****

**YAŞAR UNIVERSITY**

**FACULTY OF ENGINEERING**

**DEPARTMENT OF COMPUTER ENGINEERING**

**COMP4910 Senior Design Project 1, Fall 2019**

**Supervisor: Dr. Gizem Kayar**

**POF: Performance Optimized Fluid System**

**Final Report**

**By:**

**Baran Budak -15070001012**

**Cihanser Çalışkan -16070001020**

**İsmail Mekan -15070001048**

# PLAGIARISM STATEMENT

This report was written by the group members and in our own words, except for quotations from published and unpublished sources which are clearly indicated and acknowledged as such. We are conscious that the incorporation of material from other works or a paraphrase of such material without acknowledgement will be treated as plagiarism according to the University Regulations. The source of any picture, graph, map or other illustration is also indicated, as is the source, published or unpublished, of any material not resulting from our own experimentation, observation or specimen collecting.

**Project Group Members:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name, Last name** | **Student Number** | **Signature** | **Date** |
| Baran Budak | 15070001012 |  |  |
| Cihanser Çalışkan | 1607000020 |  |  |
| İsmail Mekan | 15070001048 |  |  |

**Project Supervisors:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name, Lastname** | **Department** | **Signature** | **Date** |
| Gizem Kayar | Computer Engineering |  |  |

# ACKNOWLEDGEMENTS

We thank Gizem Kayar for discussions and suggestions in the development of the project. This project is supported by Yasar University Computer Engineering Department.

# 

# KEYWORDS

|  |  |
| --- | --- |
| **Term** | **Description** |
| **Cell** | Axis aligned bounding box is divided into small identical cubes. |
| **Color field quantity** | It is a functionthat calculates how each particle is affected by all the other particles. |
| **Gradient** | The directional derivative of a scalar field gives a vector field directed towards where the increment is most, and its magnitude is equal to the greatest value of the change. |
| **Grid** | Series of vertical and horizontal lines that are used to subdivide AABB vertically and horizontally into cells in three-dimensional space. |
| **Iso-surface** | 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. |
| **Marching Cubes** | Marching cubes is a computer graphics algorithm, published in 1987 for extracting a polygonal mesh of an isosurface from a three-dimensional discrete scalar field. |
| **NVIDIA Flex** | NVIDIA Flex is a particle-based simulation technique for real-time visual effects. |
| **Polygonal Mesh** | A polygon mesh is the collection of vertices, edges, and faces that make up a 3D object. |
| **Unity 3D** | 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. |
| **Visual Studio** | Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. |

**Table 1:** Keywords

# ABSTRACT

The POF system offers more efficient and better performance particle-based fluid simulation. To increase efficiency, we use hashing to reach particles faster. We used the Marching cubes algorithm to visualize particles. We only render the surface particles by using Marching cubes to increase system performance.

\*\* burayı düzenle.

# ÖZET

POF sistemi daha efektif ve daha iyi performanslı bir parçacık temelli sıvı simülasyonu sunar. Hash algoritmasını taneciklere daha hızlı ulaşmak ve verimliliği arttırmak için kullanıyoruz. Tanecikleri görselleştirmek için ise Marching cubes algoritmasını kullanıyoruz. Marching cubes ile yüzey taneciklerini görselleştirip diğer tanecikler sıvının içinde kaldığı için çizmiyoruz ve bu da performansı arttırıyor.

# TABLE OF CONTENTS

[PLAGIARISM STATEMENT ii](#_Toc533517651)

[ACKNOWLEDGEMENTS iii](#_Toc533517652)

[KEYWORDS iv](#_Toc533517653)

[ABSTRACT v](#_Toc533517654)

[ÖZET vi](#_Toc533517655)

[TABLE OF CONTENTS vii](#_Toc533517656)

[LIST OF FIGURES viii](#_Toc533517657)

[LIST OF TABLES ix](#_Toc533517658)

[LIST OF ACRONYMS/ABBREVIATIONS x](#_Toc533517659)

[1. INTRODUCTION 1](#_Toc533517660)

[1.1. Description of the Problem 1](#_Toc533517661)

[1.2. Project Goal 1](#_Toc533517662)

[1.3. Project Output 1](#_Toc533517663)

[1.4. Project Activities and Schedule 1](#_Toc533517664)

[2. DESIGN 1](#_Toc533517665)

[2.1. High Level Design 1](#_Toc533517666)

[2.2. Detailed Design 1](#_Toc533517667)

[2.3. Realistic Restrictions and Conditions in the Design 1](#_Toc533517668)

[3. IMPLEMENTATION, TESTS and TEST DISCUSSIONS 2](#_Toc533517669)

[3.1. Implementation of the Product 2](#_Toc533517670)

[3.2. Tests and Results of Tests 2](#_Toc533517671)

[4. CONCLUSIONS 2](#_Toc533517672)

[4.1. Summary 2](#_Toc533517673)

[4.2. Cost Analysis 2](#_Toc533517674)

[4.3. Benefits of the Project 2](#_Toc533517675)

[4.4. Future Work 2](#_Toc533517676)

[References 3](#_Toc533517677)

[APPENDICES 4](#_Toc533517678)

[APPENDIX A: REQUIREMENTS SPECIFICATION DOCUMENT 5](#_Toc533517679)

[APPENDIX B: DESIGN SPECIFICATION DOCUMENT 6](#_Toc533517680)

**Table 2:** Table of Contents

# LIST OF FIGURES

Figure 1: The P.O.F. system…………….2

# LIST OF TABLES

Table 1: Keywords…………………3

Table 2: Table of Contents …………………3

Table 3: List of acronyms/abbreviations……..3

Table 4: Cost Analysis of Workers…………3

# LIST OF ACRONYMS/ABBREVIATIONS

|  |  |
| --- | --- |
| **AABB** | Axis Aligned Bounding Box. Bounding volume for a set of objects is a closed volume that completely contains the union of the objects in the set. |
| **API** | Application Programming Interface. |
| **CPU** | Central Processing Unit. |
| **GPU** | Graphic Processing Unit. |
| **OPENGL** | Open Graphics Library is a cross-language, cross-platform application programming interface for rendering 2D and 3D vector graphics. |
| **POF** | The Performance Optimized Fluid system. |

**Table 3:** List of acronyms/abbreviations

# 1. INTRODUCTION

# 1.1. Description of the Problem

The main problem of the particle-based fluid simulation system is excessive numbers of the particles. There are millions of particles in a small number of liquids such as water. A particle is a rigid body sphere. Simulation applies physics to particles and these particles act as a liquid. Simulation having difficulties in calculations predicated on a surplus of particles. Indirectly, time and memory complexity increasing.

# 1.2. Project Goal

The main goal of the project researches whether there is a way to enhance fluid simulation. Increasing the efficiency and performance of an existing particle-based fluid simulation is a major goal. We aim to achieve these goals by implementing a variety of methods to the POF system such as using special structures to find store particles and visualize it by using various methods like the Marching cubes. In our project, there is no certain way because it is a research and development project and new more effective ways can be found during the project. Various methods and techniques will be researched and implemented while the project is in the development process.

# 1.3. Project Output

* Better performance.
* Better memory efficiency.
* Fluid-like appearance and behaviour.
* Testing of different algorithms for performance and efficiency.
* Higher frame rates per second.

# 2. DESIGN \*\* DSD den sonra bakılacak

**//// EVERY MAIN TOPIC NEEDS FORMAL DESCRIPTION LIKE 🡪 THIS SECTION DESCRIBES SYSTEM ELEMENTS AND HOW COMBINED EACH OTHER BLA BLA BLA.**

# 2.1. High Level Design

# 2.2. Detailed Design

This section will be completed in future works.

**/// Give a little brief about detailed design and talk about dsd maybe some samples don’t know…**

# 2.3. Realistic Restrictions and Conditions in the Design

We had to neglect some aspects of the project to implement the project in a year. The security issue is ignored because the project aims to help everybody who has interested fluid simulations and contribute to science. We assumed that users of the POF system have the necessary equipment and software and know to how to use them.

# 3. IMPLEMENTATION, TESTS and TEST DISCUSSIONS

# 3.1. Implementation of the System

**3.1.1. Research Papers and System Structure**

This project based on these two research papers, surface reconstruction algorithm that implemented by Zhu et al [ZB05] and marching cubes algorithm [WH87], but when system structure creation stage is started there are some problems occurs. Our main problem was performance and memory efficiency on that point we added two algorithm to our project for passing over with performance and memory issues.

**Main Problem Solution of Problem**

|  |  |
| --- | --- |
| Searching particle data linearly due to 3D space positions and vector3 to integer translation. | Spatial hashing algorithm provides reaching particles by put them into cell data. |
| Too many particle appears in simulations and handling all of them occurs performance problems. | Do not put into calculations inactive and unnecessary particle on visualization (surface particle finding algorithm). |

After these solutions, we have started implement our system structure by creating classes but we realize complexity getting higher due to interconnection between classes, so we create a handler class for control every classes in the one class.

**Main Problem 2:**

**3.1.2. Implementation of Hash Algorithm**

**3.1.2 Implementation of Neighbor Finding Algorithm**

**/// Firstly write a pdf’s and algorithms inside them as you know firstly we use a some equations in these pdf not all of them. Talk about this stage.**

**/// Secondly talk about searching problem and how we find a solution for this problem why we use spatial hashing instead of the other hash algorithms. (Some images can be good ask baran not code images)**

**/// Thirdly talk about our neighbor algorithm and discussion about sphere or cubic usage we did this discussion. (Give an image for neighbor cell ask baran.)**

**/// Talk about cell creation way in neighbor and hash algorithm usage.**

**/// fourthly write our cell creation way (If you think you know algorithm you can talk about imaginary cells and imaginary (not storaged) numbers.). A little code will be enough. If you do not ask Baran.**

**/// Ask baran is there any deficiency or excess in there if you think there is, discuss about them(But you have to think we should not write all of them for second semester.)**

**/// Lastly talk about its not a completed, will completed in second semester bla bla bla.**

**P.S: I think we do not have to talk about all code and diagram staffs.**

**P.S2: Each case have their own test situations some of them just a math and others just a code.**

**P.S3: Ask tests to baran for visualisation(Probably I’ll not remember in first half hour).**

**// kod atma bölümü burası**

This section will be completed in future works.

# 3.2. Tests and Results of Tests

**3.2.1 Availability of the Necessary environment**

Before finding NVIDIA Flex, we have tested three particle based fluid simulation and we disqualify for these reasons;

|  |  |
| --- | --- |
| **uFlex** | Had small bugs and errors in the code, even though we have fixed minor bugs, the particles were not recognizing the collider of the objects. Collider of the simple primitive objects was not recognized by the Uflex and particles were penetrating the objects. The only plane object was being recognized by the uFlex. The problem could not be solved, and we have changed the fluid simulation. |
| **Obifluid** | Obifluid is eliminated because of performance problems. The expected result was not satisfied by the Obifluid compared to other fluid simulations our expectation was reaching 30fps with a hundred thousand particle but in three thousand particles we have 3fps. |
| **Screen Space Fluids Pro** | Like uFlex we recognize small bugs and errors in the code and we fixed it but performance was very low on higher particle count. |

**3.2.2. Testing of Implemented Codes**

**/// Talk about our not simulation test system (They’ll like this idea because it’s test with coding dudes!!!!!!!!!!!!!!!)**

**/// And talk about zhu and bridsons fake system testings.**

This section will be elaborated vastly in the future.

# 4. CONCLUSIONS

In this section, the cost table of workers is given and explained. The cost of software and hardware is given with details and benefits of the projects are explained. This part of the final report summarizes our project and gives a cost analysis for the project. Future works mentioned.

# 4.1. Summary

The massive amounts of particles can be a computational hardship for the computer.

We implement various methods to get better results by making a research. Our project focuses on catalyzing computational difficulties by increasing the performance and efficiency. Project should make easier to simulate with higher quantities of particles or getting better results with the same number of particles.

# 4.2. Cost Analysis Table

**4.2.1 Cost of workers**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Members** | **Day/Hour** | **Week/Hour** | **Semester/Hour** | **Salary/Hour** | **Salary/Monthly** | **TOTAL** |
| Member | 8 | 40 | 560 | 30 TL | 4800 TL | 16800 TL |

**Table 4:** Cost Analysis of Workers

As shown in the cost analysis table, three people works in the project. Every people work equally as workload. Therefore, only one member is represented on the cost table.

Every member works 8 hours a day and 5 days a week. A semester consists of 14 weeks and salary is 30 Turkish lira per hour. Each member costs 4800 TL per month and costs 16800 in a semester. The salary costs of all three members are 50400 TL per semester. The equivalent of 16800 TL is $2894,67. Currency translation has made from Dollar / Turkish Lira = 1 / 5.80 in 10 December 2019.

**4.2.2 Cost of Software**

**4.2.2.1 uFlex**

Unity uFlex is an asset, software purchased for our project. We used the Uflex fluid simulation asset to study on particle-based fluid simulation. This asset is eliminated because of the adaptive feature cannot be closed. The adaptive attribute is a change in the particles to look particles more realistic and physics based. However, the POF system is not able to work with the adaptive feature. uFlex asset costs 30$ in the unity asset store.

**4.2.2.2 Obi Fluid**

Obi fluid is another assed particle-based fluid simulation. This asset not selected because performance was very low compared to NVIDIA flex. Obi Fluid asset costs 30$ in the unity asset store.

**4.2.2.3 Screen Space Fluids Pro (SSF)**

Screen space fluids asset is used for fluid system visualization. It recommended that SSF works with uFlex best. Since uFlex is elected in our project, SSF is not necessarily needed. Besides, visualization is a part of our project and we will use the Marching cubes algorithm to visualize. SSF asset costs 7$ in the unity asset store.

**4.2.2.4 Technie Collider Creator (TCC)**

Technie collider creator asset is a collider asset for the objects. TCC aimed to use create better interactions with the fluids and rigid body objects. It is not decided to use this asset, but it can be used in future works. TCC asset costs 30$ in the unity asset store.

Total Software Costs: 97$

**4.2.3 Cost of Hardware**

**4.2.3.1 PC component costs that used in Project (PC 1)**

Operating System: Windows 10 (64-bit)

Processor: Intel Core i7-4700 HQ CPU

Memory: 16 GB RAM – DDR3L-1600 MHz

GPU: NVIDIA GeForce GTX850M 4GB DDR3

Cost of PC 1 in project per user: $1693, 21

Total cost = Total employee cost + Total software cost + Total Hardware cost (Hardware that used in project.)

Total cost per user = $2894,67 + $97 + $1693, 21 (PC1) = $4684,88

Total cost = Total cost per user \* 3 = ($2894,67 + $1790, 21) \* 3 = $14054,64

**4.2.3.2 Optimal Simulation Computer (PC 2)**

Operating System: Windows 10(64-bit Pro)

Processor: 8-core Intel i7 5.1 GHz

Memory: 32 GB RAM- DDR4- 2666MHz

GPU: NVIDIA Quadro P2200 5GB

Cost of PC 2 per user: $5017

Total cost = Total employee cost + Total software cost + Total Hardware cost (Hardware that used in project.)

Total cost per user = $2894,67 + $97 + $5017 (PC2) = $8008,67

Total cost = Total cost per user \* 3 = ($2894,67 +$5114) \* 3 = $24026,01

# 4.3. Benefits of the Project

Our project can benefit in all areas where liquid simulation is available.

**4.3.1 Animations and Movies:** The POF system can be used in any movies, animations that used fluids.

**4.3.2 Scientific work:** Our project benefit scientific areas the most because the project is heavily research and development based of the research papers about the particle-based fluid simulations. Scientist and researchers can use the POF system for their scientific researches

**4.3.3 Games:** Some games need a fluid simulation system to make more realistic games. The POF system can be a good factor for the makes realistic games. For instance, in sailing simulator game is a perfect match for our system.

**4.3.4 Construction:** The construction and Architecture sector can benefit from our system because the simulation is physics-based which means the POF system is almost realistic. The POF system neglects some imperceptible elastic deformations. For instance, a civil engineer can build a barrage and want to test endurance, on the computer simulation. Therefore, our system can be used for construction and architecture testing.

# 4.4. Future Work

We will develop our project in order to achieve performance and efficiency goals. The functionality of the project will remain the same. However, small changes in the calculations will be changed to get better results. For instance, the research paper [SOL??] will be implemented to our project.

# References

Requirement Specification Document revision 1.0 (RSD 1.0)

Requirement Specification Document revision 2.0 (RSD 2.0)

Design Specification Document revision 1.0 (DSD 1.0)

Solenthaler paper