Dickinson College
Dickinson Scholar

VPython for Introductory Mechanics

2019

01 Introduction

W.A. Morgan
Dickinson College

L.Q. English
Dickinson College

Follow this and additional works at: https://scholar.dickinson.edu/vpythonphysics

 Part of the Astrophysics and Astronomy Commons, Curriculum and Instruction Commons, Numerical Analysis and Computation Commons, Physics Commons, and the Science and Mathematics Education Commons

Recommended Citation
Morgan, W.A. and English, L.Q., "01 Introduction" (2019). *VPython for Introductory Mechanics*. 11.
https://scholar.dickinson.edu/vpythonphysics/11

This Book is brought to you for free and open access by Dickinson Scholar. It has been accepted for inclusion in VPython for Introductory Mechanics by an authorized administrator of Dickinson Scholar. For more information, please contact scholar@dickinson.edu.

Introduction

This book is intended to supplement the course materials of a first-semester introductory course in physics by adding a computational dimension into the course content through the use of VPython - a "visual" extension of the *Python* programming language that incorporates 3D animation and vector math. The book is not designed to replace the regular course textbook. Since it is intended to be used in conjunction with other material presented in the course, one of the goals was to make its integration into the course flow as seamless as possible. To this end, we have organized the topics and exercises so that they should more or less follow the traditional sequencing of topics encountered in most introductory mechanics courses.

The exact order of topics was chosen to enable its easy integration into our *Workshop Physics* curriculum.[1] While the pedagogical approach, format and presentation of Workshop Physics is very different from the more standard lecture-based course, its content selection is fairly conventional. Thus, we believe that this presentation should also work well in conjunction with a more traditional introductory mechanics course (either at the senior high-school or first-year college level). Furthermore, swapping neighboring chapters or omitting certain sections should create only minimal disruption.

Having said that, introducing computation in the first semester does allow for some curricular innovation, such as in the form of inclusion of types of problems that are not typically encountered otherwise. In particular, it allows students to examine problems that have no closed-form analytical solutions, to engage in more creative and open-ended problems,[2] and to practice physical modeling. One example we include in this book has to do with the law of universal

[1] *Workshop Physics* was developed at Dickinson College, principally by P. Laws, as an inquiry-based, active-learning curriculum for introductory physics

[2] E.F. Redish and J.M. Wilson, "Student programming in the introductory physics course", *Am. J. Phys.* **61**, 222 (1993).

gravitation. After a sequence of exercises where students discover Keplerian orbits, we ask them to explore what would happen to planetary orbits if gravity were instead a "1/r" kind of force. Such a question would be quite difficult to answer analytically, and would require an upper-level arsenal of mathematical tools, but intro students can tweak the code very easily to explore this scenario.

Thus, the use of VPython has the pedagogical virtue of empowering students to engage in more physical modeling, and it enables them to better visualize processes, both of which in turn strengthen students' conceptual understanding.[3] Finally, a very straightforward benefit for introductory students is that they are exposed to a language, Python, that is fast becoming the premier tool for scientific computation.

The structure of the book is as follows. We start in Chapter 2 with some preliminary exercises to get students familiar with the interface and basic coding in VPython. In Chapter 3, we then turn our attention to one-dimensional motion and motion diagrams. Quite a lot of emphasis is placed in Workshop Physics, as well as *RealTime Physics*,[4] on describing basic motion in multiple representations: verbally, in graphical form, and with mathematical equations. Physics education research has shown that real conceptual and problem-solving progress occurs when a student can move fluidly between these various representations of motion.[5] The idea in this chapter is to bolster students' comprehension of position- and velocity-graphs, which are one layer removed in abstraction from the literal description of the motion, by having them program an object performing a simple kind of motion on the screen and then

[3] R. Chabay, B. Sherwood, "Computational Physics in the introductory calculus-based course", *Am. J. Phys.* **76**, 307 (2008).

[4] D. Sokoloff, R. Thornton, P. Laws, *RealTime Physics - Active Learning Laboratories* (John Wiley & Sons, 2012).

[5] A. Van Heuvelen, "Learning to think like a Physicist: A review of research-based instructional strategies," *Am. J. Phys.* **59**, 891 (1991).
L.C. McDermott, "A View from Physics" in *Toward a Scientific Practice of Science Education* (Lawrence Erlbaum Assoc., 1990).

having VPython generate the corresponding motion graphs. In a sense, it is the computational analogue to the common laboratory exercise of using a motion sensor and walking in front of it.

In Chapter 4 we assume the course has by now advanced to a discussion of average and instantaneous velocity and acceleration. (In a traditional intro course, this may come very quickly.) We use some of these ideas to introduce the basic Euler method at the heart of all numerical simulation encountered here.

In Chapter 5, we return briefly to the realm of plotting formulas, as the course is now at the point of exploring projectile motion, i.e., two-dimensional motion under constant acceleration. This topic is naturally suitable for adding a computational facet. Exact analytical results (such as the range formula) can be easily tested numerically, for instance, as outlined in the exercise section of this chapter. Then, additional complications can be added (such as a sloped ground). We end by having students code a "hit-the-target" game.

Chapter 6 in many ways strikes at the heart of mechanics. We are finally ready to simulate Newton's laws, and the full power of computational physics is revealed for the first time. We can see the kinds of motion that Newton's laws actually give rise to, given the forces acting on a particle and the particle's initial state. We specifically highlight three important problems in order of increasing difficulty: the 1D mass-spring problem, projectiles traveling through air, and the orbits of planets and comets. What makes this approach so potent in the context of an introductory course is that the analytical solutions are too mathematically challenging to derive and therefore remain beyond reach for this audience. Yet, the numerical pathway is accessible.

For Chapter 7, it is assumed that the course has now reached a discussion of momentum and energy. We chose the binary star system

as a vehicle for introducing the concept of center-of-mass and to tie it to momentum conservation. We then return to the mass-spring problem to illustrate the conservation of mechanical energy in this system. The Rutherford experiment can be skipped, but it provides a nice preview to electrostatics.

Maybe unsurprisingly, we now follow up with rotational motion and angular momentum, which often represents the last material covered in the semester. We end with what we believe to be two engaging final examples - the gravitational "slingshot" and chaotic orbits - as a kind of capstone project for students to explore.

Lastly, before we get started, all the VPython codes featured in this book can be accessed online at
https://scholar.dickinson.edu/vpythonphysics/ .