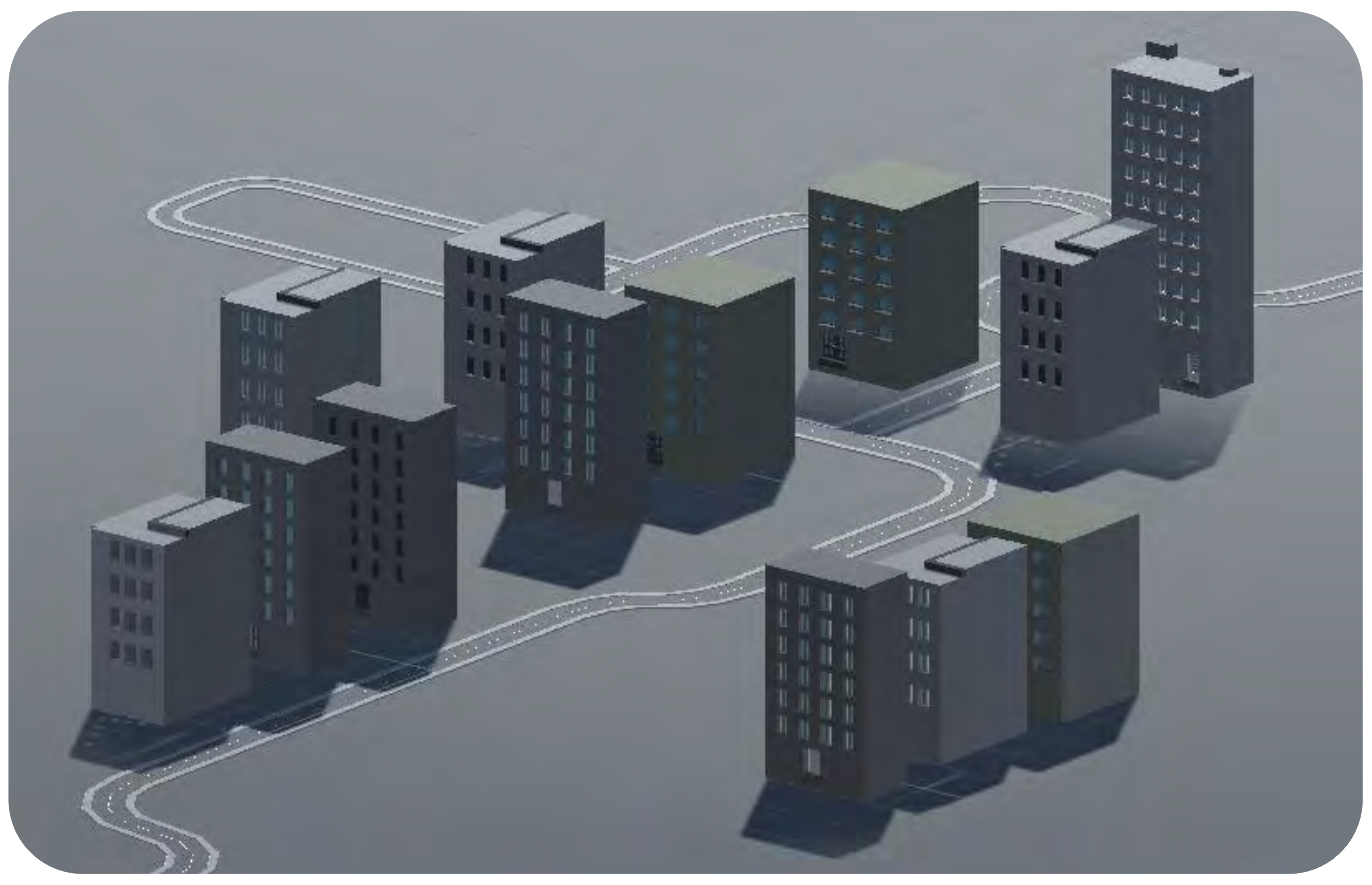
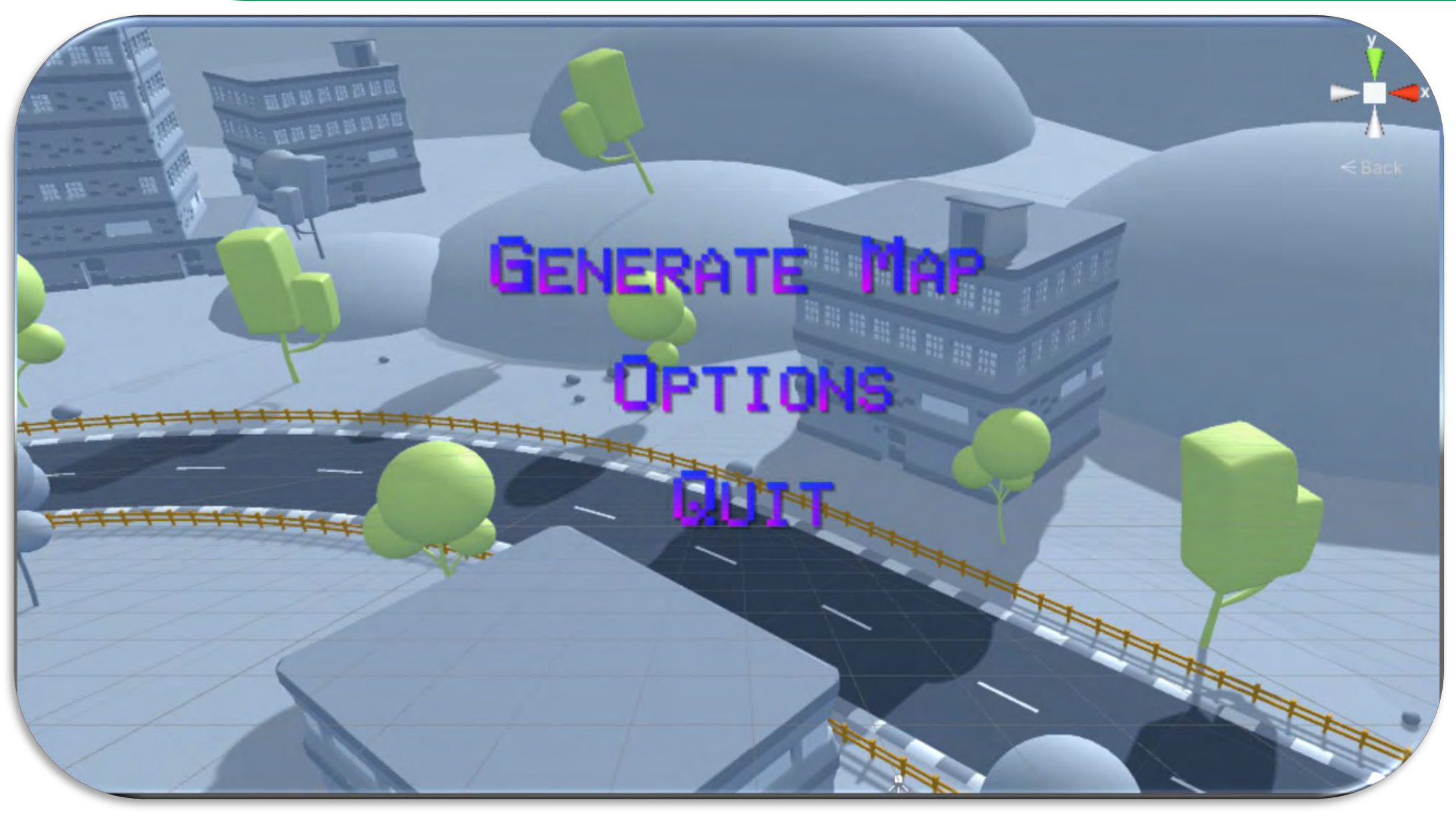
Automated Environment Generator

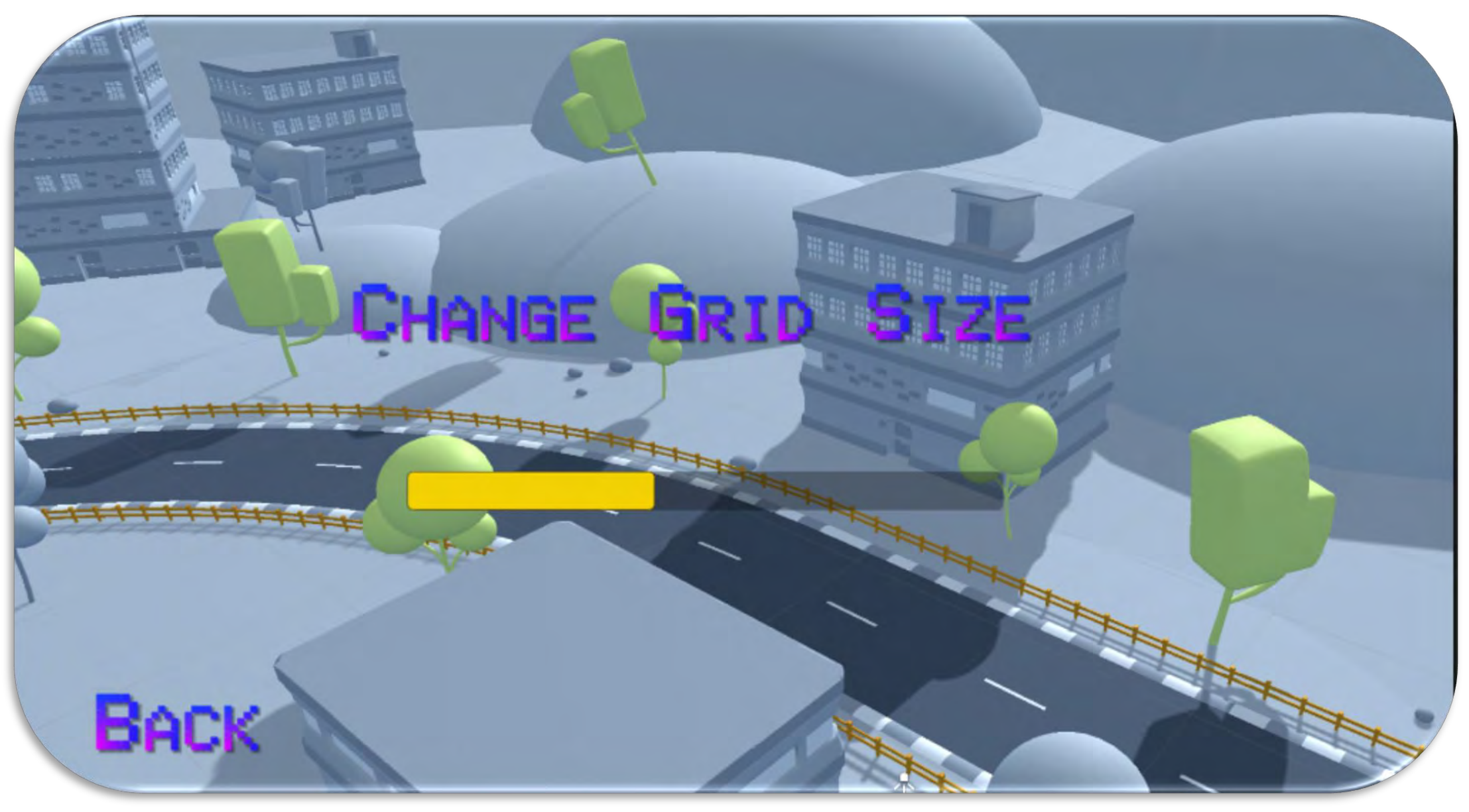
Waleed Ayyash, Jeremy Kracy, James Nicholas, Austin Reeves, Jonathan Sanchez
The Core Dumps
University of North Texas
Sponsored by: Kaushik Madala

Background

Unity game engine has been used for autonomous vehicle simulation software these days. However, testing autonomous vehicles requires diverse environments, which include different road types, different vehicles, different environmental settings, different types of pedestrians, etc. All these environments are currently being created manually. In this project, the major goal is to create such environments automatically. The project group members can simply use various artifacts available in the Unity asset store. They will need to create a UI that offers options based on the assets available, using which the users must be able to specify options of what elements and sizes of maps they want. The program will need to consider these inputs and any additional constraints given by the users and need to generate maps automatically.







System requirements

The system will need the Unity game engine installed, Windows/macOS, and a high-performance system capable of generating roads and environments in a decent amount of time

User's profile

Engineers in the autonomous driving industry, or those interested in advanced AI and driving.

List of Features

F1. Generate roads given set specifications, randomly or given options from the user.
F2. Generate environments for roads to be placed into, randomly or given options.
F3. Generate stop lights, stop signs, etc.
F4. UI that offers options based on assets available.
F5. Pause Menu - basically the Main Menu accessible when not at the Main Menu.

Materials & Methods

The materials used for our project consisted of Unity game engine software, GitHub desktop, and Trello for planning and organization. Unity is a cross platform game engine, it gives users the ability to create expansive games in 2D and 3D.

We used primarily C# as our language of choice. This choice was made due to the fact that Unity is well built for C# programming.

GitHub desktop was also a great tool because it seamlessly connected to Unity and made it really easy for us to push and pull code from the team repository. It allowed us to all work separately on the same project.

On Trello we used a Kanban board that really helped with project planning. We could easily assign people tasks and label them in terms of importance, and we had different columns depending on the state of the task "in progress", "done", etc.

We used Scrum methodologies for our project development. Breaking up the work in "sprints", having weekly meetings about what has been worked on what needs to get done, and having a scrum leader who kept everything flowing was essential for our success.

Technologies Used

unity
Trello
GitHub