Software Requirements Specification
for
Non-Euclidean Game Engine

Prepared By: Ömer Buğra İNCE

Barış MERT

Ozan BAYRAKTAR

Advisor: Dr. Faris Serdar TAŞEL

1. Introduction

1.1 Purpose

The purpose of this document is to explain the Non-Euclidean game engine. The goal of this game engine is to create games by using non-Euclidean geometric elements by going beyond the boundaries of current reality. This document contains detailed information about the project's requirements, reflecting the defined restrictions and recommended software functionality.

1.2 Scope of project

Most games today are made with game engines created based on known physics rules. For this reason, games are designed and created in accordance with Euclidean geometry. Whenever we want to make a project that is not suitable for Euclidean geometry, we have to design the mechanics of this game from the beginning. The main purpose of our project is to design a game engine ready to be used in the creation of games based on non-Euclidean geometry mechanics.

Most mechanics available are actually about smoothly moving the player from one location to another, directing the player to another location as opposed to the one they think they will reach, or altering the surrounding items or some material about the user indiscriminately during this smooth teleport to create an illusion.

While working on this project, it will be more beneficial for us to use the Unity Game Engine and C# language, which we are familiar with in this sector, and we think that it can be more efficient while getting help about the problems we face in the rest of the project.

2. Overall Description

2.1 Product Perspective

Non-Euclidean Game Engine is for game developers who are making games about Non-Euclidean geometry. These games generally use portals and camera tricks to change perspective. Game engine has 2 parts which are portals and camera tricks.

2.2 Development Methodology

For developing the project, we meet twice a week in the beginning and end of the week and we divide project parts among project members.

2.3 User Characteristics

2.3.1 Game Developer

User could be anybody but preferably a game designer or game developer.
User must be able to read and understand English language due to engine being English
Users must have knowledge of game making or game developing and read the guide provided beforehand.

3. Requirements Specification

3.1 External Interface Requirements

3.1.1 User Interfaces

User interfaces will be integrated to Unity as an add-on and will use Unity's own tab system.

3.1.2 Hardware Interfaces

There are no external hardware interfaces.

3.1.3 Software Interfaces

Product requires Unity 2019.1 to work.

3.1.4 Communications Interfaces

There are no external communications interface requirements.

3.2 Functional Requirements

3.2.1 Create_Portal Function

Create Portal function will be used to create a portal in the game world when the user activates it from the add-on menu.

Process:

1. Game Developer clicks the appropriate button.
2. A pre-made portal is put into the world.
3. Optionally, Game Developer can set;
4. its width and height,
5. where the user will be teleported with coordinates,
6. a connected portal, where users will be teleported.

3.2.2 Create_NonEuclidean_Area Function

Create Non-Euclidean Area function will be used to put a non-euclidean area attribute to the selected GameObject when the user activates it from the add-on menu.

Process:

1. Game Developer will select a GameObject, that they pre-made,
2. Game Developer will click the appropriate button,
3. Non-Euclidean Area attribute will be put into the selected GameObject.
4. After this, Game Developer can set and change;
5. its type and status.

3.2.3 Teleport_Object Function

Objects in the game world will be teleported when they collapse with the Portal object.

Process:

1. Object collapses with the portal
2. If portal has a connected portal, object is teleported to that portal,
3. if portal doesn't have a connected portal but has coordinates for teleportation, object is teleported to that coordinates,
4. if the portal doesn't have both, the object doesn't get teleported.

3.2.4 Warp_Objects Function

Objects in the Non-Euclidean Area will be warped differently depending on the type of the Non-Euclidean Area, if the Non-Euclidean Area is turned on.

Process:

1. Game Developer needs to define an object as a player and/or camera.
2. If Player/Camera is collapsing with the Non-Euclidean Area;
3. System will check if the Non-Euclidean Area is turned on,
4. If it is turned on, objects will be warped according to type of the non-euclidean area,
5. if it is turned off or a player/camera is not identified, nothing will happen.
6. When the Player/Camera leaves the Non-Euclidean Area, the area will return to normal.

3.2.5 Change_Camera Function

Game Developers may want to use their own camera to change perception of the Non-Euclidean Area so depending on the type of the Non-Euclidean Area, the system will change the camera defined to the one the user wants.

Process:

1. Game Developer needs to define an object as a player and/or camera.
2. Game Developer needs to define a camera that will be changed with the current one.
3. When Player/Camera collapses with the Non-Euclidean Area, the camera will be changed with the one the Game Developer has chosen.
4. When Player/Camera leaves the Non-Euclidean Area, the camera will be changed with the default one.

3.2.6 Basic Game Engine Functions

Other than our project's functionality, a basic game also needs to contain some functionality. These are;

- Gravity,
- Object Collision,
- Character movement,
- Character-Object Interaction and many more individual functionalities based on the needs of the game Game Developer plans to develop.

3.3 Performance Requirement

Performance is highly relevant to the project the user is working on, scaling with its size. But as a minimum, Unity 2019.1 requires:

- Windows 7, 8, 10, 64-bit operating system,
- graphics card with DX10+ capabilities,
- and an appropriate amount of space in the disk.

3.4 Design Constraints

Software will be designed as a general non-Euclidean add-on to help developers build non-Euclidean game projects. As the plot or gameplay of their project may require specific elements that we can't predict, this project will only cover common elements that can be used in a variety of non-Euclidean game projects.

3.5 Software System Attributes

3.5.1 Portability

Non-Euclidean Game Engine is an add-on for Unity thus it can be installed to any Unity instance in any computer with ease.

3.5.2 Performance

Performance scales with the needs of the user, the base add-on only requires Unity's system requirements to work.

3.5.3 Usability

Add-on will have an understandable and easy to use interface.

3.5.4 Availability

Add-on will work on Unity 2019+ and all operating systems that Unity 2019+ works on.