Student Predictions Of Functional But Incomplete Example Programs In Introductory Calculus-Based Physics

Shawn Weatherford[*] and Ruth Chabay[†]

[*]*Department of Mathematics and Sciences, Saint Leo University, St. Leo, FL 33574*
[†]*Department of Physics, North Carolina State University, Raleigh, NC 27695*

Abstract. Computational activities for Matter & Interactions were redesigned to focus student attention to the interpretation and prediction of functional, but incomplete example programs before modifying the program with additional program code. Evaluation of the instructional activities in experimental lab settings reveals student predictions that are based on the expected dynamics of real-world systems in addition to the information gathered through the interpretation of the example program code.

Keywords: physics education research, computation, modeling
PACS: 01.40.-Fk, 01.40.G-, 01.50.H-, 01.50.Lc

INTRODUCTION

The Matter & Interactions[1,2] curriculum for introductory calculus-based physics students integrates computational programming exercises using VPython[3] to support the conceptual development of the application of fundamental physics principles[4]. Prior research investigated the effectiveness of computational activities regarding how students construct the computational algorithm[5] and use as student homework assignments[6].

Observations of physics students working on computational assignments in collaborative groups during lab sessions revealed missed opportunities for students to connect the end product with the motions of analogous real-world systems.[7] The computational activities were redesigned to (1) focus student attention on the connection between the application of physics principles and the constraints of motion in the visual output and (2) to help students make connections between the visual output and the predicted motions of real-world systems.

MINIMAL WORKING PROGRAMS (MWP)

Previous iterations of computational activities asked students to take a shell program and fill in missing lines of code, free from errors, to produce a program that runs to generate a visual output to display a specified result. Any coding errors would have to be resolved before the program would display the visual output. Therefore, students tend to use the visual output screen as an indication that their program was free of these coding errors and would move on without interpreting the visual output to evaluate the validity of the physics principles added to the program. For example, students may not interpret a visual output displaying the effects of repulsive gravity as an indication that there is an error with the formulation of the algorithm used to calculate the gravitational interaction between two objects. The activity relied on experienced teaching assistants to observe and question student groups who failed to make the connection between the visual output and the program code.

Motivated by this specific instructional issue and observations from an earlier pilot study, the activity was redesigned to have students begin from an example program that will run without any changes to the program code. The program is not complete in the sense that lines of code representing fundamental physics principles, calculations of interactions between 3D objects, or both are strategically omitted from the program. Figure 1 shows the Space Voyage MWP code and the visual output displayed if you were to run the program in a VPython editing environment. Notice that due to the omission of lines of code defining the gravitational interaction between the `craft` and `Earth`, as well as the line of code using the interaction to update the craft's momentum, the spacecraft travels at constant velocity in the $+y$ direction across the visual scene.

Instructional Tasks Using An MWP

First, students work together to read the program code to negotiate an agreed interpretation of the code, including the code's function in the visual output. Then students are asked to draw a prediction on a

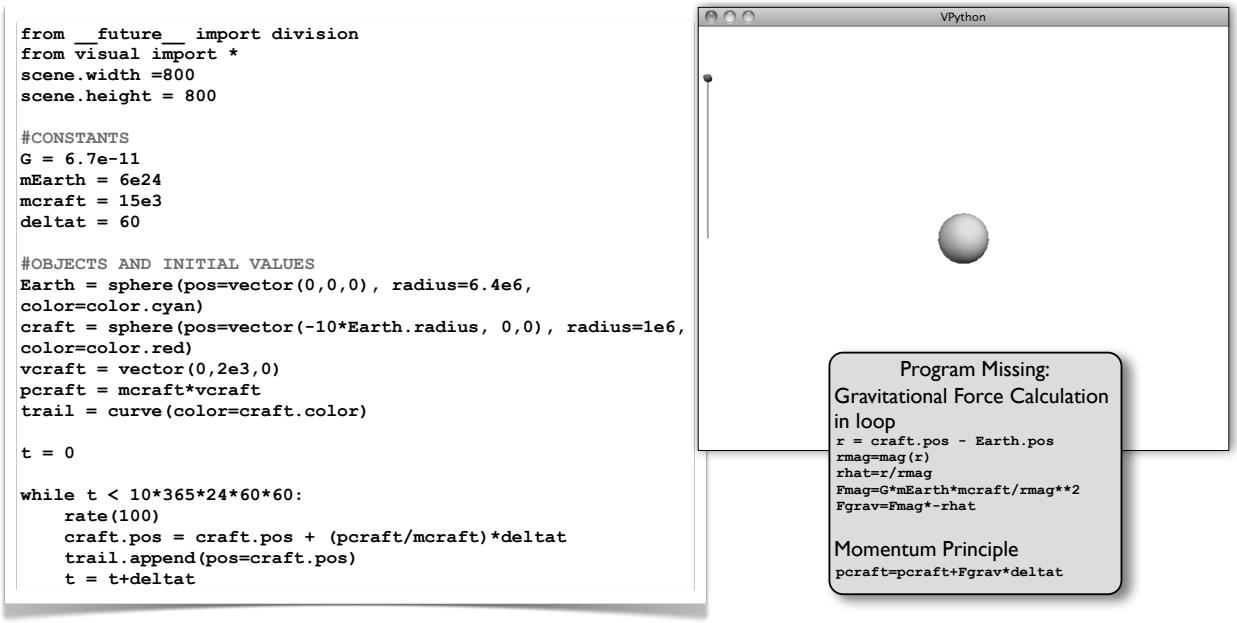

FIGURE 1. The program code, visual output, and listing of the missing lines of code for the Space Voyage Minimal Working Program.

whiteboard illustrating what they expect to see when they run the program for the first time. Only after the whiteboard prediction is complete are the students permitted to run the MWP program and observe the visual output. The whiteboard illustration serves as an artifact representing the group's interpretation of the MWP and students compare the whiteboard prediction to the visual output displayed before them. Students discuss any differences between the visual output and their prediction. Finally, students collaborate to determine what is missing from the program code and construct the lines of program code representing missing physics principles or interactions.

METHODOLOGY

The MWP activity was piloted in the 1st author's lab sections and later deployed in an experimental lab environment developed to closely monitor group conversations, computer input and output, and drawings on whiteboards[8]. Two experimental lab sections, each capped with an enrollment of twelve, were offered along with traditional lab sections for Matter & Interactions students during the Fall 2009 semester Twenty-three students voluntarily enrolled and provided consent to participate in the evaluation of the MWP activities. Students were assigned to collaborative groups for all labs. The 1st author served as the teaching assistant for the two experimental lab sections.

Three computational activities were modified to make use of the MWPs and follow same sequence of tasks described above. Students worked on the first MWP activity during lab week 5, following weeks of activities introducing VPython and basic computational concepts. This paper reports the data collected for the experimental lab sections while students completed the reading and prediction tasks using the Space Voyage MWP. This data consists of six episodes of collaborative group discussions as students completed the interpretation and prediction tasks. Transcripts of each episode were segmented and coded to determine (1) which line of program code a speaker is speaking about, (2) the nature of the interaction with the program code, (3) and the type of information the speaker brings to the discussion during the interpretation task and the prediction task. Each coding scheme met reliability and validity standards.[9]

RESULTS

Examining the coding data across all episodes reveals varying levels of program comprehension, use of physics knowledge to complete the instructional tasks, and communication among group members. Groups chose two different approaches to complete the read and predict tasks with five of the six groups making predictions of the visual output as the group reads through the program code together. For this paper, two illustrative episodes are presented and discussed, both combining the read and prediction tasks.

Episode 1: Two Possibilities

Surprisingly, students did not find it necessary to decide upon a single prediction at runtime. Three of the six groups ran the program with more than one prediction for the motion of the craft. This episode begins with Otis[10] interpreting the function of the lines of code used to draw 3D objects in the visual output and begins to draw filled-in circles representing the Earth and craft on the whiteboard, as shown in figure 2. Otis correctly identified and applied the information in the program to sketch the relative location of the two 3D objects.

FIGURE 2. Whiteboard prediction showing the location of the Earth at the center of the board with the craft to the left of the Earth. An arrow pointing towards the top of the whiteboard is attached to the craft.

Then Fernanda begins to interpret the line of code defining the upward initial velocity of the craft:

Fernanda: It would be, craft would be up, wouldn't it?
Ramon: Yeah…
Ramon: and it's velocity is going straight up,
Ramon: **so its going up**…
Otis: [Draws up arrow at location of spacecraft]
Otis: I'm going to run the program [laughs]
Fernanda: yeah [laughter]
Ramon: Ah, then it's multiplying by days, hours…
Otis: **So yeah it's got to be going around**…right?
Otis: or… I dunno.
Ramon: I thinks it's gonna…
Ramon: I don't think, **it's not curving around it, it's just shooting straight up**…
Ramon: **so I think it's going to keep going up**…
Otis: okay
Ramon: But you think it's gonna curve…
Ramon: it might curve… I dunno.

Notice at runtime that two predictions exist. Both predictions agree that the craft initially moves in the $+y$ direction with Otis's prediction adding that the craft moves around the Earth. Although Ramon's prediction is correct, it is not clear that the reasoning developed from a careful analysis of the existing program code or from fundamental physics principles. Otis's prediction states an expectation that the craft moves through the visual scene in a manner that is consistent with the real-world interactions between the Earth and an orbiting spacecraft. Later, the group acknowledges Ramon's correct prediction and moves on to the modification task where the goal of getting the craft to orbit the Earth emerges from the group discussion, beginning with focusing on coding a line to calculate the gravitational force acting on the craft.

Episode 2: Evidence-Based Prediction

One of the six groups justified their prediction from their interpretations of the program code.

Norma: Oh yeah, let's draw that vector…
Norma: it's the velocity,
Lidia: right
Norma: That's the direction of its velocity, so just draw an arrow going up [draws upward arrow]
Norma: yeah…
Lidia: and the momentum…is the mass times the velocity, obviously
Lidia: trail…makes a curve?
Norma: curve…
Lidia: **so obviously it's going in a curve**
Norma: you wanna…
Lidia: maybe it's gonna curve around the earth…[gestures in a circle]

Lidia reads the 3D object called `curve` and interprets the function as a command to the craft to follow a curved path. The group recalls seeing this in action in lecture and corrects the misinterpretation of the function of the line of code to, as Norma states, "gonna show that where its been." That is, the curve object shows previous locations of the craft in the visual output. Finally the group revisits the prediction:

Dora: like [gestures in a circle with marker]
Norma: yeah…
Norma: **cause we've got gravitational constant and stuff…**
Dora: um hum um hum…
Norma: Okay I think that's a good…
Dora: Run the program … [Presses F5]

Norma justifies her prediction with the appearance of the symbol or numerical value representing the gravitational constant near the beginning of the program. Although this is an incorrect interpretation of the function of the constant, it represents a shift away from predicting orbital motion based on a command to follow a curved path. This physics quantity will be used to formulate the calculation of the gravitational

force during the program modification task. Later, while the group views the visual output and recognizes that the craft continues moving in the $+y$ direction, Norma inspects the program code again and connects the visual output to the program code, saying, "I guess there's nothing in the program about the force from the Earth."

DISCUSSION

Running the program served two purposes in the aforementioned episodes. During episode 1, the visual output was the referee in deciding which prediction was correct. Two other student groups not reported in this paper run the program with two competing predictions: one prediction matching the visual output and one prediction matching the real-world dynamics of a spacecraft-Earth system.

During the second episode, the visual output revealed a misinterpretation of the program code with students looking back to the code to determine why their interpretations were incorrect. Three of the six student groups justified the straight-line motion of the craft observed in the visual output with the omission of program code. Norma's group and one other group mentioned the failure to use the gravitational constant in a way to have an effect on the motions of the craft.

None of the groups discussed the omission of the momentum principle in the while loop. Possible explanations include the difficulty in recognizing what code is omitted while at the same time interpreting the code that is present in the program. Although two groups acknowledged the presence of the position update line in the computational loop, these groups, like all others, interpreted the motion of the spacecraft from the initial value of the craft's velocity. An alternative explanation is based on the evidence that students blended the read and prediction tasks together, interjecting a prediction of the visual output before reading the program code thoroughly. Students who made predictions while working through a comprehension of the program stopped reading the code once they had enough information to generate a prediction about the motion of the craft. Therefore, these groups ended work on the read/predict task before discussing the loop structure.

Since the predictions blended interpretations of program code (locations of 3D objects and initial velocities) with the dynamics of real-world systems (orbits), the prediction task was revised to include drawing a whiteboard prediction of the analogous real-world system directly beside a prediction of the visual output informed by the program code. Instructors who use these activities report that it helps students consider the possibility that the MWP may produce an output that is different from the real-world environment.

Most surprisingly, students interpret the 3D curve object as a command to direct the craft to follow a curved path through the visual scene. Considering the consistency of this misinterpretation, developers of VPython have added attributes of 3D objects called `make_trail` to replace the functionality of the 3D curve object.[11]

The Space Voyage MWP computational activity and two more MWP activities modeling 3D springs and scattering experiments are included in the instructor's resources for adopters of the Matter & Interactions textbook. Data from this same cohort of student participants working on these two MWP activities later in the semester shows improvement in discussing interpretations of the computational loop.[9] There is ongoing research investigating how students modify the MWP example to produce a working program that includes all missing physics principles.[12]

ACKNOWLEDGMENTS

This work was financially supported by National Science Foundation (DUE-0618504). The authors would like to thank the contributions from former and current members of the NC State PER&D group, the NC State STEM Initiative, and the participants of this study.

REFERENCES

1. R.Chabay and B. Sherwood. *Am.J.Phys*. **72** 439-445 (2004)
2. R. Chabay and B. Sherwood. *Am.J.Phys* **74** 329-336 (2006)
3. see, http://www.vpython.org
4. R. Chabay and B. Sherwood. *Am.J.Phys* **76** 307-313 (2008).
5. M. Kohlmeyer. "Student performance in computer modeling and problem solving in a modern introductory physics course" Ph.D. Thesis, Carnegie Mellon, 2005
6. M. Caballero "Evaluating and extending a novel course reform of introductory mechanics" Ph.D. Thesis, Georgia Institute of Technology. 2011
7. A. Buffler et. al., *Am.J.Phys* **76** 431-437 (2008).
8. http://www.ncsu.edu/physics/physics_ed/facilities.html
9. S. Weatherford "Student use of physics to make sense of incomplete but functional VPython programs in a lab setting." Ph.D. Thesis, North Carolina State University, 2011
10. all participant names in this paper are pseudonyms.
11. make_trail included in Visual 5.70 and later
12. B. Lunk "A framework for understanding physics students' computational modeling practices" Ph.D. Thesis, North Carolina State University, 2012