Final Project: Realistic Fire Simulation System

Ashley Huo, x2huo, 20875817

Figure 1: some running result

Description

This project aims to develop a realistic fire simulation system using rasterization with a particle system, incorporating interactions with objects and environmental factors. The simulation will demonstrate realistic fire behaviors, including combustion, fuel vaporization, and interactions with the environment using advanced rendering techniques.

Requirements

- **GLM**: OpenGL Mathematics library for vector and matrix operations.
- **GLFW**: Library for creating windows and handling input.
- **GLEW**: OpenGL Extension Wrangler Library for managing OpenGL extensions.

Implementation Details

Algorithms and Data Structures

- Particle System: Utilizes a particle-based approach to simulate fire and smoke.
- Level-Set Method: Represents thin reaction zones using level-set techniques.
- Vorticity Confinement: Enhances turbulence by computing vorticity and applying confinement forces.
- Grid-Based Methods: Efficiently handle particle interactions and vorticity confinement using spatial grids.
- Ray Marching: Used for realistic rendering of fire and smoke effects.

Platform and System Dependence

- The simulation is implemented in C++ using OpenGL for rendering.
- It is platform-independent as long as the required libraries are supported.
- Use the most recent version for `std_image.h` and `std_image_write.h`.

Input/Output Syntax

- Inputs: Initial parameters for the particle system, such as the number of particles, initial fuel, and temperature.

- Outputs: Visual rendering of the fire simulation and additional effects like smoke, wind, color changes, reduction,
- and interactions with objects.

Data and Code Sources
- Fire and Flame Simulation using Particle Systems and Graphical Processing Units by T.S. Lyes and K.A. Hawick.
- Level Set Methods and Fast Marching Methods by Sethian.
- Physically Based Modeling and Animation of Fire by Nguyen, Fedkiw, and Jensen.
- Some texture images are from [Textures.com](https://www.textures.com/) as well as github.

Caveats, Bugs, and Assumptions
- Caveats: Performance may vary depending on the hardware capabilities.
- Bugs: Ensure all required libraries are correctly installed to avoid runtime errors.
- Assumptions: The simulation assumes ideal conditions for fire propagation and interactions.

Technical Outline

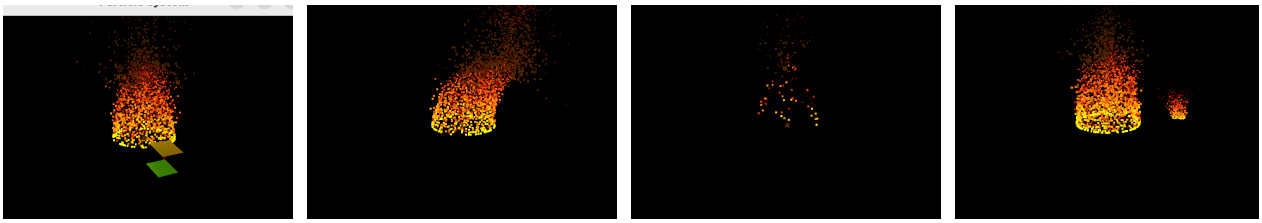

Figure 2: left to right: smoke, wind, reduction, interaction with enviroment

1. Develop a Basic Particle System for Fire and Smoke Simulation
- Particles are born in a random position inside a circular area with upwards velocity and slightly deviated acceleration along the x and z axes.
- Use a particle system where particles are born within a specified area with an upwards velocity.
- Implement GPU rendering for faster performance.

2. Model Combustion and Color of Fuels
- Add a function `updateColor` to update particle color based on lifetime.
- Add a temperature field and modify the color of particles in the system.

3. Thin Flame Model for Reaction Zone Simulation
- Use level-set methods to represent the thin reaction zone.

4. Swap to GPU Rendering
- Create vertex and fragment shaders for GPU rendering and add volume for particles to enhance realism.

5. Apply Vorticity Confinement to Enhance Turbulence
- Use vorticity confinement techniques to simulate small-scale turbulence and swirling effects.

- $fc\ f = \varepsilon h(N \times \omega)$

6 smoke simulation
- Enabled by commenting in the `#define SMOKE` in the `fire.h` file. Add some type particles.

7. Enable Interaction of Fire with Environmental Objects
- Implement interactions of fire with environmental objects such as cubes or other flammable objects.
- Allow fire to interact with objects, including ignition and burning of flammable objects.
- Enabled by commenting in the `#define CUBE` in the header file.

8. Optimize for Faster Simulation
- Optimize simulation algorithms and leverage GPU acceleration for real-time performance. Use grid-based methods to improve efficiency. Reduce time from O(n^2) to O(n + k).
- Use efficient data structures like grids for fluid simulation and particles for fire and smoke.

9. Handle Collisions
- Implement collision handling between particles to ensure realistic interactions.

10. Add Wind Effects
- Add wind forces to particles to simulate wind effects.
- Enable by commenting in the `#define WIND` in the header file.

Objectives
1. Develop a basic particle system for fire and smoke simulation.
2. Model the combustion and color of fuels.
3. Implement the thin flame model for the reaction zone simulation.
4. Swap to GPU rendering.
5. Apply vorticity confinement to enhance the turbulence of fire.
6. Smoke simulation.
7. Enable interaction of fire with environmental objects.
8. Optimize the simulation for faster performance.
9. Handle collisions between particles.
10. Add wind effects to the system.

Bibliography

- Fedkiw, R., Stam, J., & Jensen, H. W. (2001). Visual Simulation of Smoke. SIGGRAPH 2001, ACM.
- Nguyen, D. Q., Fedkiw, R., & Jensen, H. W. (2002). Physically Based Modeling and Animation of Fire. ACM Transactions on Graphics.
- Sethian, J. A. (1999). Level Set Methods and Fast Marching Methods. Cambridge University Press.
- Osher, S., & Fedkiw, R. (2003). Level Set Methods and Dynamic Implicit Surfaces. Springer-Verlag.
- Persson, P. O. (2005). The Level Set Method. MIT Numerical Methods for Partial Differential Equations.
- [Visual Simulation of Fire](https://physbam.stanford.edu/~fedkiw/papers/stanford2003-11.pdf)
- [Realistic and Interactive Simulation of Fire](http://graphics.ucsd.edu/~henrik/papers/fire/fire.pdf)
- [Fronts Propagating with Curvature Dependent Speed](https://math.berkeley.edu/~sethian/2006/Papers/sethian.osher.88.pdf)
- [Navier–Stokes Equations](https://en.wikipedia.org/wiki/Navier%E2%80%93Stokes_equations)