Software Design Document

for

INART VR Project

Version 1.0 approved

Prepared by: Kevin Diaz-Lopez, Taha Kamran, David Hermosillo, Zudong Li, Kiet Hoang, Jessy Francisco, Steve Galvan, Noah Castro, Daniel Ramirez

Institute For Interactive Arts, Research, & Technology (INART)

May* 14, 2021

Revision History

Name	Date	Reason For Changes	Version
Preliminary Draft	12/09/20	First draft	1
Final Draft	5/10/21	Final changes	2

1. Introduction

1.1 Purpose

The purpose of this document is to outline the design plan and specifications for the functionality of INART VR. This document will expand on the features described in the Software Requirements Specifications and explain the functionality of each one. The architecture, design, and implementation of INART VR will be fully defined in this document.

1.2 Document Conventions

Typographical Conventions	Description
Bold text	Title/Section heading
Bullet points	Unordered lists
Numbered lists	Ordered lists

1.3 Intended Audience and Reading Suggestions

This document is intended for:

- Developers who can utilize this document to easily overview the design of the system to improve or adjust features..
- Liaisons who wish to overview the document in order to suggest design improvements.
- Advisors for evaluation to determine if the document is sufficient.

1.4 System Overview

The INART VR system is a 3D photo realistic environment to be used in a film production developed in Unreal Engine 4. The environment shall act as an interactive book for the user. The system will also provide features to simplify the virtual film production process by giving producers tools that will automate tasks and reduce technical effort. The system will provide adjustable camera settings and automated camera movements. The system will also be able to adjust environmental lighting. The system will also provide features to adjust environment lighting.

2. Design Considerations

2.1 Assumptions and Dependencies

This product assumes the following:

- The user has access to a computer with the required hardware to run Unreal Engine 4.

- The user has access to video and audio equipment.

- The user has access to chroma key technology.

- The user has access to the internet.

2.2 General Constraints

- Computers lacking sufficient hardware will struggle to run Unreal Engine 4
- Development is limited to C++

2.3 Goals and Guidelines

- INART VR will provide a photorealistic 3D environment to be used as a setting for a film.
- INART VR will provide tools to simplify camera operations.
- INART VR will provide tools to easily adjust environment visuals.
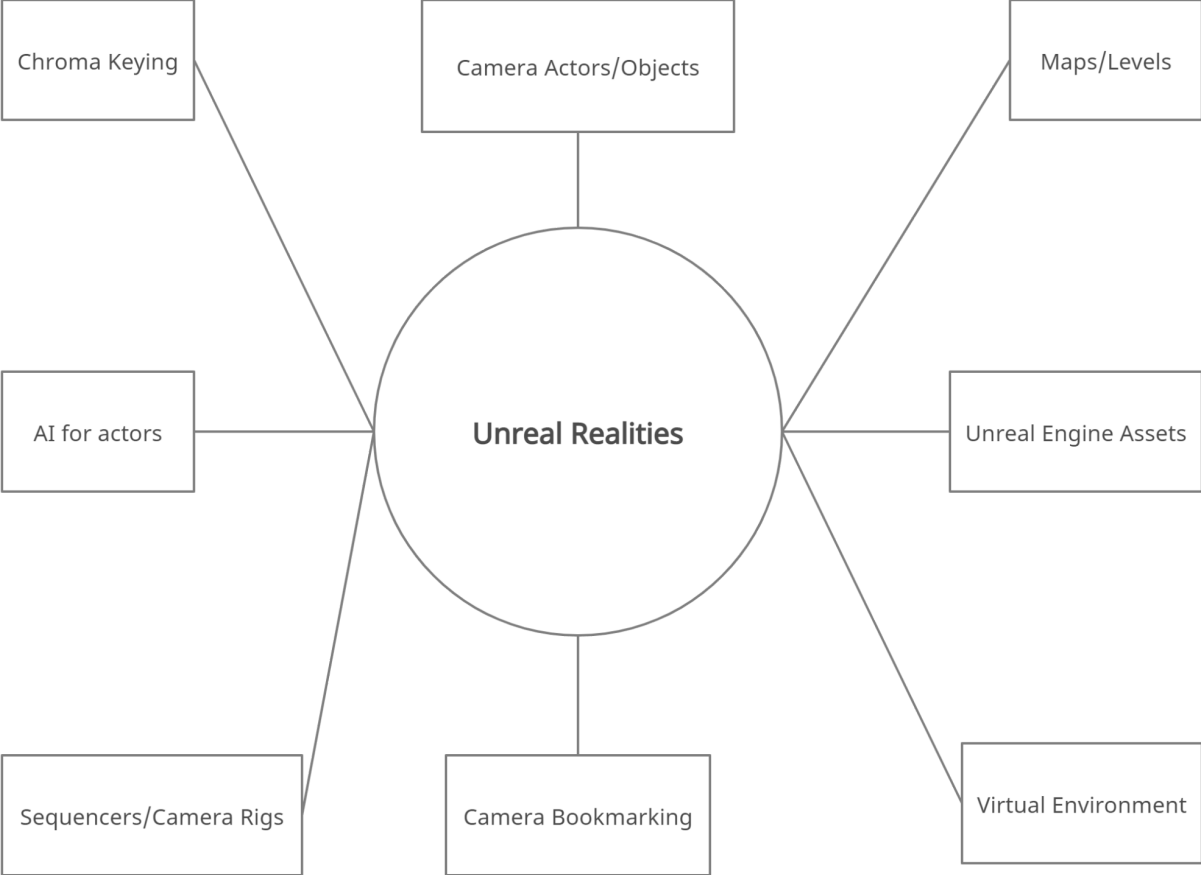
2.4 Development Methods

The development method that will be used is the Agile Development method in which developers will determine solutions through collaboration and communication.

3. Architectural Strategies

- For the 3D environment
    - The environment was developed in Unreal Engine 4 in order to create a photorealistic setting.
    - The user shall be able to interact within the environment through various objects.
    - Assets obtained from the Unreal Engine Marketplace shall be modified and repurposed in order to create unique settings.
    - The environment shall contain the following settings: a desert, city, office, book nook, and hallway.
- The system needed to deploy accurate lighting effects for each scene
    - The lighting needed to accurately match the scenery around them
    - Lighting reflection needs to reflect onto the actors to add to the photo-realism
    - The lighting in virtual worlds needs to be transferable to other actors and environments
- For the camera tools
    - C++ was used for development of the tools as it is used as a base for blueprint classes.
- For the virtual characters
    - The virtual characters shall function through artificial intelligence in order to direct their movement animations and interaction with the user.
    - The virtual characters follow behavioral trees that determine its actions.
    - Algorithms that check the characters radius and moves their location.
    - Determine a predefined area in which the characters can traverse.
- The system will be reusable with any 3D environment.

4. System Architecture

Chroma Keying

Camera Actors/Objects

Maps/Levels

AI for actors

Unreal Realities

Unreal Engine Assets

Sequencers/Camera Rigs

Camera Bookmarking

Virtual Environment

5. Policies and Tactics

5.1 Choice of which specific products used

Unreal Engine 4, Visual Studio, C++, Open Broadcaster Software

5.2 Plans for ensuring requirements traceability

The development team will use Github to do version control.

5.3 Plans for testing the software

Testing the system will be done through interactive use of the system to manually detect errors and through feedback from users.

6. Detailed System Design

6.1 Environment Module

6.1.1   Responsibilities

This module will be a photorealistic 3D environment and will serve to have interactive objects and ability to modify textures.

6.1.2   Constraints

Environment must be run on Unreal Engine 4.

6.1.3   Composition

Portable environment that can be imported onto Unreal Engine 4.

6.1.4   Uses/Interactions

The user will be able to use this module as their virtual background for filmmaking.

6.1.5   Resources

The Unreal Marketplace for necessary assets.

6.2 Camera Module

6.2.1 Responsibilities

The camera module allows for filmmakers to record inside the environment and serve the responsibilities of; automating camera movements, adjusting camera focus, adjusting camera depth, ability to add multiple cameras, bookmark cameras, creation of defined paths, and ability to change camera angles.

6.2.2 Constraints

The camera's capabilities may be limited due to software restraints in Unreal Engine 4.

6.2.3 Composition

The camera module will be contained in a menu that allows the user to adjust camera settings defined in 6.1.1.

6.2.4 Uses/Interactions

The use of this module will be to record inside the 3D environment to mimic the use of a real physical camera. The user will have start and stop buttons that begin or pause camera recording. The user will also have the ability to change camera settings or add multiple cameras.

6.2.5 Resources

Unreal Engine 4 camera classes.

6.3 Lighting Module

6.3.1 Responsibilities

This module will be responsible for adjusting environment lighting, transfer lighting to the actors, change light sources, allow the ability to change the SkySphere, and allow for adding multiple light sources.

6.3.2 Constraints

The lighting capabilities may be limited due to software restraints in Unreal Engine 4.

6.3.3 Composition

The lighting module will be contained in a settings menu in which the user can adjust the lighting to their needs.

6.3.4 Uses/Interactions

The user will be able to use this module to adjust lighting visuals.

6.3.5 Resources

Unreal Engine Lighting Settings

6.4 Recording Module

6.4.1   Responsibilities

The recording module will allow the user to screen record, import video, import audio, and have the option to chroma key videos in the set.

6.4.2   Constraints

For chroma keying, the user needs proper chroma key technology such as a green screen.

6.4.3   Composition

The lighting module will be contained in a settings menu in which the user can adjust the lighting to their needs.

6.4.4   Uses/Interactions

The user will interact with various buttons that function to fulfill the responsibilities of this module.

6.4.5   Resources

Open Broadcaster Software for video recordings

6.5 Mannequin Module

6.5.1   Responsibilities

This module will serve as a stand-in to allow producers to test for technical purposes. The mannequin shall be allowed to be placed in the virtual environment, adjustable to scaling, ability to change the mannequins position

6.5.2   Constraints

The lighting capabilities may be limited due to software restraints in Unreal Engine 4.

6.5.3   Composition

The lighting module will be contained in a settings menu in which the user can adjust the lighting to their needs.

6.5.4   Uses/Interactions

The user will be able to place the mannequin wherever they desire in the environment.

6.5.5   Resources

Unreal Engine Character Blueprint

6.6 Virtual Character Module

6.6.1   Responsibilities

The virtual character module allows for the filmmaker to use it to interact with the user and serve the responsibilities of; moving around the environment, different animations and moving to set locations.

6.6.2   Constraints

The module's capabilities may be limited due to software restraints in Unreal Engine 4.

6.6.3   Composition

The module will be contained in the blueprint menu

6.6.4   Uses/Interactions

The use of this module will be to interact with the user. The user will have the ability to interact with the character by pressing specific keys to start the animation, move it to a set location, or let it roam around.

6.6.5   Resources

Unreal Engine 4 behavior trees

7. Database Design

Not applicable.

8. User Interface

8.1 Overview of User Interface
The main user interface allows the user to enter the environment or enter a separate interface in which the user will be allowed to change various settings such as environmental, camera, and lighting aspects. The interface will provide options for tools such as camera bookmarking and camera rig rails to be placed in the environment.

8.2 User Interface Flow Model
A discussion of screen objects and actions associated with those objects. This should include a flow diagram of the navigation between different pages.

10. Requirements Validation and Verification

6.1.0	**Mannequin**
6.1.1	The system shall be able to place mannequin(s) in the virtual set
6.1.2	The system shall allow the user to change the mannequin's position and other body parts individually
6.1.3	The system shall allow the user to animate the mannequin
6.1.4	The mannequin(s) shall be adjustable in scaling

6.2.0	**Recording**
6.2.1	The system shall allow screen recording
6.2.2	The system shall allow the client to move the recording camera within the software

6.2.3	The system shall allow the use to import video
6.2.4	The system shall allow the user to import audio
6.2.5	The system shall allow livestream recording
6.2.6	The system shall allow the use of multiple cameras
6.2.7	The system shall allow the user to switch between multiple cameras
6.2.8	The system shall have the option to chroma key videos in the virtual set

6.3.0	**Camera Movement**
6.3.1	The system shall allow camera movement on the X-axis
6.3.2	The system shall allow camera movement on the Y-axis
6.3.3	The system shall allow camera movement on the Z-axis
6.3.4	The system shall allow camera rotations
6.3.5	The system shall provide automated camera movements
6.3.6	The system shall allow the option to change camera movement speed
6.3.7	The system shall allow the camera to track a virtual object.

6.4.0	**Lighting**
6.4.1	The system should have the option to transfer the virtual environment's lighting onto the actors
6.4.2	The system shall allow the user to adjust the environment's lighting
6.4.3	The system shall allow the user to change light sources
6.4.4	The system shall allow the user to change the SkySphere
6.4.5	The system shall allow the user to add more light sources

6.5.0	**Environment**
6.5.1	The system shall create objects that the actors can interact with
6.5.2	The system shall create objects that the client can interact with
6.5.3	The system shall allow the user to import levels

6.6.0	**Virtual Character**
6.6.1	The system shall be able to place virtual character(s) in the virtual set
6.6.2	The system shall allow the user to change the character's position and other body parts individually
6.6.3	The system shall allow the user to animate the character
6.6.4	The character(s) shall be adjustable in scaling
6.6.5	The character should be intractable by the user and actor
6.6.6	Virtual characters shall be capable of interacting with the user through artificial intelligence

11. Glossary

Chroma key - A technique in which a block of color can be replaced by a video image. Often done through a green screen.

3D - Three dimensional

UE4 - Unreal Engine 4

OBS - Open Broadcast Software

AI - Artificial Intelligence

C++ - A object-oriented programming language

GitHub - A code hosting platform for collaboration and version control.

12. References

Unreal Engine 4 Documentation: https://docs.unrealengine.com/en-US/index.html

Unreal Engine 4 asset store: Marketplace - UE Marketplace (unrealengine.com)

Open Broadcaster Software: https://obsproject.com/docs/

C++: https://docs.microsoft.com/en-us/cpp/?view=msvc-160