RiftBlade - A Virtual Reality Sword Fighting Game

SWP SS 2014 - Team D

RWTH Aachen University

6. August 2014

Concept

- Goal: Avoid genres that are common in low budget virtual reality games
- Spontaneous idea: First person swordfighting
- Focus on technology and implementation rather than complicated gameplay design - simple arena gameplay, player fights a single enemy in a limited environment

Goals

- Primary goal: Graphical fidelity and solid mechanics (i.e. stable physics, animations, etc.)
- Make the simple gameplay (hit the enemy faster than he hits you) look and feel good by nailing the technical implementation

Features that were planned

Features that were planned

Deferred Rendering Pipeline

Features that were planned

Deferred Rendering Pipeline

- No real *need* for deferred pipeline as chosen environment does not expose many potential light sources, but it's a bit more complex than simple forward rendering, so...

Features that were planned

Deferred Rendering Pipeline

- No real *need* for deferred pipeline as chosen environment does not expose many potential light sources, but it's a bit more complex than simple forward rendering, so...
- HDR with bloom and tonemapping, shadowmapping, surface effects i.e. normal mapping, parallax occlusion mapping, ...

Features that were planned

Deferred Rendering Pipeline

- No real *need* for deferred pipeline as chosen environment does not expose many potential light sources, but it's a bit more complex than simple forward rendering, so...
- HDR with bloom and tonemapping, shadowmapping, surface effects i.e. normal mapping, parallax occlusion mapping, ...

Skeletal Animations

Features that were planned

Deferred Rendering Pipeline

- No real *need* for deferred pipeline as chosen environment does not expose many potential light sources, but it's a bit more complex than simple forward rendering, so...
- HDR with bloom and tonemapping, shadowmapping, surface effects i.e. normal mapping, parallax occlusion mapping, ...

Skeletal Animations

- In character based games, animations are key to immersion

Features that were planned

Deferred Rendering Pipeline

- No real *need* for deferred pipeline as chosen environment does not expose many potential light sources, but it's a bit more complex than simple forward rendering, so...
- HDR with bloom and tonemapping, shadowmapping, surface effects i.e. normal mapping, parallax occlusion mapping, ...

Skeletal Animations

- In character based games, animations are key to immersion

Motion Controlled Input

Features that were planned

Deferred Rendering Pipeline

- No real *need* for deferred pipeline as chosen environment does not expose many potential light sources, but it's a bit more complex than simple forward rendering, so...
- HDR with bloom and tonemapping, shadowmapping, surface effects i.e. normal mapping, parallax occlusion mapping, ...

Skeletal Animations

- In character based games, animations are key to immersion

Motion Controlled Input

- WiiMotionPlus controller for sword control

Features that were planned

Deferred Rendering Pipeline

- No real *need* for deferred pipeline as chosen environment does not expose many potential light sources, but it's a bit more complex than simple forward rendering, so...
- HDR with bloom and tonemapping, shadowmapping, surface effects i.e. normal mapping, parallax occlusion mapping, ...

Skeletal Animations

- In character based games, animations are key to immersion

Motion Controlled Input

- WiiMotionPlus controller for sword control
- Treadmill for player movement control

Implementation: The rendering pipeline

The GBuffer

Implementation: The rendering pipeline

The GBuffer

- Originally: 3 color attachements (fragment positions, normals, diffuse color) and a depth buffer

Implementation: The rendering pipeline

The GBuffer

- Originally: 3 color attachements (fragment positions, normals, diffuse color) and a depth buffer
- Store positions and normals in view space

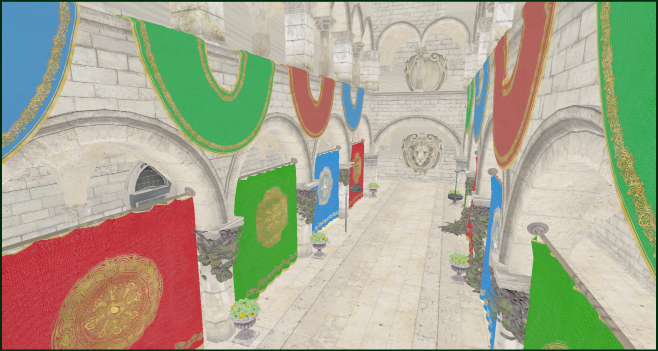
Implementation: The rendering pipeline

The GBuffer

- Originally: 3 color attachements (fragment positions, normals, diffuse color) and a depth buffer
- Store positions and normals in view space
- Problem: Positions require a floating point buffer - proved to be imperformant; Positions are now reconstructed from depth

Implementation: The rendering pipeline

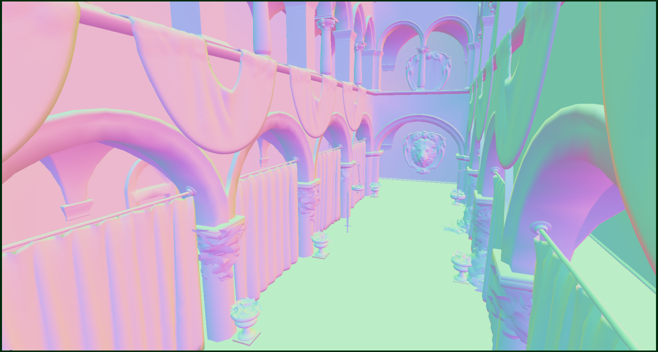
Diffuse Buffer



- GL_RGB color texture

Implementation: The rendering pipeline

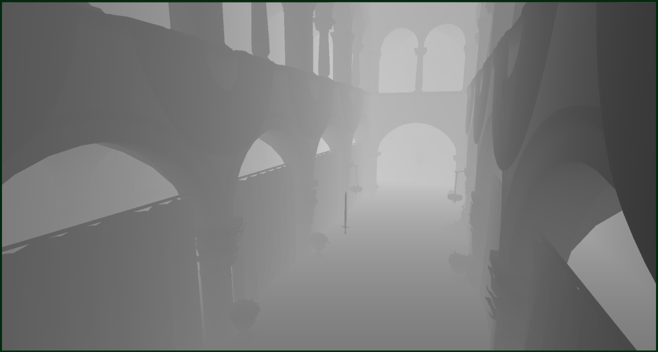
Normal Buffer



- GL_RGB color texture

Implementation: The rendering pipeline

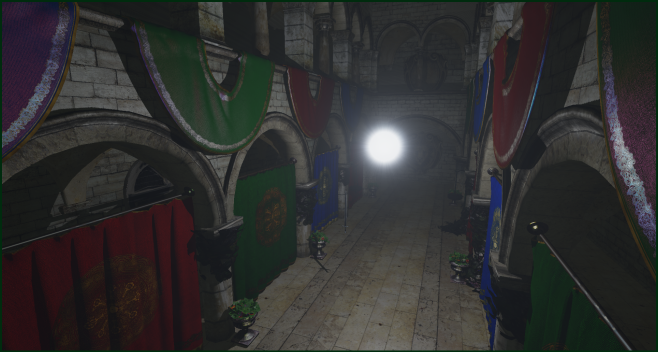
Depth Buffer



- 24 bit depth texture

Implementation: The rendering pipeline

Final Image



Implementation: The rendering pipeline

Implementation: The rendering pipeline

- Render lights using attributes from gbuffer

Implementation: The rendering pipeline

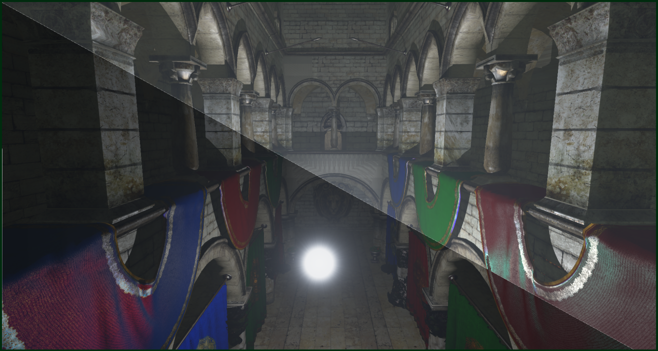
- Render lights using attributes from gbuffer
- Light contributions are accumulated into a floating point buffer for HDR values

Implementation: The rendering pipeline

- Render lights using attributes from gbuffer
- Light contributions are accumulated into a floating point buffer for HDR values
- Final pass: Apply tonemapping (no dynamic adaption though)

Implementation: The rendering pipeline

Tonemapping: Enabled (Left) vs. Disabled (Right)

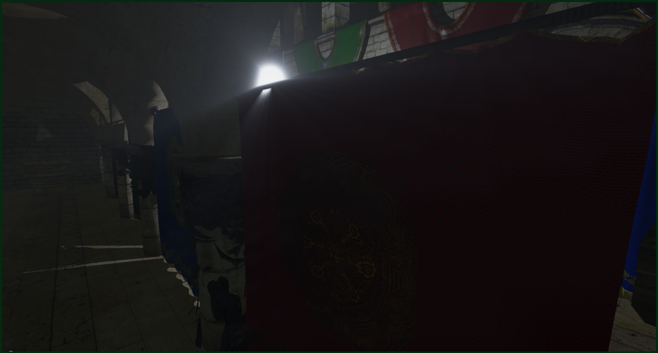


Implementation: The rendering pipeline

- Volumetric lights are rendered in another pass (several passes actually)

Implementation: The rendering pipeline

Volumetric lighting

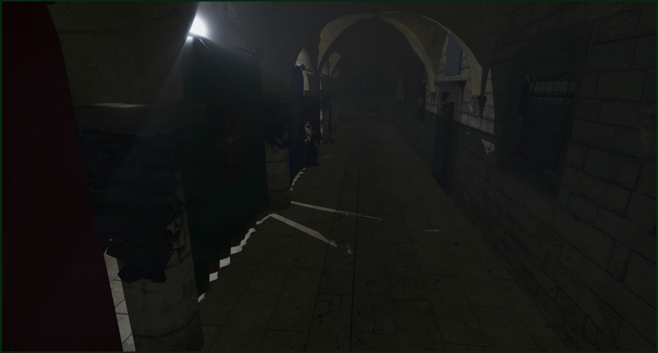


Implementation: The rendering pipeline

- Shadow maps are created in a pre-pass, then taken into account in the light pass

Implementation: The rendering pipeline

Shadow Mapping

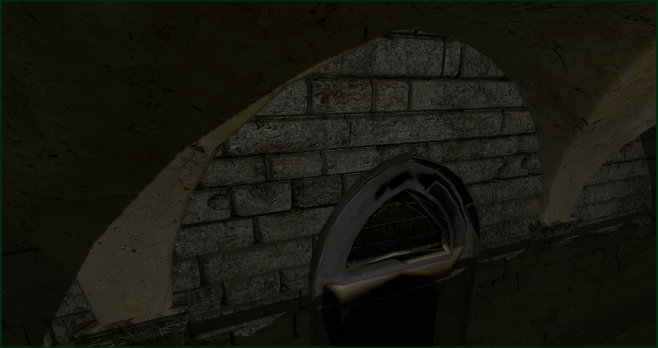


Implementation: The rendering pipeline

- We also tried our hand at Parallax Occlusion Mapping

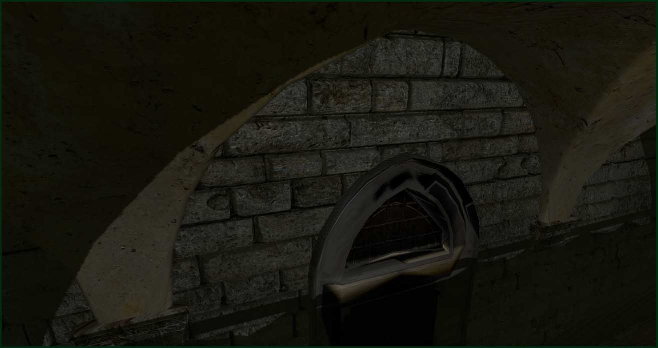
Implementation: The rendering pipeline

Shadow Mapping



Implementation: The rendering pipeline

Shadow Mapping



- …but as it turns out, some tangent-vectors were corrupted.

Implementation: Animations

- Skeletal animations are performed in the vertex shader, using matrix arrays send per frame

Implementation: Animations

- Skeletal animations are performed in the vertex shader, using matrix arrays send per frame
- Effect can best be seen in our gameplay presentation ;)

Implementation: Motion Input

- WiiMotionPlus integration proved to be quite a challenge, but we got it working rather well

Implementation: Motion Input

- WiiMotionPlus integration proved to be quite a challenge, but we got it working rather well
- We actually implemented inverse kinematics to allow for free sword control with correct arm movement...

Implementation: Motion Input

- WiiMotionPlus integration proved to be quite a challenge, but we got it working rather well
- We actually implemented inverse kinematics to allow for free sword control with correct arm movement...
- ...but it turned out much worse looking than we hoped

Implementation: Motion Input

- WiiMotionPlus integration proved to be quite a challenge, but we got it working rather well
- We actually implemented inverse kinematics to allow for free sword control with correct arm movement...
- ...but it turned out much worse looking than we hoped
- We now use a more straightforward approach with two WiiMotionPlus controllers instead of one that allow for better tracking of the whole arm

Implementation: Motion Input

- WiiMotionPlus integration proved to be quite a challenge, but we got it working rather well
- We actually implemented inverse kinematics to allow for free sword control with correct arm movement...
- ...but it turned out much worse looking than we hoped
- We now use a more straightforward approach with two WiiMotionPlus controllers instead of one that allow for better tracking of the whole arm
- Treadmill integration turned out quite trivial. There never was any treadmill to integrate.

Things we'd like to add or improve...

- We were able to achieve most goals...

Things we'd like to add or improve...

- We were able to achieve most goals...
- ...however some things didn't exactly work out as planned

Things we'd like to add or improve...

- We were able to achieve most goals...
- ...however some things didn't exactly work out as planned
- Normal Mapping and Parallax Occlusion Mapping look a bit strange...

Things we'd like to add or improve...

- We were able to achieve most goals...
- ...however some things didn't exactly work out as planned
- Normal Mapping and Parallax Occlusion Mapping look a bit strange...
- ...not only, but also because some of our UV maps are exported incorrectly

Things we'd like to add or improve...

- We could need some more polished animations and physics

Things we'd like to add or improve...

- We could need some more polished animations and physics
- Ragdolls would be a nice addition

Things we'd like to add or improve...

- We could need some more polished animations and physics
- Ragdolls would be a nice addition
- Some bugs couldn't be resolved in time...

Things we'd like to add or improve...

- We could need some more polished animations and physics
- Ragdolls would be a nice addition
- Some bugs couldn't be resolved in time...
- ...but in the end, what game is bug-free at release? ;)

Thanks for listening...

...now please enjoy our gameplay presentation. :)