VIP Senior, 2017, Summer

School of Computing & Information Sciences

<Web-VR Towards Virtual and Augmented Reality for the Web Education>

Student: <Daniel Khawand>, Florida International University

Mentor: <Francisco Ortega>, *Florida International University*

Instructor: Masoud Sadjadi, Florida International University

Problem

With the emergence of VR technology, researchers in education are now showing interest in education via virtual and augmented reality. ***The intention is to increase the retention rate of Computer Science students.***

Solution

Develop the scenes for each Data Structure using **three.js and the WebVR API.** Keep the architecture simple but allow the user to visualize the data being manipulated in real time along with the actual Data Structure in the Javascript code. Some static scenes will also be created.

Requirements

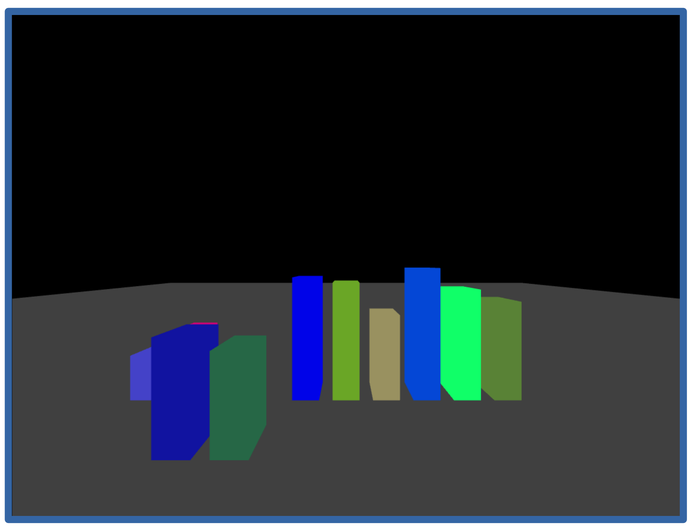
- User should be able to load scenes
- User should have input
- User should have the option between VR and non-VR
- All objects of the scene must load
- Data Structures visualizations should be clear and proper.

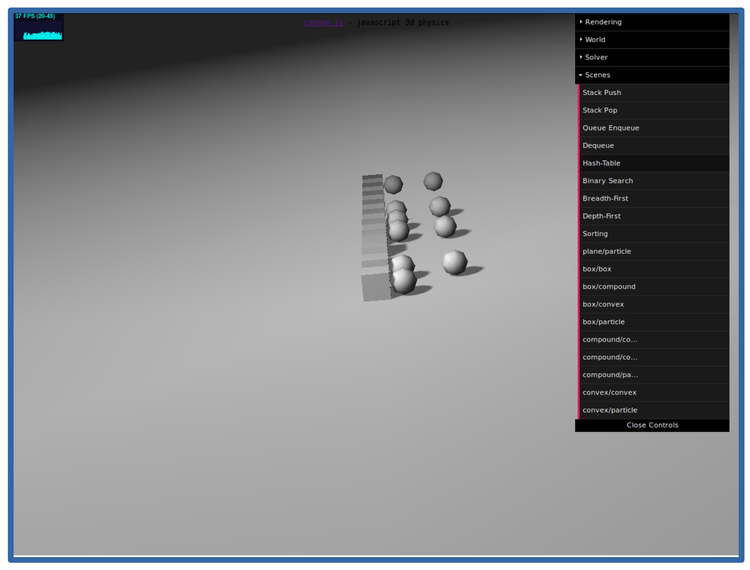
Current System

- One html page per scene
- Works with VR headsets on VR-specific browsers
- Non A-Frame scenes use the WebVR API directly
- Pulls objects from three.js and cannon.js
- Entity-Component system for A-Frame scenes
- A-Frame Scenes are loaded via npm run serv
- Developers will easily be able to implement new I/o features for users.
- Cannon.js scene selector for physics-based Data Structure representations (non-VR)
- MVC Architectural pattern for VR scenes that use Threejs

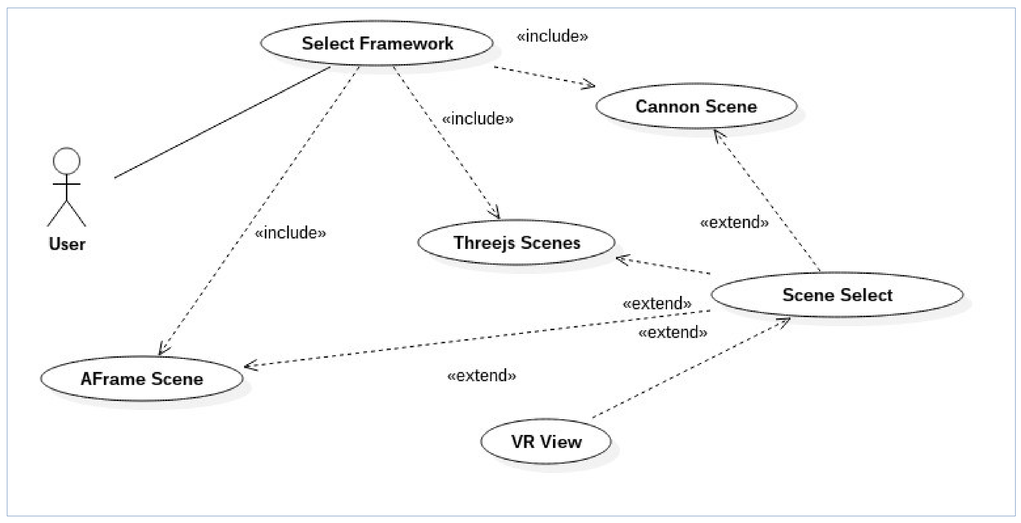
Screenshots

Left: Selection Sort Scene using WebVR API in threejs
Right: Hash-Table Collision 3D Scene using Cannon.js





Use-Case Design

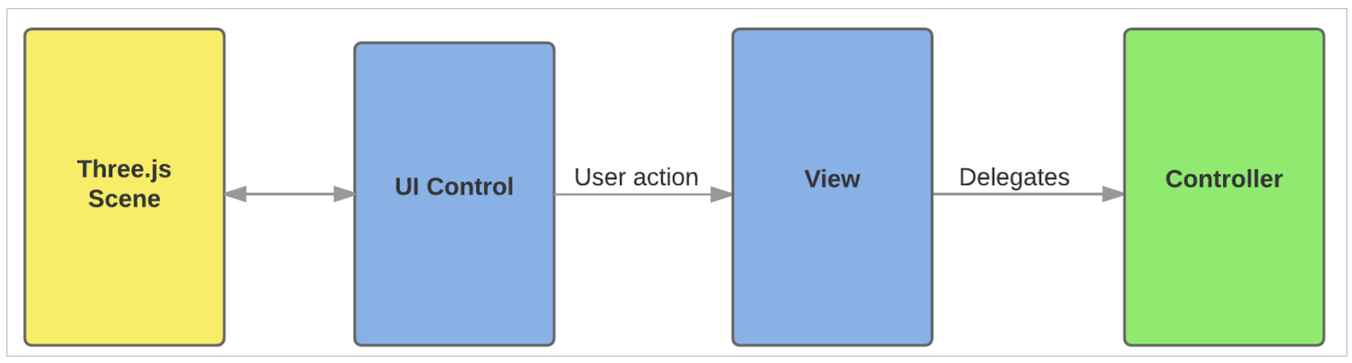


Summary & Objectives for Next Release

- Virtual and augmented reality tools will help students learn Data Structures.
- This version of the system is the first release.
- Future developers will have a lot of room to improve on this project
- A lot of potential for improvement and optimization, including making the programs more dynamic to user input and VR-Touch or Vr-Gamepad controls.
- Improve **Control** component of application and provide more **Views**

Implementation

The system is developed entirely in Javascript and HTML.
Threejs and Cannonjs were used for the scenes, using **MVC Architecture.**



Acknowledgement

The material presented in this poster is based upon the work supported by Dr. Francisco Ortega. I am thankful to the help that I received from my group members Hamilton Chevez, Bernardo Pla, Pachev Joseph, and Daniel Rivero.