**A picture containing clipart

Description automatically generated COMP4910 Senior Design Project 1, Fall 2019**

**Advisor: Gizem Kayar**

**POF: Performance Optimized Fluid System Requirements Specification**

**16.11.2019**

**Revision 2.0**

**By:**

**Baran Budak-15070001012**

**Cihanser Çalışkan-16070001020**

**İsmail Mekan-15070001048**

**Revision History**

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Explanation** |
| **1.0** | **03.11.2019** | **Initial requirements** |
| **2.0** | **16.11.2019** | **Requirements Model** |

**Contents**

Revision History2

Contents3

1.0 Introduction4

1.1 Purpose4

1.2 Scope4

1.3 Overview4

2.0 Diagrams **5**

2.1 Sequence Diagram5

2.2 Use Case Diagram6

2.2 Activity Diagram6

3.0 General Description**7**

3.1 System Functions7

3.2 NVIDIA Flex7

4.0 Functional Requirements **7**

4.1 Non-Functional Requirements8

5.0 Glossary9

6.0 User Characteristics 9

7.0 General Constraints **9**

8.0 References **9**

**1.0 Introduction**

**1.1. *Purpose***

The purpose of the performance optimized fluid (POF) system is research and apply surface reconstruction methods with intend of create more efficient particle based simulation system. The POF system should increase the efficiency of simulation by means of running it faster while occupying less memory of the computer. In detail, POF system is reconstructing the surface particles by benefiting from various research papers mentioned. The POF system approaches particles as a continuum and inspect the fluid as a whole object. Herewith, system approaches to fluids as there are no separate particles but rather the fluid is a continuous material.

**1.2. *Scope***

The POF system shall help to increase performance for simulating fluids. System reduces the necessary computation operation for particles during the simulation. Inıtially, the POF system runs the NVIDIA flex because the POF system need particles position data. The POF system works with unity engine for visualization but another program can be used for simulation.

The POF system computes colour field quantity of each particle and marks all the surface particle. Surface particles are calculated and marked for the 2r distance which is the two times of a particle radius and each particle is exactly same. Marked vertices controls by handler in our code. For every surface vertex we compute every small cells in the 4 times of radius area in axis aligned bounding box. After we find the cells, we find the particles in those cells and calculate how the every particle effect the other particles as a scalar value of the vertex which the method is defined in Zhu&Bridson. Lastly, for all the vertex data, we draw a cube and check those eight vertices of cube and if cell is on the surface we pass the information of the vertices for triangulation stage.

**1.3. *Overview***

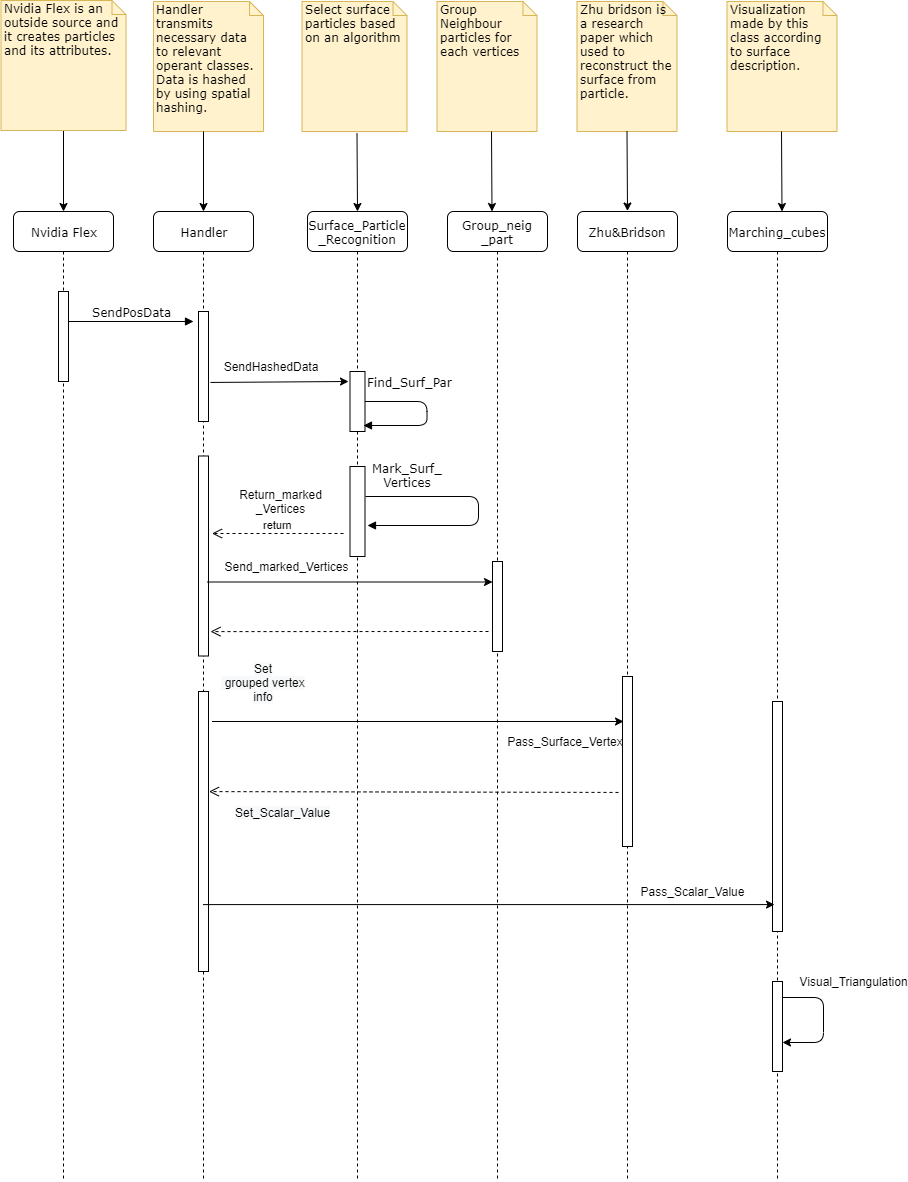
This document provides a description of the POF system. Requirements specification document involves diagrams which defines user roles in system and more importantly, explaining how system operates in the background. Document mostly focuses on the specified requirements. System functions are defined by expressing functional and non-functional requirements. The function of NVIDIA flex and how it is used in the POF system is described. User characteristics and constraints specifies that how POF system can work under which circumstances.

**2.0 Diagrams**

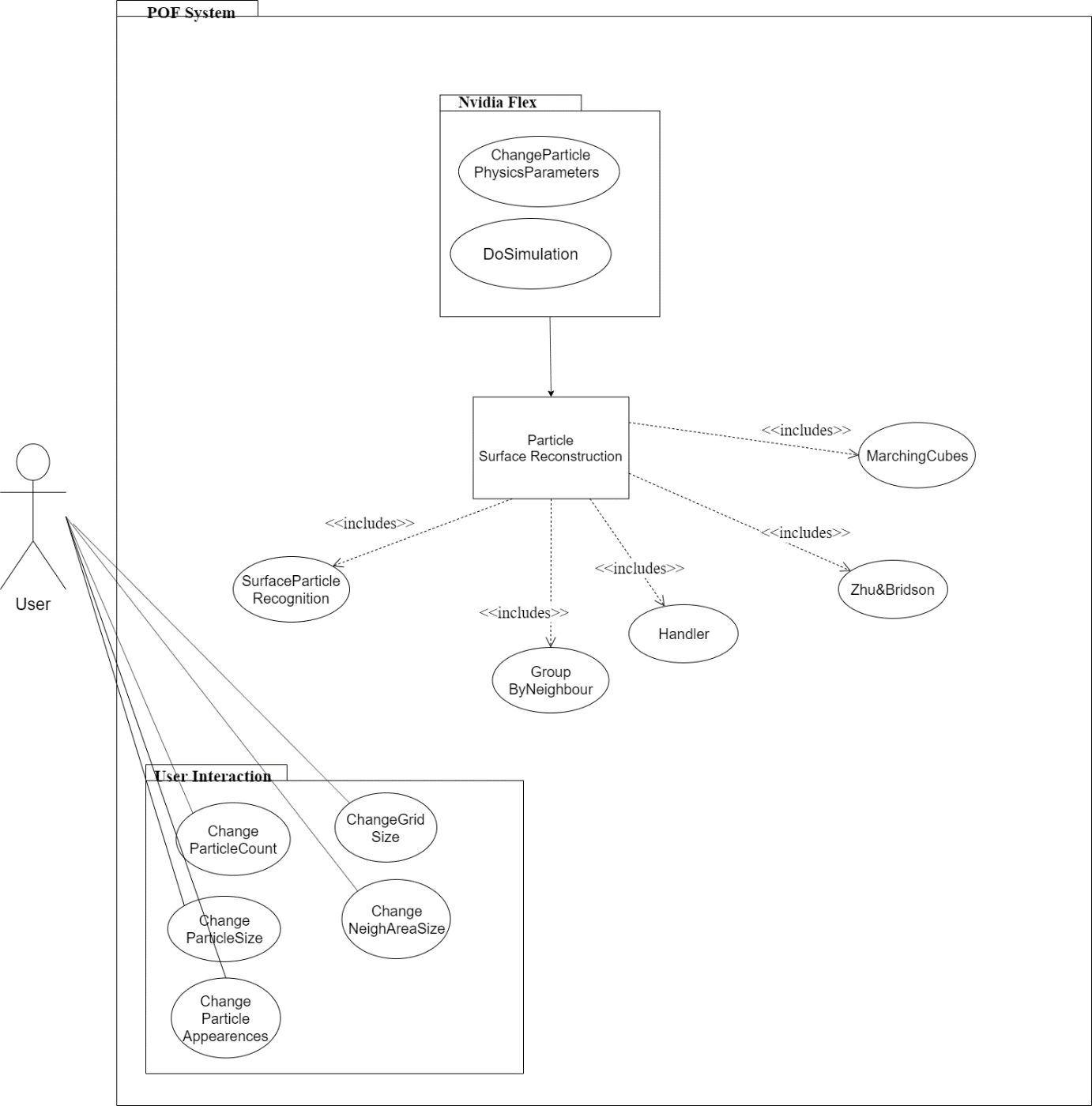
**\* activity diagram ekle**

**2.1 Activity Diagram**

**2.2. *Sequence diagram***



**2.3. *Use case diagram***



**3.0 General Description**

This system is used to provide a high-level description of the system, as well as identify the users involved and help explain their roles.

***3.1. System functions***

The POF system shall retrieve the position data of the particles which is created by NVIDIA flex particle-based fluid simulation system. System detects the required boundaries that particles occupied in three dimensional space. Grids are created after the bounding box. System will detect the surface particles for the specific algorithm and use the hash function for accessing surface particles more effective.

***3.2. NVIDIA flex***

NVIDIA Flex is a particle-based simulation technique for real time visual effects. It is an outside source tool for our simulation enhancement. We will use NVIDIA flex for creating particles and using particle data to process it for our algorithm. Besides, it is unnecessary to strive with particle physics for our project because it is aimed that enhancing the performance of the already existed particle-based fluid system by surface construction and it is not aimed to create a fluid simulation system from scratch. In simulation, a stack of water consists from millions of water particles.

1. **Functional Requirements** 
   * 1. ***Take the particle data from NVIDIA flex:*** NVIDIA Flex creates the particles and we retrieve the data to another function which will use these data to apply our algorithm.
     2. ***Find the boundaries*:** Particles occupies a space in three-dimensional coordinate system. Particles’ minimum and maximum boundaries should be found for the specifying the volume that particles occupy. Simply, boundaries imply an Axis Aligned Bounding Box which is memory efficient way of representing a volume. It is necessary preliminary step for the dividing into small cubes.
     3. ***Divide into grids:*** Axis aligned bounding box should have divided into small cubes to analyse the particles and apply the algorithm. According to our researches, it was decided to use ratio of one-eighth of the particle radius, but it will change in during project time for testing performance and efficiency.
     4. ***Surface recognition:*** The algorithm detects surface particles and their cells so we can discard inactive cells (for marching cubes of vertices). With this method we have more efficient and better performance by discarding unnecessary cells.
     5. ***Animating Sand as a Fluid:*** Animating Sand as a Fluid is a research paper that explains animating sand as a fluid. However, research paper also mention about it can be used in fluids which is the significant part that it will used in project. The paper mentions about surface reconstruction from particle and gives the functions and formulas in order to applying the method.
     6. ***Marching cubes:*** The algorithm is used for extracting a polygonal mesh of an isosurface from a three-dimensional discrete scalar field. In this project, marching cubes algorithm is used with Zhu-Bridson algorithm. Zhu-Bridson algorithm is used in marching cubes algorithm in order to get better visual outputs.
     7. ***Performance:*** This requirement can be accepted as both kind of requirement type. The project does not give this requirement as mandatory, but to achieve performance has significant importance. This requirement explained in non-functional requirements.

**4.1 Non-functional Requirements**

**Performance:** The system’s performance should be increased after the application POF to the system. Due to POF system, particle simulation has higher fps rate, or it can be run at lower end devices. The existed methods will be checked whether it can be developed or not.

**Usability:** Similar fluid systems are developed in OPENGL or another various platform. However, our project will be deployed into Unity game engine which is supported on windows and macOS.

**Efficiency:** The aim of the POF system is efficient memory usage.

1. **Glossary**

**NVIDIA Flex:** NVIDIA Flex is a particle based simulation technique for real-time visual effects.

**POF:** An Acronym stands for performance optimized fluid system.

**Unity:** Unity is a cross-platform game engine developed by Unity Technologies. Unity is used for developing video games and simulations for consoles and mobile devices.

**Axis Aligned Bounding Box (AABB):** Bounding volume for a set of objects is a closed volume that completely contains the union of the objects in the set.

**Zhu and Bridson:** Names of the researcher’s last names which owner of the research paper called “Animating sand as a Fluid”

**Marching Cubes**: Marching cubes is a computer graphics algorithm, published in the 1987 for extracting a polygonal mesh of an isosurface from a three-dimensional discrete scalar field.

**OPENGL:** Open Graphics Library is a cross-language, cross-platform application programming interface for rendering 2D and 3D vector graphics.

**GPU:** Graphic Processing Unit

**CPU:** Central Processing Unit

**API:** Acronym for Application Programming Interface

**Isosurface:** An isosurface is a 3D surface representation of points with equal values in a 3D data distribution which is the 3D equivalent of a contour line.

**Polygonal Mesh:** Polygon mesh is the collection of vertices, edges, and faces that make up a 3D object.

**Grid:**

**Cell:**

**Color field quantity:**

**6.0 User Characteristics**

There are two kinds of people who will use our system. The performance optimized fluid system can be used from scientists which they can examine our system, develop it or apply to another related research. Besides, it can be used from students who have interest about surface reconstruction.

**7.0 General Constrains**

A D3D11 capable graphics card with the following driver versions:

NVIDIA: GeForce Game Ready Driver 372.90 or above.

AMD: Radeon Software Version 16.9.1 or above.

In order to build the demo at least one of the following is required:

Microsoft Visual Studio 2013 or above.

G++ 4.6.3 or higher

CUDA 8.0.44 or higher

DirectX 11/12 SDK

**8.0 References**

**\* Düzgün formatta referansları yaz ve maddeler alfabetik sırada olacak.**

1. [ASF05]

Zhu, Y., and Bridson, R.

https://www.cs.ubc.ca/~rbridson/docs/zhu-siggraph05-sandfluid.pdf

Vink, G. E., Morgan, W. J., and Vogt, P. R., 1985, The Earth's hot spots, *Scientific American,* v. 252, p. 50- 57.

2) NVIDIA Flex Documentation

https://docs.nvidia.com/gameworks/content/gameworkslibrary/physx/flex/index.html

3) Marching cubes

http://www.cs.carleton.edu/cs\_comps/0405/shape/marching\_cubes.html

4) Surface recognition

https://cg.informatik.uni-freiburg.de/publications/2012\_CGF\_surfaceReconstructionSPH.pdf

5) particle based simulation of fluids

http://www.sci.utah.edu/~tolga/pubs/ParticleFluidsHiRes.pdf

6) Optimized Spatial Hashing for Collision Detection of Deformable Objects

Matthias Teschner Bruno Heidelberger Matthias M¨uller Danat Pomeranets Markus Gross

· December 2003

7)