorly in relation to student satisfaction.

Six questions are asked along with the opportunity to provide written comments associated with the unit. The following summarizes the results from 2006 to 2010 for both On and Off campus modes.

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

Figure 1: SETU results for both On and Off Campus

"I was satisfied with the quality of teaching from this teacher in this unit."

The introduction of the video tutorials in 2008 indicates 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 fail rate remained relatively unchanged. The improvement in the student's satisfaction was reflecting a positive perception for the video's, especially from Off Campus students who rarely received an interactive teaching experience and potentially because the use of the technology was new and not being used in other subjects they were studying. Despite this 40% of students were still not engaging in the unit content and were not utilizing the videos as expected regardless of their value and accessibility.

6. Refinements

In 2009, the design of the video tutorials was adjusted to more closely align them with the four assignments. Assessment 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. Regardless 40% of students were electing to not complete all the assignments or were completing the assignments poorly.

Guessing a solution to a programming problem is unlikely. To code a solution you must understand the problem and then understand how you can code the solution using the

required programming language. To learn students must practice the concepts and generally the time and effort required can be demanding.

Not all university students studying engineering can muster the motivation to learn programming and as a consequence either lose confidence and drop out or simply cannot use the resources efficiently within the timeframes to learn the unit material as required. As a consequence a new strategy was developed to streamline the learning process and focus all the videos on how to complete the assignments.

The subject for each video was based around a problem that was similar to the assignment problem the students were expected to complete. Designing the content of the videos to relate more closely with the assignment problem implies that by watching the video you will get more help in completing the assignment. This strategy was used specifically to appeal to those students that were not engaging in the learning. If the students were attempting and completing the assignments they are engaging in the learning whilst gathering marks and consequently passing the unit.

Students were instructed to watch the videos and then attempt the assignment. Each video stepped the students through the concepts required to complete the assignment. Students can pause the video, then apply what they watched to the assignment, test and confirm the code then move to the next concept. The result would be two - five hours of programming practice and a part or completed assignment.

The danger in this strategy is that the videos would compromise the learning by giving the student too much information relating the assignment solution and bypass quality learning and when it came to the exam the students would not have the required

knowledge to solve the exam questions using their acquired knowledge without the aid of an accompanying video.

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

The exam component of the unit assessment was conducted at the end of the unit and consisted of a closed book, three hour supervised exam worth 60% of the overall assessment marks for the unit. The exam consisted of multiple choice questions that required student 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. The exam was important as it provided an unassisted environment that required a student to reflect their knowledge of the unit and their ability to write logical, well-structured code.

7. Improved outcomes

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

The SETU results for 2009 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.

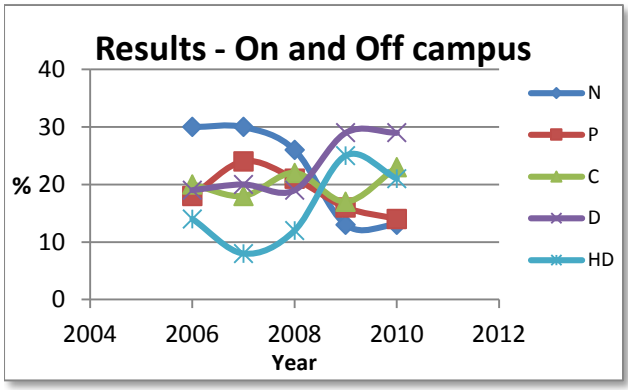
Results were as follows:

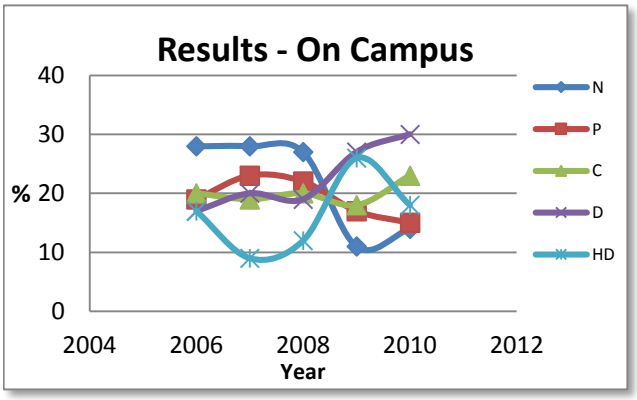
Grade	Mark out of 100
N	0 – 49 (Fail)
P	50 – 59 (General pass)
C	60 – 69 (Credit pass)
D	70 – 79 (Distinction pass)
HD	80 – 100 (High Distinction pass)

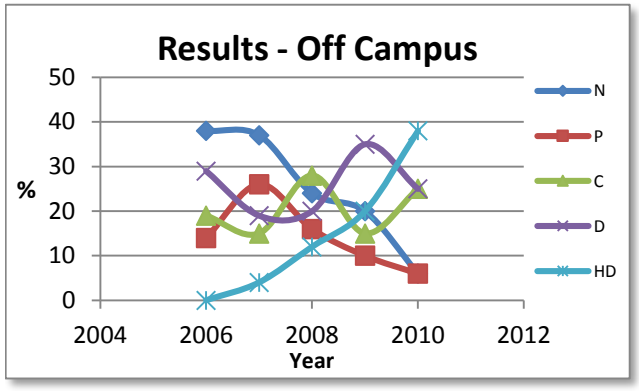
Figure 2: Results Table

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

Figure 3: Results







Students studying in off campus mode appeared to gain the most benefit from the video tutorials as fails dropped to 6% and Results 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 practical's and were left to their own devises to learn the unit contact. The video

tutorials filled the void and were directly responsible for the dramatic turn around to excellent results and unit perceptions.

8. Validation

To determine in more detail what resources were being utilized and valued by the students and to gain a better understanding of the student perceptions of the unit a survey was created and distributed to all students in 2009 and 2010. 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 how important each resource was and to provide clear guide to where effort should be directed in the future to ensure the teaching methodology being used best suited the students and the way they wanted to learn.

The results from the surveys clearly indicates 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 subsequently enabling students to learn the unit material as indicated by the improvement in the unit results.

Year	Count	On%	Off%
2009	56	96	4
2010	23	65	35

Figure 4: Survey response rate

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

Figure 5: Survey results

The unexpected outcome of introducing the video tutorials to the unit resources has been the devaluing of the lectures conducted for the unit. The surveys clearly indicate video tutorials are the most valued resource but face to face lectures hardly rate. This, in hindsight, was being reflected in the low attendance rates to the lectures and indicates that students prefer to use resources to help them complete the unit expectations when they want or need them. Lectures require a student to attend a class at a specified time and place and the content presented may not immediately relate to the assessments tasks they are required to complete. The video tutorials allow the student to obtain the information they need to learn in order to complete the assessment when they want, where they want and as many times as they need.

9. Opportunity

This approach is not just beneficial to teaching programming but any discipline that requires students to learn the process that is required to create a product or outcome []. Static content struggles to pass on experience, technique, process and an application of knowledge. This is especially relevant where the application of the knowledge requires the use of unfamiliar tools, environments and pitfalls that have the potential to frustrate and distract the learner.

10. Conclusion

Traditional teaching styles where students are expected to attend lectures and practical classes to gather an understanding of the unit material, works for those that attend but has the capacity to impact on those students that do not attend. The introduction of the video tutorials has enabled those students that do not attend classes and consequently

struggle to pass the unit and opportunity to engage in the learning when it is most important. Where the assessment is the motivation to learn, video tutorials designed to focus on the concepts required to complete the assessment are desired, used and have very positive outcomes. The video tutorials not only benefit those that are not attending the classes but all students appear to benefit. This is reflected in a general shift upwards of the unit results.

There are many issues that students must overcome in order to complete coursework and there are many issues lecturers faces to ensure students receive the service and resources required to guide them through this process. This study has confirmed the value 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 the assessment, encouraging them to watch the videos and thereby engaging them in the learning. Simply providing video tutorials relating to the course material, although valuable to some, will not provide the motivation to those students that are not engaging in the 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 the learning but this alone is not enough to assist all of the failing students. There remains a cohort of students that are enrolling in units but simply not engaging at all. Future research should focus on the specific qualities of these ‘failing’ students and not general student failure and performance. This information would allow better understanding of their learning styles, appropriate content delivery, and as a result action can be taken to more accurately to assist the students where required.

11. References

[] Bennedsen, J.B. and Caspersen, M.E.: ‘Exposing the Programming Process’, Reflections on the Teaching of Programming, LNCS 4821, Springer-Verlag, 2008, pp. 6-16

[1] Caruso, JB 2004, 'ECAR Study of Students and Information Technology, 2004: Convenience, Connection, and Control', Educause Center For Applied Research.

[2] Carver, CA, Howard, RA & Lane, WD 1999, 'Enhancing Student Learning Through Hypermedia Corseware and Incorperations of Student Learning Styles', IEEE Transactions on Education, vol. 42, no. 1, pp. 33-8.

[3] Cummings, D & Buzzard, C 2002, 'Technology, Students, and Faculty... How to Make It Happen!', in 2002 ASCUE Conference, Myrtle Beach, South Carolina.

[4] Hagan, D & Markham, S 2000, 'Does it help to have some programming experience before beginning a computing degree program?', SIGCSE Bull., vol. 32, no. 3, pp. 25-8.

[5] Jenkins, T. (2002). On the difficulty of learning to program. Presented at the 3rd Annual Conference of the Learning and Teaching Support Network Centre for Information and Computing Sciences, August 27-29, Loughborough, UK.

[6] Kennedy, GE, Judd, TS, Churchward, A, Gray, K & Krause, K-L 2008, 'First year students' experiences with technology: Are they really digital natives?', Australian Journal of Educationional Technology, vol. 24, no. 1.

[7] Lister, R & Edwards, J 2010, Teaching Novice Computer Programers: bringing the scholarly approach to Australia, australian Learning & Teaching Council, Sydney.

[8] Lohnes, S & Kinzer, C 2007a, 'Questioning Assumptions About Students' Expectations for Technology in College Classrooms', Innovate: Journal of Online Education.

[9] Lohnes, S & Kinzer, C 2007b, 'Questioning assumptions about the students' expectation for technology in college classrooms.', Innovate, vol. 3, no. 5.

[10] Mahmoud, Q. H., Dobosiewicz, W., & Swayne, D. A. (2004). Making computer programming fun and
accessible. IEEE Computer, 37(2), 106–108.

[11] McCracken, M, Almstrum, V, Diaz, D, Guzdial, M, Hagan, D, Kolikant, YB-D, Laxer, C, Thomas, L, Utting, I & Wilusz, T 2001, 'A multi-national, multi-institutional study of assessment of programming skills of first-year CS students', SIGCSE Bull., vol. 33, no. 4, pp. 125-80.

[12] Nicholson, J & Nicholson, DB 2010b, 'A stream runs through IT: using streaming video to teach information technology', Campus-Wide Information Systems, vol. 27, no. 1, pp. 17 - 24.

[13] Picciano, AG 2002, 'Beyond Student Perceptions: Issues of Interaction, Presence, and Performance in an Online Cource', Journal of Asynchronous Learning Networks, vol. 6, no. 1.

[14] Prensky M, (2001) "Digital Natives, Digital Immigrants Part 1", On the Horizon, Vol. 9 Iss: 5, pp.1 - 6

[15] Sheard, J & Hagan, D 1998, 'Our failing students: a study of a repeat group', SIGCSE Bull., vol. 30, no. 3, pp. 223-7.

[16] Smith, LER 2009, 'Does digitally enhanced instruction benifit student learning', Masters Paper thesis, Masters of Science in Information Information Science thesis, University of North Carolina.

[17] Winslow, L. (1996). Programming pedagogy - A psychological overview. SIGCSE Bulletin, 28(3).