WATOSCENGEN

WATonomous Road Generation

Gregory Desrosiers

Description

WATOSCENGEN creates multiple test scenarios for the simulation program WATonomous utilizes to test the feedback controller, sensors, and the robot operating system of a car, against randomized 3D environments to identify bugs that can't be reproduced in predefined situations.

WATonomous is one of the eight student design teams in North America participating in a three-year competition called the SAE AutoDrive Challenge. The team retrofits a Chevrolet Bolt with sensors and a computer to drive on its own instead of human intervention.

This module allows the team to perform sanity, regression, and system tests on the car in random environments with less requirements than on-site tests.

Specifications

Several road standards are followed:

Lane Width: 3.7 m
Minimum Stop Sign Height: 2.9 m [for a stop sign with area is 30 in x 30 in.]
Minimum Lateral Space from Road: 99 cm [stop signs in residential / commercial zones]

Additional specifications were made during the project as follows:

Maximum Angle of Turning: 20°
Straight Road Length: 100 m - 500 m
Minimal Space between Stop Signs: 20 m
Length of Curved Roads: 950 - 1050 m
Length of Individual Segments for Curved Roads: 10 - 100 m

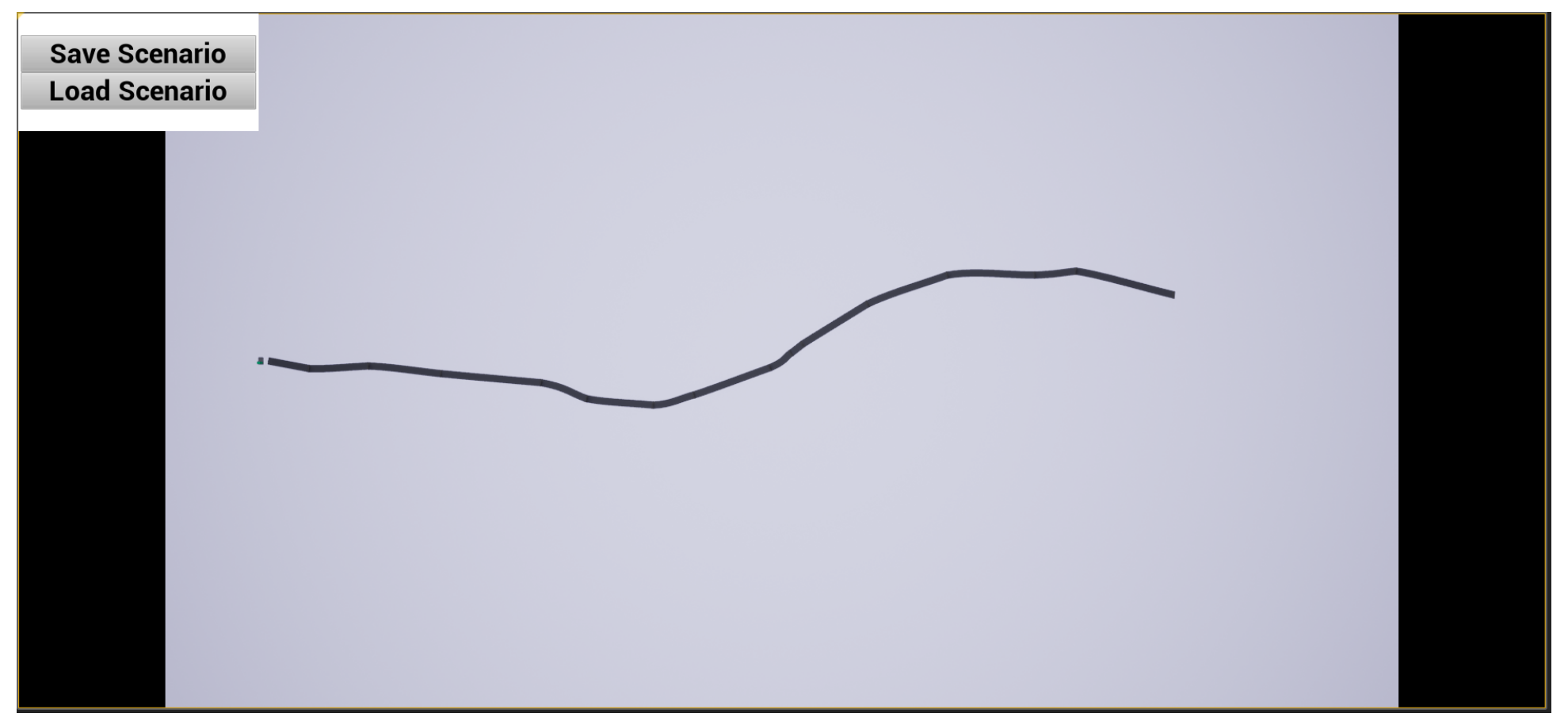
Highest speed in the challenge: 40 km/h
Curvy road navigation task speed: 3 - 8 km/h.

Elicitation Techniques

The project was developed using the agile development model, providing an opportunity to outline what's to be worked on, what's the progress, and to reprioritize tasks as needed. Short meetings were done daily.

Some elicitation techniques were utilized for agile effectiveness:

- Document Studies (documentation of Unreal Engine 4)
- Mockups and Prototypes (proof of concepts)
- Understanding the Work (prioritizing work)
- Creativity-based Elicitation (creative thinking involved for algorithms used to generate roads)
- Systemic Thinking and Brainstorming (consulting with team faculty advisors)
- Task Demo (tasks shown in person with discussion)
- Norms (making the output of the roads as realistic as possible)



Project Scope

This module creates scenarios based on three dynamic challenges from Year 1 of the SAE AutoDrive Challenge: a straight road with at most three stop signs, a straight road with static objects, and navigation on a curved road.

On May 5, 2018, the Year 1 competition took place in Yuma, Arizona. WATonomous won awards in the static challenges, but their modified Chevrolet Bolt couldn't reach the highest potential in the road challenges. To aid the team in tests with future changes, a procedural generation module was built for their existing simulator using Unreal Engine 4, a C++ based game engine produced by Epic Games, Inc.

The original project scope was to create an environment generator for full physical tests. As the competition is mainly for simple scenarios, though, it was cut down to the three dynamic challenges.

Motivation

Simulations provide a cost-effective mechanism for testing prototypes of designs and control. Because WATonomous can't know in advance what the road layouts are in the physical challenges, a mechanism to create road scenarios efficiently must be built while the program can recall scenarios where WATonomous identifies bugs with their self-driving code.

Computer simulations have less requirements than actual driving. To drive the Bolt, WATonomous needs drivers with licenses, a test track must be booked, and transportation must be meticulously planned. In addition, if the car breaks down, or accidents occur, repairs are required. Performing software tests thus becomes impractical because of time and money. With simulations, alone, the team only pays for computer hardware, electricity, and software. Using computers allows WATonomous to conduct tests in only minutes. This project provides a road generation model where the team won't have to create road environments by hand.

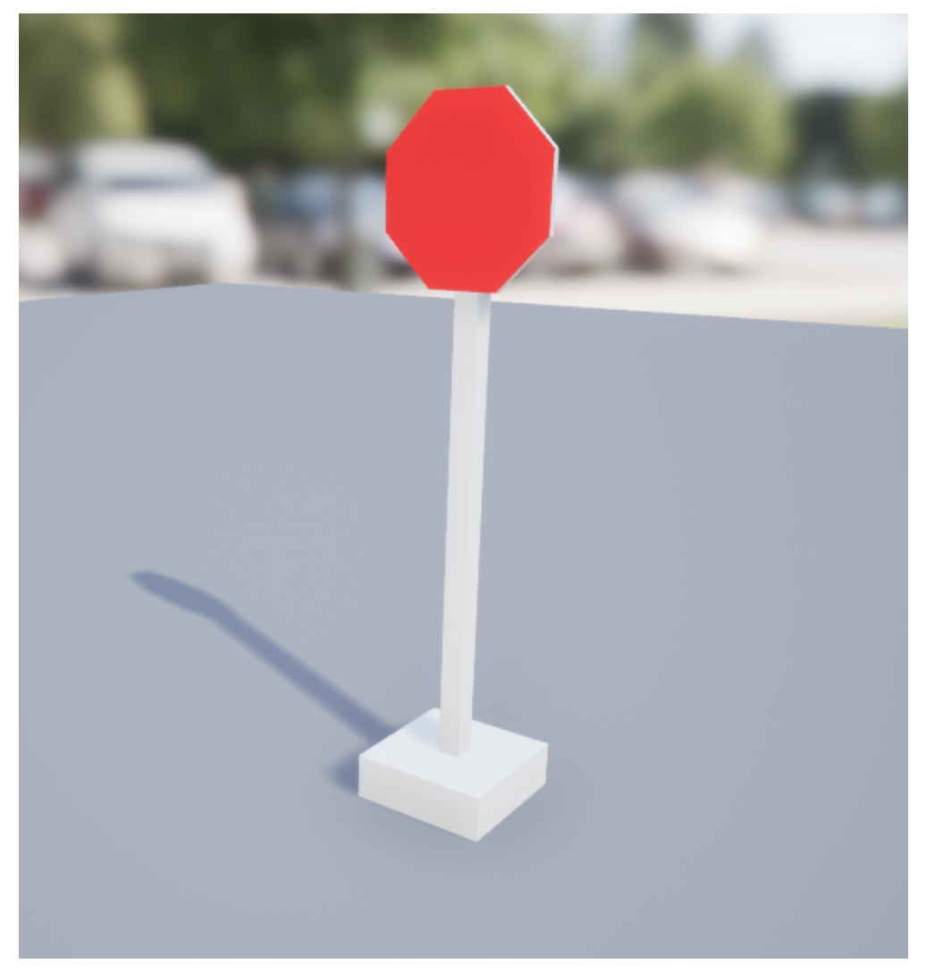
Graphics Implementation

Three key components in computer graphics are used: meshes, splines, and materials.

Meshes are a collection of 3D points called vertices, along with edges and faces, that defines the geometry of a 3D polygon, such as cubes. A spline, in mathematics, is a piecewise function defining a curved line passing several points called control points. A material is a group of coefficients defining how light interacts with the mesh; the base color, roughness, and so on, can be set to give the material a certain look and feel. Textures can also be applied to a mesh or a material; these are two-dimensional images that allows a mesh to simulate real world objects.

Meshes are used for road segments, a stationary bus, stop signs, and grass fields. Splines define the road's shape, using meshes as physical components. Materials are used to provide a mock look and feel in the generated environments.

On the right is a stop sign modelled and coloured using Blender, a 3D modelling program. Three individual meshes are used: two rectangular prisms make up the base and the post of the sign, while the octagon plate is a flat eight-sided cylinder. The sides are coloured with materials appropriately. No letters are added because the car recognizes signs by colour filtering.



Technologies

Main Programming & Rendering:

UNREAL ENGINE

3D Modelling:

blender

Testing Platform:

Windows

Research Programming:

C++

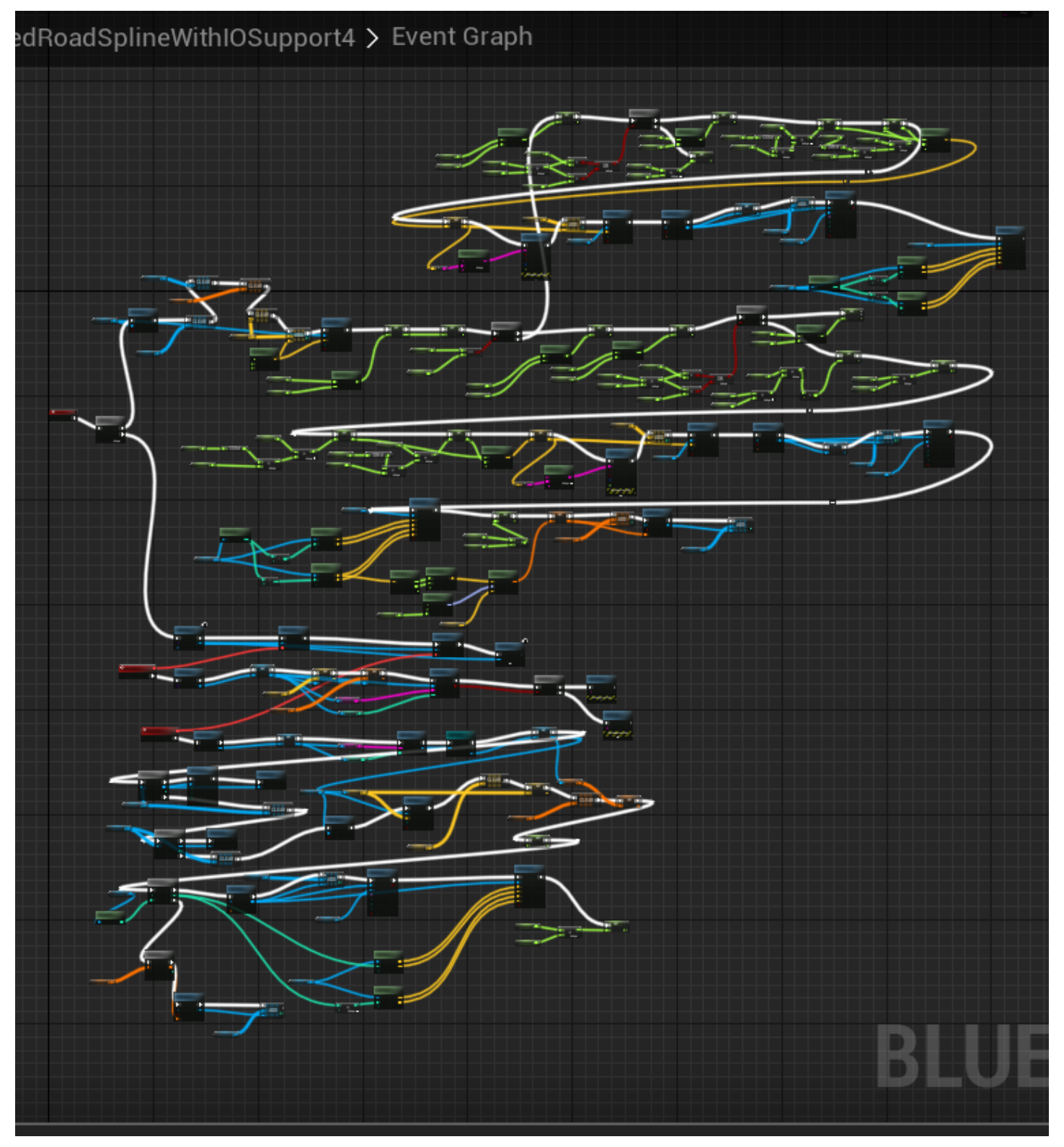
Acknowledgements

Team Captain: Rowan Dempster
Project Judges: Derek Rayside, Joanne Atlee, Patrick Lam
Design Pattern & Architecture Consultancy: Derek Rayside, Werner Dietl
Development Communities: Unreal Slackers [Discord Server], Unreal Engine 4 AnswerHub & Forums, Dave Pagurek van Mossel (SE 2019)

Car specifications provided by Chevrolet. Standards of road design provided by the U.S. Department of Transportation.

Architecture

To build environments, three individual levels with blueprint classes were created. These three classes follow the pipe-and-filter architecture style. Classes are programmed using UE4 Blueprints, a visual method. On the right is the code that generates curved roads.



The pipe-and-filter style provides scaffolding for all three classes. Random numbers are used as input for angles and road length of individual segments. Filters include creating control points for straight and curved roads, instantiating multiple meshes that make up the road, placing meshes between two control points, and so on.

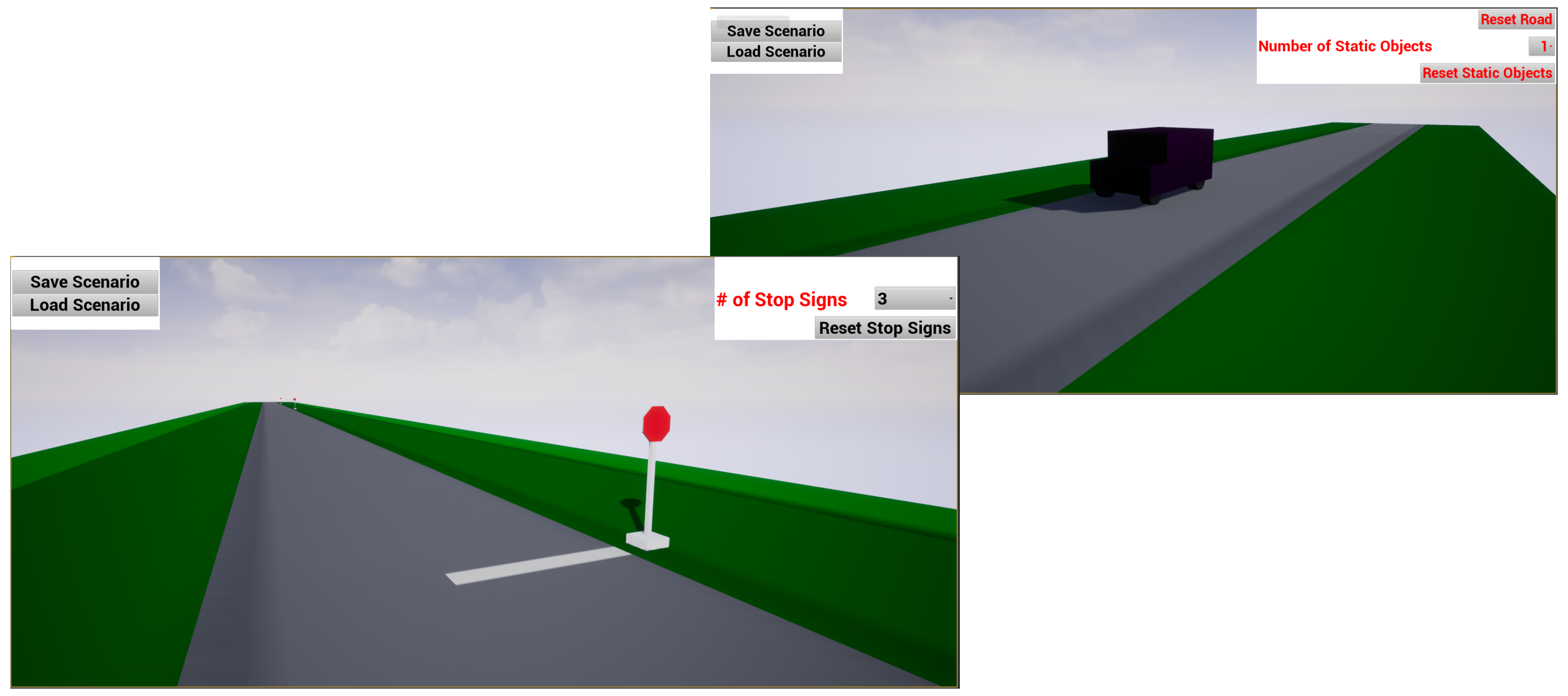
Essentially, data and execution flows from left to right, up to down. Nodes represent data and functions, while the lines connecting them are execution and data flow.

Design Patterns

In Unreal Engine 4, classes use composition to define the components of what it has. For instance, the class that builds curved roads has three mesh components and one spline component instead of inheriting from the classes for meshes and splines.

The classes that generate the environments, as well as those for saving and loading them, use composition and aggregation relationships to store object and array references, such as two control points for straight roads, and an array of structures representing position, rotation, and scale of road patches.

Lastly, user interfaces for saving and loading scenarios, resetting stop signs, and making a new straight road with a stationary bus from scratch, use the model-view-controller pattern. Part of the implementation is done in UE4, but event dispatch, and data manipulation, was built from scratch.


Save Scenario
Load Scenario
Reset Road
Number of Static Objects
1
Reset Static Objects
Save Scenario
Load Scenario
# of Stop Signs
3
Reset Stop Signs