Is It Feasible to Incorporate VPython Programming into Introductory Physics? A Case Study at Bridgewater College

Deva O'Neil

BRIDGEWATER COLLEGE

Introduction

The AAPT recommends incorporating at least three types of computational tools into undergraduate physics programs.[1] Bridgewater College aims to familiarize physics students with Excel, Mathematica, and at least one language suitable for scientific programming. For the past two years, our introductory physics class has incorporated programming in VPython. The students in this class complete a survey at the end of the first semester.

Research Goals

- Assess student comfort level and perceived value in programming assignments
- Identify obstacles or challenges in incorporating programming into introductory physics course

Programming Environment:

Glowscript (web-based VPython)

Student Population:

Physics, Biochemistry and Chemistry Majors

Total Class Time Used (first semester):

Recitation (5 hours), Lab (2 - 3 hours)

Benefits Reported by Students

"It was helpful for deriving equations and seeing the physical results of manipulating certain numbers"

"It helps me visualize how changing a variable's value affects the movement / force."

Abstract

In 2016, students in General Physics at Bridgewater College, most of whom had no previous programming experience, were introduced to programming in Glowscript. In homework and in class, students simulated objects moving under the influence of forces.

A written survey probed student comfort level with the programming activities (N = 30), with mixed results. While most students felt adequately prepared, a significant minority of students expressed frustration with the coding assignments.

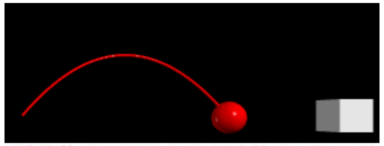
In response, two changes were implemented in the 2017 course offering: more code templates were provided to students, and in-class activities replaced the homework. In this cohort (N = 22), only one student reported feeling confused. Overall, the majority of students found the computational aspect helpful.

VPython

An extension of python designed for visualization of objects in motion.

VPython Code Sample

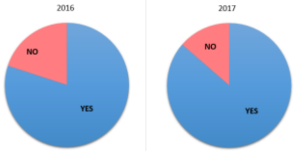
Students simulate launching a ball. Initial conditions are adjusted until target is hit. Motion is produced iteratively in a while loop.



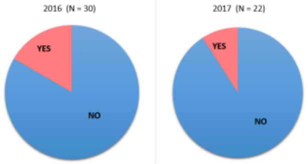
```
while(ball.pos.y >= 0): #stop when ball hits the ground
    rate(100)
    p = p + Fnet*delta_t #update momentum
    ball.pos = ball.pos+(p/m)*delta_t #update position
    t = t + delta_t #update time
```

Student Responses

Did you feel that the programming aspect of this course helped your learning of the physics content?



Would you have needed more previous programming experience to meet the expectations of the assignments?



Student Perceptions of Cost/Benefit

Frequently-cited Benefits

- Visualizing the physics
- Gaining programming experience
- Practice applying the equations

Frequently-cited Challenges

- Learning curve required for novice programmers
- Too much time spent learning programming
- Frustration during debugging (especially when working on assignments at home)

Conclusions

Improving the efficiency of student programming activities by doing them in-class and providing templates was helpful in reducing student frustration. Perceived value remained highly variable, but overall students found the activities helpful.

Recommendations

- Use class time for programming to ensure students have access to the instructor while developing and debugging their code.
- Encourage concurrent or prior enrollment in a programming course
- Participate in PICUP (Partnership for Integration of Computation in Undergraduate Physics[2])
  - Offers summer workshops
  - Reviews exercise sets and templates and makes them publicly-available

References & Suggested Resources

[1]AAPT Recommendations for Computational Physics in the Undergraduate Physics Curriculum, 2016 (aapt.org)

[2]PICUP (gopicup.org),