BSP Project Description:
OrbitCalc - A simplified version of the solar system

Friday 6th October, 2023 - 10:21

Henrik Klasen
University of Luxembourg
Email: henrik.klasen.001@student.uni.lu

This report has been produced under the supervision of:
Gabriel Garcia
University of Luxembourg
Email: gabriel.garcia@uni.lu

***Abstract*—With scientific endeavours becoming more cost intensive, there is a need to plan specific missions into minor details. For having the means to plan such missions, for example to the edge of the solar system, to the depth of the oceans and onto the surface of other celestial bodies, there has to be a computational model, which virtually has any property, any side factor and any physical property of the environment the mission planners expect to encounter. The development of one such computational model with a potential use case in space exploration is the topic of this Bachelor Semester Project. (100 words)**
In total 315 words, ca. 15%

1. Main required competencies (107/100-200 words 5.35%)

1.1. Scientific main required competencies (44 words)

Before starting the Bachelor Semester Project, the student had some experience in orbital mechanics, which includes the basics of (gravitational) force vectors.
Next to that, the student had some experience in constructing mathematical and physical models with certain simplifications for computational ease.

1.2. Technical main required competencies (63 words)

As the technical deliverable is about the implementation of a basic solar system simulation in Unity, the student already knows how to work with Unitys' Scaling, Transformation and Rotation system and their Script APIs in C#. Also the basics of game and simulation development are known, as it was part of a small course in the Computer Science Summer Camp at Hochschule Trier.

2. A computational model for a solar system simulation (108/800 words)

The scientific part of this Bachelor Semester Project is about modelling a suitable, simplified model of the solar system.
It will consist of three parts. The first part being about the fundamentals of computational modelling, which will take a look at the approach of breaking real world data down to a computable model, which can be emulated by the computer. As the title of this Bachelor Semester Project implies, that it is about the simulation of orbital mechanics in a virtual solar system, the second part of the scientific part is about modelling a model with some constraints and simplifications for implementing it later in the technical part.

3. Implementation of a virtual solar system (0/800 words)

The description of what will be presented in the technical deliverables section of the final report. This section must present and be based on a state of the art of the topics addressed by the technical aspects of the project.

4. Plagiarism statement

I declare that I am aware of the following facts:

- As a student at the University of Luxembourg I must respect the rules of intellectual honesty, in particular not to resort to plagiarism, fraud or any other method that is illegal or contrary to scientific integrity.
- My report will be checked for plagiarism and if the plagiarism check is positive, an internal procedure will be started by my tutor. I am advised to request a pre-check by my tutor to avoid any issue.

- As declared in the assessment procedure of the University of Luxembourg, plagiarism is committed whenever the source of information used in an assignment, research report, paper or otherwise published/circulated piece of work is not properly acknowledged. In other words, plagiarism is the passing off as one's own the words, ideas or work of another person, without attribution to the author. The omission of such proper acknowledgement amounts to claiming authorship for the work of another person. Plagiarism is committed regardless of the language of the original work used. Plagiarism can be deliberate or accidental. Instances of plagiarism include, but are not limited to:

    1) Not putting quotation marks around a quote from another person's work
    2) Pretending to paraphrase while in fact quoting
    3) Citing incorrectly or incompletely
    4) Failing to cite the source of a quoted or paraphrased work
    5) Copying/reproducing sections of another person's work without acknowledging the source
    6) Paraphrasing another person's work without acknowledging the source
    7) Having another person write/author a work for oneself and submitting/publishing it (with permission, with or without compensation) in one's own name ('ghost-writing')
    8) Using another person's unpublished work without attribution and permission ('stealing')
    9) Presenting a piece of work as one's own that contains a high proportion of quoted/copied or paraphrased text (images, graphs, etc.), even if adequately referenced

    Auto- or self-plagiarism, that is the reproduction of (portions of a) text previously written by the author without citing that text, i.e. passing previously authored text as new, may be regarded as fraud if deemed sufficiently severe.

References

[BiCS(2021)] BiCS Bachelor Semester Project Report Template. https://github.com/nicolasguelfi/lu.uni.course.bics.global University of Luxembourg, BiCS - Bachelor in Computer Science (2021).

[BiCS(2021)] Bachelor in Computer Science: BiCS Semester Projects Reference Document. Technical report, University of Luxembourg (2021)

[Armstrong and Green(2017)] J Scott Armstrong and Kesten C Green. Guidelines for science: Evidence and checklists. *Scholarly Commons*, pages 1–24, 2017. https://repository.upenn.edu/marketing_papers/181/

5. Appendix

All images and additional material go there.

5.1. Source Code

The following environment shows the correct and mandatory way to insert your code.

Listing 1: Caption example.

```
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None #to become the incidence matrix
    VT = np.zeros((n*m,1), int) #dummy variable

    #compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m-1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

                if M is None:
                    M = np.copy(VT)
                else:
                    M = np.concatenate((M, VT), 1)

                VT = np.zeros((n*m,1), int)

    return M
```