Python Programming for Geomatics – GEO1000

Assignment 4: C++ and github

Deadline: Friday, October 29, 2021, 18h00 CEST

Authors: Rianne Aalders, Marieke van Esch

Student Numbers: 4593987, 4545508

GIT REPOSITORY link: https://github.com/mariekeve/assignment4-nbody

Differences between the python and C++ program. (200 words)

Python is a high-level programming language. Python is run through an interpreter, whilst C++ is pre-compiled. Python is an easy to read program for people, and is a dynamically typed language. Whilst C++ is an statically typed language which make use of classes. Information of python is short written. C++ is an extended program with classes and therefore less readable for humans at first sight. All the variables have to be initialized in order to work with it in a proper way.

The information of the simulation in C++ is presented in a way of using classes datatypes. In nbody.cpp "body state" is therefore an array containing all the planets, which in turn use the class "body". "body" makes use of vector3d class with methods position and velocity. Vector3d makes use of pointers.

In contrast to C++ is the nbody.py structure is presented more in a global way. BODIES is an dictionary data type containing the keys of the planets. The values are presented as a tuple containing first its x,y,z-positions, secondly its x,y,z-vectordirections and lastly its respect to the solarmass. (181 words)

Reflection paragraph how you went about solving the task (max 400 words)

Github operations

First we needed to clone the GIT-repository of bmmeijers. After obtaining the files, we could add our cloned files to our index area. Next, we committed the change of cloning our data to our repository. Finally, we pushed our file towards our own assignment4-nbody remote on Github. When we made changes to the files, we pulled our remote again to our own working directory. Because we worked on one computer, we only had to pull at the end to check if it worked on another computer too.

Program

Nbody.cpp
For C++ we had to install a debugger and a release version in order to compile the files. Next, the python interpreter was installed to run python on Clion. After that, we included more pre-processor directives to load the files: fstream, iomanip, string and cstdlib. To print the csv file, we initialized a void function, print_csv, which takes the stringtype filename, and the array bodystate[]. The filename will be opened. It gets its header and afterwards it gets its items through an iteration of the number of BODIES_count in body state[]. In main, print_csv is called with nbodyCPP.csv as filename.

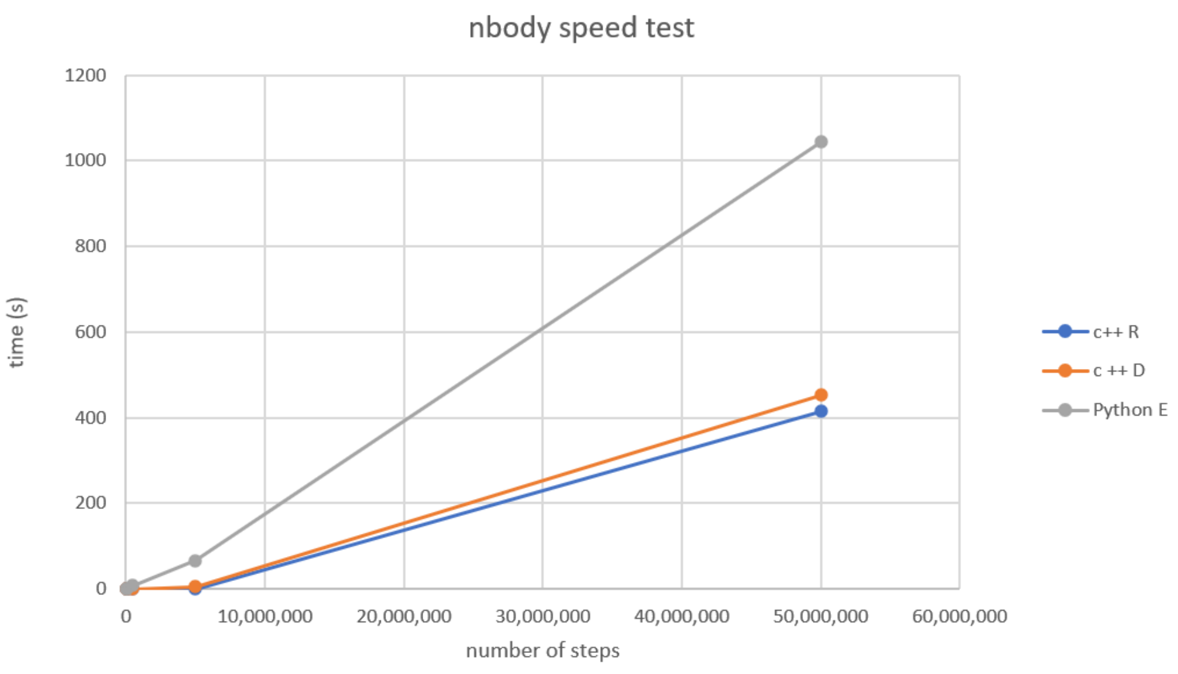
Nbody.py
In python interpreter the function write_csv is established with parameters file_name, bodies=SYSTEM and names=NAMES. Bodies=System are the values of the bodies' global dictionary. Names are the keys of the global dictionary. file_name will be opened and instructed to work in write mode. After writing the header, a for-loop creates the table with the names and the respective x,y,z-positions. In the function main, iterations for the python executable are used and the function write_csv is called to write the nbodyPython.csv.

Runner.py
To compute the runtime of the debugger C++, the Release C++ and the python executable, writing a runner.py was needed. Importing of os and perf_counter to be able to calculate the perf_counter. After initializing the steps the several programs can be printed and a table can be obtained of the speed timing (table 1: varying from 5,000 till 5,000,000) (graph1: varying from 5,000 and 500,000).

Table 1: Speed timing of nbody (s)

Steps	C++ release	C++ debug	Python script	Python executable
5,000	0.020	0.047	0.062	0.448
50,000	0.023	0.087	0.409	1.030
500,000	0.062	0.518	3.778	7.577
5,000,000	0.466	4.628	44.650	66.306
50,000,000	414.568	453.097	415.539	1044.075

Conclusions derived from the table and graph are that C++ runtimes are significantly faster than python times with increasing steps. This is because of C++'s better memory management compared to Python. The debugger is slower than the release version since it disables its optimization, in order to work in debugging mode. The release version enables its optimisation which makes the program run faster. (400 words)



Graph 1: nbody speed test

APPENDIX A

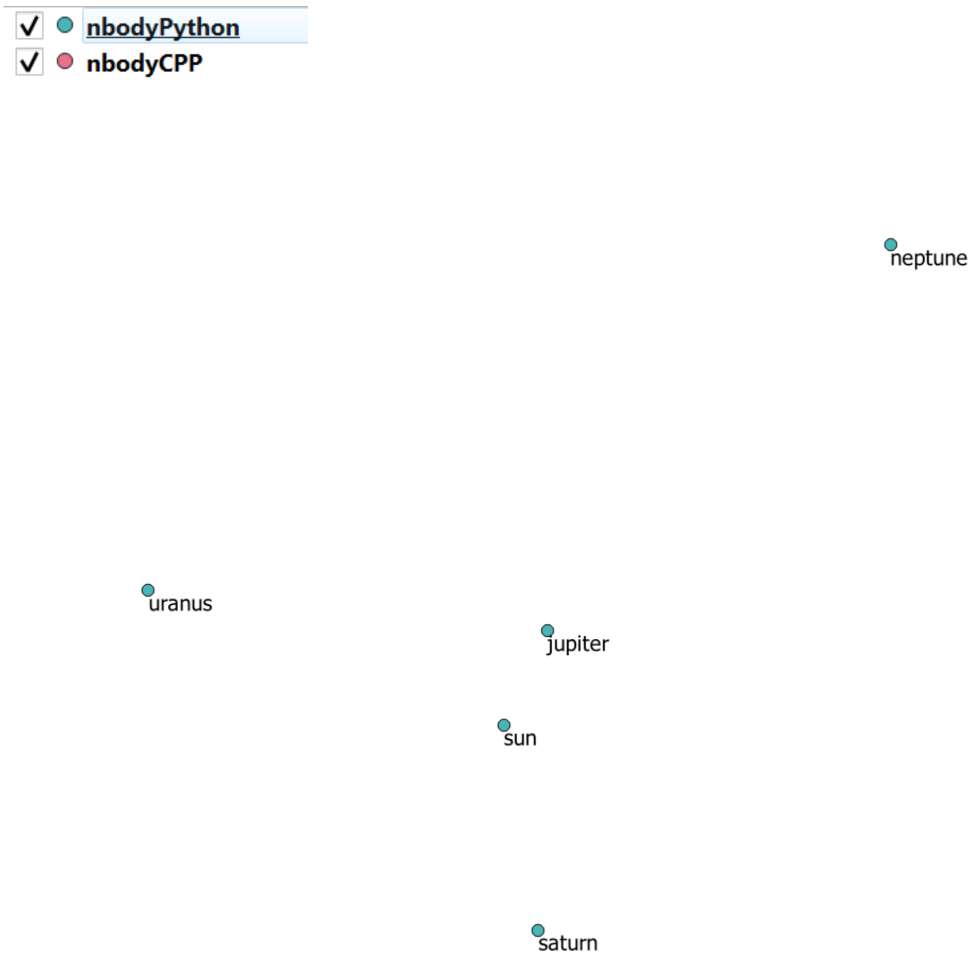


Figure 1: nbody Python in QGIS

APPENDIX B

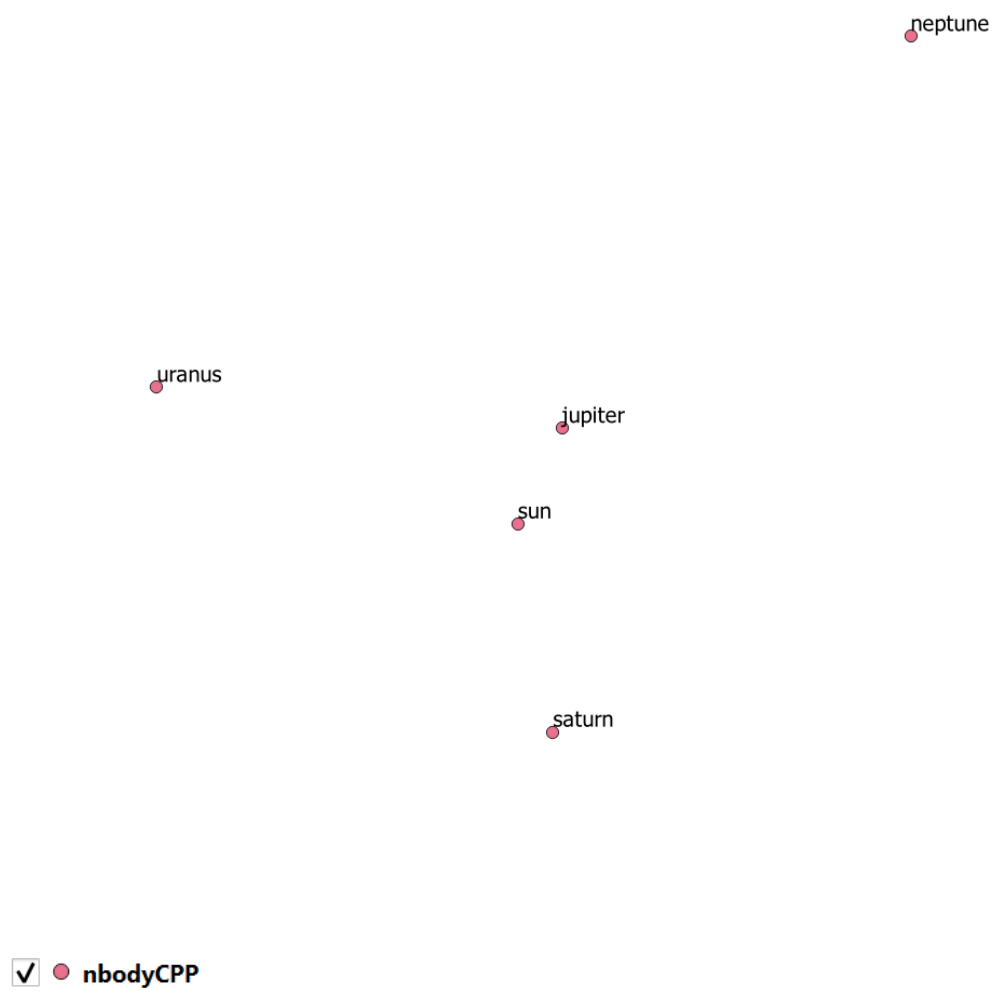


Figure 2: nbody CPP in QGIS

Literature whenever we got stuck

Stackoverflow