Video Versus Source Code Lab Solutions

Aidan Mcgowan*
Queen's University, Belfast, United Kingdom
aidan.mcgowan@qub.ac.uk

Neil Anderson
Queen's University, Belfast, United Kingdom
n.anderson@qub.ac.uk

Paul Sage
Queen's University, Belfast, United Kingdom
p.sage@qub.ac.uk@qub.ac.uk

Leo Galway
Queen's University, Belfast, United Kingdom
l.galway@qub.ac.uk@qub.ac.uk

Janak Adhikari
Queen's University, Belfast, United Kingdom
j.adhikari@qub.ac.uk

Giuseppe Trombino
Queen's University, Belfast, United Kingdom, g.trombino @qub.ac.uk

ABSTRACT

Traditionally university programming modules have been delivered using a blend of lectures, tutorials, and practical lab sessions. Although the lab sessions offer valuable hands-on practice, they are constrained by time, limited individualised pacing, and insufficient feedback opportunities. The solutions for the labs are normally provided as static source code, with students reviewing their attempts against the model answer. The use of video-based solutions for lab exercises has the potential to enhance flexibility and interactivity for the lab. This study explores the attitudes, experiences, and impact of the wholesale provision of video-based lab solutions in improving the student performance of a cohort of postgraduate novice programmers. It reports high student engagement with the video solutions with a clear preference for a dynamic build-up style. It also identifies separate engagement styles with the videos as well as overall improvement in module averages compared to previous cohorts. The findings highlight the potential of video-based lab solutions to enhance student learning in programming modules and adds to the literature in a relatively under-researched area and presents potential of further adoption and adaption in programming and other engineering disciplines.

CCS CONCEPTS

• **Applied computing**; • **Education**; • **E-learning**;

KEYWORDS

Programming, university, lab, video, solutions

ACM Reference Format:
Aidan Mcgowan, Neil Anderson, Paul Sage, Leo Galway, Janak Adhikari, and Giuseppe Trombino. 2024. Video Versus Source Code Lab Solutions. In *Computing Education Practice (CEP '24), January 05, 2024, Durham, United Kingdom.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3633053.3633056

*Corresponding author.

CEP '24, January 05, 2024, Durham, United Kingdom

ACM ISBN 979-8-4007-0932-6/24/01.
https://doi.org/10.1145/3633053.3633056

1 INTRODUCTION

While the traditional methods of lectures, tutorials, and practical lab sessions used for the delivery of university programming remain commonplace, there is increasing adoption of technology-enhanced approaches [1], such as online learning platforms, interactive coding environments [2], and video-based instruction [3]. Variations in lecture delivery including flipped, online, broadcast, etc. have become more common in recent years. These newer approaches aim to enhance the learning experience in programming education [4].

Practical lab sessions remain a cornerstone of programming module delivery and provide a dedicated environment for students to apply the knowledge gained from lectures and tutorials [5]. In lab sessions, students have access to computers with programming software and they are expected to work on coding exercises, applying the programming concepts learned in lectures. Lab sessions allow for immediate feedback and assistance from lecturers or lab assistants [6]. While lab sessions in programming modules offer valuable hands-on experience, there are a few potential disadvantages [7], including limited availability, lack of individual pace, and insufficient feedback.

When it comes to providing solutions to lab exercises, programming lecturers employ various approaches based on their teaching style and the learning objectives of the course [8]. The provision of the source code solution is typically the common baseline, however, there are several alternative methods including In-Class Demonstrations, Written Solution Guides [9] and Office Hours [10]. While providing solutions to lab exercises is essential, similar to flipped classroom activities there are potential disadvantages associated with the timing of availability of solutions [11]. These include reduced problem-solving skills, limited learning from mistakes, reduced engagement and active learning [12]. To mitigate these disadvantages, lecturers commonly implement strategies such as delayed solution release, providing partial solutions, or promoting peer discussion and collaboration to encourage independent thinking and problem

1.1 Video-based Lab solutions

There is significant research on the use of coding videos in programming courses [13]. These include the positive use of coding video quizzes [14] finding that students who were given coding video quizzes performed better on written assessments, especially in final exams. However, the use of coding videos for flipped lectures by Horton et al [15] concluded that there was no evidence that the inverted offering helped beginners or those not fully fluent

in English. This contrasts with Moore et al [16] who concluded that students who watched more pre-lecture coding videos performed slightly better on summative assessments. There are very limited specific studies on coded videos directly related to lab activities, with [17] providing an evolved "worked example video" study that concluded that the video solutions were well received by students and increased flexibility and accessibility. The video solutions were especially helpful for struggling students whereas stronger students tended to skip the videos. Concluding that overall lab videos are a recommended valuable addition to Java/beginners programming courses.

This paper seeks to add to the literature specifically in the provision of lab solution videos by investigating and reporting on the outcomes of providing video-based solutions for lab exercises for a cohort of novice programming university students. It finds favourable outcomes in terms of academic performances and student feedback and engagement.

2 METHODOLOGY

The study was conducted with a cohort (n=108) of postgraduate students taking one compulsory module in Java programming over two semesters in a one-year university Masters Software Development (conversion) course. The module was taught over a 24-week period with typically three one-hour lectures per week, with three hours of timetabled lab sessions. Compulsory attendance was not enforced. The students were provided with a source code-only solution to each lab exercise and video solution. Both were provided before the scheduled lab time. The video solutions were in the format of i). Code Walk-through or ii). Dynamic Build-up. The Walkthrough style involved a thorough analysis of the lab exercise problem statement and code solution. This style focused on the completed code solution especially highlighting complex areas. The lecturer narrated a line-by-line code inspection, explaining the function and contextualising the code. The Build-up style involved the lecturer recording their screen as they demonstrated the construction of the solution to the lab exercise. The lecturer coded from scratch explaining the thought process, allowing students to retrospectively observe or follow along by pausing and replaying the video. Build-up videos typically taking longer to produce, consequently, in this study, roughly 40% of the video solutions in this study were in this format and the remainder were Walkthroughs. On completion of the module, the students were provided with a survey to gain an understanding of attitudinal and behaviours towards the video solutions. This included engagement with the videos including frequency and methodology, the preferred format for the solutions, and information gathered on any engagement with part-time work. The students were also provided with an open-ended question on the impact of the videos on each student's learning experience. The survey was completed by 81 of the students (75%) of the cohort. To compare the academic performance the module results for this cohort were also statistically contrasted with previous historic averages for similar cohorts that had no or very limited lab video solutions. This cohort and the previous cohort had the same course entry requirements, including having no previous formal background in programming. All students completed and passed the same aptitude test with similar cohort averages from previous years.

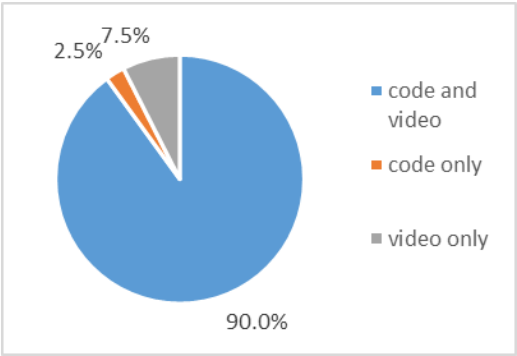

Figure 1: Preferred format for the solutions

3 RESULTS

3.1 Engagement

Based on the frequency of hits and the survey feedback, the students were highly engaged with and highly regarded the video solutions to support their learning, with only one student reporting that they rarely used the videos. Many students reported that "the videos were an essential resource for me". Engagement with the videos based on hits was highest at the time of the related lab but also a not insignificant number of hits were observed shortly after the lab (within a week of the lab) and then a further peak at assessment time, this would be a similar finding as McGowan et al [17]. The attendance at the labs was around 70-80% throughout the course, and was comparable to previous years when the videos were not available. Overall, the high engagement with the videos, and the timing of engagement beyond the timetabled lab would suggest that students who were unable to attend the lab at the time used the lab solutions to support their learning.

3.2 Lab solution format

The solutions for each lab exercise were provided as video and in the more traditional source code-only format. The students reported that they preferred to have both solution formats (90.0%), with 2.5% relying on the source code only and 7.5% preferring video only, as shown in Figure 1.

Typical quantified responses were, "I really needed the explanation the lecturer provided in the videos to enable me to understand the code and principles, the code-only solutions do not provide this" and "I found the videos very helpful. I didn't always understand the code-only solutions". However, a small number of students expressed their reasons for using both "I tend to prefer video over raw code as it forces me to engage with the process more."

Recalling that the format of the video solutions was a mixture of Walk-throughs and Build Up, the students reported (Figure 2) that they strongly (90%) preferred the Build Up style.

Typical feedback was "seeing the problem solved from scratch worked the best for me as it was useful to see the thinking behind the solution as it went along". The case for Walkthroughs was made

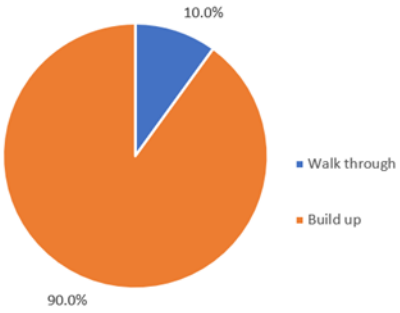

Figure 2: Preferred video solution style

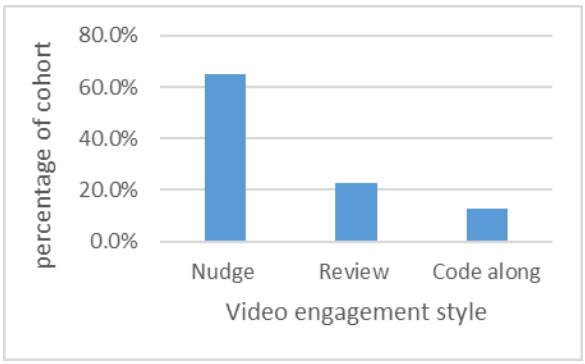

Figure 3: Video solution viewing patterns – timing of engagement

by students' responses such as: "I found the videos where the code was already built, as more efficient for my learning as opposed to building from scratch."

3.3 Video engagement timing

The timing of engagement with the videos in relation to the solution-solving identified three separate groups, namely Nudge, Review, and Code Along (Figure 3).

Most of the students (77.3%) used the videos for Nudge learning, with learners in this group typically reporting that "I attempt the labs and used the video solutions to guide me when I was stuck". A smaller number (12.5%) used the videos as Code Along, i.e., followed the video solution without tackling it independently. An interesting perspective on this was provided in the feedback "sometimes it was easy to just jump straight to them (the videos) without struggling for a while". The remaining 20.0% were categorised as the Review group i.e. tackled the problem first and then retrospectively reviewed the video solution.

An analysis of the timing of engagement groupings highlighted a difference between the groups in relation to module scores (Table 1). The Review group scored 77.3%. Those that used the video when stuck on the problem scored 68.4%, whereas those that relied on the video to provide the solution without first attempting it scored 54.0%. A one-way ANOVA was performed to compare the effect of the main three groups (Code Along, Nudge and Review) on module

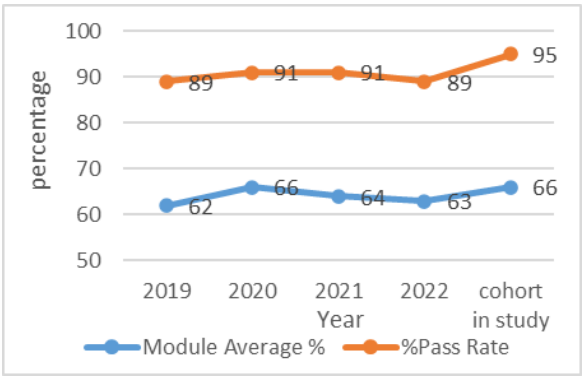

Figure 4: Trends in pass rates and module scores.

scores. It revealed that there was a statistically significant difference in mean module score between at least two groups (F(2, 27) = [4.545], p = 0.02). Tukey's HSD Test for multiple comparisons found that the mean value of module score was significantly different between the Code Along and Nudge groups (p = .00631) and Code Along and Review groups (p = .00002). There was no statistically significant difference in mean module scores between Nudge and Review groups (p = 0.12).

A comparison of the module scores and pass rates, (Figure 4) shows a moderate increase in average module score for this cohort (66.38%, SD=11.34) and historic averages (64.49% SD=16.94), with a T-Test analysis result proving not significant (p=0.13).

However, there was a marked increase in the module pass rate, with 95.1% of the 2022 cohort passing compared to the historic 90.2% pass rate. This may suggest that the videos were a factor in helping weaker students, that may otherwise have failed.

4 DISCUSSION AND LIMITATIONS

The study provides insights into the impact of video solutions for labs on student engagement, preferred learning formats, viewing patterns, and module outcomes. The findings indicate that the majority of students were highly engaged with the video solutions and regarded them as beneficial for their learning.

The availability of both video and source code solutions was preferred by most students, highlighting the importance of having multiple formats. The students expressed the significance of the explanatory content provided in the videos, enabling them to understand the code and principles more effectively compared to code-only solutions. The students overwhelmingly favoured the Build Up style of video solutions, where problems were solved from scratch. This format enabled students to grasp the engineering design and process decisions behind the solution. It is unlikely that lecturers could feasibly provide both styles for each lab and it may be that the use of Generative AI such as Chat GPT, which provides a similar but written dialogue-based description of code would be a viable alternative for the minority group of students that preferred Walkthrough video format.

The survey revealed diverse video viewing patterns, with students predominantly using the videos for Nudge learning, seeking

Table 1: Module scores and percentage of the cohort for each video engagement group

	Module score %	Percentage of cohort
Review	77.3 (SD=10.64)	22.5
Nudge	68.4 (SD=8.86)	65.0
Code Along	54.0 (SD=7.1)	12.5

guidance when stuck on a problem. The overall module score average for the cohort increased compared to historical averages, indicating potential performance enhancements, especially among traditionally weaker students.

This study is limited to this cohort for a one-year period and as such is representative only of that sample. The population itself is important but may not be comparable to students in secondary or early tertiary environments, as they likely have more mature self-regulation and learning skills. Future iterations of the study could include undergraduate novice programmers and an increase in the availability of the Build Up format for the lab videos to enable further comparisons. The viewing groups (nudge/review/code along) are self-identified by the students themselves, however, it is unclear if this is time-dependent such that are the groups truly disjointed or whether students sometimes use videos differently at different times. The use of video engagement analytics such as those provided via YouTube or Panopto would enable a more temporal-based study. Although the target of this study is programming the findings could have a generalisability into other instructional program-solving engineering disciplines.

5 CONCLUSION

These findings emphasise the value of incorporating video solutions as flexible resources for lab solutions for novice programmers to support lab exercises. This study would recommend that video solutions continue to be implemented and refined, considering the preferences and needs of students. Overall, this study suggests that the integration of video solutions positively impacted student learning and performance, contributing to the ongoing improvement of the educational experience for novice programmers.

REFERENCES

[1] Pattanaphanchai, J. (2019). An Investigation of Students' Learning Achievement and Perception using Flipped Classroom in an Introductory Programming course: Journal of University Teaching & Learning Practice, 16(5)

[2] McDowell,C. Werner, L., Bullock, H. and Fernald, J. (2002). The effects of pair-programming on performance in an introductory programming course. SIGCSE Bull. 34, 1 (March 2002), 38–42

[3] McGowan, A., Hanna, P., Greer, D., Busch, J., Anderson, N., Collins, M., Cutting, D., Stewart, D., & McDowell, A. (2019). Use of Wearable Technologies with Machine Learning. In IHIET 2019: Human Interaction and Emerging Technologies (Vol. 1018, pp. 325-331)

[4] Omer U, Farooq MS, Abid A. (2021). Introductory programming course: review and future implications. PeerJ Computer Science 7:e647

[5] Malik, S.I. Improvements in Introductory Programming Course: Syst Pract Action Res 31, 637–656 (2018)

[6] Parihar, S., Dadachanji, Z., Singh, P., Das, R., Karkare, A. and Bhattacharya, A. (2017). Automatic Grading and Feedback Program Repair for Introductory Programming. (ITiCSE '17)

[7] Malik, S., Coldwell-Neilson, J. A model for teaching an introductory programming course using ADRI. Educ Inf Technol 22, 1089–1120 (2017)

[8] Markovski, S. and Gusev, M: ICT Innovations 2012, Web Proceedings, ISSN 1857-7288 ICT ACT –

[9] Radenski, A. (2006). "Python first": a lab-based digital introduction to computer science. SIGCSE Bull. 38, 3 (September 2006), 197–201

[10] Delev, T and Dejan, G. (2012) E-Lab: Web Based System for Automatic Assessment

[11] Çakıroğlu, Ü. and Öztürk, M. (2017) "Flipped Classroom with Problem Based Activities: in Educational Technology & Society, (20:1), pp. 337–349

[12] M. Drini, (2018) Using new methodologies in teaching computer programming. IEEE Integrated STEM Education Conference (ISEC), Princeton, NJ, USA, 2018, pp. 120-124, doi: 10.1109/ISECon.2018.8340461

[13] McGowan, A., Hanna, P., & Anderson, N. (2016). Teaching Programming: Understanding Lecture Capture YouTube Analytics. ITiCSE 2016 Proceedings of the 2016 ACM

[14] Lacher, L., Jiang, A. and Lewis, M. (2018) Including Coding Questions in Video Quizzes for a Flipped CS1. 49th ACM Technical Symposium

[15] Horton, D., Craig, M., Campbell & Gries,P. (2014) ITiCSE '14: June 2014 Pages 261–266

[16] Moore, C., Battestilli, L. & Dominguez, I. (2021) Finding Video-watching Behavior Patterns in a Flipped CS1 Course. In ACM Technical Symposium on Computer Science Education

[17] Murphy, L. & Wolff, D. (2019) Creating video podcasts for cs1. Journal of Computing Sciences in Colleges Volume 25 Issue 158