Fire Hazard Simulator Project Report

Louis Declerck & Martin Leuleux

May 15, 2024

1 Introduction

The Fire Hazard Simulator, developed by Louis Declerck and Martin Leuleux, is a Python-based project designed to simulate fire propagation in a forest. This project aims to provide a working fire simulator. The simulator models the spread of fire based on various parameters and environmental conditions, allowing users to observe and analyze fire behavior in a controlled virtual environment.

2 Project Structure

The repository contains several key components:

- **Terrain Generation** (`generate_terrain.py`): This module is responsible for creating the virtual forest terrain. It uses the `random` library to introduce variability, mimicking the natural randomness of forest environments.
- **Fire Parameters** (`fire_parameters.py`): This file defines the parameters that influence fire behavior, such as wind speed, humidity, and fuel availability.
- **Terrain Parameters** (`terrain_parameters.py`): This module sets the characteristics of the terrain, including vegetation density and moisture levels.
- **Fire Propagation** (`fire_propagation.py`): This is the core of the simulation, where the logic for fire spread is implemented. It uses the parameters defined in the previous modules to simulate how fire moves through the terrain.
- **Visualization** (`main.py`): This script handles the graphical representation of the simulation, providing a visual output of the fire spreading across the terrain and to run the simulation. Users can execute this file to start the simulation process.

3 Functionality and Usage

To use the Fire Hazard Simulator, follow these steps:

1. **Clone the Repository**: Download the project files from the GitHub repository.
2. **Install Dependencies**: Ensure Python is installed on your system along with necessary libraries such as `random`.
3. **Run the Simulation**: Execute the `main.py` script to start the simulation. The simulation can be run from the command line using Python:

   ```
   python longest-trail.py
   ```

   Alternatively, the simulation can be run via PowerShell by navigating to the project directory and executing the main script with the following command:

   ```
   python -u "<your directory>/main.py"
   ```

4 Programming Choices Made

4.1 Terrain and Cell Structure

Terrain (Object) For the terrain, we went with class in order to store directly the size of the terrain and the grid itself.

Cell (Object) For the cells, we went ahead and chose a class one more in order to store all it's attributes (terrain type, fire strength, height, coordinates and whether it's burning, dying or simply alive). Therefore, it seemed more than useful to use a class of it's own for the cells.

4.2 Generating a Terrain

The terrain generation in this script is a procedural generation process that uses randomness and certain rules to create a terrain. Here's a step-by-step breakdown:

1. **Cell Generation:** The cell_generator function assigns a type (Water, Forest, or Plain) to each cell in the terrain. This is done by generating a random number and assigning a type based on the value of this number. The probabilities for each type are defined in the terrain_parameters module.

2. **House Generation:** The generate_houses function places houses on the terrain. It iterates over each cell, generates a random number, and if this number is less than the probability of a house (defined in terrain_parameter) and the cell is not water, it places a house.

3. **Harmonization:** The harmonization function changes the type of each cell based on the types of its neighboring cells. This is done to make the terrain look more natural. The change_cell function is used to change the type of a cell.

4. **Terrain Generation:** The generate_terrain function creates a new terrain object, generates the types of each cell, places houses, and harmonizes the terrain. The terrain is then returned.

4.3 Simulation's Operation

Probabilities This model is based on probabilities. Upon defined rules, the operation is simple, if the chance calculated is 90%, then the fire might at a 90 percent chance propagate.

Displaying In order to display our simulation, we chose to use the graphical interface pygame rather than simply use the terminal, to get a more user-friendly interface.

5 Acknowledgements

The project was developed by Louis Declerck and Martin Leuleux.

6 Conclusion

For more information and to access the project files, visit the Fire Hazard Simulator GitHub repository.