Project Abstract

RealityLib is an open-source virtual, mixed, and augmented reality framework developed as an extension of Raylib. The project addresses the lack of native C libraries for VR development by providing developers a fun and simple framework for building VR games. Development progressed through a series of game jams that evolved from Raylib exploration to OpenXR demos, eventually resulting in a library that maintains Raylib's philosophy as a framework rather than a full game engine.

The framework focuses on adding OpenXR support for Meta Quest devices while intentionally avoiding advanced game engine features. The result is a library that simplifies VR development through native C code, enabling the creation of 3D objects, controller input tracking, hand-tracking, and basic collision support in VR environments.

Introduction

Virtual Reality development presents significant barriers to entry, particularly for developers who prefer working in C. Current options typically require:

- Using complex game engines with steep learning curves
- Implementing OpenXR directly, with most examples in C++ rather than native C

The primary goal of RealityLib is to make VR game development easily accessible to game developers who are familiar with mobile game development, RayLib, or something similar. The other goals are to be educational, as RealityLib allows developers to modify it and learn more about VR game development

Key questions include:

- How can OpenXR functionality be integrated with Raylib?
- Can we maintain Raylib's simplicity while supporting VR?
- How can we effectively support Meta Quest devices?

Methodology

This project followed an iterative development approach conducted throughout the 2025-2026 academic year:

Learning Phase:

- Individual game jams to master Raylib fundamentals
- Exploration of OpenXR specifications
- Review of VR frameworks and requirements

Development Phase

- Extension of Raylib API with OpenXR integration
- Implementation of VR input handling systems
- Development of build pipeline for Android APKs

Testing Phase

- Series of VR-specific game jams
- Iterative refinement based on development experience
- Performance testing on Meta Quest 3S hardware

Tech Stack

- Native C for all development
- OpenXR for VR/MR/AR capabilities
- Gradle and CMake for Android builds

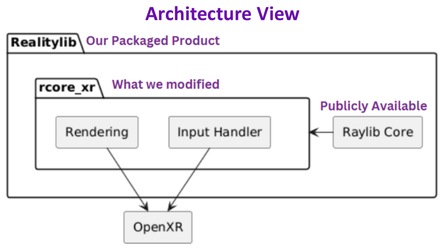
ROSE-HULMAN INSTITUTE OF TECHNOLOGY

RealityLib

CSSE Capstone Project

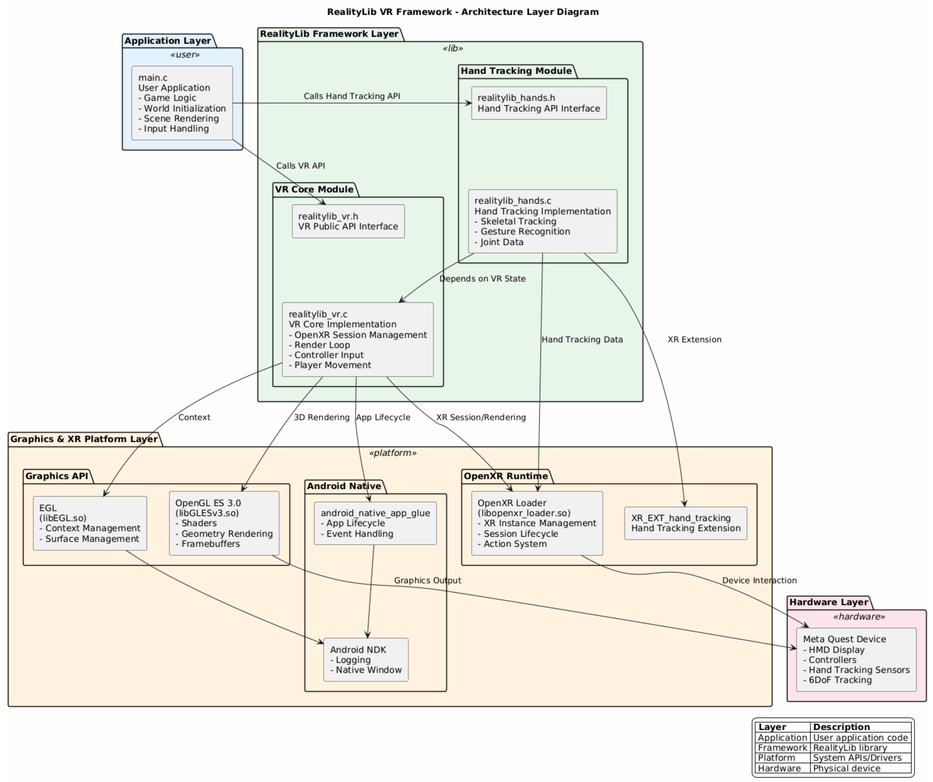
Harrison Carpenter, Irvan Wang, Wenbo Yang, Ian Zhu
Department of Computer Science Software Engineering
Client: Steve Hoelle
Advisor: Kim Tracy
Rose-Hulman Institute of Technology
5500 Wabash Ave, Terre Haute, IN, 47803

Basic Architecture



All code written in native C, with OpenXR integration for VR/MR/AR capabilities

Detailed RealityLib Architecture



Current Progress

- Successfully rebuild RealityLib
- Setup RealityLib on Meta Quest 3s
- Enable VR hand-tracking feature
- Enable VR controller feature
- Setup initial world
- Write demo game design doc
- Complete demo game

Recommendations

Based on our development experience, we propose the following strategies
for VR/MR/AR framework development:

- **Maintain API simplicity:**

Raylib is popular of its ease of use. Keep VR functions follow similar patterns and naming conventions.

- **Focus on core functionality:**

Prevent adding too many features toward a game engine; provide only core VR control capabilities while letting developers design their own solutions.

- **Prioritize examples and documentation:**

Include comprehensive examples of each VR capability to guide and encourage developers new to VR.

- **Embrace iterative development:**

Validate new features through quick prototypes and game jams prior to integration
into the core framework.

Challenge & Solutions

Technical Challenges:

- Previous Framework does not work

Solution: Rebuild everything with latest version of dependencies

- Android deployment for desktop library

Solution: Developed hybrid Gradle/CMake build system

Design Challenges:

- Balancing simplicity with VR features

Solution: Carefully selected features aligning with Raylib's philosophy

- Scope creep toward game engine territory

Solution: Maintained focus on framework functionality only

Future Work

- **Vulkan Integration:**

Replace OpenGL with Vulkan for improved performance and modern XR features.

- **Expanded Examples Library:**

Create more demonstration applications showcasing framework capabilities.

- **Cross-Platform Support:**

Extend beyond Meta Quest to other VR/MR platforms.