Flipped Project-Based Learning for Drone Programming in Undergraduate Computer Science: An Instructor's Design Perspective & Experience

Felix Mulei Mule, Texas Tech University

Felix Mulei Mule is a doctoral candidate in Instructional Technology at Texas Tech University. My interests are in student-centered learning and project-based and blended learning. My work bridges evidence-based design with workforce-aligned training to improve learner engagement and outcomes.

Deniz Bulut, Texas Tech University

Deniz Bulut is a doctoral student and research assistant in the Educational Instructional Technology Program at Texas Tech University, specializing in teacher education and instructional design research.

Burcu Stone, Texas Tech University

Burcu Stone is a doctoral candidate in Instructional Technology at Texas Tech University and a Program Manager at Texas Tech University Online. Her research focuses on student-centered learning approaches such as game-based learning and blended/personalized learning, and design-based research in higher education.

Sungwon Shin, Texas Tech University

Sungwon Shin is an Associate Professor of Instructional Technology at Texas Tech University.

Mihwa Park, Texas Tech University

Dr. Mihwa Park is an Associate Professor of STEM education at Texas Tech University. Her research interests involve developing measurement instruments to assess students' understanding of scientific concepts and their emotions when learning science. She is also interested in teachers' emotions about teaching science and teacher identity.

Prof. Sunho Lim, Texas Tech University

He is an Associate Professor in the Department of Computer Science, Texas Tech University. His primary research interests are in the areas of Cyber Aerial Computing, Mobile Data Privacy, and Wireless Networks and Mobile Computing. He is a senior member of the IEEE.

Flipped Project-Based Learning for Drone Programming in Undergraduate Computer Science: An Instructor's Design Perspective & Experience

Abstract

Flipped Project-Based Learning (FPBL) has gained traction in engineering education as a means of promoting active learning, problem solving, and engagement [1]–[4]. However, despite increasing adoption, less is known about how FPBL is designed, implemented, and adapted from the instructor's perspective, particularly in hardware-intensive computer science contexts where implementation and troubleshooting are central to disciplinary practice [5]–[8]. This qualitative study examines an undergraduate drone programming course designed using an FPBL approach, focusing on the instructor's design rationales, challenges encountered during adoption, and instructional strategies developed over time. Data sources included pre-course interview, weekly instructor reflection sessions, and post-course interview collected across a semester. Preliminary findings indicate that FPBL was adopted primarily to align instruction with authentic computer science practice, integrate hardware and software learning, and troubleshooting as a core learning outcome. At the same time, implementation revealed challenges related to student readiness, cognitive load, accountability in flipped components, and scalability constraints. Findings from this study contribute to engineering education by illuminating on the instructor's FPBL design perspective, identifying practical strategies for balancing lectures and labs, and describing the experience of implementing project-oriented teaching models in computer science.

Keywords: Flipped learning, project-based learning, computer science education, engineering education, instructional design and troubleshooting

Introduction

Active learning approaches such as flipped classrooms and project-based learning (PBL) are increasingly promoted in engineering and computing education to address concerns about student engagement, retention, and transfer of learning [1], [2], [4]. Flipped learning shifts direct instruction outside class time, allowing in-class sessions to be used for application and problem solving [3], while PBL emphasizes authentic, open-ended projects as the primary vehicle for learning [9].

Despite growing adoption, the effectiveness of flipped and project-based approaches varies widely across contexts. Prior research has documented mixed outcomes, particularly in undergraduate computing courses where students may struggle with self-regulation, abstraction, and troubleshooting [10], [8]. Moreover, much of the existing literature focuses on student outcomes, with relatively limited attention to how instructors design, adapt, and make sense of FPBL in practice, especially in courses that integrate hardware and software [6], [7].This study addresses that gap by examining FPBL implementation from the instructor's perspective and experiences in an undergraduate drone programming course.

The study is guided by the following research questions:
RQ1: What rationales underlie the instructor's design decisions regarding the overall course structure?
RQ2: What challenges and barriers arise during the adoption of FPBL?
RQ3: What instructional strategies and design adjustments are reported as effective or necessary during implementation?

Literature Review

Flipped learning has been widely studied in engineering education, with reported benefits including increased engagement, improved conceptual understanding, and more efficient use of class time [3], [11], [4]. However, flipped classrooms also introduce challenges, particularly related to student accountability and preparation. When students fail to engage with pre-class materials, in-class activities may be less effective [12], [13]. Recent studies in computer science education indicate that novice learners often struggle with the self-directed learning demands of flipped models, particularly when content is abstract, technically complex, or cognitively demanding [14]–[16], [8]. These challenges are compounded in contexts where debugging, integration, and system-level reasoning are required.

Project-based learning emphasizes authenticity, student autonomy, and complex problem solving [9]. In engineering education, PBL has been linked to improved motivation, teamwork skills, and professional identity development [17], [18]. Authentic learning environments that mirror professional practice have been shown to support deeper learning and knowledge transfer [19], [20]. In computer science, authenticity often involves implementation, debugging, iteration, and interaction with real systems. Recent computing education research emphasizes that troubleshooting and debugging are central professional practices but remain under-emphasized as explicit learning outcomes in undergraduate curricula [10], [21], [22]. As a result, students may complete assignments without developing the diagnostic reasoning required in professional computing contexts.

Research on scaffolding highlights the importance of providing temporary instructional support that is gradually withdrawn as learners gain competence [23], [24]. More recent work in engineering education reinforces the need for adaptive scaffolding in open-ended learning environments, particularly for novice learners [25]. Additionally, cognitive load theory cautions against overwhelming learners with excessive complexity or insufficient guidance [26], [27]. Research suggests that carefully designed struggle can promote deeper conceptual understanding when failure is followed by structured reflection and guidance [28], [29]. These perspectives are particularly relevant for FPBL environments, where students must navigate open-ended tasks and unfamiliar tools. These conditions can either support deep learning or lead to disengagement depending on instructional design.

Methodology

This study employed a qualitative case study approach [30] to examine the instructor's design perspective and experience in implementing a Flipped Project-Based Learning (FPBL) within a single undergraduate computer science course. A case study design was appropriate given the study's focus on context-specific instructional design decisions, implementation processes, and the instructor's lived experiences over the duration of a semester.

The focal course was an undergraduate drone programming course offered within a computer science program at a large public university in the United States. The course enrolled approximately 20 junior- and senior-level students and was redesigned to incorporate an FPBL structure. Students worked in pairs on a sequence of drone-based projects integrating embedded hardware and Python programming.

To support hands-on access and manage resource constraints, the class was divided into two sections Group A and Group B with approximately 10 students per group. Each group was further divided into pairs to foster collaboration and peer learning. Classes met on Tuesdays and Thursdays. On each class day, one group participated in the in-person lab session while the other engaged in the online flipped learning component. The groups alternated roles across class meetings so that, over time, all students experienced both in-lab and online components equally. In addition, optional lab sessions were offered on Wednesdays and Fridays, providing students with additional opportunities to complete projects, troubleshoot issues, or seek instructor and teaching assistant support. These optional sessions were particularly important for students who fell behind during scheduled lab times or encountered hardware-related challenges.

The first two weeks of the semester were dedicated to introductory content rather than project work. During instructor interviews, this design decision was described as critical for establishing shared expectations and preparing students for the demands of FPBL. The instructor emphasized that many students had no prior experience with flipped learning or embedded systems and might mistakenly interpret the flipped structure as reduced instructional responsibility.

The online component of the course constituted the “flipped” portion of the FPBL structure. When not scheduled for in-person lab sessions, students engaged with online materials that included: Research articles related to drone systems, privacy, sensing, and embedded computing, short, short video summaries designed to contextualize the research articles and low-stakes quizzes intended to encourage preparation and provide evidence of engagement.

The lab component of the course extended over 10 weeks and was organized around four major projects designed with intentional scaffolding. Earlier projects provided extensive guidance, sample code, and step-by-step instructions, while later projects gradually reduced support and increased complexity. The first project (often referred to by the instructor as a “Project Zero”) focused on basic drone assembly and simple flight operations, serving as a foundational framework that students could extend in subsequent projects. The course exams comprised of mid and end of semester exams.

The course design was the result of a collaboration between the College of Engineering and the College of Education. This partnership supported the integration of instructional design principles, flipped learning strategies, and systematic reflection on teaching practice.

Data Sources

Multiple qualitative data sources were used to capture the instructor’s design intentions, implementation experiences, and evolving instructional decisions across the semester. Data included pre- and post- semi-structured interview with the course instructor, which elicited perspectives on course design rationales, anticipated challenges, and retrospective reflections on

FPBL implementation. Weekly instructor reflection sessions were conducted throughout the semester to document ongoing observations, emerging challenges and instructional adjustments. Lab observation notes were collected to capture instructional practices, student–instructor interactions, and contextual factors shaping implementation. In addition, course materials such as project descriptions, flipped learning materials and lab guides, were also collected. Together, these data sources would enable triangulation, supporting a holistic examination of FPBL implementation from the instructor's perspective.

Data Analysis

Data were analyzed using thematic analysis following the six-phase approach outlined by Braun and Clarke [31]. An inductive coding strategy was employed. Inductive coding was used to capture patterns and insights emerging directly from the data, particularly those reflecting the instructor's evolving perspectives and context-specific challenges.

Coding was conducted iteratively across data sources, including pre- and post-interviews, weekly reflection sessions and observation notes. Codes were progressively refined, collapsed into categories, and synthesized into higher-order themes aligned with the research questions. Data analysis is ongoing, and the findings presented here reflect preliminary thematic patterns, as this paper represents a work in progress.

Findings and Discussion

Analysis of interview data, weekly reflections, and observations revealed that the instructor consistently framed FPBL as a means of aligning instruction with authentic computer science practice. Across data sources, instructional success was defined primarily by functionality (whether systems worked) rather than adherence to prescribed procedures. This emphasis reflects professional norms in computing, where implementation, iteration, and debugging constitute central forms of knowledge production. From the instructor's perspective, FPBL helped students learn how computer scientists think and work, not just the underlying concepts. This finding aligns with prior research on situated and authentic learning in engineering and computing education, which emphasizes learning through participation in meaningful practice [32], [19].

A key rationale underlying the FPBL design was the instructor's perception of a significant curricular gap in students' prior learning. While students entered the course with substantial experience in programming and simulation-based coursework, most had limited exposure to physical systems and embedded hardware. Drone-based projects were intentionally designed to combine hardware and software so that system limits, failures, and dependencies were visible to students. This design made learning tasks more realistic and required students to think about how the entire system worked. Rather than treating hardware as an add-on, FPBL made hardware–software interaction a core part of learning, reflecting how computing and engineering are practiced in real-world settings. In this course, troubleshooting was not merely a byproduct of project work but a deliberately cultivated practice through which students learned to reason about complex systems.

In this integrated setting, troubleshooting became both a key learning goal and a central teaching approach. Rather than minimizing errors or treating failure as something to be avoided, the

instructor framed breakdowns as expected and productive. Teaching assistants were explicitly instructed to prompt students with guiding questions rather than provide direct solutions, encouraging students to diagnose problems independently. This approach aligns with research on productive failure [28] and supports calls within computing education to foreground debugging and troubleshooting as explicit learning goals rather than incidental skills.

At the same time, the instructor described persistent tension in balancing instructional scaffolding with cognitive load. Early projects, particularly the initial "Project Zero" provided substantial guidance to prevent students from becoming stuck. This helped establish foundational workflows related to hardware setup, code deployment, and basic system operation. As the semester progressed, scaffolding was intentionally reduced to promote independent problem solving. Determining the appropriate level of support was described as iterative and context-dependent rather than formulaic. This finding highlights the complexity of FPBL design in hardware-intensive contexts, where insufficient guidance can overwhelm novice learners while excessive support can undermine authentic engagement and agency.

Challenges related to flipped learning accountability further shaped instructional decisions. Student engagement with flipped learning materials was uneven, and although quizzes provided some evidence of participation, monitoring offsite preparation remained difficult. The instructor noted limited visibility into whether students meaningfully engaged with readings and videos prior to lab sessions. In this context, mandatory in-person lab participation functioned as a structural enforcement mechanism, ensuring engagement even when flipped preparation was inconsistent. These findings suggest that in hardware-intensive computing contexts, flipped learning alone may be insufficient without embodied, place-based activities that anchor learning in shared physical experiences.

Over the semester, significant instructor effort emerged as a defining aspect of implementing FPBL. Pre-semester preparation, equipment management, safety planning, and ongoing instructional adjustments were treated as ethical responsibilities rather than optional enhancements. Care was not seen as reducing academic standards, but as supporting students to work productively with challenging, failure-prone tasks while maintaining high expectations. This finding extends discussions of caring pedagogy [33] into engineering education contexts, illustrating how care and rigor can be mutually reinforcing in FPBL environments.

The instructor implemented several adjustments throughout the semester. The instructor and TAs offered additional attention to struggling students through targeted support and one-on-one interactions, facilitated by the small class size. Rather than directly providing answers, a guided prompting approach was emphasized to encourage students to think through problems, thereby reducing reliance on direct instruction and avoiding spoon-feeding. Structural adjustments were also made to support student progress, including offering extra lab sessions and extending project deadlines based on observed progress. Collectively, these in-the-moment adjustments aimed to support completion while fostering troubleshooting skills and learner autonomy.

Finally, the instructor emphasized significant constraints on scalability. Space limitations and safety regulations restricted the feasibility of expanding FPBL beyond small cohorts. These constraints caution against assuming universal applicability of FPBL without careful consideration of context. While the instructional principles underlying FPBL may be

transferable, their enactment remains deeply shaped by physical infrastructure, resources, and institutional support.

References

[1] M. Prince, “Does active learning work? A review of the research,” Journal of Engineering Education, vol. 93, no. 3, pp. 223–231, 2004.

[2] S. Freeman et al., “Active learning increases student performance in science, engineering, and mathematics,” Proceedings of the National Academy of Sciences, vol. 111, no. 23, pp. 8410–8415, 2014.

[3] J. L. Bishop and M. A. Verleger, “The flipped classroom: A survey of the research,” in Proceedings of the ASEE Annual Conference, Atlanta, GA, USA, 2013.

[4] D. C. van Alten, C. Phielix, J. Janssen, and L. Kester, “Effects of flipping the classroom on learning outcomes and satisfaction: A meta-analysis,” Educational Research Review, vol. 28, 2019.

[5] M. Borrego, J. E. Froyd, and T. S. Hall, “Diffusion of engineering education innovations,” Journal of Engineering Education, vol. 99, no. 3, pp. 185–207, 2010.

[6] X. Du, K. K. Naji, U. Ebead, and J. Ma, “Engineering instructors’ professional agency development and identity renegotiation through engaging in pedagogical change towards PBL,” European Journal of Engineering Education, vol. 46, no. 1, pp. 116–138, 2021.

[7] L. M. Guimarães and R. D. S. Lima, “Active learning application in engineering education: Effect on student performance using repeated measures experimental design,” European Journal of Engineering Education, vol. 46, no. 5, pp. 813–833, 2021.

[8] S. Grover, R. Pea, and S. Cooper, “Designing for deeper learning in a blended computer science course for middle school students,” Computer Science Education, vol. 25, no. 2, pp. 199–237, 2015.

[9] J. W. Thomas, A Review of Research on Project-Based Learning. San Rafael, CA, USA: Autodesk Foundation, 2000.

[10] A. Robins, J. Rountree, and N. Rountree, “Learning and teaching programming,” Computer Science Education, vol. 13, no. 2, pp. 137–172, 2003.

[11] M. K. Seery, “Flipped learning in higher education chemistry,” Chemistry Education Research and Practice, vol. 16, pp. 758–768, 2015.

[12] L. L. Abeysekera and P. Dawson, “Motivation and cognitive load in the flipped classroom: Definition, rationale and a call for research,” Higher Education Research & Development, vol. 34, no. 1, pp. 1–14, 2015.

[13] C. K. Lo and K. F. Hew, “A critical review of flipped classroom challenges,” Educational Technology Research and Development, vol. 68, pp. 1–30, 2017.

[14] P. A. Kirschner, J. Sweller, and R. E. Clark, “Why minimal guidance during instruction does not work,” Educational Psychologist, vol. 41, no. 2, pp. 75–86, 2006.

[15] R. Lister et al., “A multi-national study of reading and tracing skills,” ACM SIGCSE Bulletin, vol. 36, no. 4, pp. 119–150, 2004.

[16] S. Grover and R. Pea, “Computational thinking in K–12,” Educational Researcher, vol. 42, no. 1, pp. 38–43, 2013.

[17] M. Prince and R. Felder, “Inductive teaching and learning methods,” Journal of Engineering Education, vol. 95, no. 2, pp. 123–138, 2006.

[18] D. Kokotsaki, V. Menzies, and A. Wiggins, “Project-based learning: A review of the literature,” Improving Schools, vol. 19, no. 3, pp. 267–277, 2016.

[19] J. Herrington, T. Reeves, and R. Oliver, Authentic Learning Environments. New York, NY, USA: Routledge, 2010.

[20] J. Trevelyan, “Transitioning to engineering practice,” European Journal of Engineering Education, vol. 44, no. 6, pp. 821–837, 2019.

[21] P. Ardimento, M. L. Bernardi, M. Cimitile, and G. D. Ruvo, “Reusing bugged source code to support novice programmers in debugging tasks,” ACM Transactions on Computing Education, vol. 20, no. 1, pp. 1–24, 2019.

[22] M. Fowler and J. Cusack, “Teaching debugging explicitly,” IEEE Transactions on Education, vol. 64, no. 2, pp. 130–138, 2021.

[23] L. S. Vygotsky, Mind in Society. Cambridge, MA, USA: Harvard University Press, 1978.

[24] D. Wood, J. Bruner, and G. Ross, “The role of tutoring in problem solving,” Journal of Child Psychology and Psychiatry, vol. 17, pp. 89–100, 1976.

[25] E. Bethke, A. Ansari, J. Bradley, J. R. Amos, and H. M. Golecki, “An adaptive scaffolding approach based on team dynamics in an integrated master’s and undergraduate bioengineering capstone design course,” in Proc. ASEE Annu. Conf. Expo., Portland, OR, USA, Jun. 2024.

[26] J. Sweller, “Cognitive load during problem solving,” *Cognitive Science*, vol. 12, pp. 257–285, 1988.

[27] J. Sweller, P. Ayres, and S. Kalyuga, Cognitive Load Theory, 2nd ed. New York, NY, USA: Springer, 2019.

[28] M. Kapur, “Productive failure,” Cognition and Instruction, vol. 26, no. 3, pp. 379–424, 2008.

[29] M. Kapur and K. Bielaczyc, “Designing for productive failure,” Journal of the Learning Sciences, vol. 21, no. 1, pp. 45–83, 2012.

[30] R. K. Yin, Case Study Research and Applications, 6th ed. Thousand Oaks, CA, USA: Sage, 2018.

[31] V. Braun and V. Clarke, “Using thematic analysis in psychology,” Qualitative Research in Psychology, vol. 3, no. 2, pp. 77–101, 2006.

[32] J. Lave and E. Wenger, Situated Learning. Cambridge, UK: Cambridge University Press, 1991.

[33] N. Noddings, The Challenge to Care in Schools, 2nd ed. New York, NY, USA: Teachers College Press, 2005.