Hellish Dodgeball Documentation

Description and Features:

We made an arena with a few obstacles such as big rock formations that the player can jump on as well as lava pits with animated lave where the player can fall into. The player can move freely in the arena and has to avoid touching the flaming ball and dying due to falling into the lava. The player can deflect the ball if he is close enough and score points if he successfully does so. In the left corner of the screen, we show the score of the current run. It increments 10 points for every successful hit. In the right corner of the HUD, we display the cooldown of the dash, which is green if it is available. If the game ends, the player sees his score on and the current high score which was either achieved in a previous run or was beaten by the current run. The player will also get a brief description on why he lost the game. For the lighting of the scene, we are using the ball as a point light as well as a few torches on the wall.
We are currently running into a weird error where our program does not run from time to time. This happens very irregularly and sometimes we think we have fixed it because it does not happen for a long time but it somehow always returns. The error says:“nvoglv32.pdb not loaded”. We have already asked for help in the forum but somehow could not manage to fix it.

Implementation of Features:

Collision Detection:
Has not really changed since the last documentation.

Player:
From now on a character controller, not a PxRigidDynamic anymore. This made the movement quite easy.

Hand:
The Hand is now a skeleton model, but we did not implement an animation for the LMB press.

Movement:

- WASD Movement:
  Still the same implementation as before.

- Jumping and airborne:
  We managed to improve our jumps significantly. It now has a way more natural flying curve, and we managed to make the gravity work properly.

Ball:
We did not change anything at the ball physics. The calculation of the ball trajectory still needs some work but we can't figure out how to make it work.

HUD:
The heads up display got a few more text displays. We added a new font and a cooldown for the dash.

Traps:
We have changed the traps into lava pits and added moving lava with a vertex displacement animation. We added hitboxes to the traps because they did not exist in the PhysX world before. So when the player got the ball to go into a trap it would get stuck underneath the floor.

Controls:

WASD: Movement
Shift: Dash
Spacebar: Jump
LMB: Deflect
Enter: Restart (only possible after death)
Esc: Closes the game
F1: Wireframe
F2: Culling

Implementation of effects:

Since the last documentation we added following effects:

- GPU Particle System:
  We use them to add a trail of fire behind the ball and also to make our lava pits stick out more.
- Vertex Shader Animation:
  We used it to animate our lava.
- Physically Based Shading:
  Our new render model is PBS. We implemented a Cook-Torrance model with Schlick´s approximation for the geometry function and the Fresnel factor as well as a Beckmann distribution for our specular highlights.
- Bloom:
  The entire scene is rendered twice once normally and a second time we only render parts of the scene with values over a specific threshold. The second render target is then blurred multiple times with a gaussian kernel and then

reapplied over the original render. We are running into a really weird error, where the second framebuffer for the blurry texture shrinks our image when applying the gaussian blur. This only happens when we run the game at a resolution different from 1280x720 or 1920x1080(in fullscreen mode).

- Models:
  We use the Assimp library to handle all our model loading.

Libraries and References:

We are using the NVIDIA PhysX Library, Assimp and also FreeType. PhysX does the physics calculations for us (already discussed earlier), FreeType does the font rendering for us and Assimp does the model loading.

Particles:
http://www.opengl-tutorial.org/intermediate-tutorials/billboards-particles/particles-instancing/
https://learnopengl.com/In-Practice/2D-Game/Particles
PBS:
https://learnopengl.com/PBR/Theory
https://learnopengl.com/PBR/Lighting
http://www.codinglabs.net/article_physically_based_rendering_cook_torrance.aspx
Bloom:
https://learnopengl.com/Advanced-Lighting/Bloom
Model Loading:
https://learnopengl.com/Model-Loading/Assimp
https://learnopengl.com/Model-Loading/Mesh
https://learnopengl.com/Model-Loading/Model
Vertex shader animation:
https://github.com/ashima/webgl-noise (for the noise function)
https://www.clicktorelease.com/blog/vertex-displacement-noise-3d-webgl-glsl-three-js/