On Demand Texturing and Interactivity with Generative AI and Neural Radiance Field (NERF)-like models

Group Members:
Brendan Bain: bbain7@uwo.ca
Brandon Nathan Marks: bmarks7@uwo.ca
David Alter: dalter4@uwo.ca

Supervisor:
Dr. Brent Davis
Department of Computer Science and Psychiatry

4470Y Software Maintenance and Configuration Management
February 12th 2024
Progress Report 2

Updated Project Objectives and Requirements

We've decided to focus our development efforts solely on traversable structures with collision and omit background structures and decoration structures in the interest of time. Since traversable structures are more complex and have more functionality, the other structures can easily be added to the system past the end of this project.

This scope change has resulted in the omission of old **objective 4** (Have the NeRF model generate all structure types) and old **objective 9** (Have the NeRF model generate all structure types with collision). The project goal is maintained through this change: to create a system that is capable of generating structures at runtime that look appropriate in the surrounding virtual environment, and mimic real-life objects.

OBJECTIVES:

O1: Have the project setup with a blank VR scene

O2: Create a generic wireframe representing structure and implement marching cubes algorithm to add mesh

O3: Have a trained NeRF model that generate a structure

O4: Optimize performance considering headset specifications

O5: Have the model working with the Meta Quest Pro Headset

O6: Add collision to the outside of a structure

O7: Add collision to the inside of a structure

O8: Use a heuristic to make sure the structure is navigable by an avatar

SYSTEM REQUIREMENTS:

Due to the scope change, old **feature 2** (multiple types of structures to generate) has been removed along with old **feature requirements 4.1** (the system should be capable of generating background structures without collisions with any other objects. Background structures are objects that are not meant to be interacted with) and **4.2** (The system should be capable of generating decoration structures, these structures may collide with other objects but are not meant to be interacted with). Old **feature 4** has been renumbered to 3, and old feature 4.3 has been broken down into 3.1 and 3.2

Feature 1: Virtual Reality Headset Compatibility

- **QR 1.1**: The system is compatible with the Meta Quest Pro headset and performs as expected in a real-time virtual reality environment.

Feature 2: Rendering of Objects

- **FR 2.1**: When a user looks in the direction of a particular structure, that structure should appear as rendered in the virtual reality environment.
- **FR 2.2**: Given that a user looks at a particular structure, that structure should appear to be its specified size.

Feature 3: Multiple Structure Interaction Types

- **FR 3.1**: The system should be capable of generating traversing structures, these are structures that are meant to be interacted with, such as buildings that you are able to walk into with collision on the outside and inside.
- **FR 3.2:** When a user walks into a wall on the outside or inside of a building, the user cannot pass through.

Feature 4: Performance Optimization

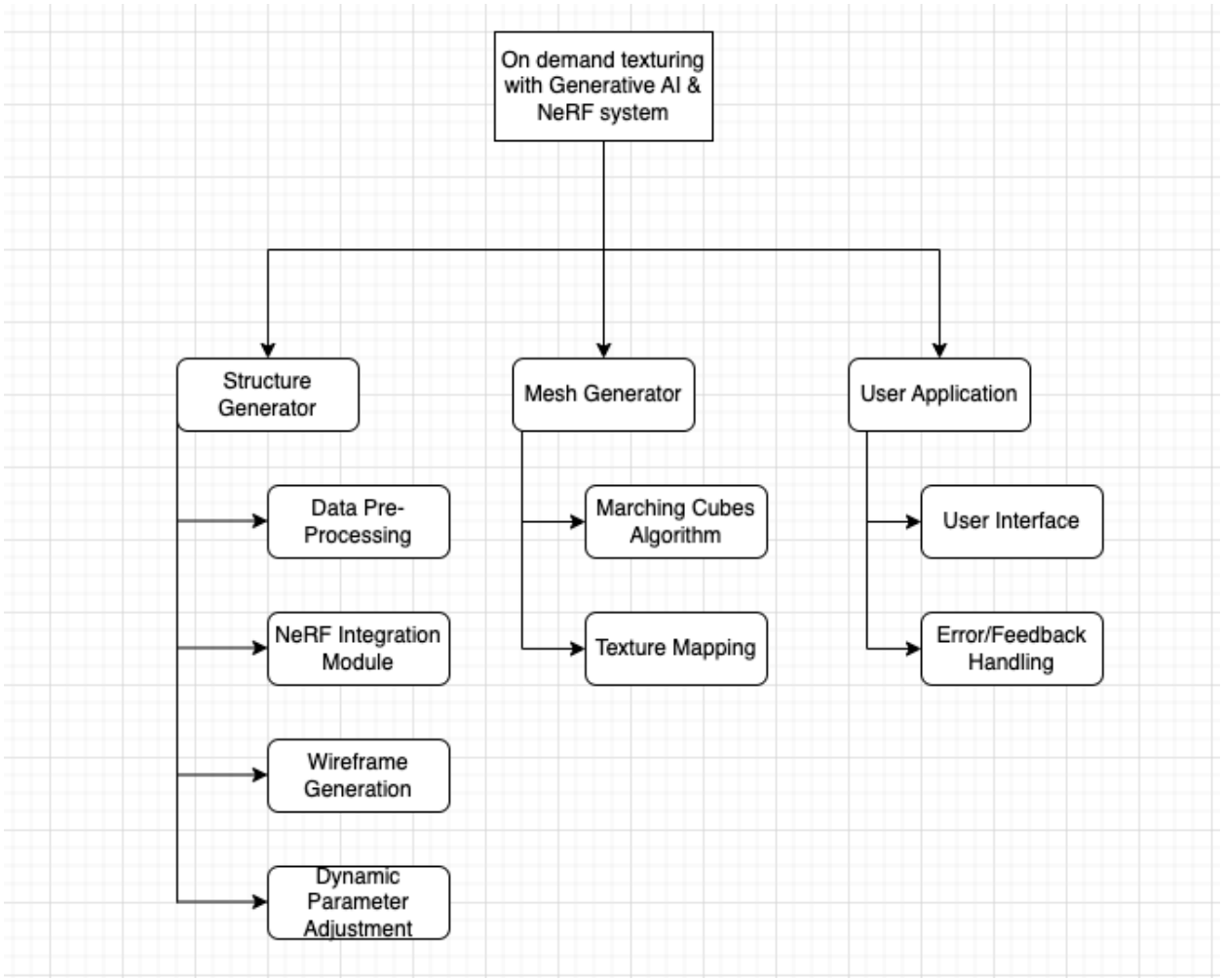
- **QR 4.1**: Optimize system performance considering the specifications of the VR headset in use (Meta Quest Pro).
- **QR 4.2**: The system as well as the environment running it should not drop below 60fps.

Feature 5: Scalability

- **QR 5.1**: Design the system to handle varying levels of complexity and scale for different virtual reality environments.

System Design Issues

Functional Decomposition Diagram



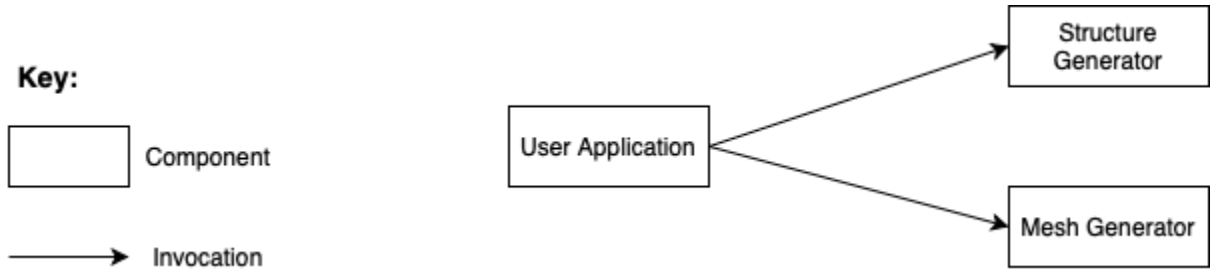
In the above diagram, we've decomposed our three main functions into smaller, more manageable subfunctions.

1. **Structure Generator** - top level module responsible for the generation of wireframe structures in the VR environment.
    a. **Data Preprocessing**: sub-function which prepares input data for NeRF model, pre-processes 3D models, textures, ect.
    b. **NeRF Model Integration Module (third party / open source)**: sub-function used to integrate the NeRF model into the system, load up the model, and set up necessary infrastructure for referencing.
    c. **Wireframe Generation**: executes the NeRF model and generates wireframes of structures based on input parameters.
    d. **Dynamic Parameter Adjustment**: dynamically adjusts NeRF model parameters based on user or system feedback, allowing for real time customization of structures.
2. **Mesh Generator** - top level function which is responsible for generating meshes for the wireframes generated by the "Structure Generator" module.

a. **Marching Cubes Algorithm (open source)**: sub-function responsible for implementing the marching cubes algorithm to convert wireframes into 3D meshes. Essential for generating meshes from volumetric data.
b. **Texture Mapping**: sub-function which applies textures and refines the generated meshes, ensures smooth surfaces and enhances visuals.

3. **User Application -** This function represents the part of the system which the user interacts with to control structure generation.
    a. **User Interface**: provides the user with the graphical user interface (GUI) allowing users to control structure generation.
    b. **Error/Feedback Handling**: handles user input and provides feedback to the user through the GUI, such as progress indicator and relevant error messages.

Component-Connections Diagram

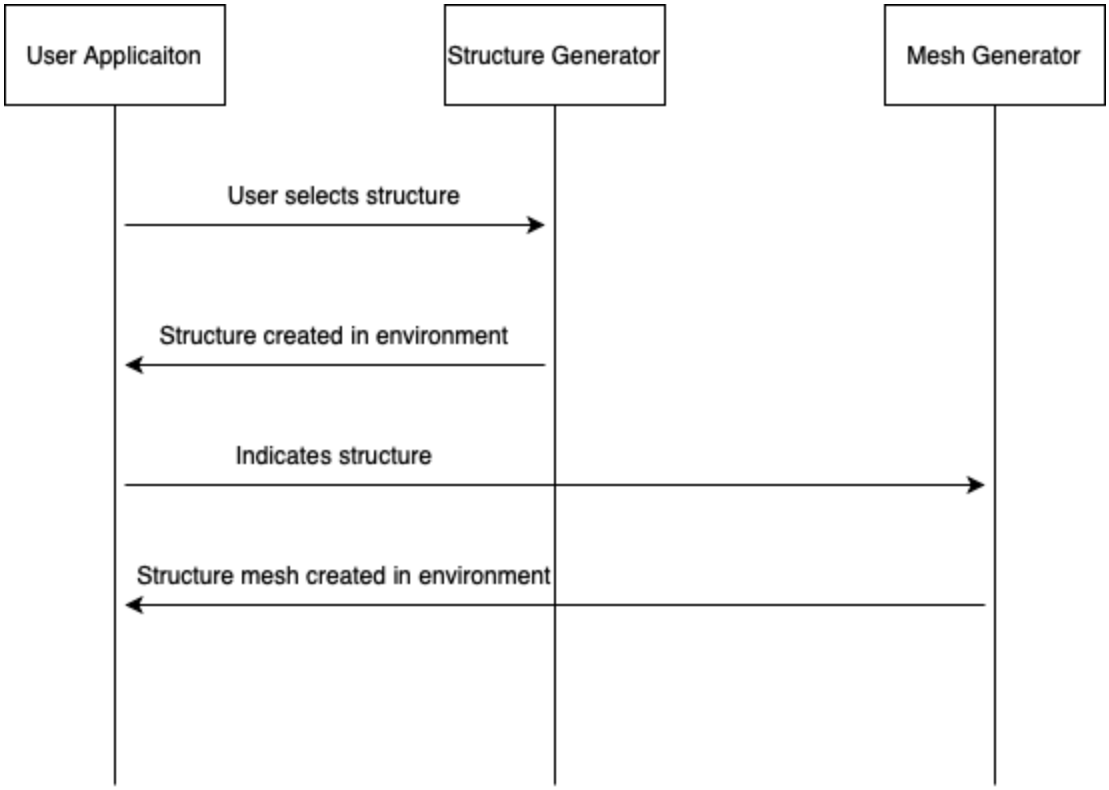


The “Structure Generator” component in this diagram contains the NeRF model that will generate wireframes representing structures in the virtual environment. In “Structure Generator”, the NeRF model will be third-party and other aspects of “Structure Generator” will be implemented from scratch. The “User Application” component here represents the parts of the system that the user directly interacts with, including the user interface where they would control what gets generated. The “Mesh Generator” component in this diagram uses the marching cubes algorithm to generate meshes for the structures created by “Structure Generator”. “User Application” will be implemented from scratch. In “Mesh Generator", the marching cubes algorithm will be third-party and other parts of it will be implemented from scratch.

The diagram has been constructed in this way because in the system, the user can select what structures get generated, so there must be a component representing this capability, hence why “User Application” was created. In this system, the creation of virtual environments involves two steps. Firstly, the system must generate wireframes representing structures which users can interact with, which is why I created “Structure Generator”. Then meshes for those wireframes must be formed, which is why I created “Mesh Generator”.

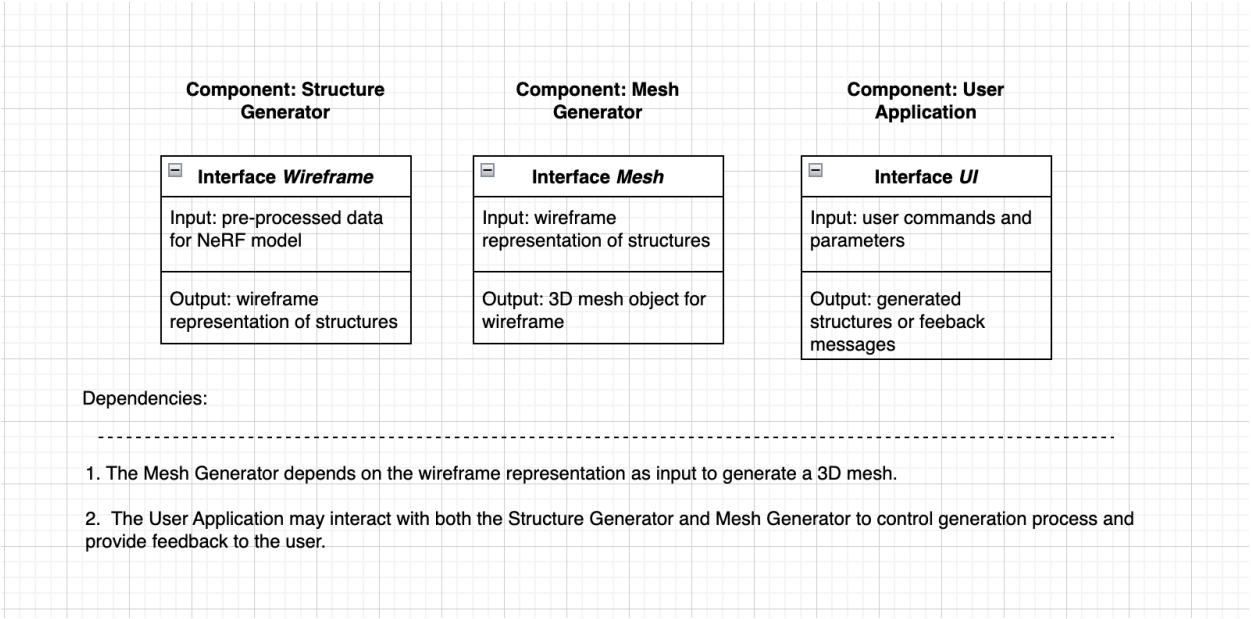
Sequence Chart

User selects structure to be generated



The components here are the same as those from the Component-Connections diagram above. This sequence chart represents the data flow that occurs when the user tries to have a structure generated in the virtual environment. Firstly, in the User Application, when the user selects a structure to be generated in the virtual environment, the User Application sends information about that structure to the Structure Generator and the Structure Generator creates a generic wireframe of that structure in the virtual environment. Then the User Application sends information about that wireframe to the Mesh Generator, and the Mesh Generator creates a mesh for that wireframe to finish completing the structure which the user can then interact with.

Interface Specifications



The Structure Generator component generates *wireframe* representations of structures based on pre-processed data. These wireframes serve as input for the Mesh Generator, which then generates 3D *meshes*. The User Application (*UI*) controls the structure generation process, sending user commands and parameters to both the Structure and Mesh Generators and receiving feedback messages. Additionally, dependencies are highlighted at the bottom of the diagram, illustrating the reliance of the Mesh Generator on the output of the Structure Generator.

System Testing

Sanity tests based on feature requirements followed by bottom-up integration testing based on the functional decomposition diagram hierarchy are planned. Afterwards, system testing with supervisor Dr. Davis will verify whether the system conforms to the requirements as specified. Tests marked with N/A under the “actual result” column have not been performed yet.

Test #	Test Type	Test Purpose	Expected Result	Actual Result	Pass/Fail	Fixed?
1	Black Box	**QR 1.1** Check the systems compatibility with the Meta Quest Pro headset	Headset loads virtual reality environment in real-time	Headset loads virtual reality environment in real-time	**Pass**	N/A
2.1		**FR 2.1**: When a user looks in the direction of a particular structure, that	Structure appears as rendered in the virtual reality			

	Black Box	structure should appear as rendered in the virtual reality environment.	environment when the user looks in that direction	N/A		
2.2	White Box	**FR 2.2**: Given that a user looks at a particular structure, that structure should appear to be its specified size.	Structure appears to be specified size	N/A		
3.1	Black Box	**FR 3.1**: The system generates a building the player can enter	The user is able to walk into the building through the entrance	N/A		
3.2	Black Box	**FR 3.2**: The system generates a building the player can collide with from the outside	The user cannot move through the building when contact is made with the outside of the structure	N/A		
3.3	Black Box	**FR 3.2**: The system generates a building the player can collide with from the inside	The user cannot move through the building when contact is made with the inside of the structure	N/A		
4.1	White Box	**QR 4.1**: Make sure CPU usage does not exceed 70% at any point while system is running	CPU usage remains under 70%	N/A		
4.2	White Box	**QR 4.1**: Make sure GPU usage does not exceed 70% at any	GPU usage remains under 70%	N/A		

		point while system is running				