# Graphical Processing Systems
# Project Documentation

**Student : Rogoz Bogdan Dorin**
**Group : 30433**
**Laboratory assistant : Dr. Eng. Bacu Victor**

## Table of Contents

1. Subject specification....2
2. Scenario....2
  2.1. Scene and objects description....2
  2.2. Functionalities....2
3. Implementation details....3
  3.1. Functions and special algorithms....3
    3.1.1. Possible solutions....3
    3.1.2. Motivation of the chosen approach....3
  3.2. Graphics model....3
  3.3. Data structures and class hierarchy....3
4. Graphical user interface presentation and User manual....4
  4.1. GUI....4
  4.2. User manual....5
5. Conclusions and further developments....6
6. References....6

# 1. Subject specification

This documentation various aspects regarding the modeling of the virtual scenario, as well as the code that powers it. The project had to be done in C++, using the OpenGL library and GLSL (GL Shading Language).

# 2. Scenario

## 2.1. Scene and objects description

The scene consists of a tropical island, an inhabitant and its shack. Occasionally, a helicopter flies above the island. Various other elements, such as an explosive barrel, bottles and a giant diamond have been included in order to add more diversity to the scenery.

## 2.2. Functionalities

The user is permitted to move around the scenery and influence certain natural aspects of it, such as enabling / disabling rain, increasing / decreasing rain density, increasing / decreasing wind power and changing wind direction.

# 3. Implementation details

## 3.1. Functions and special algorithms

A special algorithm developed for this project was the one responsible for rendering rain.

### 3.1.1. Possible solutions

One solution would have been to draw each rain drop as a 2D rectangle composed out of 2 triangles. Such as solution would provide fast and efficient results, but the overall quality would be poor at best with a simple implementation.

Another solution would be to draw multiple 3D objects, each one representing a rain drop. This solution has a great impact on the CPU during initialization and can stress the GPU if rendered in high numbers, but provides the best visual results. This one is the solution implemented in the final code.

### 3.1.2. Motivation of the chosen approach

Taking into consideration the materials made available throughout the laboratory sessions, rendering multiple rain drop objects would have seemed to be the easier solution. Initially, the model used had a large number of faces (tens of thousands), so rendering multiple instances affected the overall performance very fast. Drastically reducing the number of faces (less than 50) and drawing a down-scaled version of the object resulted in a significant improvement in performance, while retaining the same level perceivable level of detail.

## 3.2. Graphics model

Objects have been downloaded from the Internet and separately imported into the project. Some of the objects required special attention, since they came without any texture, or their material file was wrong. In some other cases, the material files had to be modified in order to be able to use the same 3D object with different textures.

## 3.3. Data structures and class hierarchy

Several data structures have been employed in order to make the development process easier:

- Mesh : represents a 3D object; includes the following data structures: Vertex (position, normal vector and texture coordinates); Texture (id, type and path); Material.

- Model3D : contains methods for printing meshes using a specified shader program.
- RainManager : contains the rain drop objects, plus methods for generating rain drop objects, drawing the rain drop objects and manipulating the coordinates of the rain drops, as well as the wind parameters. Includes a data structure named ModelPos, which contains the rain drop object and its position inside the defined rain area.
- Shader : contains methods for creating and activating shader programs.
- SkyBox : contains methods for creating and rendering skyboxes.

# 4. Graphical user interface presentation and User manual

## 4.1. GUI

## 4.2. User manual

The user can interact with the virtual world with the following keys:

- Mouse for rotation (a maximum rotation of 90 degrees up or down)
- W / A / S / D = Move forward / left / backward / right
- Q / E = Rotate small rock situated at the origin
- Keypad + / Keypad - = Increase / Decrease fog density
- 1 / 2 = Activate / Deactivate rain effect
- 3 / 4 = Increase / Decrease rain density
- C / V = Decrease / Increase rain speed
- 5 / 6 = Activate / Deactivate wireframe rendering; This feature affects only some of the objects
- [ (left bracked) / ] (right bracket) = Decrease / Increase alpha value for blending
- F / G = Activate / Deactivate full-screen
- 9 / 0 = Activate / Deactivate wind effect
- Left arrow / Right arrow = Rotate wind direction

- Up arrow / Down arrow = Increase / Decrease wind power
- 7 / 8 = Activate / Deactivate automatic mouse movement
- Z / X = Load High poly / Low poly stone object situated at the origin

# 5. Conclusions and further developments

This project presented a challenge for me, in a way that I had to familiarize myself with graphical viewing / editing tools such as Blender and also come up with solutions for problems which do not have a specific solution / compatible implementation.

Some of the further possible developments include:

- Improving the rain effect
- Animation improvement
- OpenAL implementation
- Skeletal animations

# 6. References


- http://cgis.utcluj.ro/teaching/
- https://free3d.com/
- https://www.blender.org/