POF: Performance Optimized Fluid System Requirements Specification

2.11.2019

Revision 1.0

By:

Baran Budak-15070001012

Cihanser Çalışkan-16070001020

İsmail Mekan-15070001048

**Contents**

1. Introduction……………………………..

2. General Description…………………..

3. Functional Requirements………………..

4. UML Diagrams

5. User Characteristics………………..

6. General Constraints…………..

7. References………………….

**Revision History**

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Explanation** |
| **1.0** | **03.11.2019** | **Initial requirements** |

1.0 Introduction

This section introduces the requirement specification document for the Performance Optimized Fluid System. It provides the purpose and scope of the system.

1.1 Purpose

Purpose of the requirement specification document is describing the functions and specified requirements for performance optimized fluid systems. The system should help the performance problems while simulation is running by running it faster and occupying less memory in computer.

The aim of the performance optimized fluid system is enhance the performance of the existing fluid simulation system by reconstruct the surface benefiting from research papers(other various research papers applied along the project).It was decided to use unity game engine. However, it can be used another tool program such as unreal engine in order to simulate fluid system.

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. System approaches as a whole rather than each one of particle. It takes the particles position data and Pof system simplifies it the way that requires less computation.

1.3 Overview

This document provides a high level description of the performance optimized particle based fluid system. It identifies the involved users and helps to define their roles in system. The majority of this document focuses on the specified requirements.

2.0 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.

2.1 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.

* 1. 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, striving with particle physics is unnecessary for our project because we already aiming enhance the performance of the already existed particle based fluid system by surface reconstruction, not creating a fluid simulation system from scratch. In simulation, a stack of water may be consists of 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.

3.0.2 Find the boundaries: Particles occupies a space in three dimensional coordinate system. Particles boundaries should be found for the specifying the volume that particles occupy. Simply, boundaries implies 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.

* + 1. 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
    2. Surface recognition: The algorithm detects surface particles and their cells so that, we can discard inactive cells (for marching cubes of vertices). Because of this method we have more efficient and better performance by discarding unnecessary cells.
    3. Hash function: Hash is used to access particles and particle cells more efficient and faster than linear search.
    4. (??????????????????????)Zhu-Bridson: research paper will determine that which particle should reconstruct.
    5. 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.

3.1 Non-functional Requirements

**Performance:** The system performance should be increased at the end of the pof application to the system. Because of the 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 platforms. 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. UML Diagrams

4.1 Sequence Diagram

Activity diagram, Use case, Sequence Diagram,

5.0 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 reverse engineer our system and apply it to another related research and development system however they want it. Besides, it can be used from students who have interest about surface reconstruction topics in college who researches about these issues.

6.0 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

7. References

1) Zhu & Bridson

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

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>