Final Project Report

I. Introduction

Our Virtual Flood Simulator is an immersive VR experience that allows users to visualize and understand the impact of flooding in three major global cities: Hong Kong, London, and New York. Using realistic 3D environments and dynamic water physics, the simulator demonstrates how rising sea levels, extreme weather events, and urban infrastructure interact during flood scenarios.

Flooding is a growing threat due to climate change, and many coastal cities are at risk. This simulator helps users grasp the real-world consequences in an engaging way. By exploring flood scenarios in different cities, users can compare how varying infrastructure affects resilience. This tool can be used for disaster preparedness training, urban planning discussions, and motivating action against climate change.

By experiencing floods in VR, users gain a deeper emotional and intellectual understanding of the risks, making abstract climate threats feel immediate and urgent.

II. Background

Urban Vulnerability & Adaptation Strategies

Hong Kong: Heavy rainfall and typhoons cause flash floods; the city relies on drainage systems but faces challenges due to dense urbanization.

London: The Thames Barrier protects against storm surges, but future sea-level rise may require upgrades.

New York: After Hurricane Sandy (2012), the city invested in floodwalls, wetlands, and zoning laws.

Data-Driven Flood Modeling

Projects such as FloodUp or NASA’s ARIA use real-world hydrological data to predict flood extents.

Earth Under Water VR let users save cities by choosing mitigation strategies

III. Methods

Tools

Unity the game engine for creating the VR simulation, handling interactions, and managing scenes

Blender for creating and modeling 3D assets such as buildings, roads and environmental props

VR SDKs (OpenXR, Meta XR) and possible asset libraries or plugins for dynamic simulation

How does it work?

To create 3D city models for your VR flood simulator, start by extracting OpenStreetMap (OSM) data (buildings, roads, terrain) in. osm or .geojson format using tools such as Overpass Turbo. Convert this 2D data into 3D by extruding buildings based on OSM height tags (e.g., building:levels) using tools like OSM2World (outputting .obj files) or Unity/Unreal plugins (e.g., OsmSharp). Enhance realism by merging elevation data (e.g., SRTM) and applying textures. Finally, import the 3D model into Unity/Unreal, add flood simulations (via fluid dynamics or shaders), and optimize for VR performance. This open-source workflow lets you generate any city dynamically for immersive flood visualization

A simple water shader, involves vertex displacement using a time-based sine wave function to simulate movement and a fragment shader to add color, transparency, and lighting effects. This shader also includes Fresnel effects to simulate how water appears more reflective at glancing angles, and simple UV distortion to mimic underwater warping. The shader code interacts with Unity's lighting system through surface shader techniques or through custom lighting models if done in a pure vertex-fragment shader.

Reference: https://www.youtube.com/watch?v=VgFkFZZUCbA

IV. Limitations

Data Accuracy & Completeness

OpenStreetMap (OSM) data may have missing/inaccurate building heights, incomplete terrain, or outdated infrastructure.

Performance Optimization

Challenges maintaining a high stable frame rate with complex urban detail, managing rendering loads when simulating many dynamic elements simultaneously

User Experience Constraints

Balancing immersion with comfort such as reducing motion sickness, optimizing controller responsiveness, limited resources to simulate full-scale city dynamics under strict time constraints.

Validation Challenges

Hard to verify if simulated floods match real-world behavior.

V. Future work

Enhance Simulation Complexity

Add dynamic traffic systems, realistic pedestrian behaviors, and adaptive weather patterns. Implement AI-driven simulations for city services (emergency response, public transport)

Expand Data Integration

Incorporate real-world data feeds (traffic data, population statistics) to make simulations responsive to actual city conditions. Develop analytical tools for user feedback and performance monitoring to drive continuous improvements.

GPS Integration for Real-World Location Tracking

Adding GPS functionality to enhance realism by letting users see their real-world location in the virtual city.