Department of Physics
Umeå University March 24, 2020

# Object Oriented Programming For Physics Students
# **Bungee Boy**

**Author:** Felix Djuphammar

**E-mail:** fedj0002@student.umu.se

**Guide File:** BungeeBoy.ipynb

A Python guide using Anaconda

# 1 Introduction

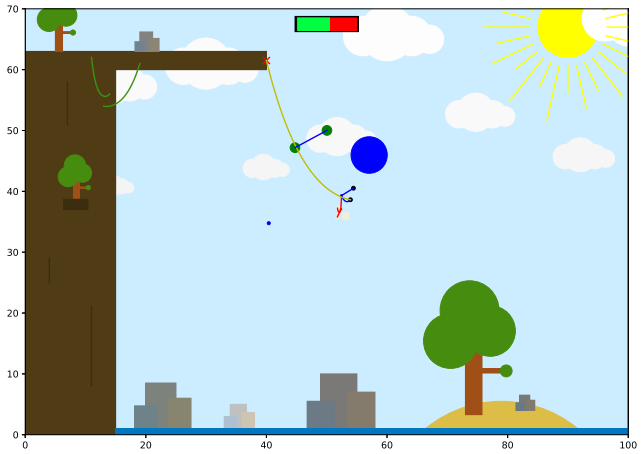
This document introduces the bungee jumping with Python guide by explaining its contents using some examples, showing how to install the program you need and how to use the guide once it's opened.

# 2 Quickstart

1) Download and install Anaconda (Python 3.7 or higher)
2) Open Anaconda Navigator, then click on Jupyter Notebook
3) Download BungeeBoy.ipynb from the location you got this .pdf
4) Open the .ipynb file from the Jupyter Notebook browser window

# 3 The Guide

The guide is written in a Jupyter Notebook. This environment allows having instructions and executable source code in one single document. The document takes you on a magical journey from creating simple shapes and simulations to creating a very vivid and realistic simulation as in the snapshot below.



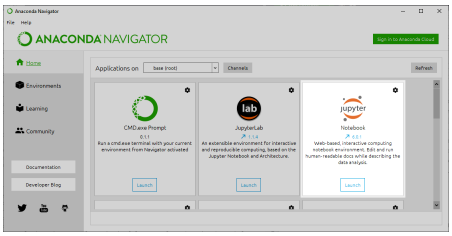

The idea is not that you follow every step in the guide to understand every bit of code, as it is a bit extensive. Instead, you can skim through the contents, reading some of the text bits and looking at the simulations to get a sense of how the guide progresses, so that you quickly reach the end. At the end, you can modify the scenery in which the simulation takes place by adding/modifying objects. You can then have some fun with the code and get more familiar with both the code and the notebook environment while

getting your imagination going. You might also run into some quirks and glitches, which is good practice to manage as well.

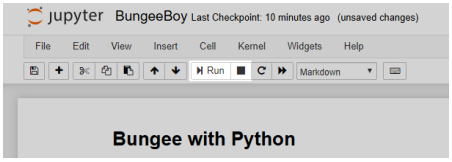
After spending a few minutes/hours/weeks giving your jumper the time of their life you can go back through the guide to read up on the bits you find interesting, simply ignoring the stuff that seems a bit too boring to deserve your time. Modify the code snippets and run the next following simulation to see the effects and learn how the code works. Then if you want, try to modify or add stuff your imagination tells you is a good idea. If you need some inspiration, check the two exercises below, but it's most likely better if you try something you think of yourself!

# 4 Jupyter Notebook

To use Jupyter Notebook, open Anaconda Navigator after installing Anaconda. Then Launch the Notebook module from the navigator window as highlighted below.




The Notebook will open in a browser window with a file explorer page. Navigate to open the .ipynb file you want to open, in this case BungeeBoy. Once opened, the most important thing to know it the "Run" button as highlighted below.




The Notebook is divided into cells, where marking one and clicking Run will execute the code in that cell. You can also see in the image above that there is a drop down menu labeled Markdown. This is when the cell is supposed to represent text, and is per default set to Code, which is when it contains Python code.

The STOP button is what you use if a cell gets stuck or you want to abort a simulation. If things freeze or something starts lagging, you can restart the Notebook by restarting the kernel, under the Kernel menu just above. You then have to rerun all the cells you need.

Double click a cell to edit the text, then click run, to try and get the hang of how it works. Experiment a bit and you'll get the hang of things.

# 5 Exercises

There are three built in tasks in the guide that you can try to complete. It is more encouraged that you do modifications you think of yourself, but you can check out these for inspiration. By doing them you will deeper understand how the code works. The first is basically playing around. The second is a bit more technical but quite simple and is recommended to try. The third is much more challenging.

**1) Build your own scene.**
In Chapter 5 there is a cell defining a function myScene. Here, you can move/modify/add objects to create your own scene, and run the simulation in the cell below. This is a good place to start because you can be creative and you will probably see some fun simulations and most likely some bugs that make the system go bonkers. As inspiration you can replicate the Clifton Suspension Bridge from an image, which is where the modern bungee jump originates from, or search for some pictures online of cliffs, waterfalls etc.

**2) Create a triangle class.**
In the beginning of Chapter 3 the basic objects are defined. Everything is currently built from circles, rectangles and lines. In CGI, objects are basically built using triangular surfaces, one corner/vertex for each spatial degree of freedom. You can create a class for a triangle in the guide and add it to a scene, and it can instantly interact with the environment as a fixed object. You can first get the visual representation working and check that it is rendered (look at matplotlib Polygon) and then add the closest point function. See the guide for more details.

**3) Complete the rod class.**
The Physics chapter introduces a class called Rod. The challenge is to use these to represent the body of the dude class instead of the bungee ropes currently used. To do this, two things must be done, and you can choose to do both, one or the other, or none. Attempt this exercise if you think the challenge sounds fun and understand why the approximation made doesn't work properly, otherwise it's probably too cumbersome to be entertaining.

**a)** Fix the improper approximations made in the existing class to be more physically accurate. There is a cell designated to pasting the current code and modify it. You can compute the force the springs should be acting with and apply it so that the energy in the Newton's cradle becomes constant.

**b)** Rewrite the Rod class so that it can be used in chapters 4 & 5. By comparing to the bungee rope class you can see what functions you might need to implement, and use a basic line to represent the rod. You can then replace the arms etc in the dude class. This should result in a stiff stick figure instead of the slightly above average flexible one that is currently used. This can be done without having done a) but it can get a bit tricky.