Framing: A lens to understand and address student difficulties with Visual Python in University Physics

Hannah M. Kramer and Scott W. Bonham
Department of Physics and Astronomy, Western Kentucky University, 1906 College Heights Blvd., Bowling Green, KY 42101

Computational thinking and modeling are recognized as important skills for educators to help engineering and science students develop. This has led to efforts to integrate computational programming into introductory physics courses, such as the implementation of Visual Python (VPython) as part of the Matter and Interactions curriculum. However, many students have difficulties integrating physics into functional code. Initially, we hypothesized that these difficulties were due to cognitive overload. However, after curricular interventions designed to address that hypothesis were of limited effectiveness, we developed a new hypothesis that focused on how students frame (mentally categorize) their activities, specifically if they are operating out of a "coding frame," a "physics frame," or both. We then tested this new hypothesis through structured observations of groups of students working through VPython exercises, obtaining evidence supporting the framing hypothesis.

2024 PERC Proceedings edited by Ryan, Pawl, and Zwolak; Peer-reviewed, doi.org/10.1119/perc.2024.pr.Kramer

I. INTRODUCTION

A. Overview of the Problem

Computational thinking and modeling are widely recognized as important skills for STEM students and there are many calls for and efforts made to integrate computational methods into introductory physics courses [1,2,3]. However, there are significant challenges in successfully integrating computational modeling in introductory physics, including adding more material to an already-full curriculum, (un)motivated students, and student difficulties developing the skills to integrate physics principles into computer code. We have found all of these to be persistent challenges in our adoption of the Matter and Interactions [4] curriculum in the University Physics I lecture and lab corequisite courses. A key component of the curriculum is the use of Visual Python (VPython) to computationally solve and visualize physics. This work reports on an effort to address student difficulties with VPython. As we implemented interventions and analyzed data, we changed our working hypothesis for the root of student difficulties from cognitive overload to challenges frameshifting between coding and physics thinking.

For many years, cognitive overload was the working assumption for student difficulties with VPython among our instructors, reinforced by a discussion with one of the developers of the Matter and Interactions curriculum [5]. Cognitive load is the "amount of mental activity imposed on working memory" [6, p. 293]. Cognitive overload occurs when the student needs to keep too many pieces of information in working memory to be able to easily complete the task. Our first experimental interventions, which were designed specifically to target issues of cognitive overload, were of limited success. While we hoped to see the experimental group have better outcomes than the control group, the biggest difference was instead due to a large gap in prior coding experience.

A further intervention was put in place to address this issue of coding experience. Analysis from this further intervention led us to develop a different working hypothesis. We observed that the most successful activity was one in which students were guided to "frameshift" multiple times between physics and coding and integrate the two frames. This led us to investigate framing as a theoretical lens for understanding student difficulties and guiding curriculum.

B. What is Framing?

Framing, a theoretical construct borrowed from anthropology and linguistics, describes how students mentally categorize what they are doing [7]. Students are constantly framing the activities they participate in to decide "what portions of information and experience are relevant for completing assignments" [7, p.149]. Therefore, if we want students to activate and build on their knowledge of physics through coding, we need our students to frame activities not only from a "coding frame" (i.e., student recalls proper syntax to use) but also from a "physics frame" (i.e. student references useful physics material from their class notes). It is important to note that sometimes frames "shift easily" while other times they are "resistant to shifts even when challenged" [8, p.102]. Therefore, it can be helpful to initially provide cues to help students shift between and utilize both frames. An example from the context of computational physics is debugging code, which requires students to switch between "doing physics" and "doing code" [9]. This switching helps students see the connection between physics and coding and enables them to use the proper tools to be successful.

Frame switching between "doing physics" and "doing code" was explored by Conlin and colleagues among high school physics students using C2STEM, a block-based computational modeling environment. They used video analysis to investigate how students framed what they were doing as they created computational models using C2STEM. They found that the students were most successful when they were able to synergistically switch between frames of "doing physics" and "doing code," which were particularly frequent when students were debugging code [9]. We expect that the framing lens would be applicable to understanding University Physics students working with VPython.

II. INSTRUCTIONAL CONTEXT

This work was carried out in a University Physics I course at an inclusive regional comprehensive university. The primary population is engineering majors and high school juniors and seniors enrolled in a high school for gifted students. The lecture and lab courses are corequisites which both integrate VPython into instruction using the Glowscript platform [10].

The laboratory component meets once a week for two hours, is capped at 20 students, and is taught by different instructors (along with a Learning Assistant) in multiple sections. Students work in pairs to collect and analyze data and develop simulations of experiments using VPython. Starter code, with commentary, is provided for students to fill in missing physics components. The goal of this structure is that students would work with the physics concepts experimentally, analytically, and computationally.

The lecture component of the course meets four times a week, has 40-50 students, and is taught by different faculty members in a SCALE-UP style classroom [11]. Most class time is spent listening to instructor-given lectures and engaging in peer-instruction questions. Students are assigned readings in Matter & Interactions [4] and online homework. Typically, one class period is dedicated to VPython activities every other week during which students work through a guided VPython exercise using their own laptop computers with assistance from the instructor, group members, and other students. VPython related questions are included on exams.

Students come into University Physics I with varied levels of prior coding experience. It is not uncommon for most students in a given section to be unable to produce working, physically correct code even when the task involves adding just a few lines of code to the provided starter code. This present work is in response to the department prioritizing addressing this wide-spread issue.

III. INITIAL APPROACH: COGNITIVE OVERLOAD

A. Methodology: Interventions in the Laboratory

To address our initial hypothesis of cognitive overload, we implemented two curricular interventions into three laboratory sections in Fall 2022. The first consisted of additional questions on the online pre-lab assignments which focused on ensuring students had basic knowledge of VPython syntax and were aware of common coding mistakes. The second consisted of worksheets integrated into student laboratory manuals that led students step-by-step through translating physics principles from algebraic expressions to VPython code. Students were asked to first write out physics expressions algebraically in one box and then translate it to VPython notation in the next one so that they would be able to focus on just the physics or the coding at a time. By breaking the task down, we hoped to reduce the likelihood of students experiencing cognitive overload.

The three experimental sections were compared to two control sections without the interventions using data from a survey on student perceptions, completed approximately a month into instruction, and additional metrics that will not be further discussed here. The primary finding was that previous coding experience was the dominant factor in student success, regardless of treatment type. For data analysis purposes, "students with coding experience" refers to students who have taken at least one coding class or have engaged in original coding activities. 27 students were categorized as having prior coding experience and 23 as without coding experience. 10 students were excluded from the analysis because it was unclear how significant their prior exposure to coding was.

B. Data and Analysis

The impact of the discrepancy between those with and without prior coding experience can be seen in Fig. 1 and Fig. 2. In Fig. 1, only 1 student (4%) without prior coding experience agreed that they felt confident in their ability to complete VPython assignments and no students strongly agreed. Meanwhile, 71% of students with prior coding agreed (41%) or strongly agreed (30%) with the statement.

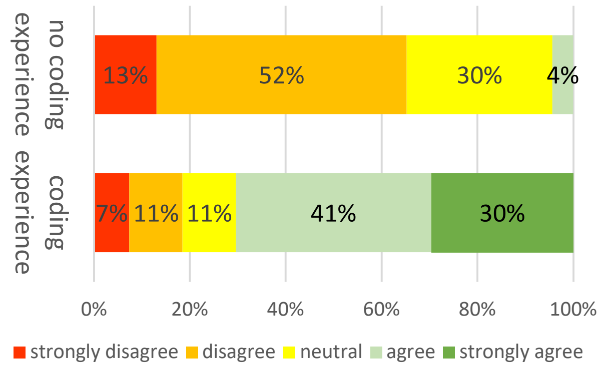

FIG. 1. Student responses to "I feel confident in my ability to complete the VPython assignments"

Figure 2 shows student responses on a free-response question, "Which parts of VPython have you learned the most from?" This was coded into four emergent categories: mechanics, visualization, an explicit statement of "nothing," and blank or "I don't know."

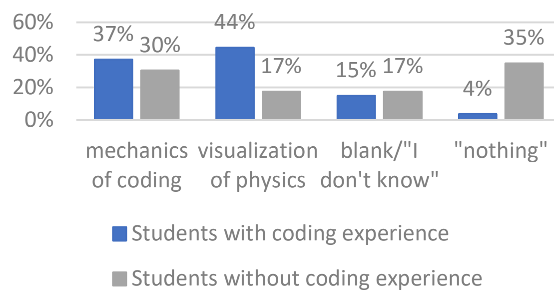

FIG. 2. Aspects of VPython students reported learning most from

Of students with no prior coding experience, 52% of the responses were "nothing" or blank/"I don't know," while only 19% of students with coding experience fell into these categories. Students without prior coding experience expressed less confidence in their ability to be successful and less perceived benefits of using VPython.

Despite the use of interventions aimed at targeting cognitive overload through scaffolding, there was little difference between control and experimental sections. Instead, prior coding experience was a better predictor of positive outcomes. This finding led us to realize that there was a larger issue than just cognitive overload that needed to be addressed. This led to further interventions in the Spring 2023 semester.

IV. FURTHER INTERVENTIONS

A. Methodology: Interventions in Lecture

As a result of these findings, we believed a more comprehensive intervention would be necessary to provide enough assistance for students lacking coding experience. A set of exercises was developed and implemented into one lecture section for students to work through during VPython sessions. The exercises included worksheets where students were guided through the activity through a series of 15-20 questions. These consisted of (1) an introductory activity introducing objects and while loops, (2) projectile motion, (3) gravitational force between the Earth and moon, (4) Tarzan on a vine (simple pendulum), and (5) conservation of energy, in which Activity 2 was revisited but with an energy focus. Students were provided with starter code and worksheets for each activity. Worksheets used a similar template to lab to guide students in identifying the physics and then translating algebraic expressions to code. Additionally, on several of the assignments, students were then asked to use their code to interpret the physics involved. The Tarzan activity included follow up questions where students were asked to modify their simulation and observe the effects. On the energy problem, students were guided to realize and correct the situation in which the initial time step was too large, leading to the accumulation of enough numerical error that the energy graphs appeared to show energy was not conserved. Student work on VPython assignments along with their responses on a survey given out approximately a month into instruction were collected and analyzed. Of the 48 students in the course, data collection was limited to the 15 students who agreed to the terms of the informed consent.

B. Results

A review of student worksheets and the Glowscript code on the in-class VPython activities demonstrated an unexpected disconnect between being able to produce functional code and understanding the physics involved. On the Tarzan assignment, some students worked in groups and others individually. All but one of the groups and individuals submitted a properly functioning code. Despite high success achieving a functioning code, the majority (4/6) submitted worksheets with physical errors (such as incorrectly stating that changing the mass would affect pendulum amplitude).

On the conservation of energy assignment, all students from whom we could collect data submitted properly functioning code. Additionally, most of the students (11/14) were able to provide on the worksheet a basic explanation that the source of the error was a coding issue, i.e. too large of a timestep introduced too much numerical error. This included several students during the activity changing their initial idea as to whether it was a coding or physics issue.

When asked the same survey questions, no students responded with "I don't know" or "Nothing" to the question, "Which aspect of VPython have you learned the most from?" One student added an explanatory note in the margins of their survey stating that "In lecture [VPython] is beneficial, in lab it's a chore that is never explained coherently."

V. REVISED HYPOTHESIS: FRAMESHIFTING

The varied response and success of students on different similarly scaffolded assignments caused us to reconsider our original theoretical framework. We noticed our most successful assignment, the energy assignment, used wording and structure that promoted frame shifting much more than other assignments, as the assignment placed emphasis on the debugging process. For example, the worksheet asked students "What do you think may be some possible causes for this unexpected behavior? Do you think this is more likely to arise from the physics itself or the computational methods we are using?" When students take the time to think about the source of the unexpected behavior, there is a key moment of synergy between coding and physics frames [9].

Meanwhile, our least successful assignment, the Tarzan assignment, used wording that may have cued students to shift *away* from the coding frame. For example, when students were directed to change various initial parameters to explore how it affected the behavior of the pendulum; two-thirds of the worksheets reported erroneous results. Although the students were supposed to use their computational model to think about physics, the section title of "Analyzing the physics," likely cued students to exclusively utilize a physics framework. This resulted in them seeing what they expected to see and not playing around with their code by changing the initial parameters, which would have integrated the two frames. This implies that students were not using their code to test and refine their physics knowledge but were instead simply relying on their (common sense) physics knowledge, (e.g., stating that changing the mass would affect the period of the pendulum).

This sharp contrast between the two assignments led to our new hypothesis: assignments that help students integrate and frameshift between physics and coding are more beneficial for student success with Visual Python activities.

A. Methodology

During the Fall 2023 semester, this new hypothesis was tested through observing groups of students working through class exercises. These were the same exercises except some of the wording was revised, in particular wording that might have discouraged frame shifting. During class time in which the whole class was working on VPython assignments, the PI (first author, who is not an instructor in the course) would pull out a group of three students to a nearby classroom and ask them to participate in a "think aloud procedure." The observational protocol utilized specified that, at two-minute intervals, the PI would record the question students were currently on, the frame the students appeared to be currently utilizing (coding, physics, a mixture of the two, or neither) and where the students' attention was focused (the computer screen, the worksheet, engaged in discussion, asking questions, or off task). The PI determined the utilized frame(s) by listening to student conversations and their externalized speech during the "think aloud" procedure. (For example, if students were talking about physics principles and ideas, it was coded as "physics.") Students were selected for group sessions from a pool of 12 students who provided contact information on the informed consent document for the overall class data collection. All observations were performed by the same researcher to ensure consistency due to the restriction of being unable to record observations.

B. Observation Data and Analysis

Figure 3 compares the framing of the questions on the exercise students were most successful with (energy conservation) to the exercise students were least successful with (pendulum, or Tarzan on the vine).

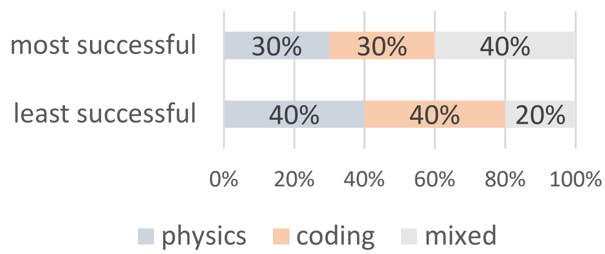

FIG. 3. Percentage of questions coded as "physics," "coding," or "mixed" for most and least successful assignments

While the number of questions coded as "physics" was equal to that coded as "coding" within each assignment, the percentage of questions that were "mixed" was double for the more successful energy assignment compared to the pendulum assignment.

VI. DISCUSSION AND CONCLUSIONS

Our data analysis, in conjunction with prior research [9], has led us to believe that framing is a useful lens to analyze student performance and perceptions on these types of activities, in particular to what extent students are switching, or not switching, between the frames of "doing physics" and "doing code" [9]. Our data indicates that activities that closely integrate the two frames are both more effective for learning and help students better appreciate the value of such exercises. However, our quantitative analysis is limited due to the small sample size and our qualitative analysis is limited due to only having two researchers in our group.

A. Student Success

Students who lacked prior coding experience reported having learned less from the VPython activities when compared to their more experienced peers (Fig. 2). Clearly students with less coding experience would have a harder time shifting between a physics frame and a coding frame. Ideally shifting frames would be a fluid process but other times it can be "sticky" [8]. Students learning how to code for the first time generally would need to give their full focus to that aspect of the task, interfering with their ability to switch frames. It is reasonable that they would get "stuck" more often.

Providing students with cues to guide their frameshifting can help increase student success—especially those who are new to coding. As anticipated, the assignment with the highest level of student success also had the highest percentage of questions coded as "mixed" (40%) between physics and coding—twice as much as the least successful assignment.

B. Student Perception & Motivation

Student perception of their ability and the value of VPython assignments can also be understood in terms of framing and shifting between physics and coding. Students with prior coding experience were confident in their ability to complete VPython assignments in lab (Fig. 1). While many of these involved different programing languages, learning to code could develop a transferable coding frame and provide experience in frame shifting—which students without prior coding experience lack.

It appears that in many cases students tended to view VPython activities entirely through a coding frame, undermining the instructional goal of learning physics through VPython. This is exemplified by a student that said, "I don't need this in my future, and am not a 'software' engineer." A different student commented on the laboratory VPython activities, "the end result tends to be fairly simplistic and doesn't end up really telling me much about the physics involved." This is perhaps in contrast, or perhaps reinforced by the student who went beyond expectations to inform us, "In lecture it is beneficial, in lab it's a chore that is never explained coherently," as mentioned previously.

All these perceptions make sense through the lens of framing and differences between how VPython activities are structured in lecture and laboratory. As already discussed, many of the lecture activities, to different degrees of success, integrated physics and coding and therefore encouraged frame shifting. During the laboratory activities, in contrast, students first collect and analyze data using graphical and analytical tools. Only after they complete that do they engage in coding, making it easy to exclusively use a physics frame for the first part and coding frame for the second. Thus, it is not surprising that students perceive the laboratory VPython activities as irrelevant for their future careers or learning physics. In contrast, activities that encourage integration of physics and coding frameworks led them to reinforce each other, promoting student success. Exercises that encourage frame integration can help students see why VPython is relevant to the physics course and develop capacity to shift frames between computational modeling and another frame, a skill of growing importance in STEM fields.

ACKNOWLEDGMENTS

This project was sponsored by Western Kentucky University through a Faculty-Undergraduate Student Engagement Grant (project ID #22-FA212).

REFERENCES

[1] Y. Li, A.H. Schoenfeld, A.A. diSessa, A.C. Graesser, L.C. Benson, L.D. English, and R.A. Duschl, On Computational Thinking and STEM Education, Journal for STEM Education Research **3**, (2020), pp.147-166.

[2] N.V. Mendoza Diaz, R. Meier, D.A. Trytten, S.Y. Yoon, J.M. Moore, A.M. Ogilvie, M. Weichold, Computational thinking in the formation of engineers (year 1), in *Proceedings of the 2021 ASEE Annual Conference & Exposition*, (2021).

[3] National Research Council, *A Framework for K-12 Science Education: Practices, Crosscutting Concepts, and Core Ideas.* (National Academies Press, 2012), pp. 64–67.

[4] B. A. Sherwood and R. W. Chabay, *Matter & Interactions (4th ed.)* (John Wiley & Sons, Inc., 2015).

[5] R. Chabay (private communication).

[6] P. Eggen and D. Kauchak*, Using Educational Psychology in Teaching* **11**, (Pearson Education, 2020), p.293.

[7] R.E. Scherr and D. Hammer, Student behavior and epistemological framing: Examples from collaborative active learning activities in physics, Cognition and Instruction **27**, 149 (2009).

[8] D. Hammer, A. Elby, R. E. Scherr, and E. F. Redish, Resources, Framing and Transfer, *Transfer of Learning from a Modern Multidisciplinary Perspective*, edited by J. P. Mestre (Information Age Publishing Inc., 2005), p. 102.

[9] L.D. Conlin, N. M. Hutchins, S. Grover, and G. Biswas, "Doing physics" and "doing code": Students' framing during computational modeling in physics," in *Proceedings of the 14th International Conference of the Learning Sciences*, Nashville, Tennessee, (2020), pp. 733-736.

[10] R. Chabay, D. Scherer and B. Sherwood, Web VPython v.3.2, https://glowscript.org/

[11] P.A. Soneral and S.A. Wyse. A SCALE-UP Mock-Up: Comparison of Student Learning Gains in High- and Low-Tech Active-Learning Environments, CBE Life Sciences Education **16**, 1 (2017).