****

**YAŞAR UNIVERSITY**

**FACULTY OF ENGINEERING**

**DEPARTMENT OF COMPUTER ENGINEERING**

**COMP4920 Senior Design Project II, Spring 2020**

**Advisor: Gizem Kayar**

**POF: Performance Optimized Fluid System**

**Product Manual**

**Revision 1.0**

**13.04.2020**

**By:**

**Baran Budak, Student ID: 15070001012**

**Cihanser Çalışkan, Student ID: 16070001020**

**İsmail Mekan, Student ID: 15070001048**

# Revision History

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Explanation** |
| 1.0 | 13.04.2020 | Initial Product Manual |

# Table of Contents

[Revision History 2](#_Toc36726022)

[Table of Contents 3](#_Toc36726023)

[1. Introduction 4](#_Toc36726024)

[2. POF Software Subsystem Implementation 4](#_Toc36726025)

[2.1. Source Code and Executable Organization 4](#_Toc36726027)

[2.2. Software DevelopmentTools 5](#_Toc36726028)

[2.3. Hardware and System Software Platform 5](#_Toc36726029)

[3. XYZAPP Hardware and Software Subsystem Testing 5](#_Toc36726030)

[4. XYZAPP Installation, Configuration and Operation 5](#_Toc36726031)

[References 6](#_Toc36726032)

**List of Figures**

[Fig 1: Use case diagram 5](#_Toc501993332)

[Fig 2: Sequence diagram 7](#_Toc501993333)

**List of Tables**

[Table 1: Description of the use case diagram](#_Toc501993332) 6

**1. Introduction**

The purpose of this product manual is to document the implementation, testing, installation and operation of the POF system as a software product.

The POF system is implemented and tested as it is described in Design Specification Document, Revision 2.0 [3], satisfying the requirements in POF system Requirements Specification Document, Revision 1.0.

Implementation, testing and operation details are given in the following sections of this document.

# 2. POF System Software Subsystem Implementation

This section describes the implementation of the POF system and its subsystems.

# 2.1. Source Code and Executable Organization

* Structure and organization of source code, including related source code filenames. Source code main program/main object, any subprogram/object names together with their methods and interface parameters, including related filenames especially if in different files.
* Structure and organization of executables, main executable and DLL if any, including related filenames.
* Properties and structure of configuration files, if any.
* Description of any scripts to install, diagnose, analyze etc, including related filenames.
* Explanation of diversions from DSD, if any.

Additional Notes About Source Code/Executables Delivery:

* **You MUST NOT list your source code in this Product Manual**.
* Your source code MUST include COMMENTS in a professional manner…!
* Source Code and executables are to be delivered as soft-copy only. Your delivery should also include any log file detailing events related to your execution of software.
* Full source code, executables, test files and log files could be zipped/rar’ed as a **single file**, not **as multiple files**, where filename starts with your project code. You must organize all of your implementation files in a reasonable and self-explanatory directory structure.   
  Example: XYZAPP-Code-2020-05-15-Rev-1.0.
* You will upload the zipped/rar’ed single code file into COMP4920 area in sakai.yasar.edu.tr.

# 2.2. Software Development Tools

In this following section, we describe software tools that we have used in the POF system project.

***2.2.1 Unity***

***\*\* 82 Düzelt \*\****

Unity is a game engine program which is used to obtain the visual output. Another program with the same function such as Unreal Engine 4 can be used. We decided to use Unity because learning speed is faster compared to the Unreal Engine. Besides, some of the members of our project team have experience with Unity engine so we have turned this into an advantage to accelerate the process in the project.

Details: Unity 3D version 2018.3.11 (29 March 2019), Unity Technologies.

***2.2.2 Visual Studio 2017***

Visual Studio is an integrated development environment (ide). We write our code in C# by using Visual Studio. The reason we use Visual Studio is we are developing project in Windows operating system and Unity has Visual Studio support. You can import Unity library to Visual Studio.

Details: Visual Studio 2017 v15.9.15 (13 August 2019), Microsoft.

***2.2.3 Github***

GitHub, Inc. is a company that provides hosting for software development version control using Git. We used Github in our project for storing safely. Keeping our data in local is not an efficient way and it confuses version order. Besides, the importance of tools as Github vastly shows its importance in telecommuting.

Details: GitHub Inc., Subsidiary to Microsoft.

***2.2.4 Gitkraken***

GitKraken is another Git GUI client is used from developers to increase productivity. It has the same operation as Github. However, Gitkraken has a reasonable advantage when it comes to code handling. Gitkraken shows the changing parts of the code and it makes easier to reduce confusions and accelerates the project speed.

Details: Gitkraken, Axosoft.

***2.2.5 NVIDIA FleX***

***\*\*86 Düzelt \*\****

We used NfleX as a third-party software which is created by NVIDIA Corporation. Flex used in as an asset for Unity. The software operates on Windows or Linux, but it is expected to operate on windows in our project. It can be executed on Unity or Unreal Engine 4 platforms, but we use unity for the reasons that we mentioned before.

NVIDIA FleX Requirements:

* Windows 7 (64-bit) or newer.
* DX11 or CUDA capable graphics cart
* Unity 2017.3 or later version

Details: NVIDIA FleX v1.0 (19 July 2018), NVIDIA company.

# 2.3. Hardware and System Software Platform

The minimum specification requirements are listed below:

|  |
| --- |
| D3D11 capable graphics card. |
| NVIDIA: GeForce Game Ready Driver 372.90 or above. |
| AMD: Radeon Software Version 16.9.1 or above. |
| Microsoft Visual Studio 2013 or above. |
| G++ 4.6.3 or higher |
| CUDA 8.0.44 or higher |
| DirectX 11/12 SDK |
| Windows 7 (64-bit) or higher |
| Unity 3D 2017.3 version or higher |

**Table 1:** Minimum Requirements of software Platform

You can see the system that we used while developing the POF system in Final Report [1].

# 3. POF System Software Testing

This section must describe how you have tested the correct operation of your software system. It must include:

* Details of module/object-method test cases
* Details of system integration test cases.
* Details of test case logs.
* Amount of effort (time and other hardware and software resources, etc.) spent for testing

Also, clearly specify,

* which parts/functions of your software have been implemented, tested sufficiently and **operating correctly** according to the most recent RSD and DSD,
* which parts/functions of your software have been implemented, tested sufficiently and **NOT operating correctly** according to the most recent RSD and DSD,
* which parts/functions of your software have been implemented but not tested at all,
* which parts/functions of your software **have not been implemented at all**, perhaps left as future work.

# 4. POF system Installation, Configuration and Operation

This section describes how user installs, configures and operates the POF system. Firstly, the user must have the required system mentioned before.

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturulduAssuming that a normal user has the POF system digital copy. User should open an empty 3D project from Unity. You can download Unity from the official website [6]. Firstly, you should import the NVIDIA Flex asset, you can download from the Unity asset store as the figure below. Access the NVIDIA Flex asset page in the Unity asset store. Click download. When download finished, import button will appear. Click import. Also, you can import the POF system so you do not necessarily need to import NVIDIA FleX from the Unity asset store. However, importing from the Unity asset store is shown here.

**Figure 2:** Nvidia Flex in Unity asset store

Another window will appear, and you select all files and click import again.

ekran görüntüsü, elektronik eşyalar, bilgisayar içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Figure 3:** Import Unity package window

NVIDIA Flex is imported, the next step is making ready our scene by setting up the Flex. Make right-click on the project files which positioned below of the screen. Select create, from the new window select Flex and then select flex container.

A screenshot of a computer

Description automatically generated

**Figure 4:** Creating Flex container in Unity

A screenshot of a cell phone

Description automatically generatedWe have created flex container and a new table of flex container should appear as the figure below.

**Figure 5:** Flex container table

Next step is making a right-click in the hierarchy window and select create empty and Game object is created. Make right-click and change its name to Flex array actor. Click the Flex array actor and at the inspector menu click add component. Write Flex Array Actor to the search bar and press enter. On the new open menu, we need to specify flex container which is not assigned right now. Drag and drop Flex Container to space across Container of the Container title in the Flex Array Actor inspector menu. It should look like this in the figure below.

**Figure 6:** Assigning Flex container to the Flex Array Actor table

A screenshot of a cell phone screen with text

Description automatically generated**Figure 6:** Assigning Flex container to the Flex Array Actor table

Make right-click and select flex and from there select flex array asset. We create Flex array asset from the same place that we created flex container, so this step is similar to the previous one.

A screenshot of a cell phone

Description automatically generated

**Figure 7:** Make right click and select create

A screenshot of a cell phone

Description automatically generated

**Figure 8:** Select Flex Array Asset

Click the small button to select the boundary mesh as shown in the figure. You can select different boundary meshes however, it is preferred to select a cube.

A screenshot of a computer

Description automatically generated

**Figure 9:** Selecting Boundary Mesh

Next, we click our flex array actor and assign flex array asset by drag and drop to the array asset row. Enable the Fluid checkbox so our particles can act as a fluid. This is a method of creating particles to our scene. There are particles which are stored like an array in somewhere. When you play the scene, particle-based fluid simulation behaves as the NVIDIA Flex operates with the CUDA supported GPU in the back end. Another way is creating a Flex source asset. Flex source asset operates a kind of tap or fountain. Here is how you can set up Flex source asset into your scene.

Right-click in the project file window and create a flex source asset as shown in the figure.A screenshot of a cell phone

Description automatically generated

**Figure 10:** Creating Flex Source Asset

Change the value of mesh tessellation to 1 and then click flex source actor and select surface mesh as shown in the figure.

A screenshot of a cell phone

Description automatically generated

**Figure 11:** Selecting surface mesh of flex source asset

Select mesh from the newly opened menu. We prefer quad mesh however you can prefer different meshes such as a cylinder. It will change the way of the release of the particles.

A screenshot of a cell phone

Description automatically generated

**Figure 12:** Surface mesh type menu

Create an empty game object in scene hierarchy and rename it as flex source actor. Then, click add component and add Flex source actor script to our game object. Flex source actor script has not assigned parts so we should assign Flex container to the container row by drag and drop. Assign the flex source asset to the relevant row in flex source actor script as shown in the figure with a red square and an arrow.

A screenshot of a cell phone

Description automatically generated

**Figure 13:** Creating Flex Source Actor and adding component

Click flex array actor and click add component. Write flex fluid renderer and press enter. Do this part again for the flex source actor. Particles look like water or fluid-like because of the fluid renderer.

////////

Tüm hata kodlarını ve mesajları açıklayabilirsin alınan en yagın hataları vs gibi.

// Flexin tutorial videoya referans gönderebilirsin. Sonra da bizim POF systemini nasıl koyacağımızı ve active edeceğimizi gösterirsin.

**ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu**Now, you can import the POF system. Click assets, from the opening menu please click import package. Click custom package as shown in the figure.

**Figure 14:** Select custom package in Unity.

Find POF unity package file location in your computer and select.

A screenshot of a social media post

Description automatically generated

**Figure 15:** Find POF unity package in file explorer.

New window will open as a next step. You can select which files to be imported on your unity project because of this window. Please select all and click import.

A screenshot of a cell phone

Description automatically generated

**Figure 16:** Import package file window.

You may wait a little while; it depends on your computer speed. After the process finished, you can see the POF package is set up to your unity project. As you can see, all components are included, NVIDIA FleX is one of them. However, you can import FleX asset from both Unity asset store and just import from our project. Then, you prepare the NVIDIA FleX because POF is dependent on the fact that FleX is executable. Marching cubes file is about visualization part which we described with all details in the Design Specifications Document [3]. SimuSystem is most of the part that our project covers. Visualization makes visual output completely different so we think that user can change the main setting from Marching cubes folder.

A screenshot of a computer

Description automatically generated

**Figure 17:** POF system set up on your unity project.

# References

1. Final Report revision 1.0
2. Requirement Specification Document revision 2.0 (RSD 2.0)
3. Design Specification Document revision 2.0 (DSD 2.0)
4. NVIDIA FleX manual

<<https://docs.nvidia.com/gameworks/content/gameworkslibrary/physx/flex/manual.html#manual>> (Last review: 4.4.2020)

1. Unity 2018.3 manual

<<https://docs.unity3d.com/2018.3/Documentation/Manual/index.html>>

(Last review: 4.4.2020)

1. Unity download web page <<https://unity3d.com/get-unity/download>> (Last review 4.4.2020)