OPTIMIZE SCENE

Robin 141266

TABLE OF CONTENT

INTRO

This block I joined the project Plungers. This team wanted to have a smoother running game. The performance was heavy. So the team wanted me to improve the performance without making the art worse.

Since this Is the goal I want to do this block. I pick up this task. The team want me to work on optimization and implement the skins in game.

I definitely want to make instances materials. And focus on LOD's, however a good suggestion from Lothar want me to do a broader range of optimization. Like looking in to lighting. I want to follow this advice and hope to find time for a broader range.

I should profile the game as well.

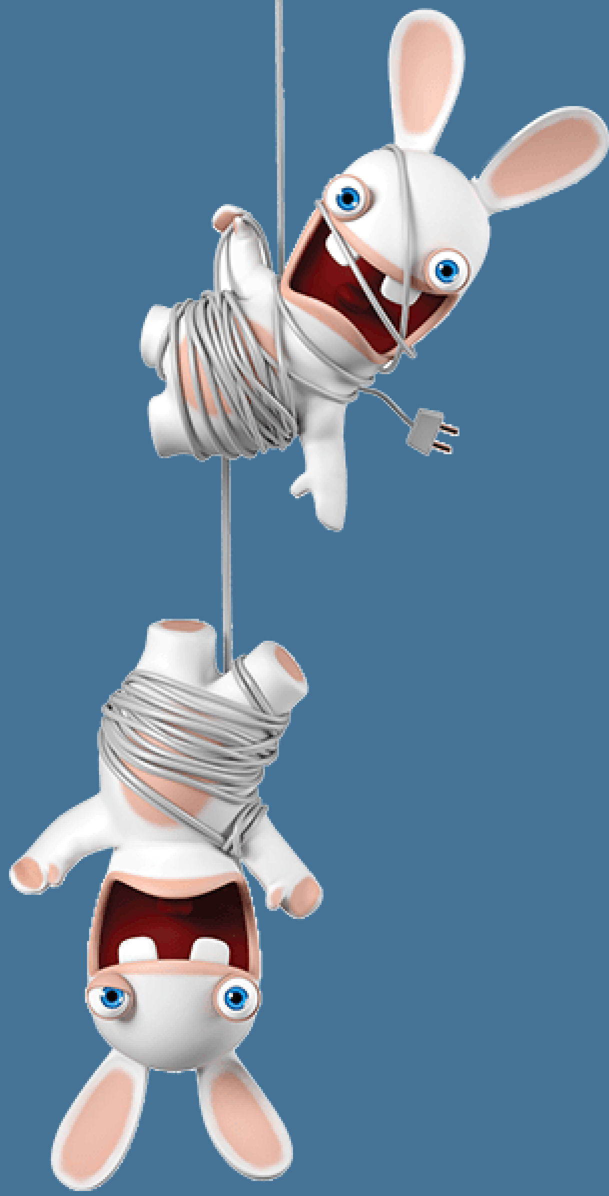
PLAN

I first should analyze the scene. Find the things that cost the most performance. Set those things on a high priority. Find out what the real problem is. And look if there is a solution that does not impact the visuals.

One of the things I should do is make a master material. This material should cover the common options. And I make a material instance for it. Apply it to different objects.

Next to the materials I am going to develop the LOD system. This reduce geometry over distance. Also the shading get cheaper over distance

Other thing I want to focus on is lighting optimization. Lightmaps, lighting optimization. Lighting baking fix

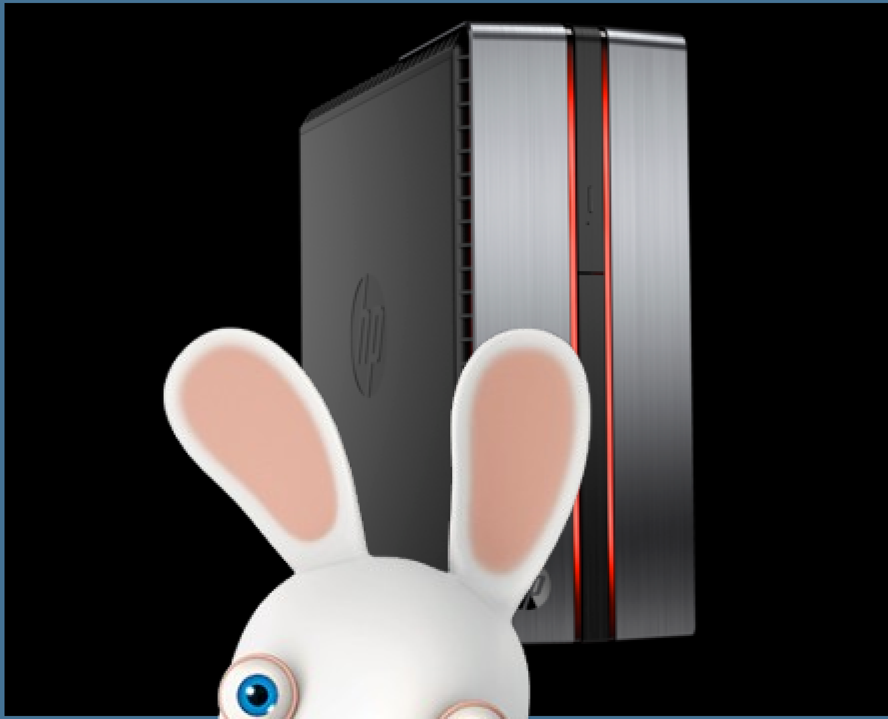


GOAL

For platform reference we picked a product recommended by the htc vive company, picked the cheapest model so the people wont be dissatisfied when buy the game. The target is a "HP ENVY Phoenix©". This machine has the following spec's:

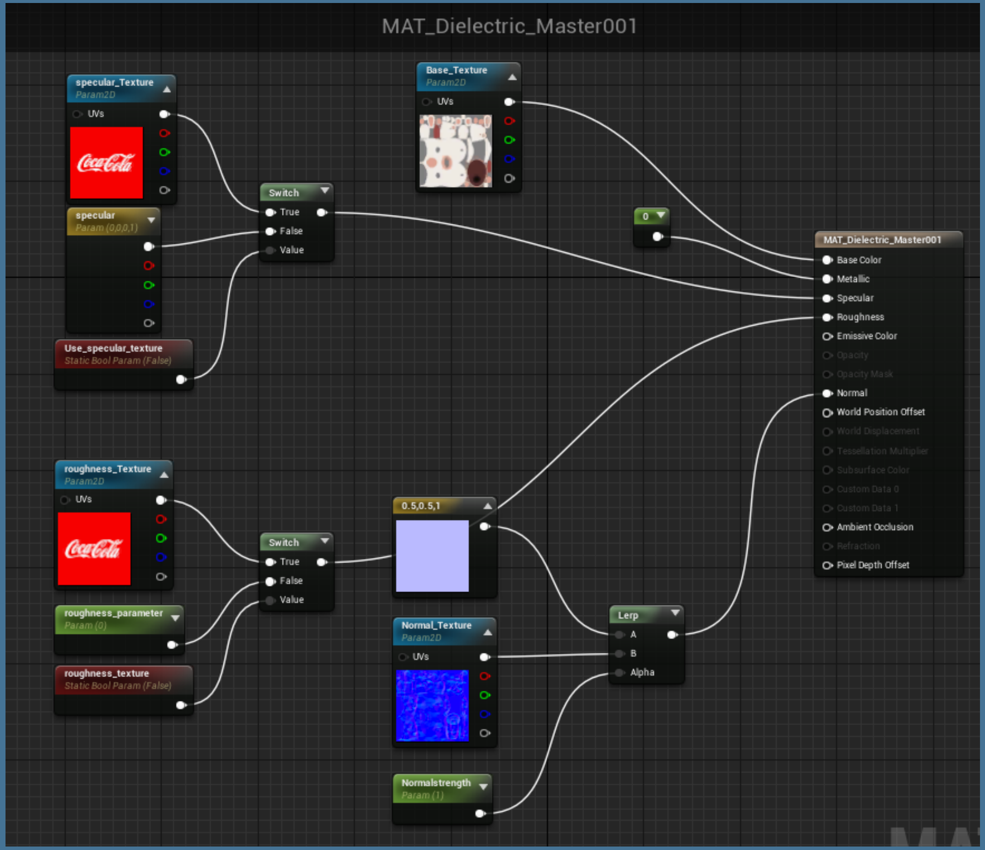
Target platform:

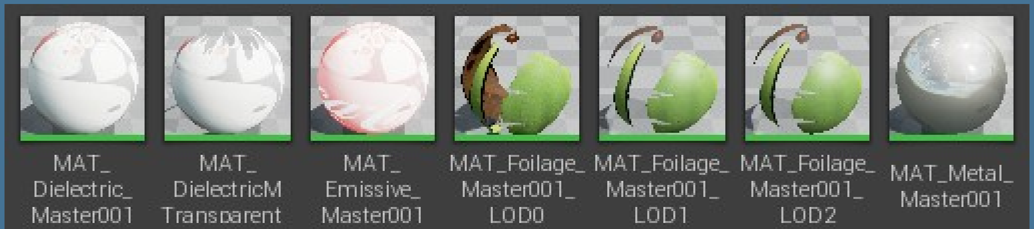
3 GB RAM
Processor family
    6th Generation Intel® Core™ i5 processor
Processor
    Intel® Core™ i5-6600K with Intel® HD Graphics 530 (3.5 GHz, up to 3.9 GHz, 6 MB cache, 4 cores)
Chipset
    Intel Z170
Special features
    Landing pad for: 2 USB 2.0, 2 USB 3.0, audio jack, and memory card reader; Air cooling solution
Internal drive
    256 GB SATA SSD
    2 TB 5400 rpm SATA
    Graphics
    NVIDIA® GeForce® GTX 970 (4 GB GDDR5 dedicated)
for more information:
http://www8.hp.com/ie/en/products/desktops/product-detail.html?oid=11413208#!tab=specs



MASTER MATERIAL

The Master!

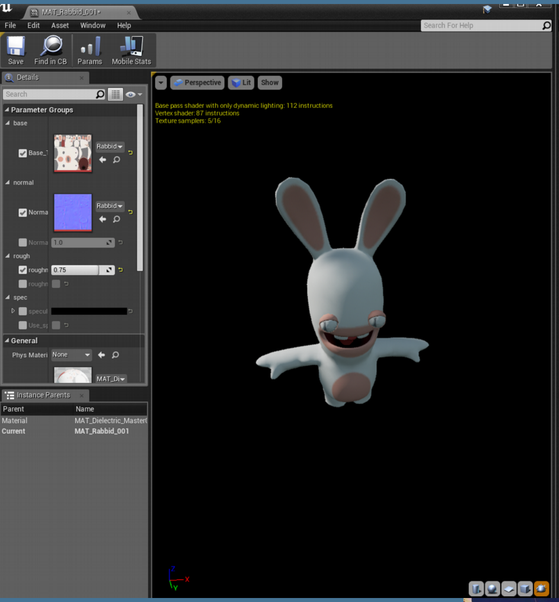




Start off making 1 material, however this proven inefficient since previews materials seem to different to make a master material without to much of a difference to the original result.

Because of this I decided to make multiple master materials. So I won’t need to many switches to take
I would make them anyway and this scopes down on the material parameters.

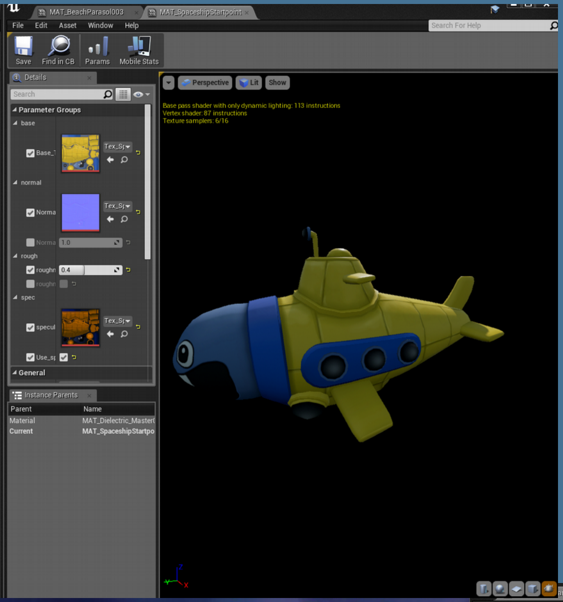
MASTER MATERIAL
APPLIED



Base pass shader with only dynamic lighting: 112 instructions
Vertex shader: 87 instructions
Texture samplers: 5/16

Instance Parents

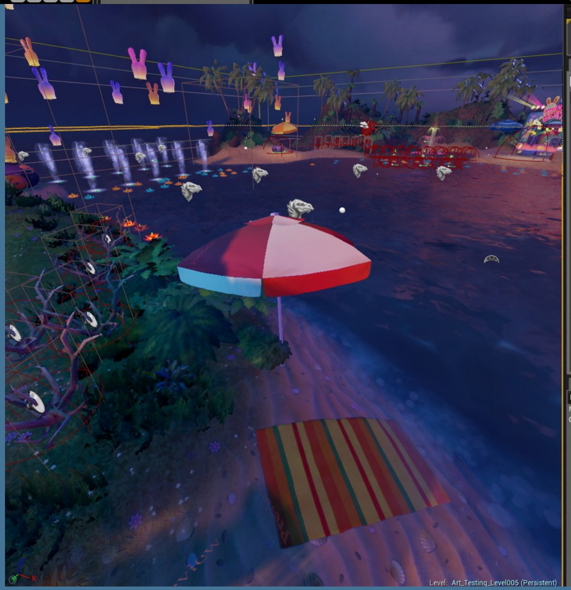
Parent	Name
Material	MAT_Dielectric_Master(
Current	MAT_Rabbid_001

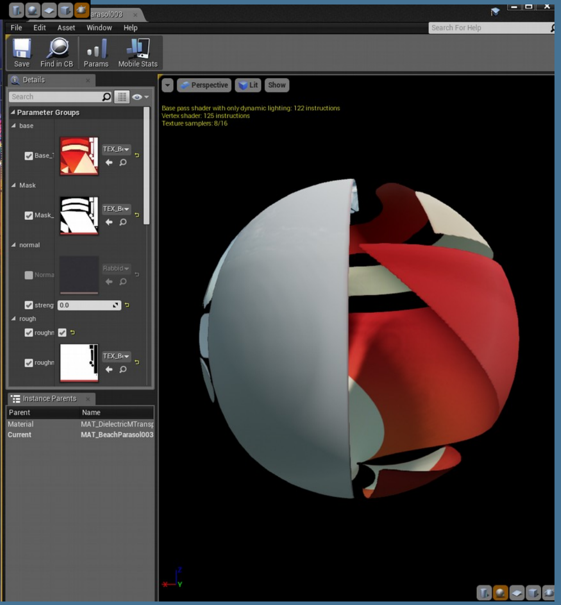


Base pass shader with only dynamic lighting: 113 instructions
Vertex shader: 87 instructions
Texture samplers: 6/16

Instance Parents

Parent	Name
Material	MAT_Dielectric_Master(
Current	MAT_SpaceshipStartpo





Base pass shader with only dynamic lighting: 122 instructions
Vertex shader: 125 instructions
Texture samplers: 8/16

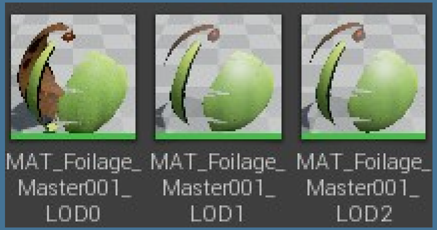
Instance Parents

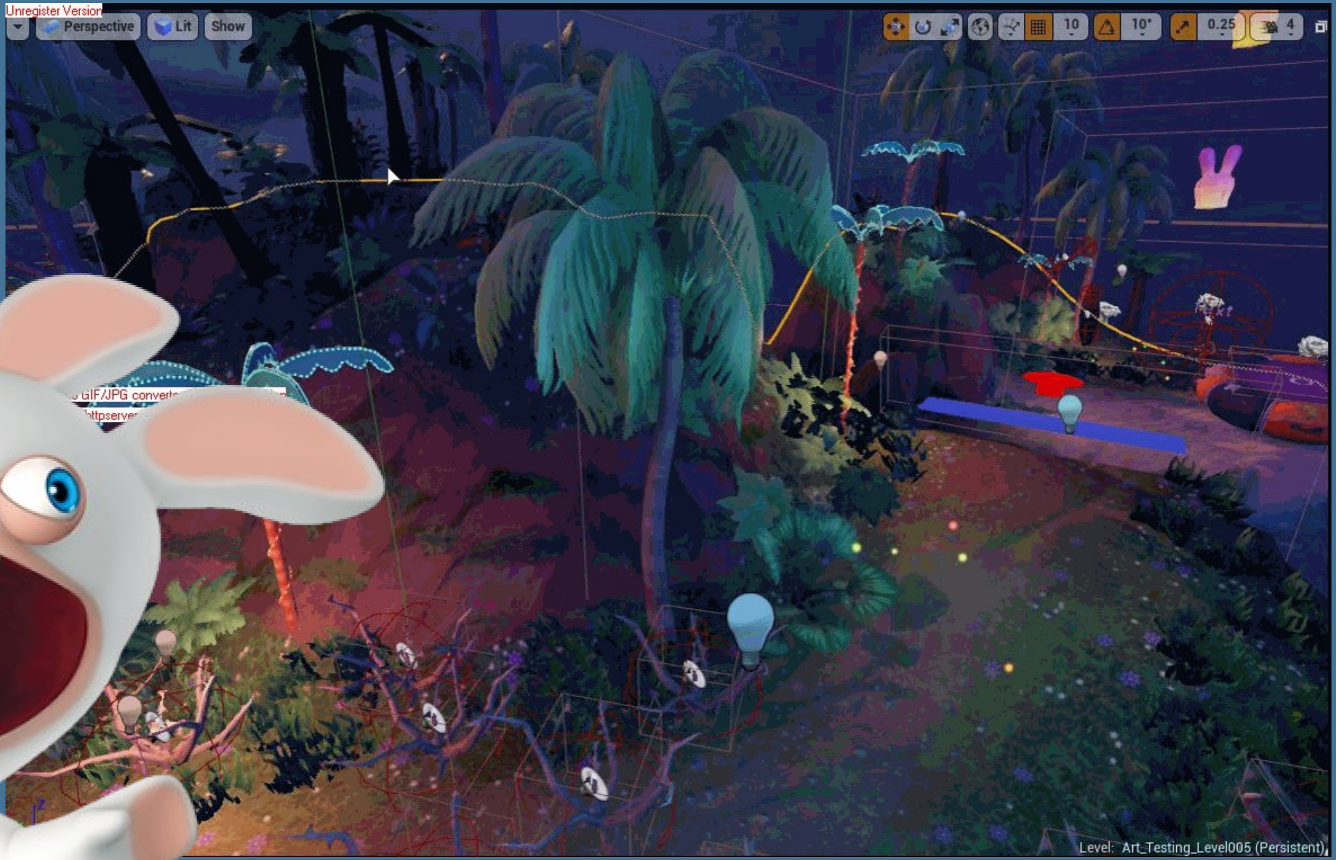
Parent	Name
Material	MAT_DielectricMTransp
Current	MAT_BeachParasol003

Here the material is applied, in a material instance way.

This should reduce load time when you play the game

FOLIAGE
THE PROBLEM


MAT_Foilage_ Master001_ LOD0
MAT_Foilage_ Master001_ LOD1
MAT_Foilage_ Master001_ LOD2

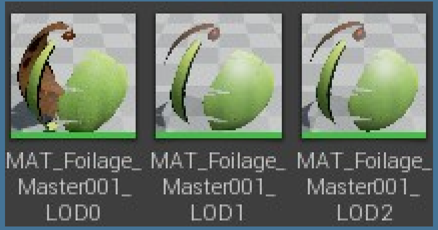


For the next challenge I optimized the foliage. One because the art lead wanted the Foliage to move, as fast as possible.

Next to this the foliage is one of the most appeared Objects in the scene.

I am going to make 3 master materials and then apply them. To the lod system. This way the shader get cheaper at distance

FOLIAGE MATERIALS


MAT_Foilage_Master001_LOD0
MAT_Foilage_Master001_LOD1
MAT_Foilage_Master001_LOD2

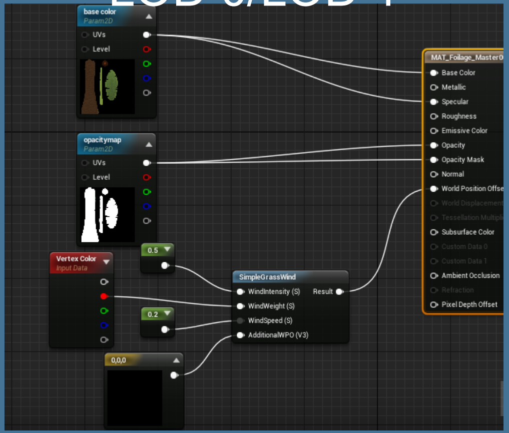
I made 3 master materials for the foliage.

LOD system in foliage is the best investment. Since these objects are the most repetive.

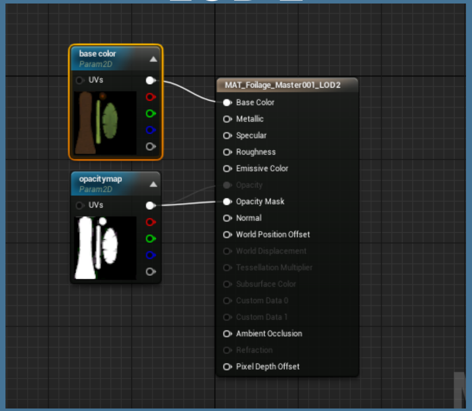
I also applied the LOD strategy to the shader/material since it requires a expensive waviness.

LOD0 uses the general waviness strength. Which the player should see. Next to that is double shaded and has foliage shading.

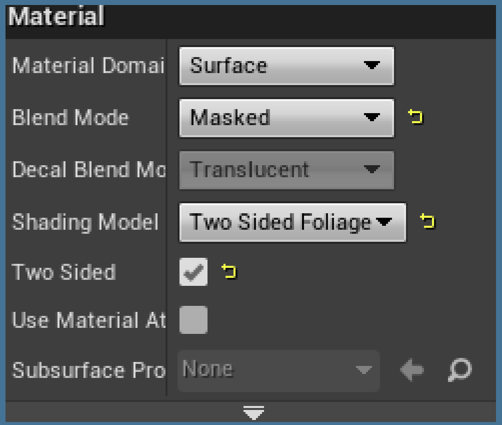
LOD 0/LOD 1



LOD 2



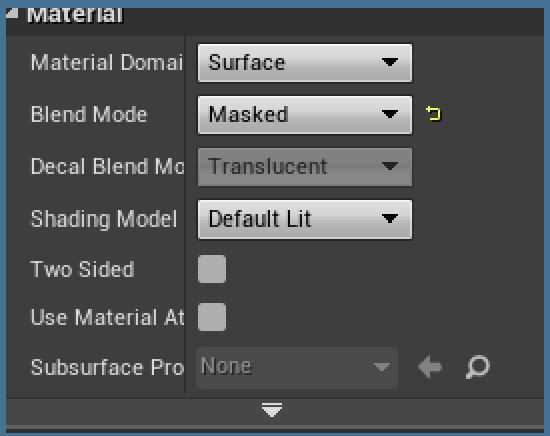
LOD 0



Material

Material Domai	Surface
Blend Mode	Masked
Decal Blend Mo	Translucent
Shading Model	Two Sided Foliage
Two Sided	✓
Use Material At	
Subsurface Pro	None

LOD 1/ LOD 2



Material

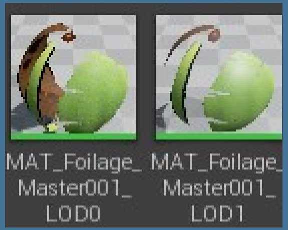
Material Domai	Surface
Blend Mode	Masked
Decal Blend Mo	Translucent
Shading Model	Default Lit
Two Sided	
Use Material At	
Subsurface Pro	None

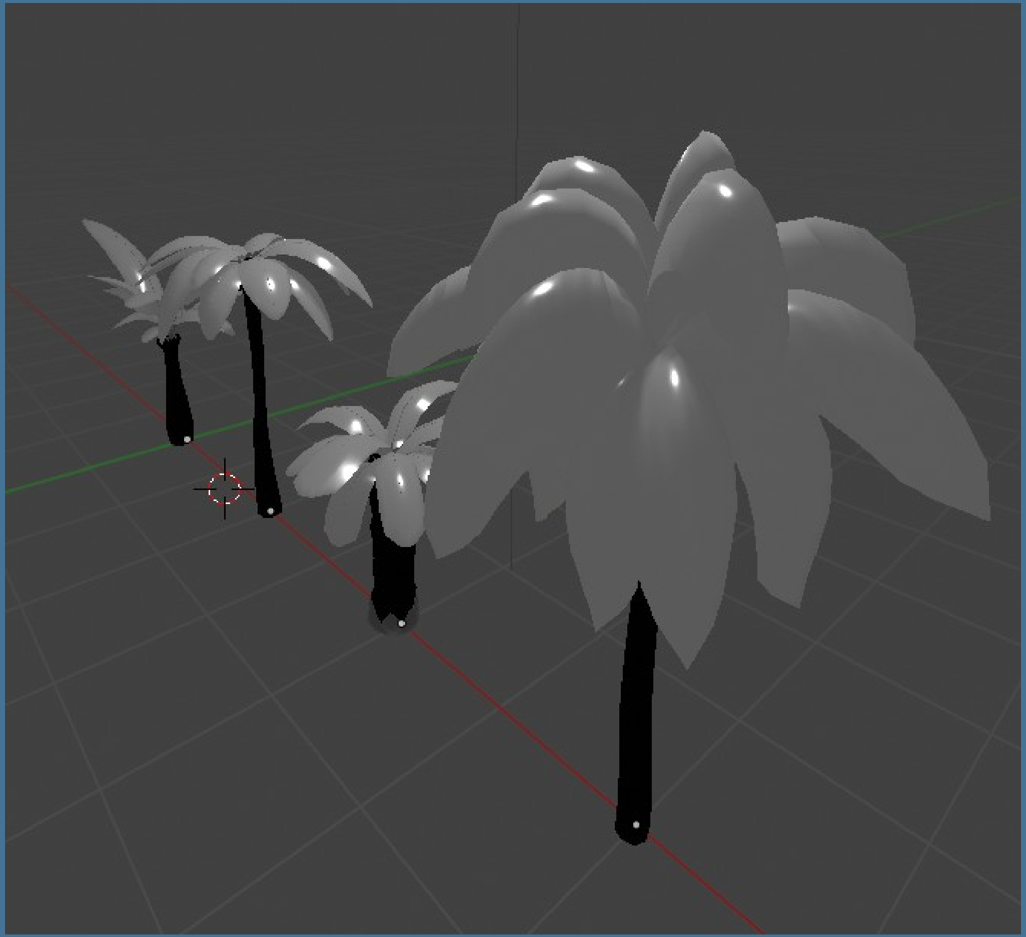
LOD1 is more a transition shader. Since you see the waviness less we make these trees less wavy this way you have a gradient of the waviness and makes more sense the next LOD has no wind. Next optimization to this is that it is simpler shaded ”default lit”. And the shader is “single sided”.

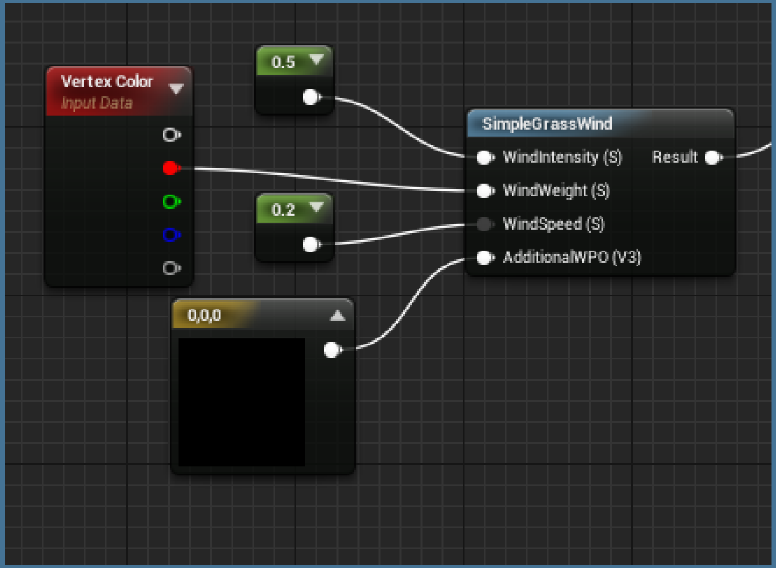
LOD2 is the simples shader. This one does not move the foliage. And has the simplest shading.

Parameters of the strength and speed are done in this master material so we have a easy workflow by just tweaking the values here.

FOLIAGE
MATERIAL PREPARATION





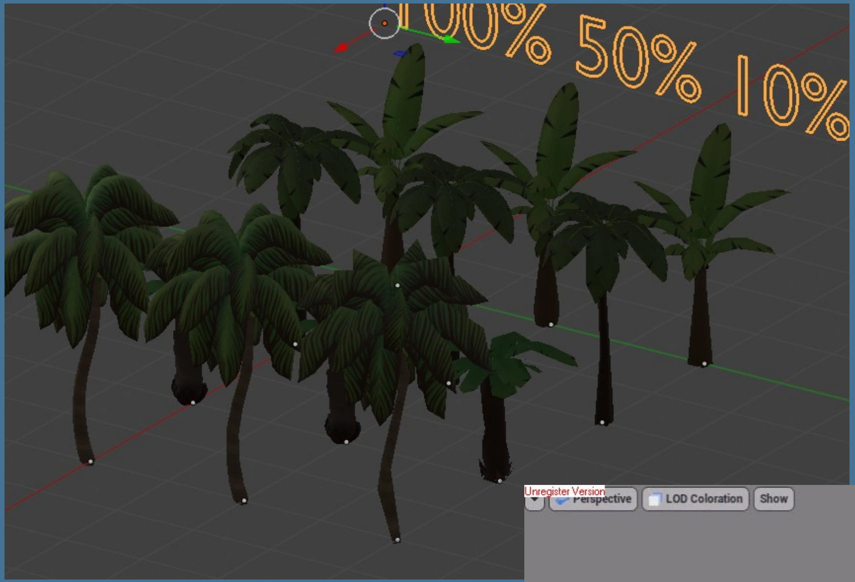


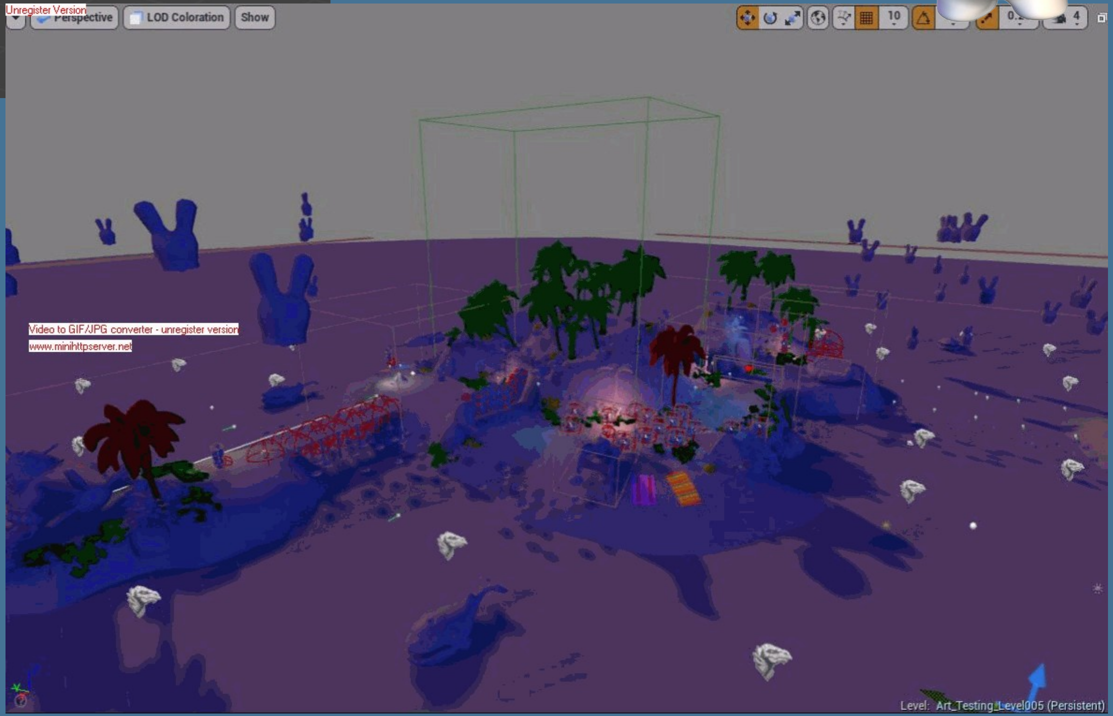
For the material influence. We did not want the full body to move. At least not for the tree's. the coconuts and bark should not move.
And the stem of the leaves should move less.

So in the shader we we need to specify what part of the object should move more and what should move less. We use the vextex data to demeter this. Because the data that's needed to save it would be less. And the data get's in a earlier cheaper stage in the render pipeline collect.

FOLIAGE

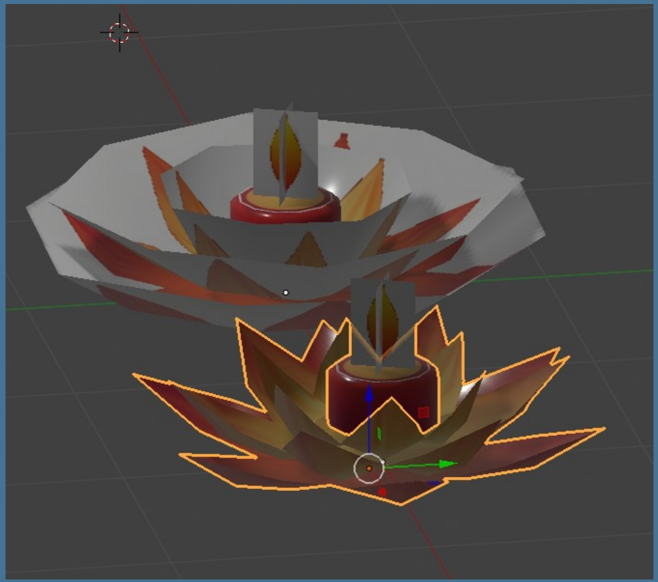
LOD SYSTEM




Unregister Version
Perspective
LOD Coloration
Show
10
Video to GIF/JPG converter - unregister version
www.minihttpserver.net
Level: Art_Testing_Level005 (Persistent)

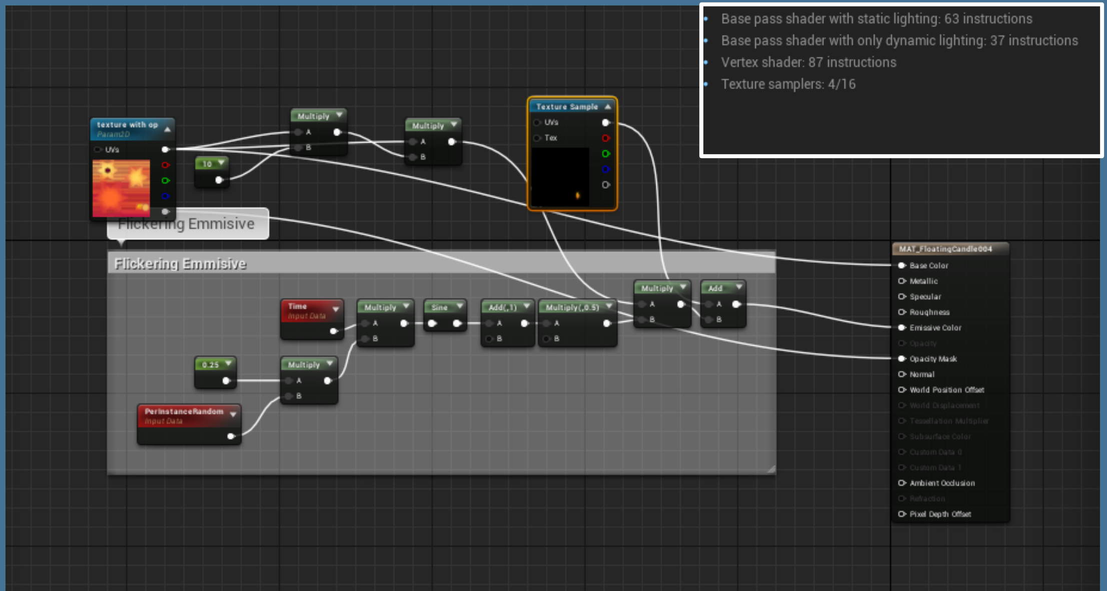
As for more optimization
By using a tool to generate the lod's I did not need to spend much time on the objects in order to make the lod's.
generally I disadvice using a generic tool to reduce the geometry without checking the output.
Howeve since I have a short time span to work on every object. I picked the solution and roughly check for not to much of a difference.
I picked the geometry change of 50% tris and 10% tris.
Halfen the geometry when the distance is dubbeld is generaly the advice.

FLOATING CANDELS

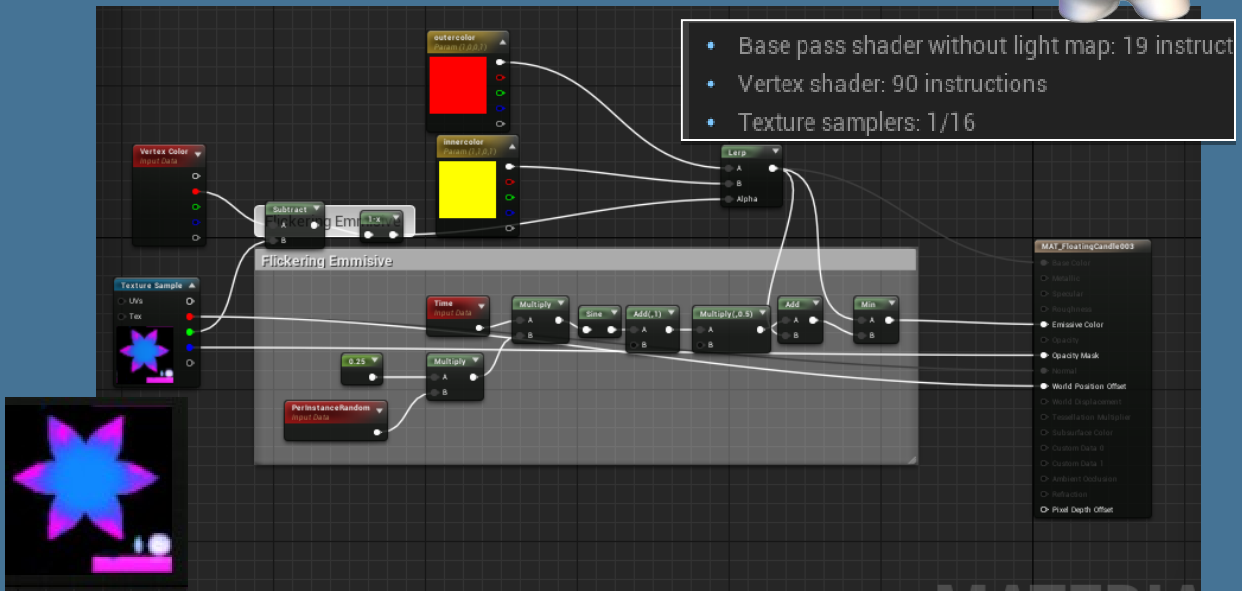


For the lower lod I made use of the transpancy what is already needed for the flame. I did not do that for the highest LOD since that would be seen in the shading.

For far away I made the object unlit since you wont see it in the distance anyway. I combined all the textures as well to 1 texture. To reduce the texture pool. Unfortunate the visuals where hurt to much by it and that why the high lod's keep the old texture methode.



- Base pass shader with static lighting: 63 instructions
- Base pass shader with only dynamic lighting: 37 instructions
- Vertex shader: 87 instructions
- Texture samplers: 4/16

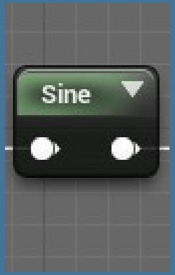


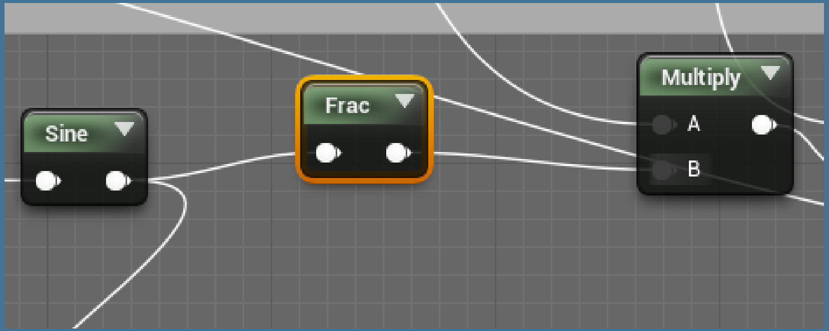
- Base pass shader without light map: 19 instruct
- Vertex shader: 90 instructions
- Texture samplers: 1/16

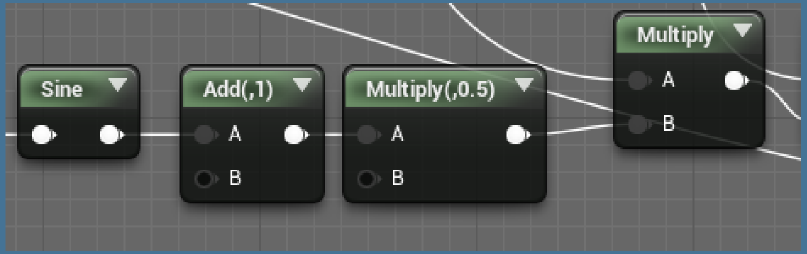
FLOATING CANDELS

SINEWAVE IMPROVEMENT

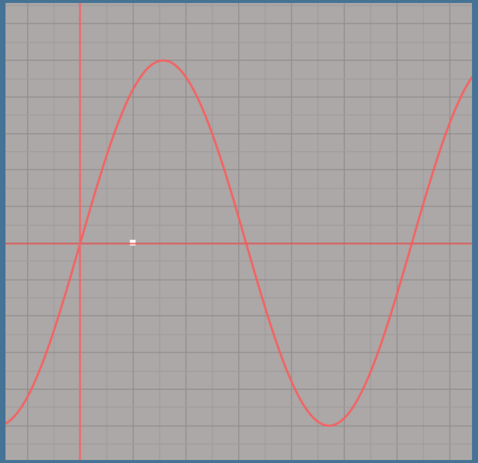
Art lead wanted to make the floating candles glow and fade In a sine wave pattern.
However the old methode gave the wrong output. I improved the output by changing the algorithm
A simple add and multiply(divide is heavier) did the trick.



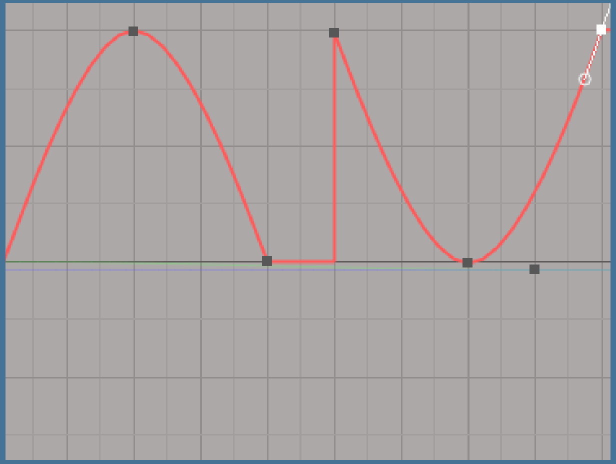




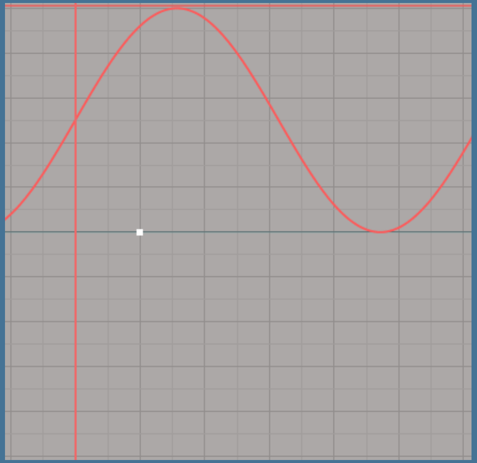
Output:



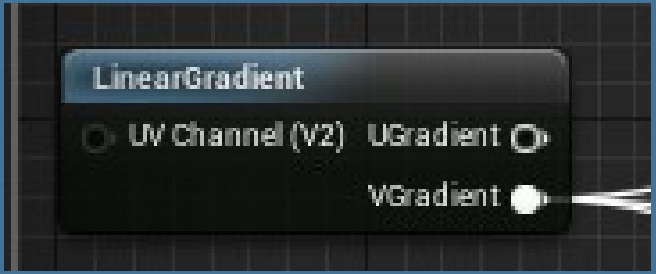
Output:

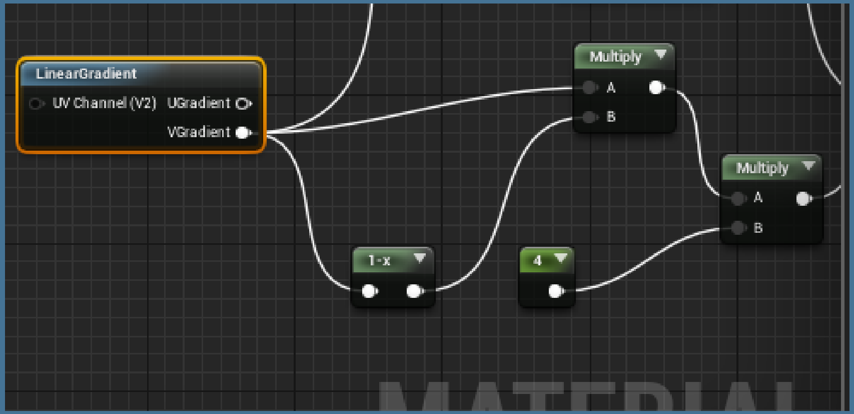


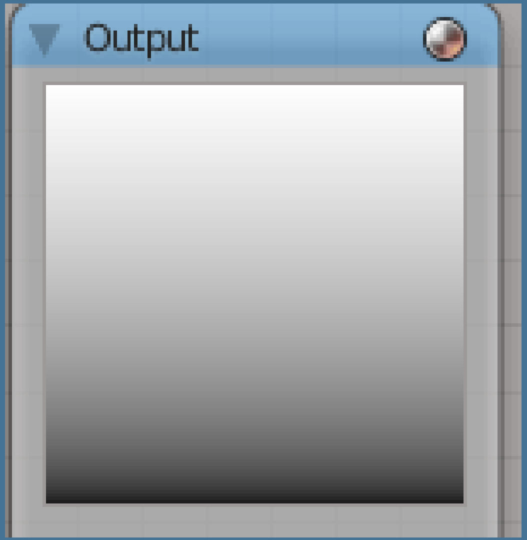
Output:

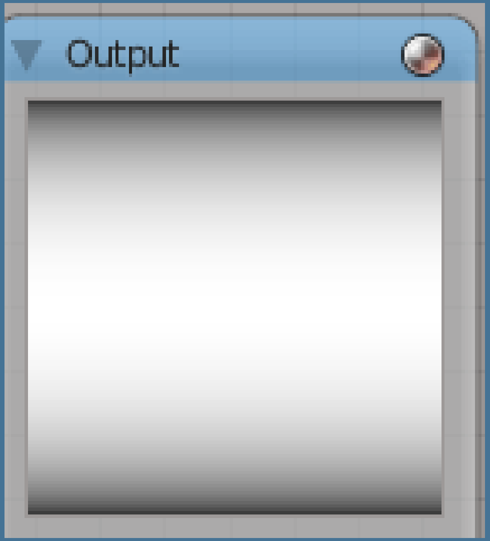


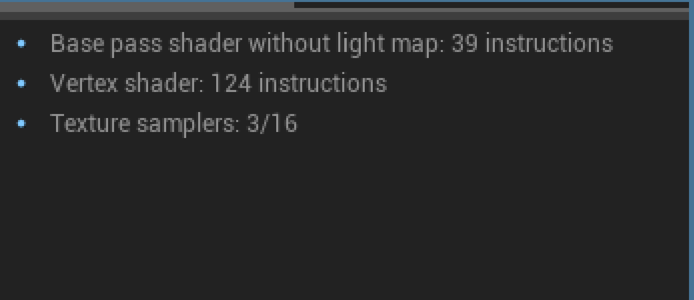
FOUNTAIN
SHADER











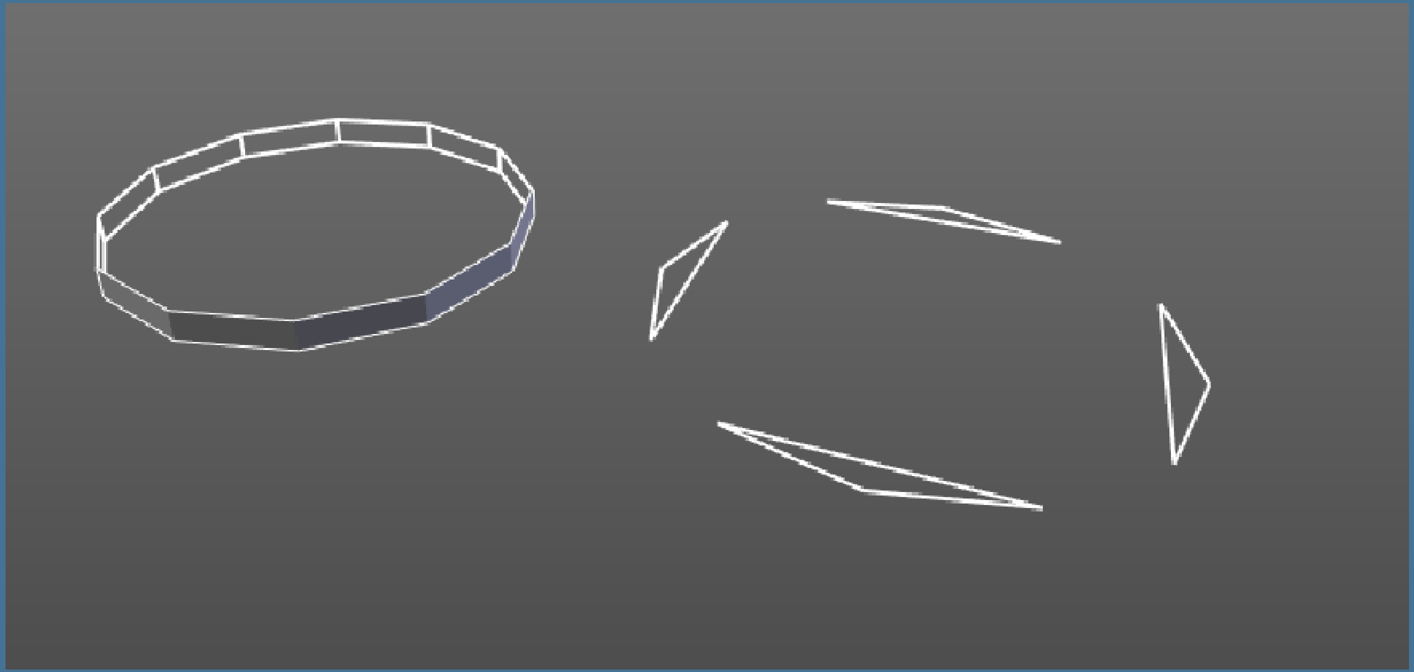
- Base pass shader without light map: 39 instructions
- Vertex shader: 124 instructions
- Texture samplers: 3/16

I reduced the texture pool size by using procedural texture.

Next to that I reduced a simple programming. By baking calculations.

Last thing I did was making the material unlit to reduce the instructions for the shader

FOUNTAIN
PARTICALS



I reduced the overdraw by letting the particle system spawn less particles.

In the system every vertex in the object spawns a vertex.
Also the vertex needs a face attached to it.
Previews model to spawn had 48 vertexes(hard surface dubbel the count). I designed this pattern with the flipped faces for less rendering and less data. And the exact points needed

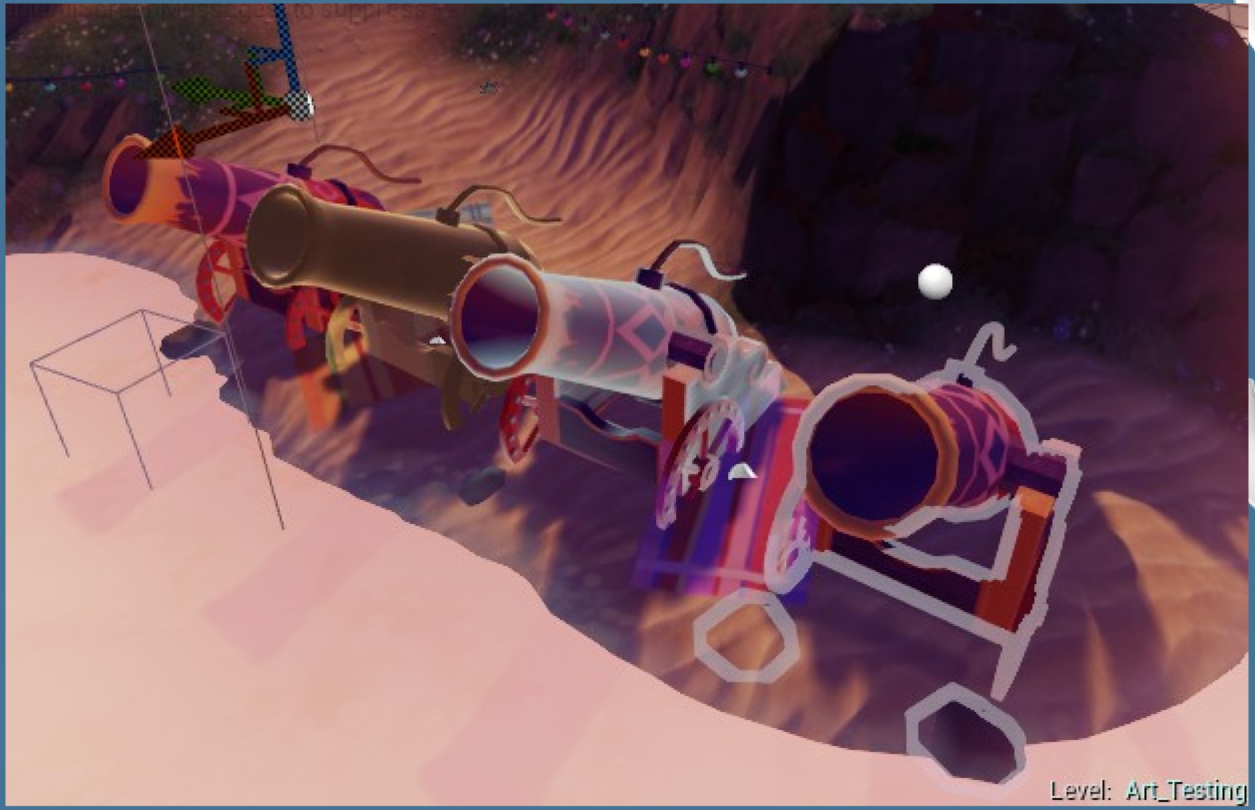
OPTIMIZATION RESEARCH

These are things to look at, for optimization
- light complexity: Amount of lights touching an object
- draw calls
- mesh complexity
- shader complexity
- is the mesh casting shadow?
- shadowmesh complexity
- proper streaming
- effective culling system
- amount of lights in scene
- amount of static meshes
- amount of dynamic meshes
- overdraw and overshading

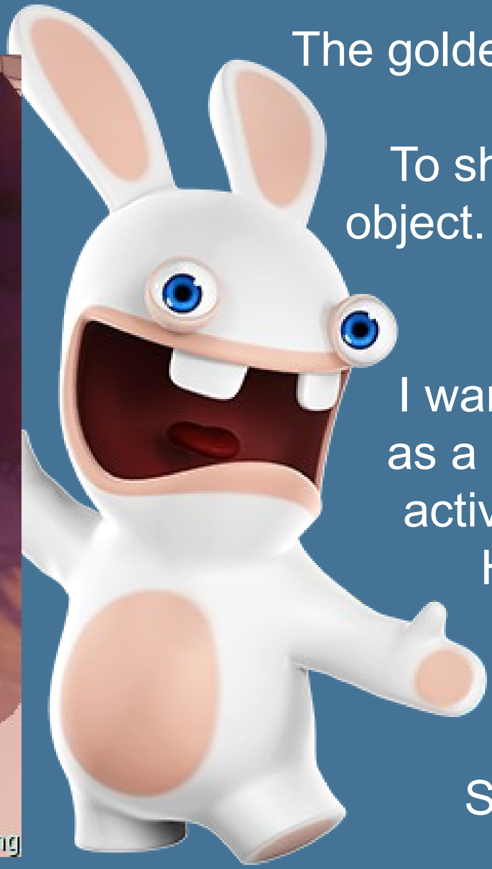
- good LOD use
- LODproxies
- amount of particles
- texture resolution
- shadow resolution
- amount of lights casting shadow
- size of mesh versus use of shader
- batching

Links:
https://discourse.techart.online/t/resources-on-optimizing/4453/7
http://discourse.techart.online/t/batch-rendering-optimization/900

Selection Shader



From left till right: normal render, gold replace, Fresnel material version, outline postprocessing effect.

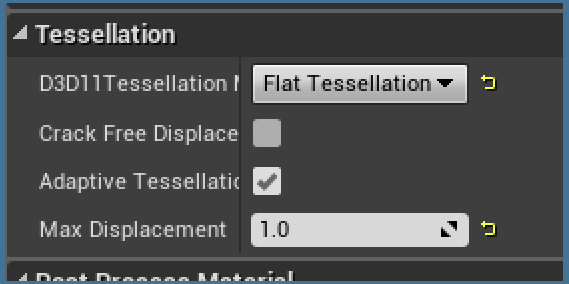


The golden selection shader is not what the art lead want as a shader.
To show the player that they grabbed a object. We wanted a outline or a Fresnel effect on the object.

I wanted to use post processing effect, as a pipeline since then it is a matter of activating the custom depth rendering. However we heard there are some troubles rendering post processing effects in VR.

So we also made a master material for the shading.
So you still can replace the material with a other one. But now blended.

WATERSHADER


Tessellation

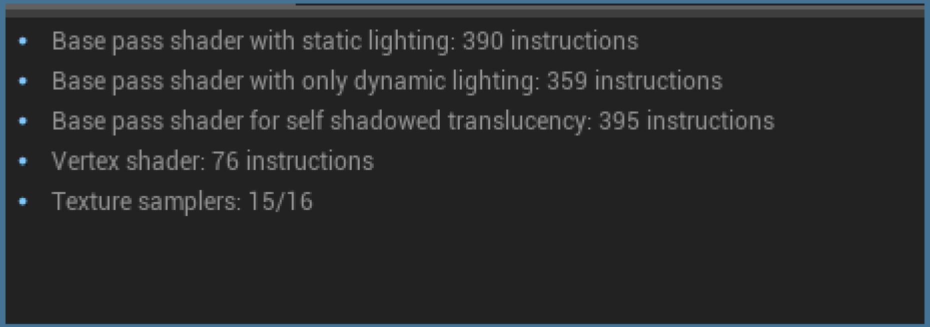
D3D11Tessellation I: Flat Tessellation

Crack Free Displace: ☐

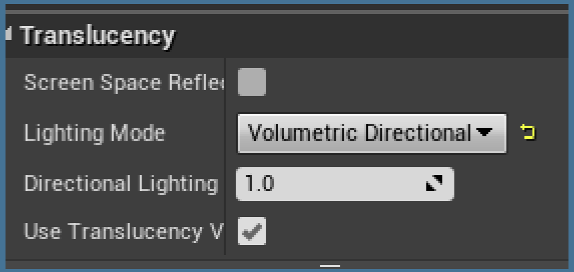
Adaptive Tessellatio: ✓

Max Displacement: 1.0

Since next to the displacement no modifications needed to occur we dropped the smoothing. The mesh was 1 shape so no need for the crack free displacement. And to reduce the amount of triangels and the amount of tessellations iterations we added adaptive tesselation


- Base pass shader with static lighting: 390 instructions
- Base pass shader with only dynamic lighting: 359 instructions
- Base pass shader for self shadowed translucency: 395 instructions
- Vertex shader: 76 instructions
- Texture samplers: 15/16

Before I started optimizing the water this was the outcome of the shader computations.


Translucency

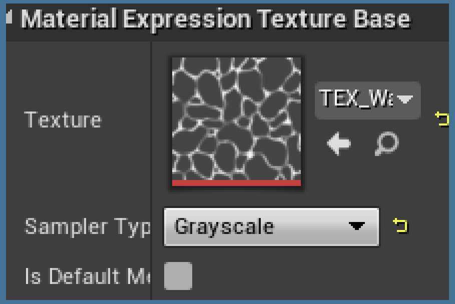
Screen Space Refle: ☐

Lighting Mode: Volumetric Directional

Directional Lighting: 1.0

Use Translucency V: ✓

Translucency is lighter with these settings

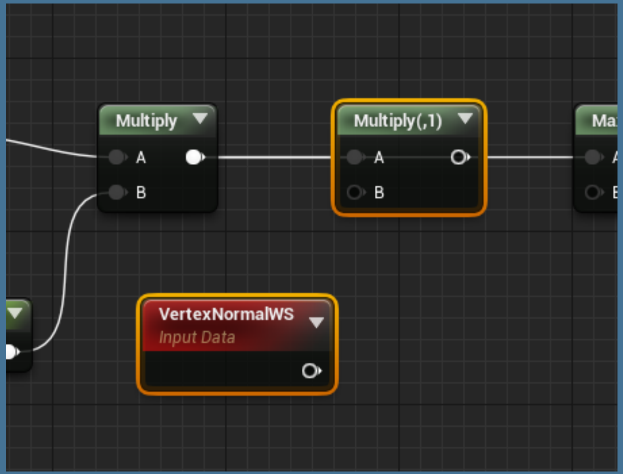

Material Expression Texture Base

Texture: TEX_Wa

Sampler Typ: Grayscale

Is Default M: ☐

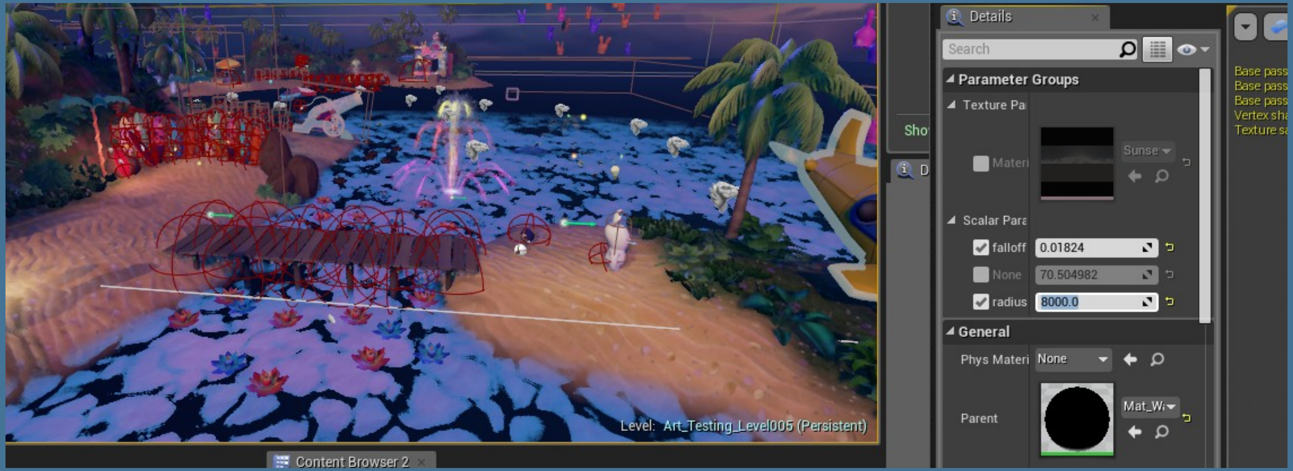
Textures do not need need to be sampled at color when they are grey. This reduces the output with 3 channels less. Pick the right sampler type



Since the start vertex normal are the same as the world coordinates. There is no reason to input the vertex normal. It reduces calculations.

WATERSHADER

Distance there is no displacement



Details

Search

Parameter Groups

Texture Pa

Materi — Sunse

Scalar Para

falloff 0.01824

None 70.504982

radius 8000.0

General

Phys Materi None

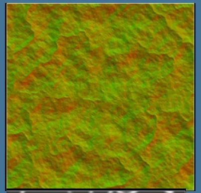
Parent Mat_Wi

Level: Art_Testing_Level005 (Persistent)

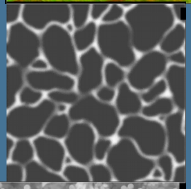
Content Browser 2

Compress textures to 1 texture

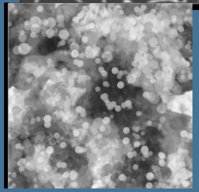
RG- Normal

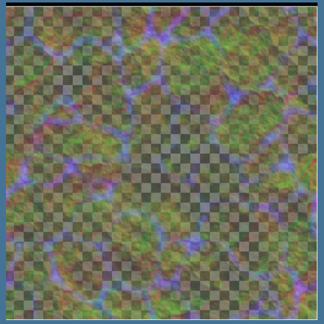


B-generalFoam

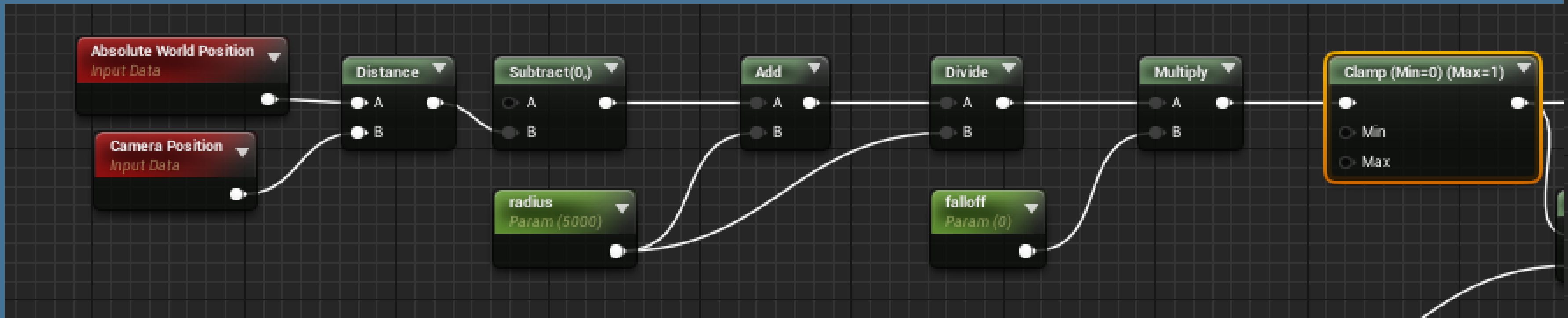


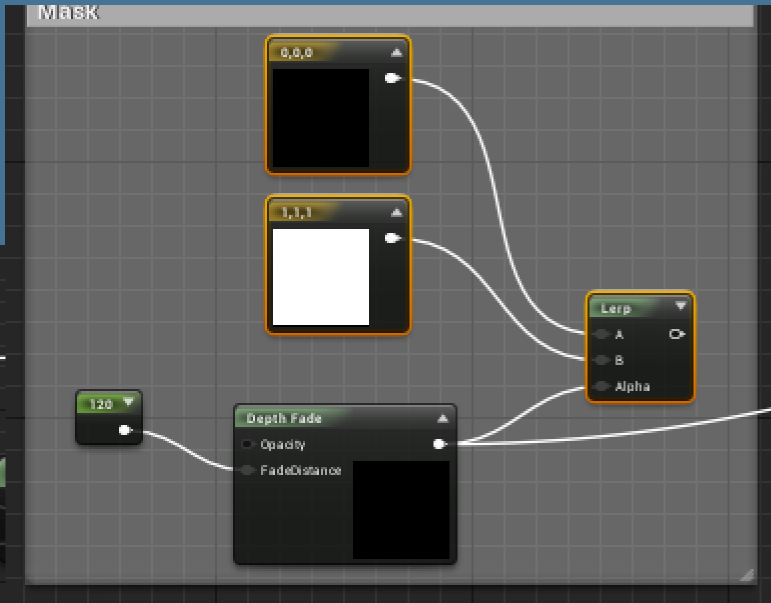
A-EdgeFoam





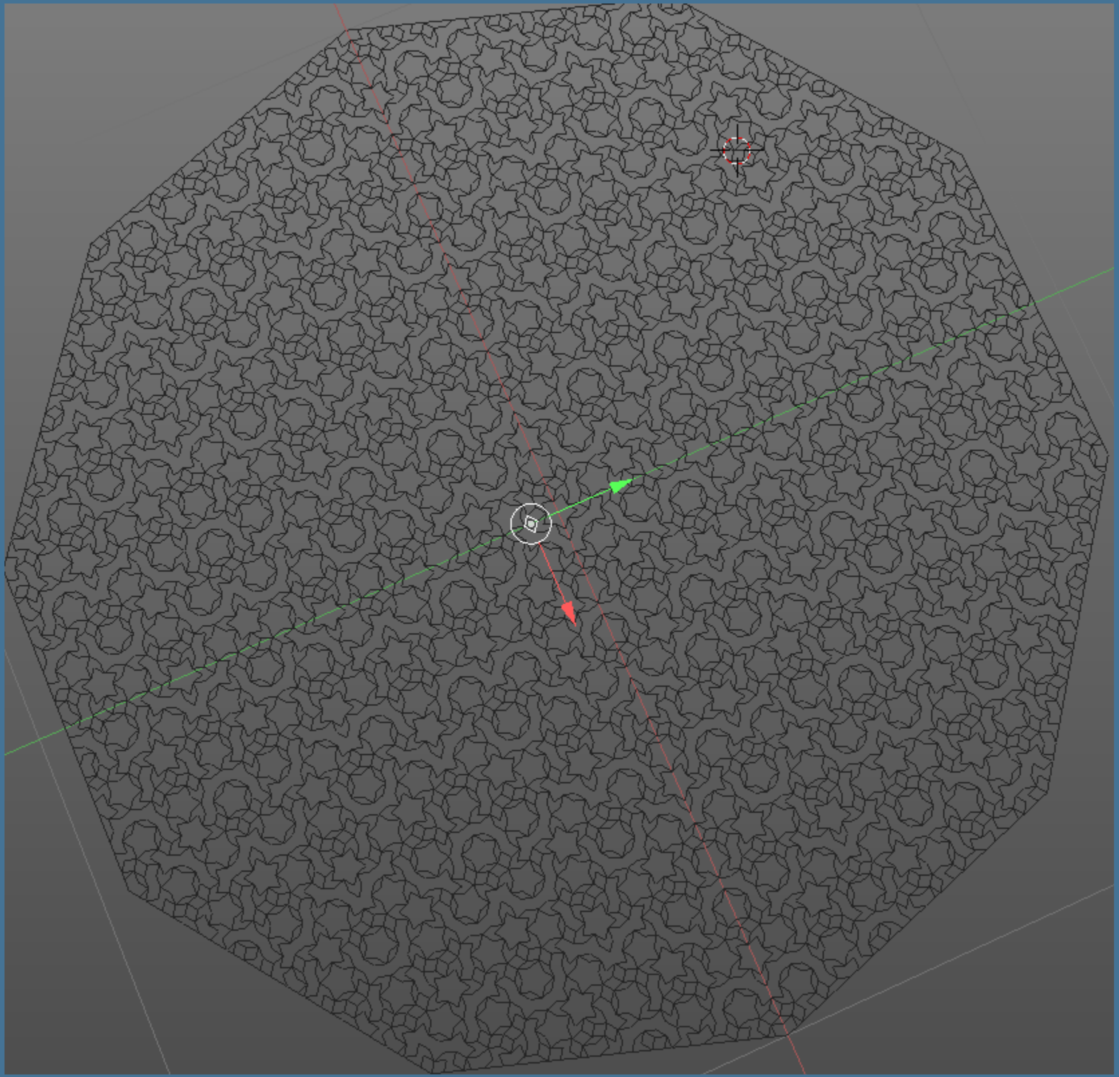
This calculates the distance at a cap from your camera. Making it possible to calculate the distance



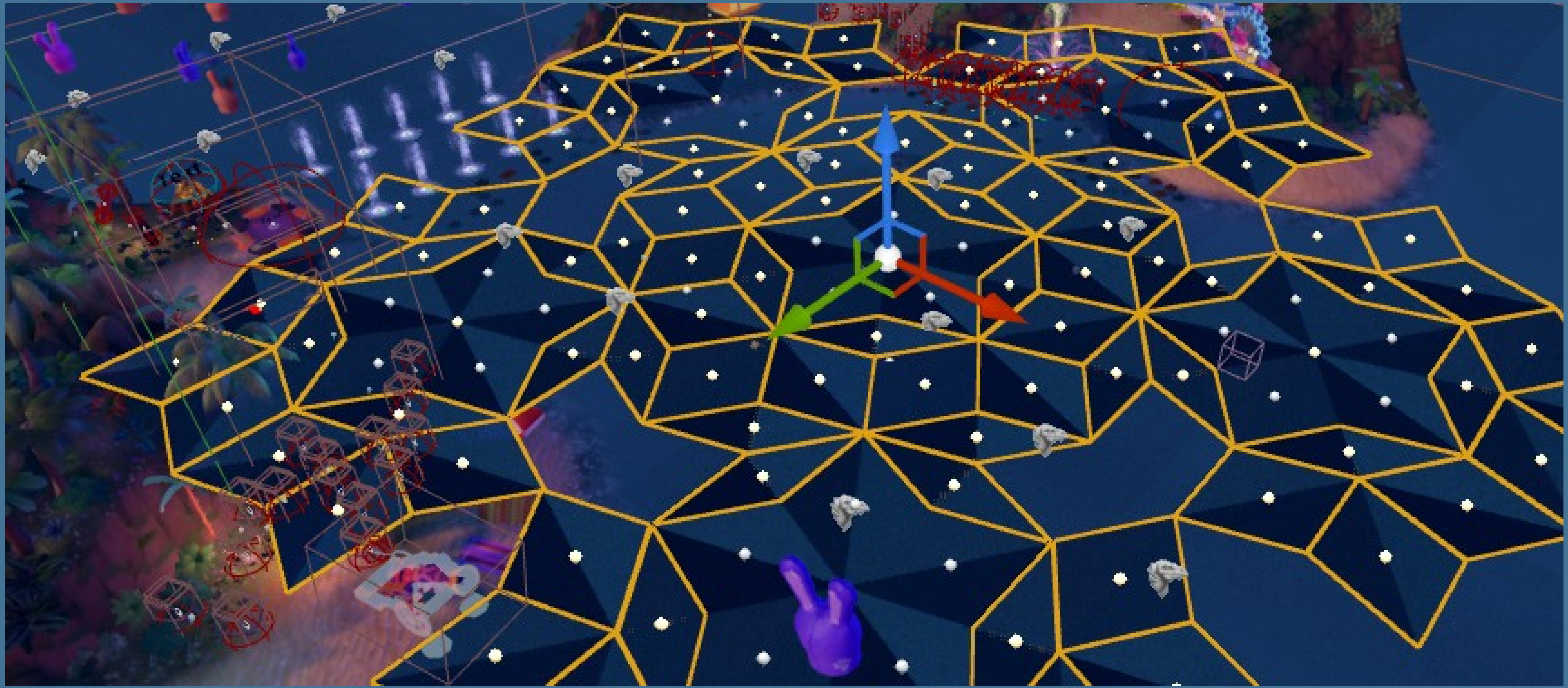


Tiling Water Planes

- First planned to use Penrose tiling and subdivide that. Since it is more round over distance.
- However it is hard to create and took to long to implement. So I go over to hexagon tiling



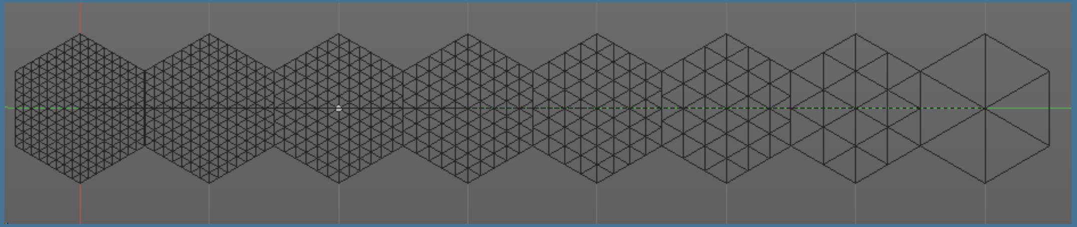
• IMPLEMENTATION

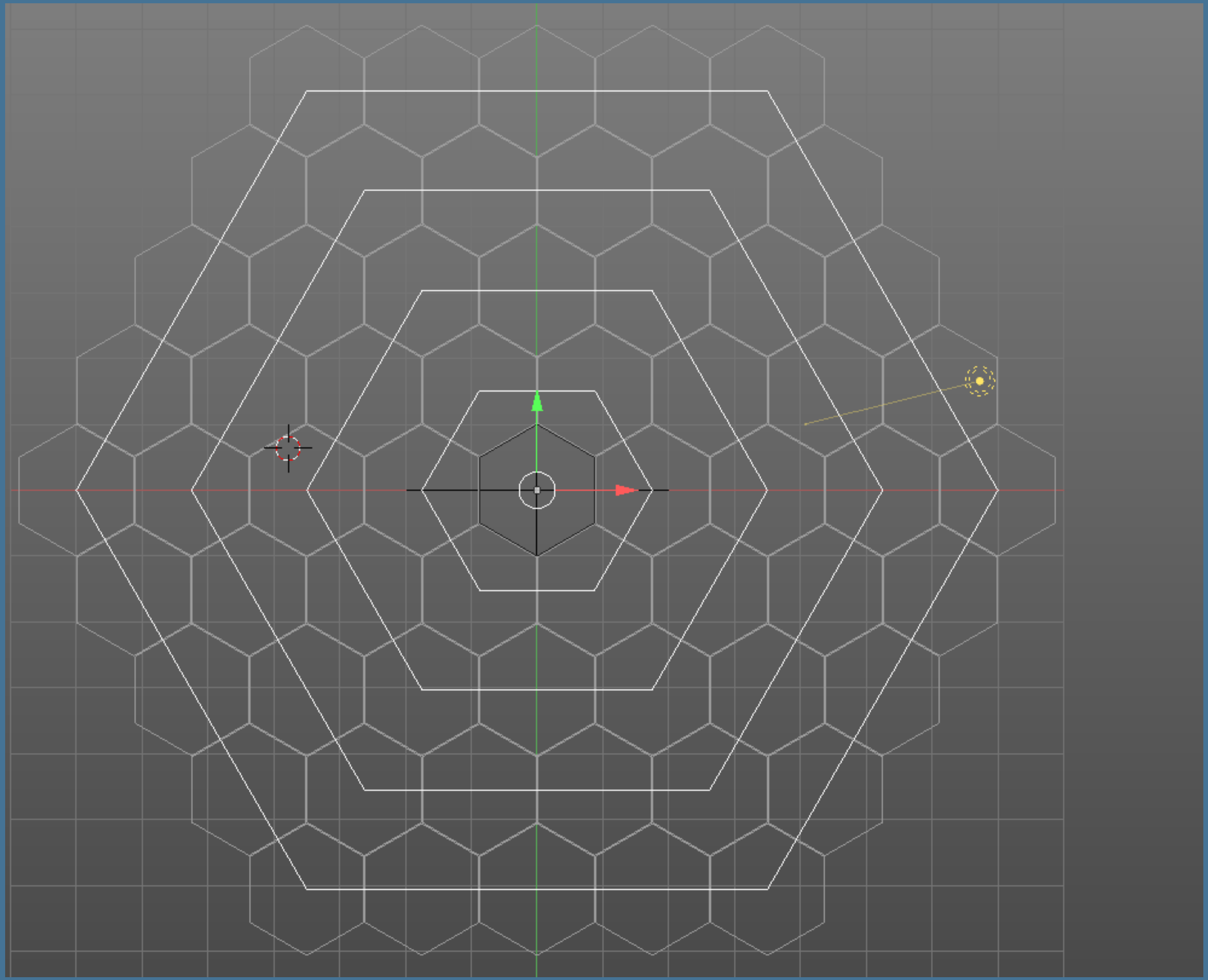


Made blueprint to use repeated shapes easier. However make the tiles fit perfectly was a hassel. So needed to find a new solution.

• HEX TILES

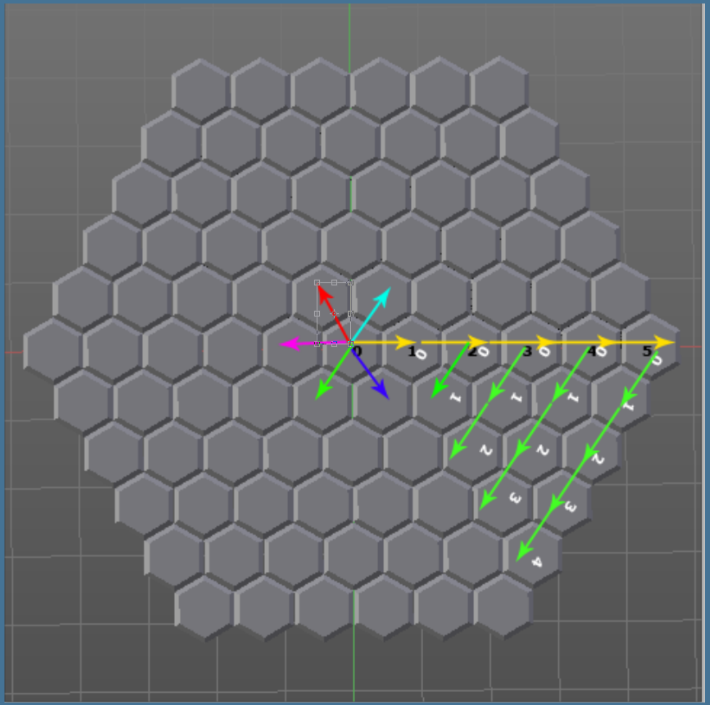
Hex tiling is a easy placement.
Only thing required is a scaling of positions
 and sub diving the placements.



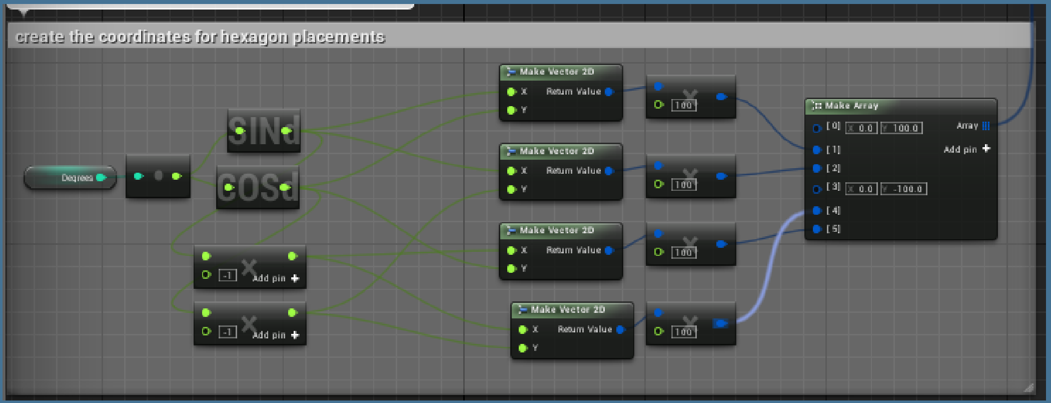


HEXAGON LAYOUT

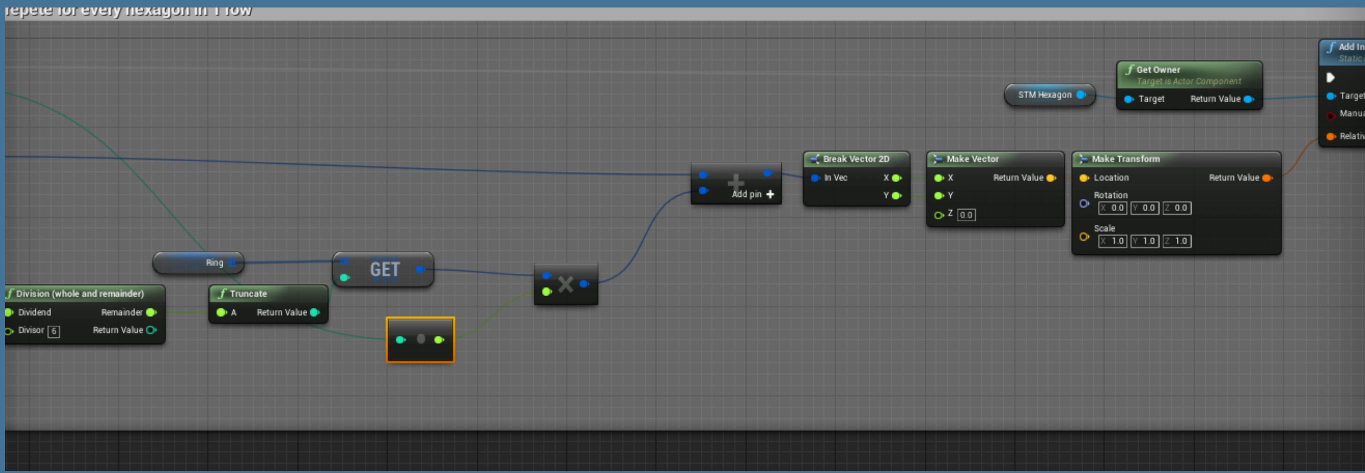
Made a algorithm that places the copy's of the hexagon at the exact place that it is tiling.
Made 6 vectors and added them together to place them at the right spot.



Add vectors.

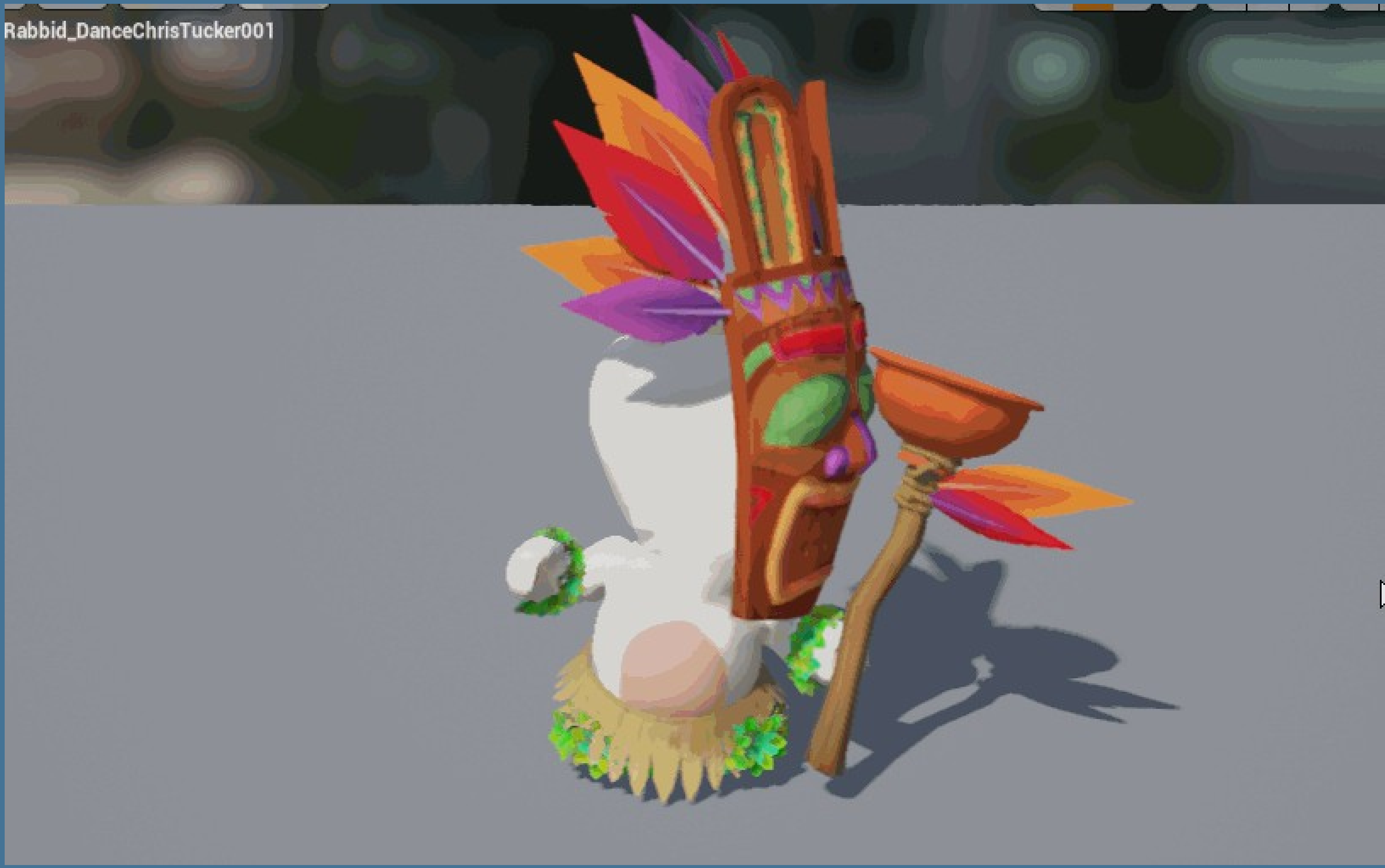


Combines vectors and place the hexagon there.

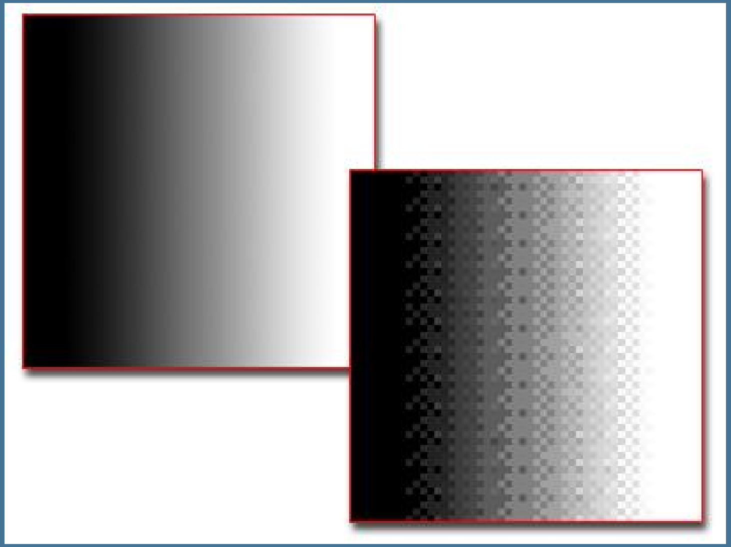


IMPLEMENTING A SKIN

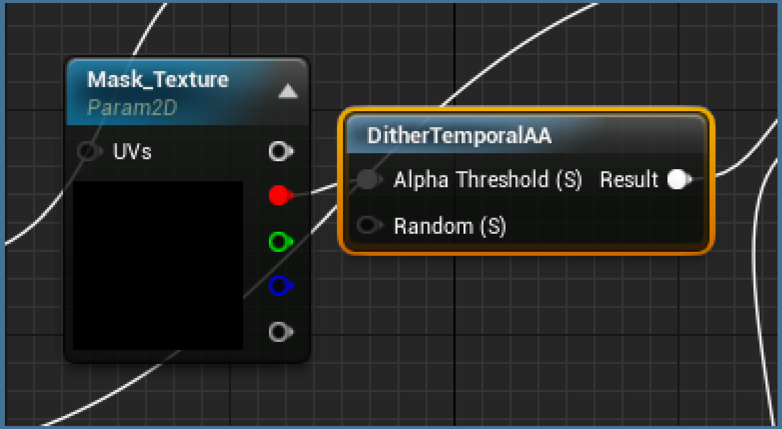
- For the game we wanted the characters to be able to have a skin. I used sockets for the skin implementation. Later I learned about a optimized skinned version.



SHADER OPTIMIZATION

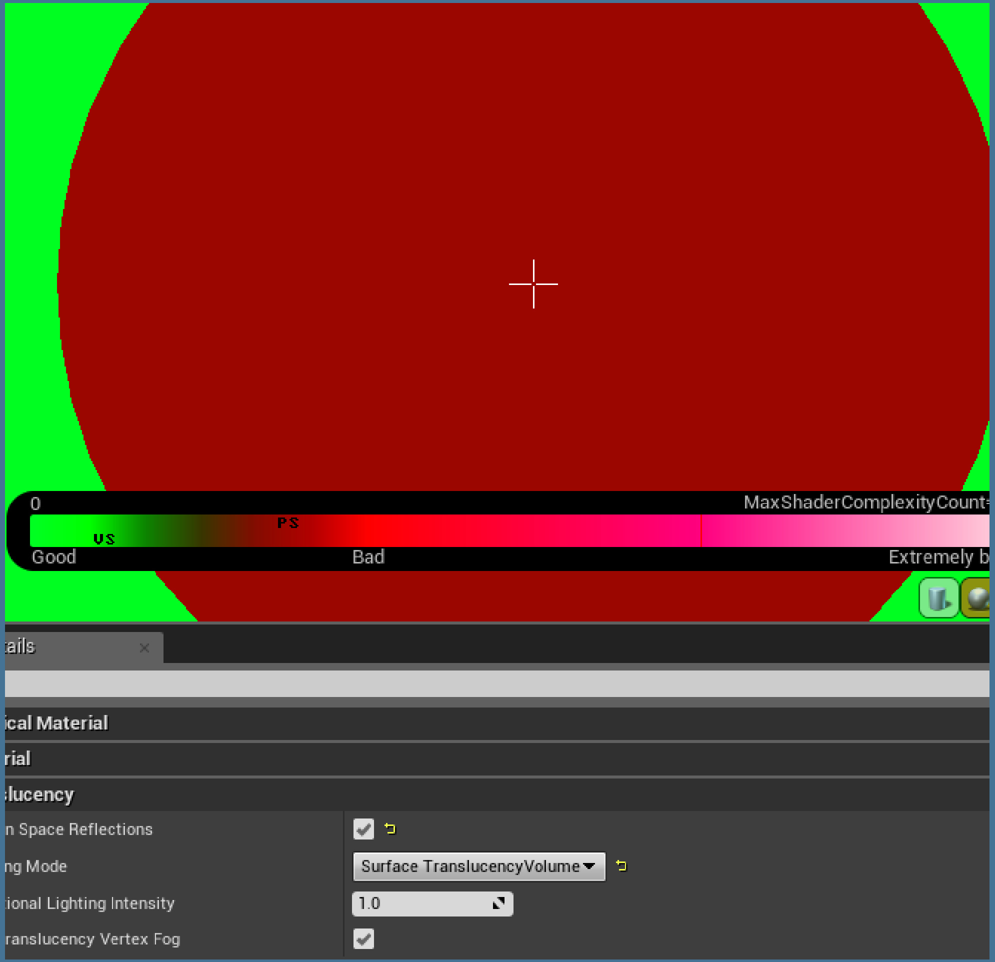


- Dithering. We now have non mask transparency in the game.
- We use Dithering to keep the shader simple but that there still a transition between transparency and opacue



SHADER OPTIMIZATION

- For the water the biggest optimization was the calculation of the tranparancy. Changing this dropped no quality but did make the shader half as heavy.



ical Material
rial
slucency

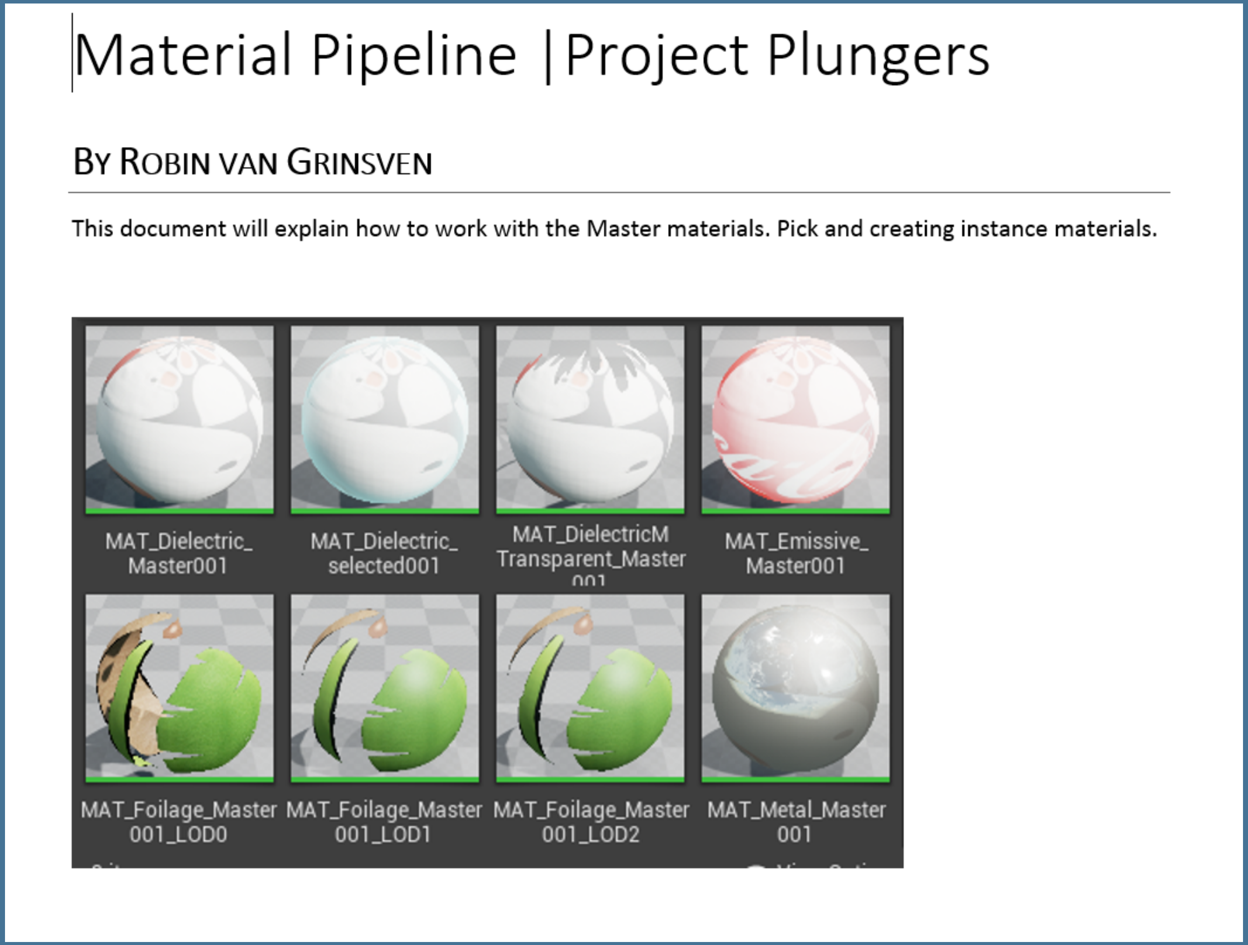
n Space Reflections	✓
ng Mode	Surface TranslucencyVolume
ional Lighting Intensity	1.0
ranslucency Vertex Fog	✓

MASTER MATERIAL DOCUMENTATION

As requested by a artist. I made

A small documentation on how you make instance materials in unreal.

https://swarm1.nhtv.nl/l/6gpds0



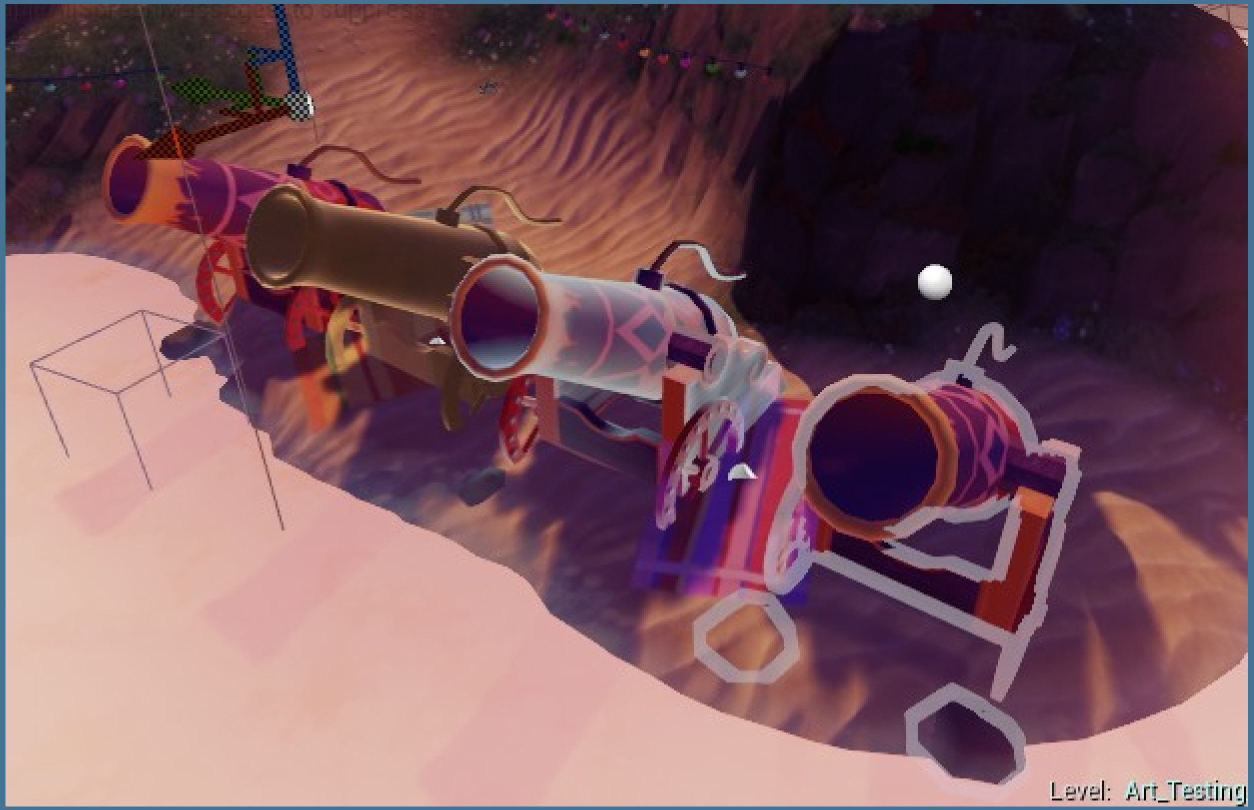
Material Pipeline |Project Plungers

BY ROBIN VAN GRINSVEN

This document will explain how to work with the Master materials. Pick and creating instance materials.

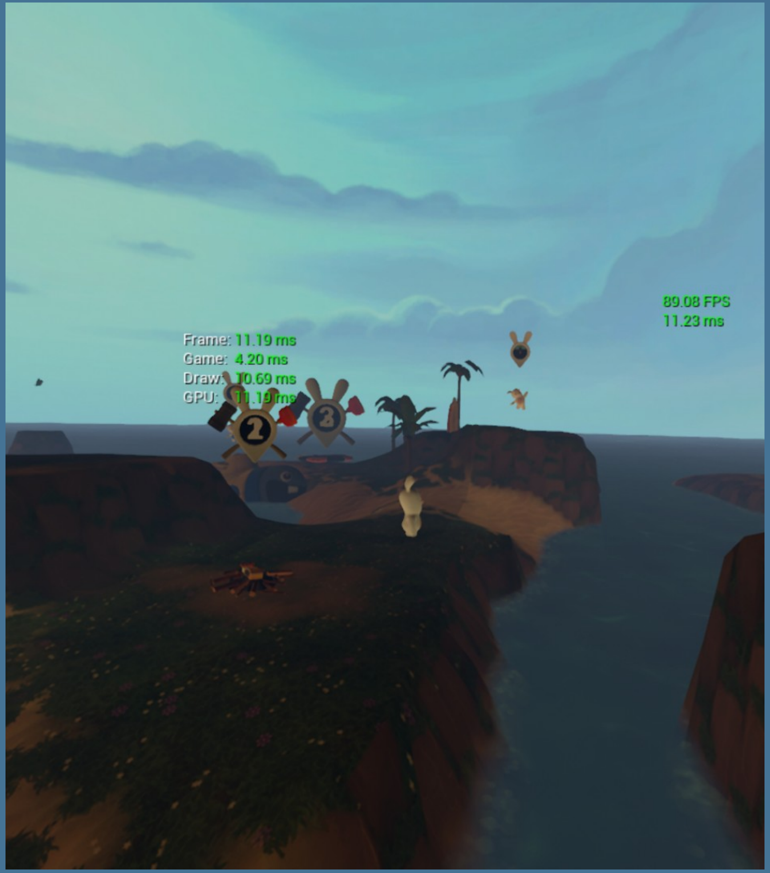
Selection Shader

- The vision of the selection Shader is clear. Only thing that needs to happen now is defining the pipeline. This can be done via postprocessing or material switch. We go for material switch, since then we do not have to spend to much time on it since we already got the mastermaterial.



VR FORWARD RENDERING

Forward- rendering
89 fps, game: 4.20 ms
draw: 10.69 ms
gpu: 11 ms



```
89.08 FPS
11.23 ms
Frame: 11.19 ms
Game: 4.20 ms
Draw: 10.69 ms
GPU: 11.19 ms
```

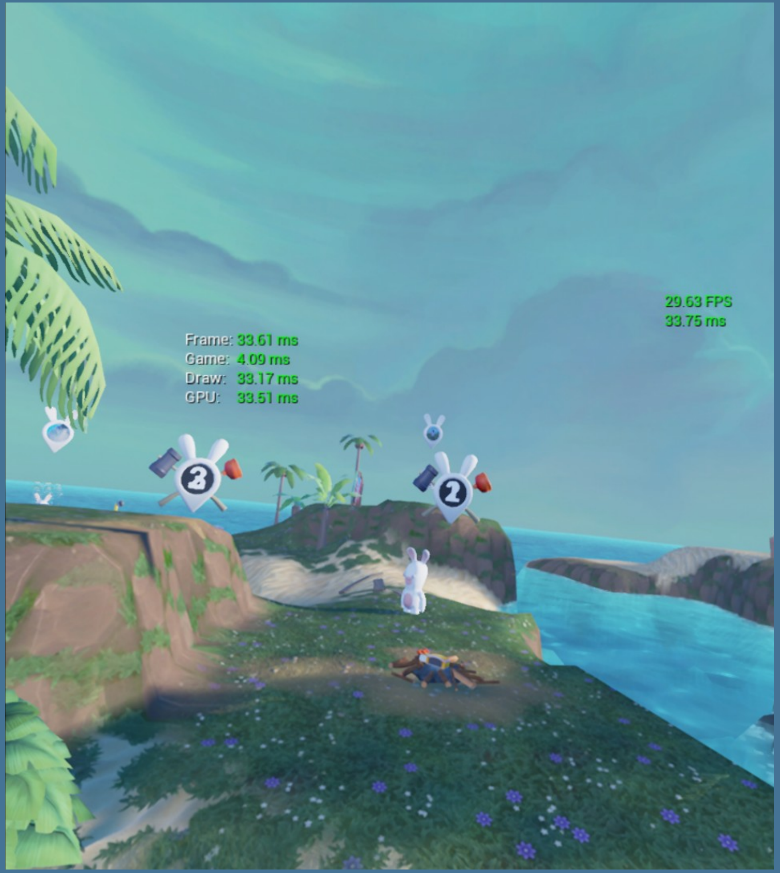
- As adviced by Unreal I need to use forward rendering for VR. So setting this I need to check for improvement and bugs.

- When I checked I noticed the art is destructed to much.

- Mainly the lighting would have been to much off.

- If we started with forward rendering. It was possible to take the lighting in to account.

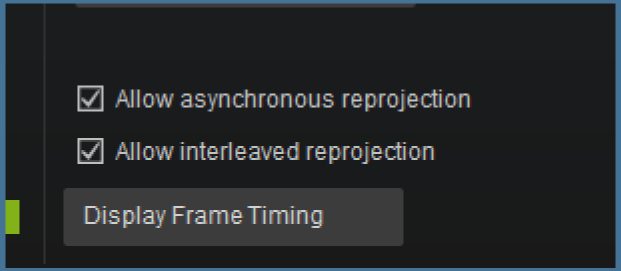
Forward- rendering
29 fps, game: 4.09 ms
draw: 33.17 ms
gpu: 33.51 ms



```
29.63 FPS
33.75 ms
Frame: 33.61 ms
Game: 4.09 ms
Draw: 33.17 ms
GPU: 33.51 ms
```

- TESTED SETTING IN VR.
- DID IMPROVE FRAMERATE BUT LAGGED EVEN MORE

The setting that fixes the lagging the object had a mayor downside on the framerate overall.

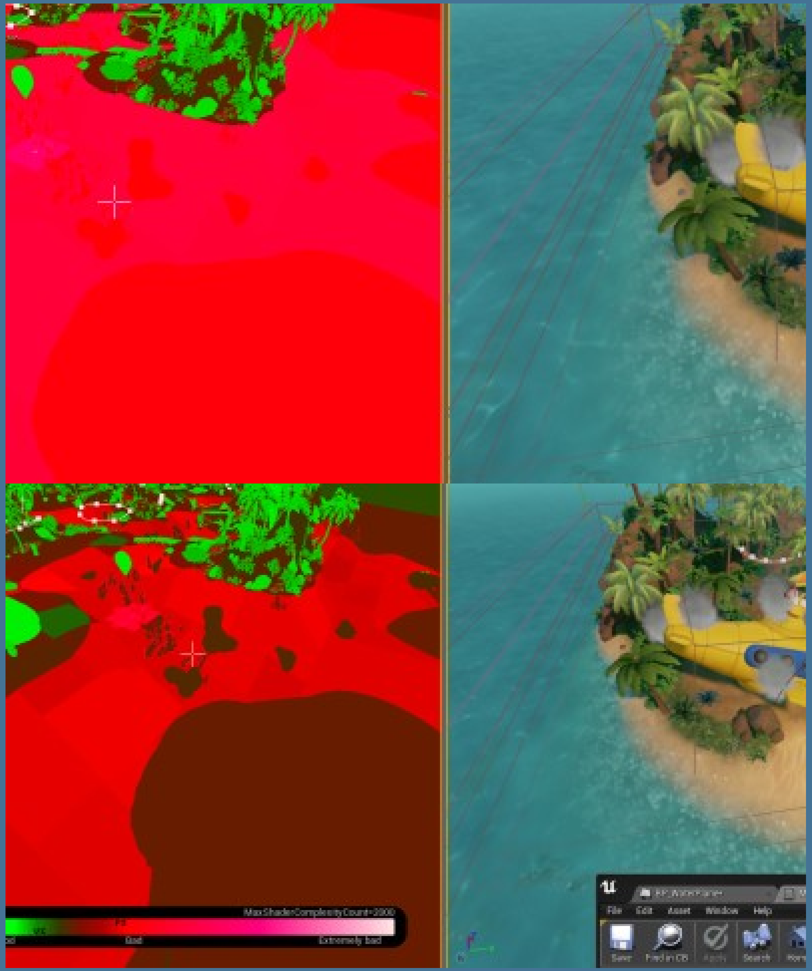


☑ Allow asynchronous reprojection

☑ Allow interleaved reprojection

DAYTIME WATERSHADER

- The nighttime shader has the requirement to have reflection for the shader. Day time shader was able to not have this. The cheaper calculation of tranluency could be applied



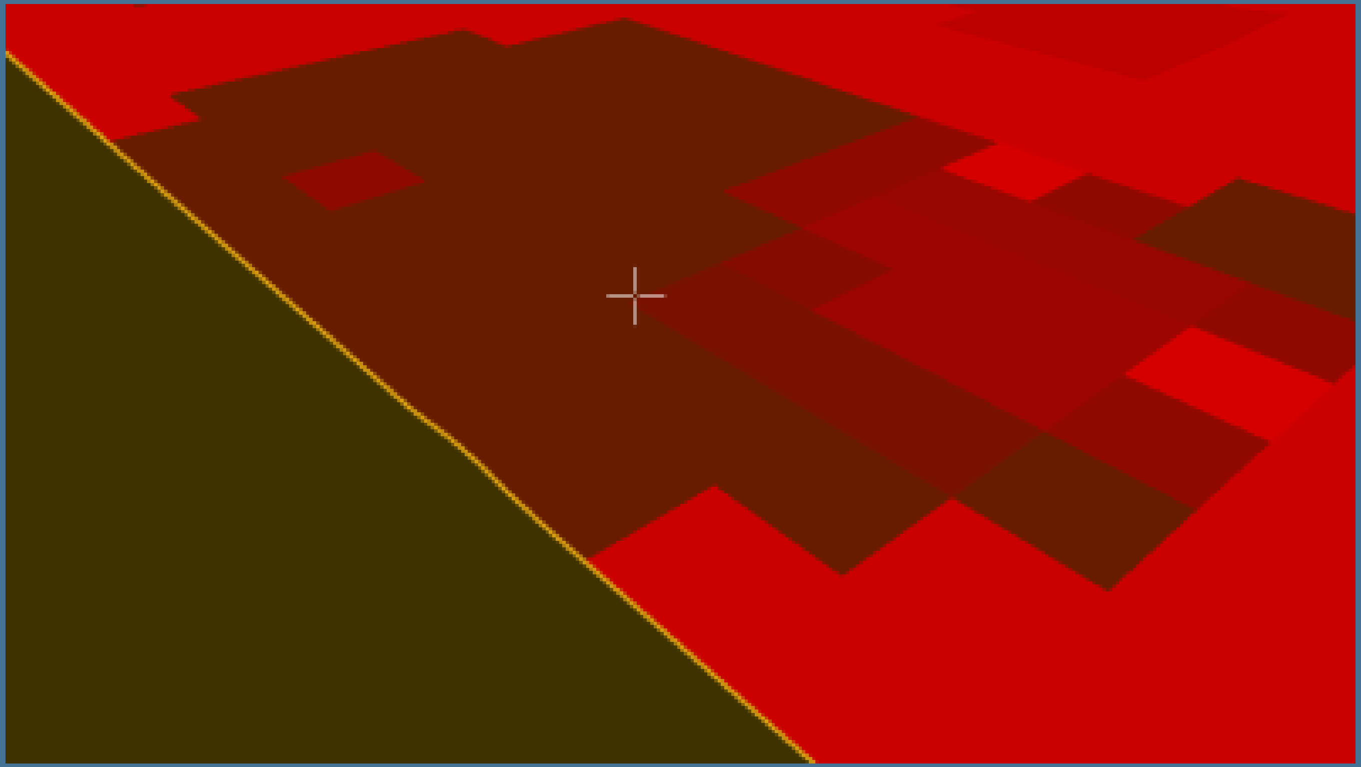
This time I took time to lay the effect side by side. To find what visual impact the optimization has. Here u see there is visually none. And it got a go by femke

LANDSCAPE IMPROVEMENT

Trancluency add's shader the shadercomplexity from the underlaying shader.

On the land shader. Complexity increases when more then 1 shader is applied on the tile.

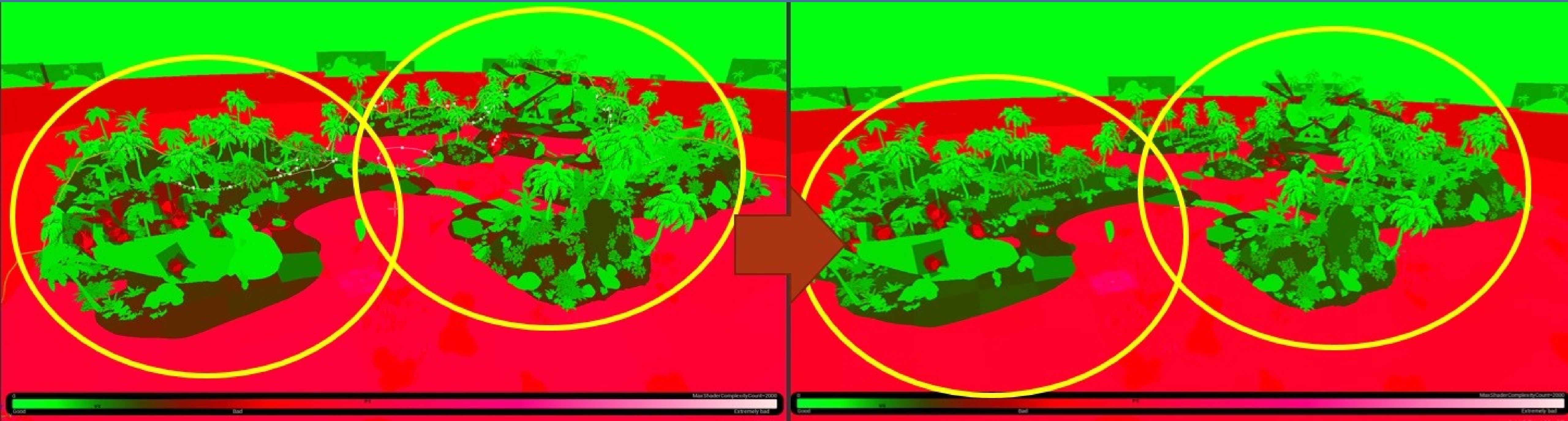
So the plan was to make islands of shaders where on the edge there is a transition. Between one and the other island.



LANDSCAPE IMPROVEMENT

OUTSOURCED

As better improvement I asked our envoirment artist. To make 1 grayscale texture that blend between the 2 textures of the 2 materials. This way all under the water is simple material and the watershader should not be to heavy for this.



NIGHT TIME SHADER

Since we wanted to have a the night time shader to have reflection, however the quality is hard to recreate