Strafed

Controls

- [A] [D] strafe left and right
- [Space] jump
- [R] zoom in
- [Esc] exit the game
- [F3] toggle HUD

Options

- To see a list of options, run the game with the `--help` flag

```
Allowed options:
  -h, --help            Show help message
  -f, --fullscreen      Run in fullscreen mode
  --width arg (=1280)   Window width - only applies when not fullscreen
  --height arg (=768)   Window height - only applies when not fullscreen
  --cheat               Cheat mode - enables F12 to cheat
```

Secret controls

- [F12] enter cheat mode (only when `--cheat` is enabled)
  - This enables [Space] to fly
  - ... and [W] [S] to move back and forth

Features

Effects

- [x] Blobby Object using Raycasting (12)
    - The goal of the level is a blobby object which is rendered using raymarching in the fragment shader.
- [x] Vertex Shader Animation (8)
    - Vertex shader animation is used to animate fluids such as lava or water but also creates big waves (in the slide in our level). This is done by displacing the vertices in the vertex shader based on time and position
- [x] Procedural Textures (8)
    - We use a noise function in the fragment shader to procedurally generate textures for water and lava
- [x] Motion Blur (16)
    - We create a second rendering pipeline to apply post processing shaders to the entire output image, onto which we apply a simple motion blur filter based on the movement of the camera matrix.

Gameplay (Optional)

- [x] Advanced Gameplay (5)
    - The advanced physics allow for complex platforming interactions, and the timer

creates competitiveness by inciting users to get the lowest time possible.

- [x] Collision Detection (6)
    - The player collides with the world around them and can die when they hit lava or fall out of the world. The movement is fluid and allows for fast-paced gameplay.
- [x] Advanced Physics (4)
    - We implemented dynamic obstacles like the rolling balls on the slide to make the level more interesting.
- [x] HUD (4)
    - The HUD displays a Splash Screen and a Win Screen aswell as the timer throughout the level.

Libraries used

- stb_image - Loading images
- nlohmann-json - JSON parsing
- assimp - Loading 3D models and scenes
- freetype - Loading fonts
- popl - Command line argument parsing

Development Notes

These are notes for designing levels, we implemented "object features" and "shader features" which can be toggled by editing the name of the object in blender. This allows us to quickly design levels and automatically toggle game engine features.

Blender

- Object names can contain metadata
    - `NAME||{"ft":["FEATURE1"],"sft":["FEATURE2"]}`
    - Loads `FEATURE1` as an object feature and `FEATURE2` as a shader feature

Shader features

- `NO_TEX`: Disables texture processing
- `AMB`: Use ambient background color
- `PROC`: Enable the procedural texture
    - `P_LAVA`: Lava procedural texture
    - `P_WATER`: Water procedural texture
    - Defaults to debug texture
- `ANIM`: Enables vertex animation
    - `ANIM_SM`: Small animation, makes the waves tinier
- `BLOBBY`: Renders a blobby object

Object features

- `NO_COLL`: Disable object collision/physics
- `IS_DYNAMIC`: Makes an object a rigid dynamic body
- `IS_VULN`: The object can "die" and will respawn
- `GOAL`: Marks the object as the goal of the level