

Rapport Informatique Graphique

Gauthier Bouyjou

fait le 16 novembre 2019

Sommaire

Description du moteur.....	2
1. Compilation.....	2
2. Description.....	2
Partie Rendu.....	3
1. Shadow map.....	3
2. Cubemap.....	7
3. Environment map.....	7
4. SkyBox.....	8
5. Percentage closer filtering shadow map.....	9
6. Rendu HDR.....	10
Partie Géometrie.....	11
1. Nurbs (édition interactive).....	11
2. PN Triangles.....	12
Bonus.....	12
1. Camera.....	12
2. Texture.....	13
3. Shader program et vues multiples.....	16
4. Lumière.....	16
5. Matériaux.....	18
Optimisation.....	20
1. World Space Frustum culling.....	20
2. Real time point light map calculation.....	20

Description du moteur

1. Compilation

CMake classique

```
$> mkcd build && cmake -DCMAKE_BUILD_TYPE=Release .. && make &&  
.applications/mainApplication/MainApplication
```

Vous devez avoir sur votre système assimp et glm installés (pas de sous module git)

Sinon spécifiez l’include path des deux librairies en modifiant le CMAKE_CXX_FLAGS en ajoutant -I votrePathAssimpEtGlm/include/

2. Description

Le moteur a une composante UI très proche de Blender, donc pour la fenêtre de vue vous pouvez retrouver les différents modes (Object, Edit, Pose) et le shader program courant de rendu (Rendered, Depth, LookDev, Normal, Solid, etc).

En sélectionnant un objet vous pouvez effectuer diverses transformations avec les mêmes shortcuts de Blender. (g : translate, r : rotate, s : scale)

Je vais faire souvent référence dans la suite de ce rapport à ces modes de rendu (Rendered, Depth, ...) qui correspondent à ces divers shader programs spécifiés plus tôt.

Page github : https://github.com/hiergaut/Goliath-Engine_standalone

Partie Rendu

J'implémente l'algorithme du forward rendering.

Voir fichier `src/engine/scene/Scene.cpp` méthode `draw`

1. Shadow map

Source code :

```
src/engine/scene/light/*  
resources/shader/shadow/*
```

Référence :

<https://learnopengl.com/Advanced-Lighting/Shadows/Shadow-Mapping>

<https://learnopengl.com/Advanced-Lighting/Shadows/Point-Shadows>

Code personnel :

Du code personnel a été rajouté (fragment shader `resources/shader/shadow/*`) dans cette partie concernant les ombres des objets partiellement transparents, utilisation de textures d'opacités sur Sponza ou de la transparence en règle générale alpha < 0.5 de la texture diffuse.



Figure 1: gauche: sans opacité, droite: avec opacité, remarquez la différence sur l'ombre projetée par les herbes sur le tissu bleu.

a) Directionnel

Création d'un fbo autre que celui par défaut (screen) pour pouvoir écrire dans une texture de profondeur.

La matrice de projection orthographique va venir englober la scène avec l'aide de sa boîte englobante, quelle que soit la direction du soleil on va venir minimiser les zones de la shadow map qui sont hors scène.

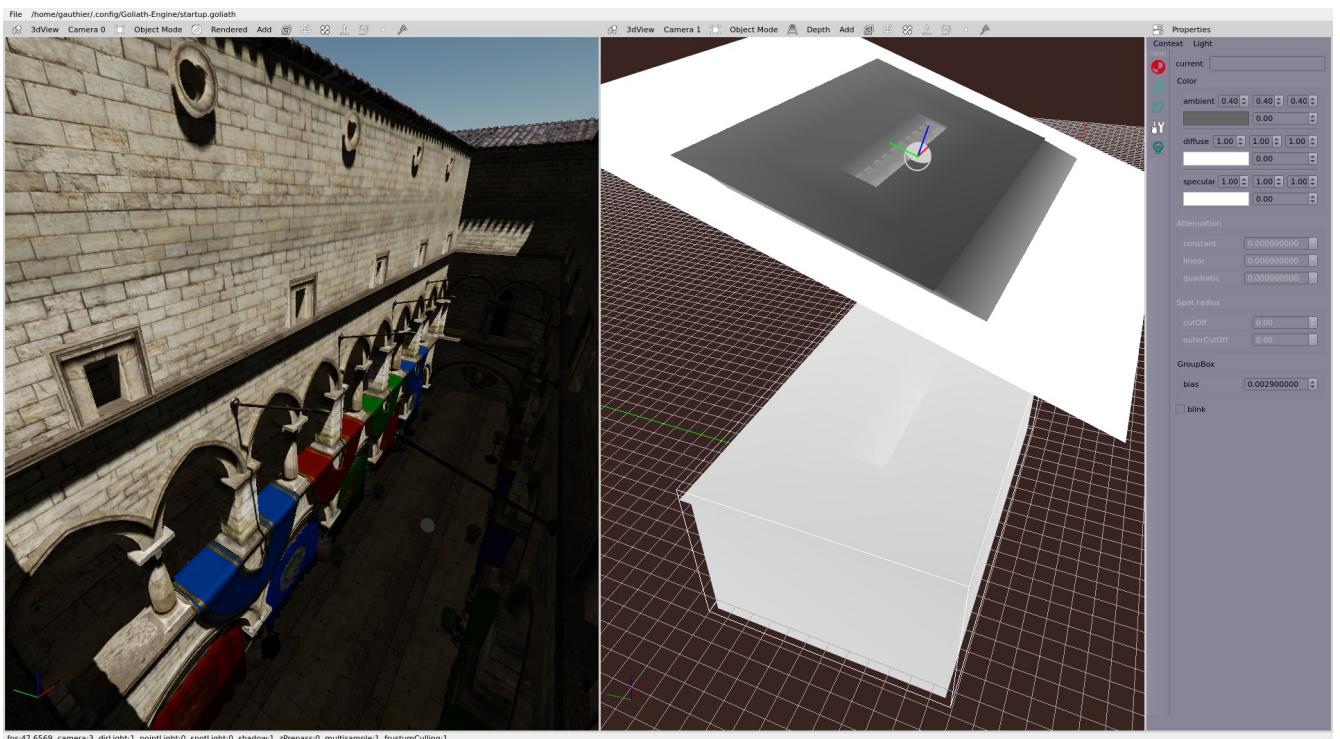


Figure 2: gauche: une lumière directionnelle en mode Rendered, droite: affichage de la shadow map en mode Depth (débogage, optimisation de la projection orthographique de la shadow map englobant la scène)

On peut voir la carte de profondeur du soleil, et donc voir comment elle se comporte en changeant la direction du soleil, le but étant de minimiser les pixels inutiles de fond qui sont représentés en blanc car représentant une profondeur infini.

On peut remarquer aussi la boîte englobante de la scène en blanc sur la vue de droite.

b) Point light

Calcul d'un cube map avec qu'un drawCall à l'aide du geometry shader réduisant ainsi le nombre de sampler dans les shaders (nombre limité de texture), car en effet on doit stocker chaque carte de profondeur pour déterminer si un fragment est bien dans une zone d'ombre ou pas.

Affichage du cubemap du pointlight avec le shader programme Depth activé.

Une liste des plus proches point lights pour chaque caméra est maintenue, qui permet de mettre à jour les éclairages ponctuels les plus proches de chaque caméra à chaque rendu et non tous les point lights de la scène car avec 20 point lights sur Sponza il y a un ralentissement des fps majeur même en réduisant fortement la résolution du cube map.



Figure 3: gauche: affichage en mode Rendered du point light et de son environnement, droite: visualisation du cubemap du point light en mode Depth, on remarquera que l'utilisation d'une environment map sur Sponza n'a vraiment aucun intérêt car le ciel est difficilement visible sur Sponza. Regardez le cubemap, la zone blanche représente une distance infinie, donc le ciel et qui serait un réel reflet d'un environment map mais tout le reste sera un non reflet. Il serait difficile de calculer s'il y a bien une intersection directe avec la scène ou l'environnement map sans faire de lancé de rayon (solution coûteuse).

c) Spot light

Fov de la projection perspective directement calculé avec l'ouverture de la torche.

Ce qui génère une texture de shadow map carrée, or la zone éclairée est circulaire, donc on a une zone calculée inutile à l'extérieur du cône d'éclairage de la torche.

Affichage du shadow map en mode Depth sous forme de texture carrée comme on peut le voir à droite sur la Figure 4 suivante.

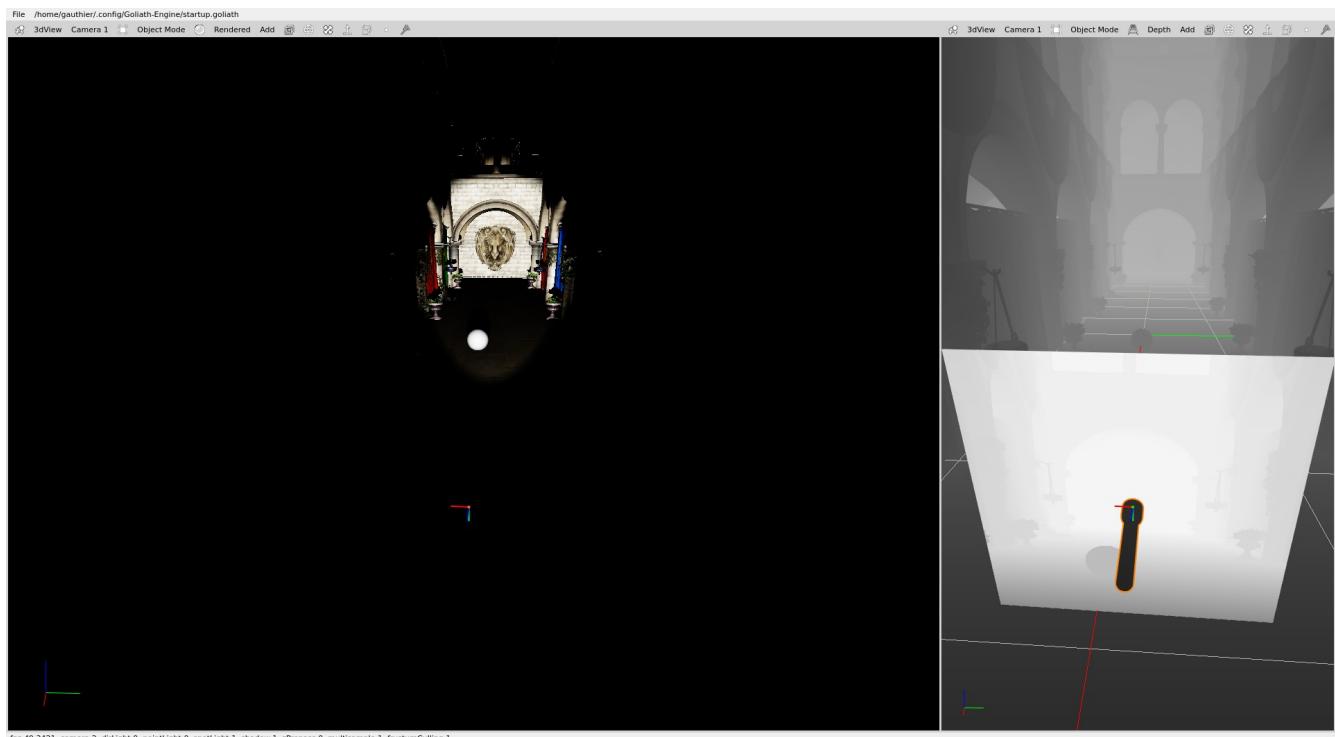


Figure 4: gauche: projection d'un spot sur Sponza en mode Rendered, droite: visualisation de la texture d'ombre en mode de rendu Depth, on remarquera que le spot a une projection conique et que donc une texture carrée n'est pas vraiment idéale. Il deviendrait intéressant d'étaler la projection circulaire du cône sur le carré afin d'optimiser la résolution (sans perte) de la shadow map.

2. Cubemap

Comme vous pouvez le voir plus haut sur la Figure 3, j'ai affiché les cubemaps des spot light pour débogage des cartes de profondeurs.

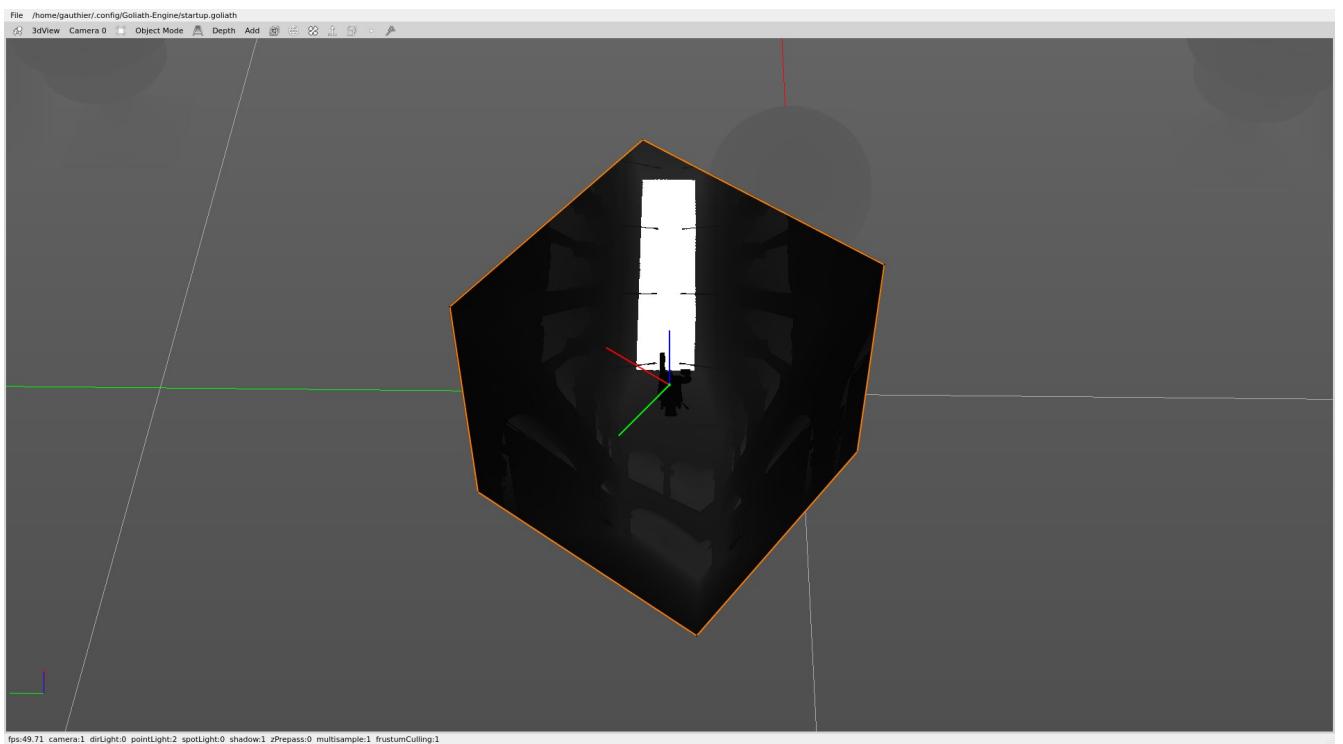


Figure 5: visualisation en mode Depth de la texture cube map représentant les 6 projections permettant d'obtenir les cartes d'ombrages du point light

3. Environment map

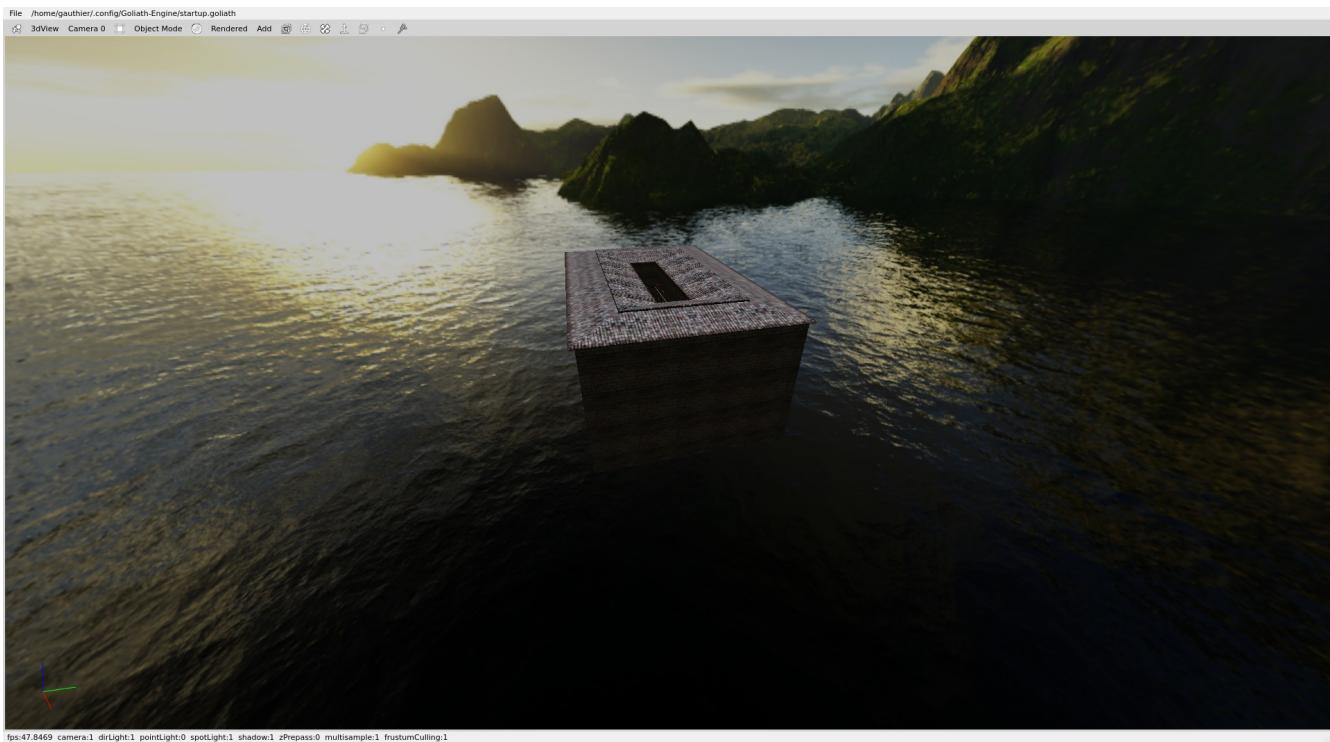
Utilisation des environment maps pour calcul de reflet non réalisé car sur Sponza il faudrait faire des calculs d'intersection si c'est bien un reflet de la cube map et non les murs des cotés.

Très compliqué à mettre en œuvre.

4. SkyBox

Référence : <https://learnopengl.com/Advanced-OpenGL/Cubemaps>

Skybox : http://www.custommapmakers.org/skyboxes/zips/hw_morning.zip



5. Percentage closer filtering shadow map

Référence : <https://learnopengl.com/Advanced-Lighting/Shadows/Shadow-Mapping>

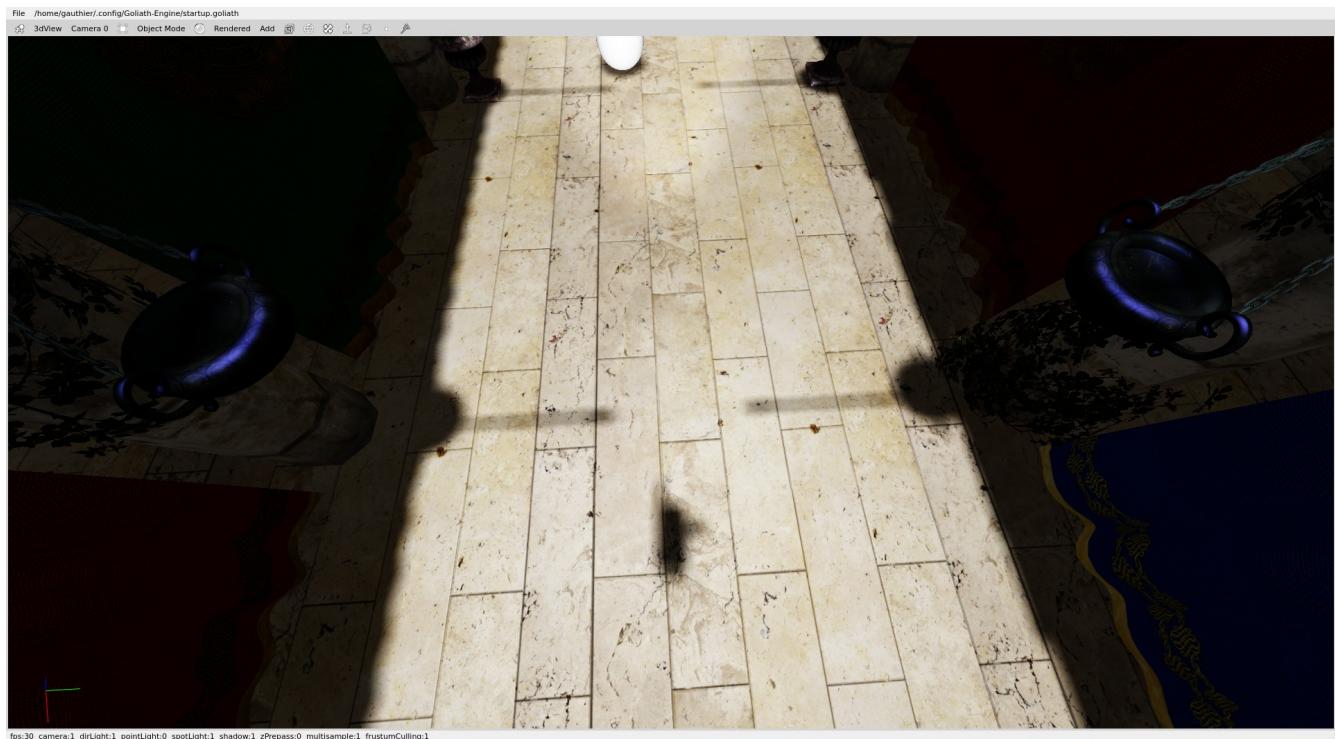


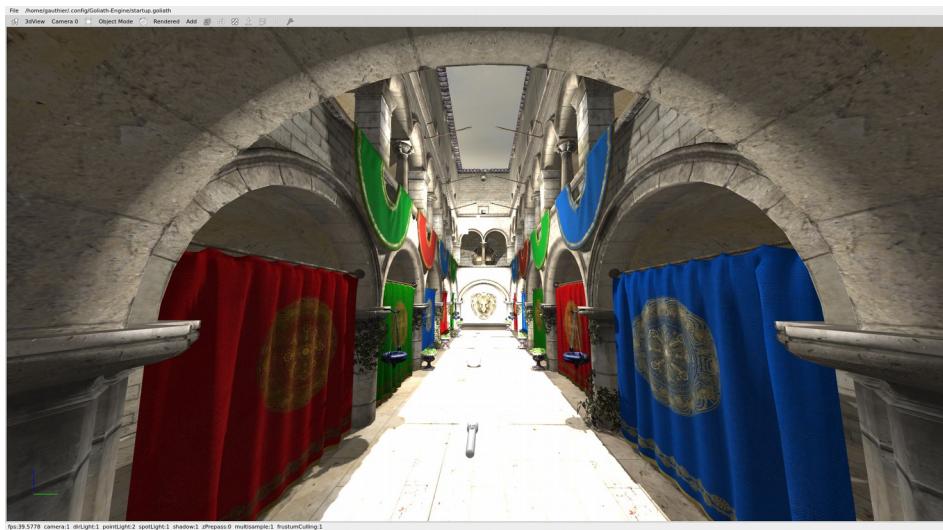
Figure 6: Taille du moyenneur de 10x10 fragments

Envie d'implémenter du Percentage-Closer Soft Shadows (taille du noyau proportionnelle à la distance objet obturateur/recepteur mais sans succès).

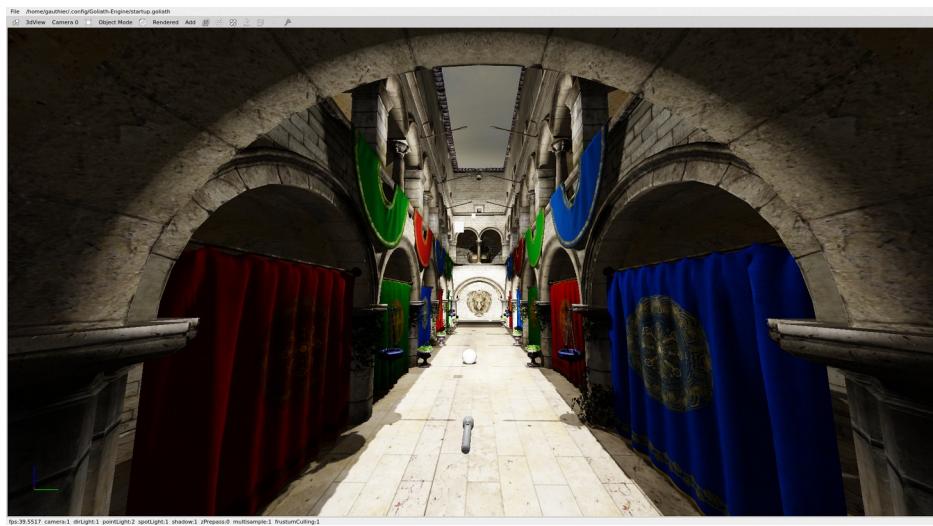
6. Rendu HDR

Référence : <https://learnopengl.com/Advanced-Lighting/HDR> <https://learnopengl.com/Advanced-Lighting/Bloom>

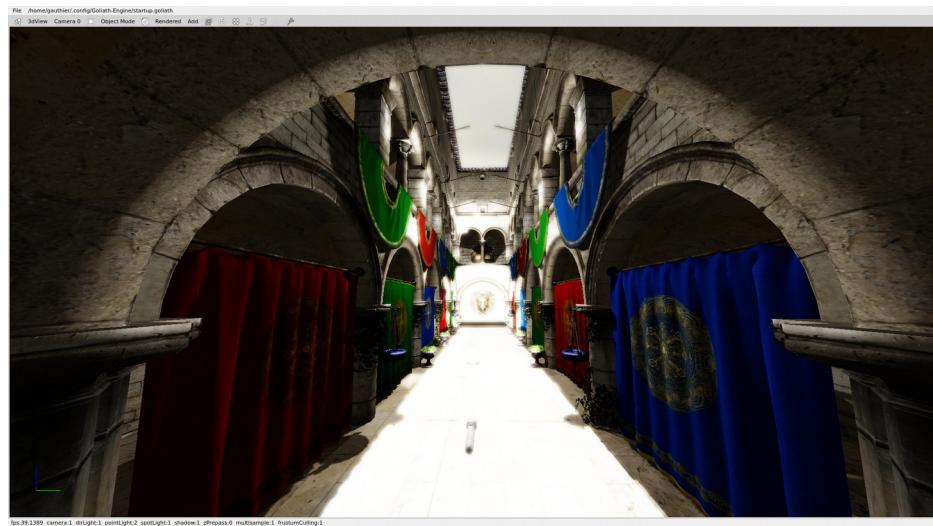
Rendu normal : touche h pour activer/désactiver le mode HDR



Rendu avec Tone mapping :



Rendu avec Bloom : touche F10 pour activer/désactiver le bloom



Partie Géometrie

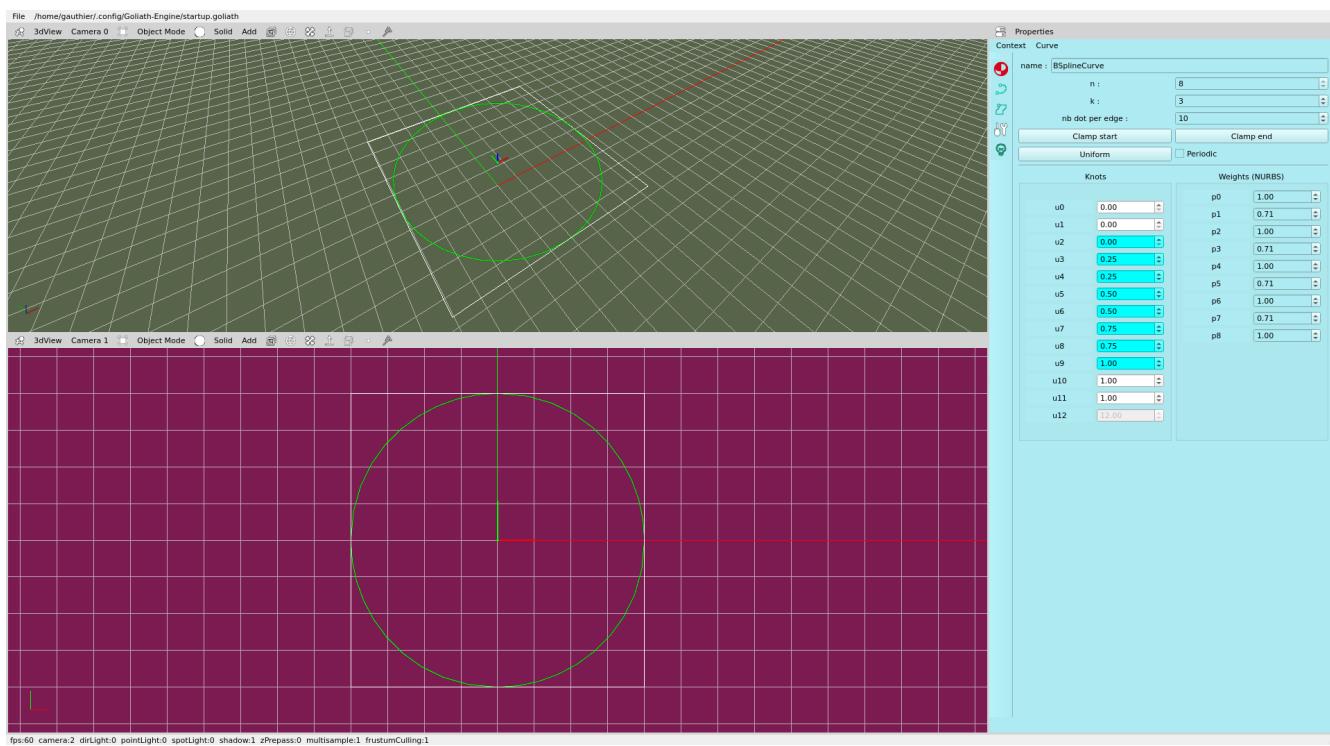
1. Nurbs (édition interactive)

Dessin d'un cercle parfait d'ordre 3 (continuité C1) avec une BSpline avec poids sur chaque point de contrôle.

Il faut sélectionner l'objet, puis avec tab changer de mode (Object → Edit) pour pouvoir bouger les différents points de contrôle.

Lors de la sélection de l'objet, l'édition du vecteur nodal et des poids sont directement activés.

L'ajout des courbes se fait via l'onglet Add (barre de menu de chaque vue) → Curve → Circle8 par exemple pour ce cercle.



Vidéo de démo Bspline : <https://youtu.be/Ms513wlBTy4>

2. PN Triangles

Référence : <http://ogldev.atspace.co.uk/www/tutorial31/tutorial31.html>

Source code : *resources/shader/shading/pnTriangle.**

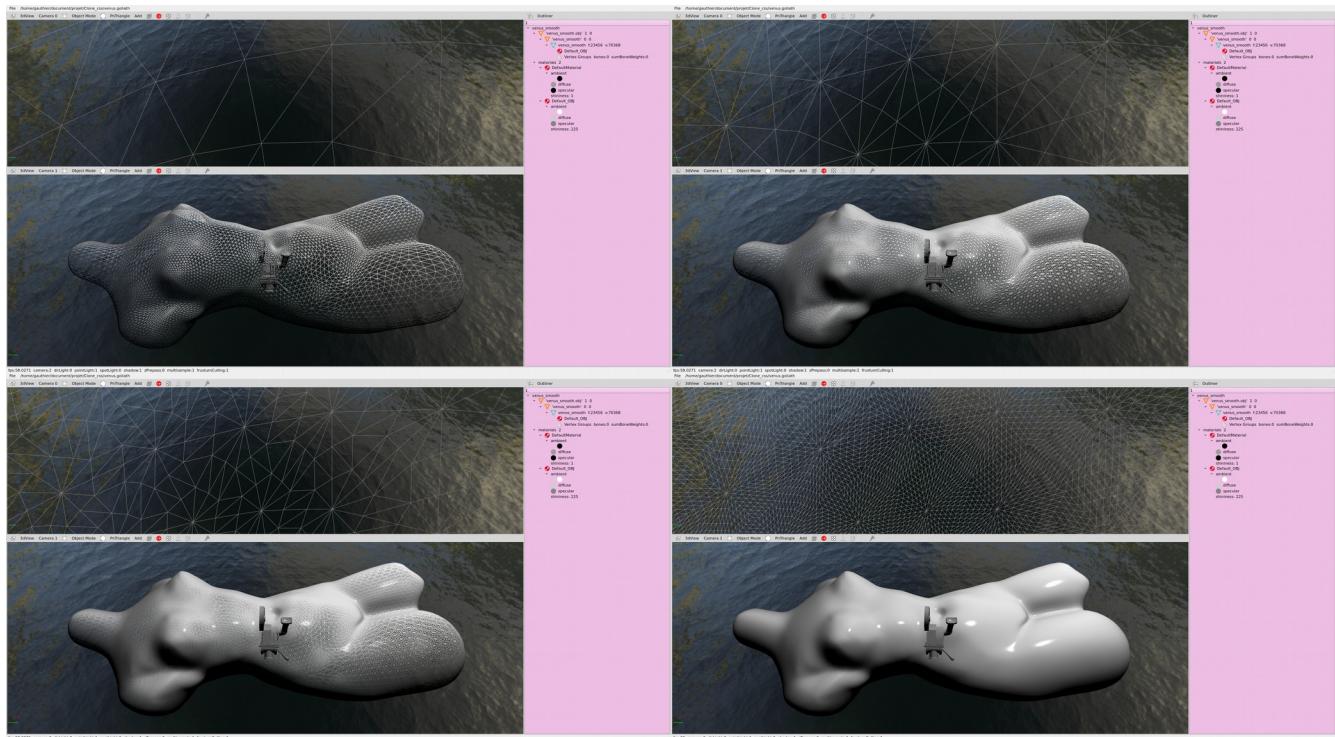


Figure 7: dans l'ordre de lecture de gauche à droite puis de haut en bas les différents niveaux de tessellation sont respectivement de 1, 2, 3, 10.

Visualisation en mode fil de fer

Vidéo de démo: <https://youtu.be/Ck42FhEDYWU>

En mode PnTriangle, touche u pour augmenter le niveau de tessellation, shift+u pour diminuer

Bonus

1. Camera

Gestion de 2 types de camera, camera fps et world (déplacement autour d'une cible à la manière de Blender ou logiciel de CAO). Touche f pour switcher entre elles.

Gestion des deux projections (perspective, orthographique).

Touche 5 pour switcher entre elles.

Source : src/engine/scene/camera/*

2. Texture

a) Diffuse (touche F1 pour activer/désactiver)



Figure 8: gauche: sans albedo, droite: avec albedo (texture diffuse)

b) Specular (touche F2 pour activer/désactiver)

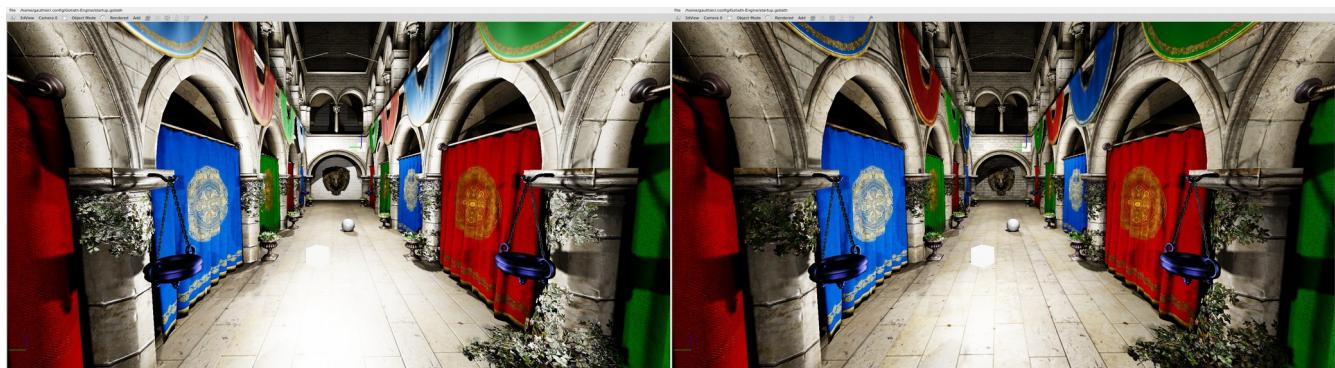


Figure 9: gauche: sans texture spéculaire, droite: avec texture spéculaire, remarquez une différence majeure sur les reflets des tapis circulaires en haut sur les côtés ou le mur du fond et le sol.

c) Normal (touche F3 pour activer/désactiver)

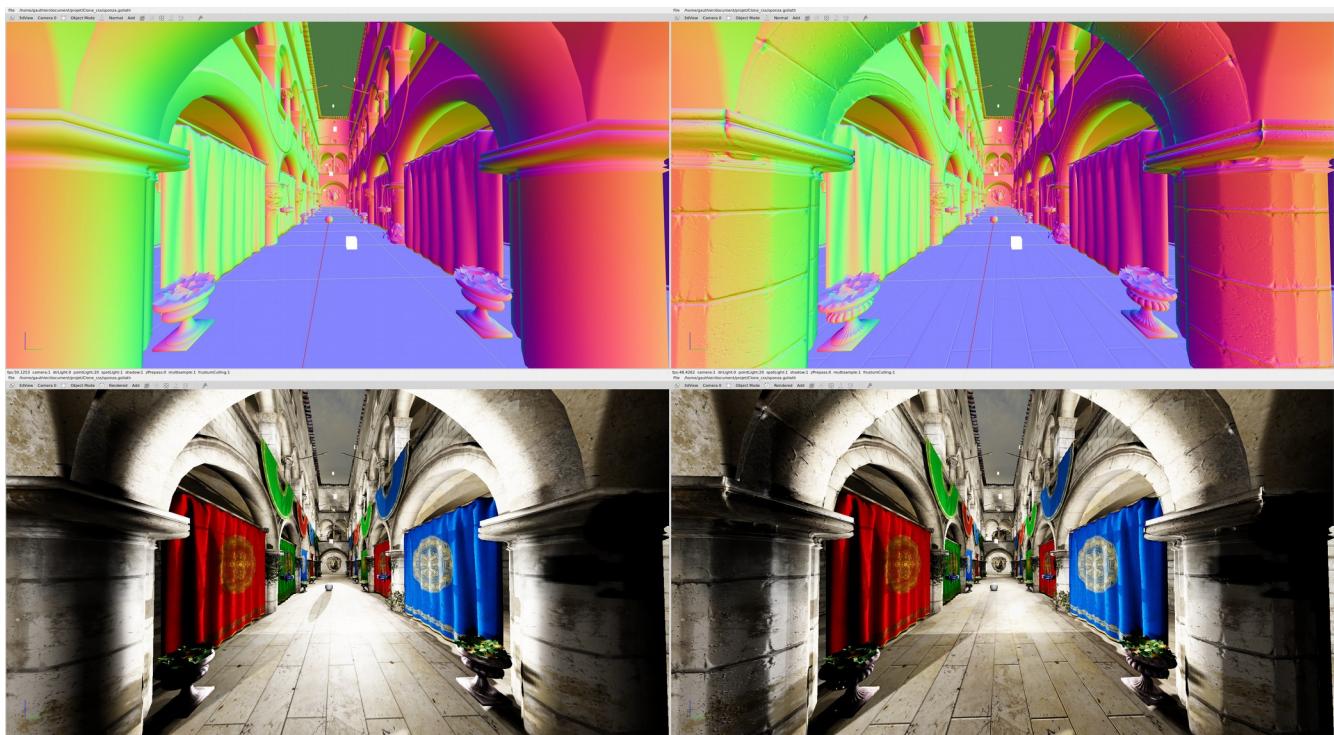


Figure 10: images du haut: mode *Normal*, images du bas: mode *Rendered*
images de gauche : sans normal map, images de droite : avec normal map
remarquez un meilleur détail des colonnes et reflet spéculaire sur le carrelage qui donne plus de réalisme/relief à la scène.

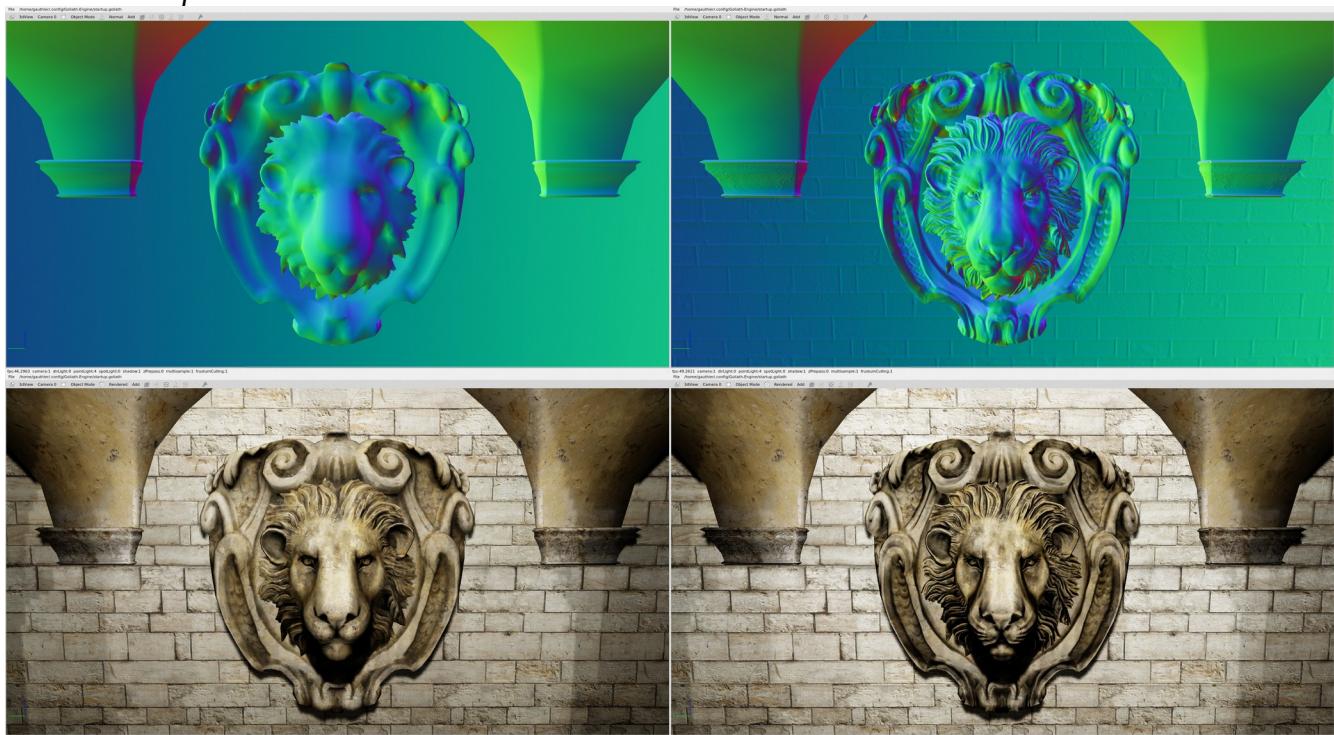


Figure 11: agencement idem que la Figure 10

d) **Height (Parallax Occlusion Mapping)** (touche F4 pour activer/désactiver) (touche p ou shift+p pour changer le niveau)

Référence : <https://learnopengl.com/Advanced-Lighting/Parallax-Mapping>

Source code : *resources/shader/shadow/rendered.fsh*



Figure 12: gauche: sans displacement mapping, droite: avec displacement mapping

En récupérant les textures de déplacement de Sponza source Crytek, le parallax Mapping donne de mauvais résultats, peut être que ces cartes de déplacement sont essentiellement utilisées lors de la tessellation de la géométrie de la scène et non pour faire du bump mapping.

e) **Opacity (touche F5 pour activer/désactiver)**

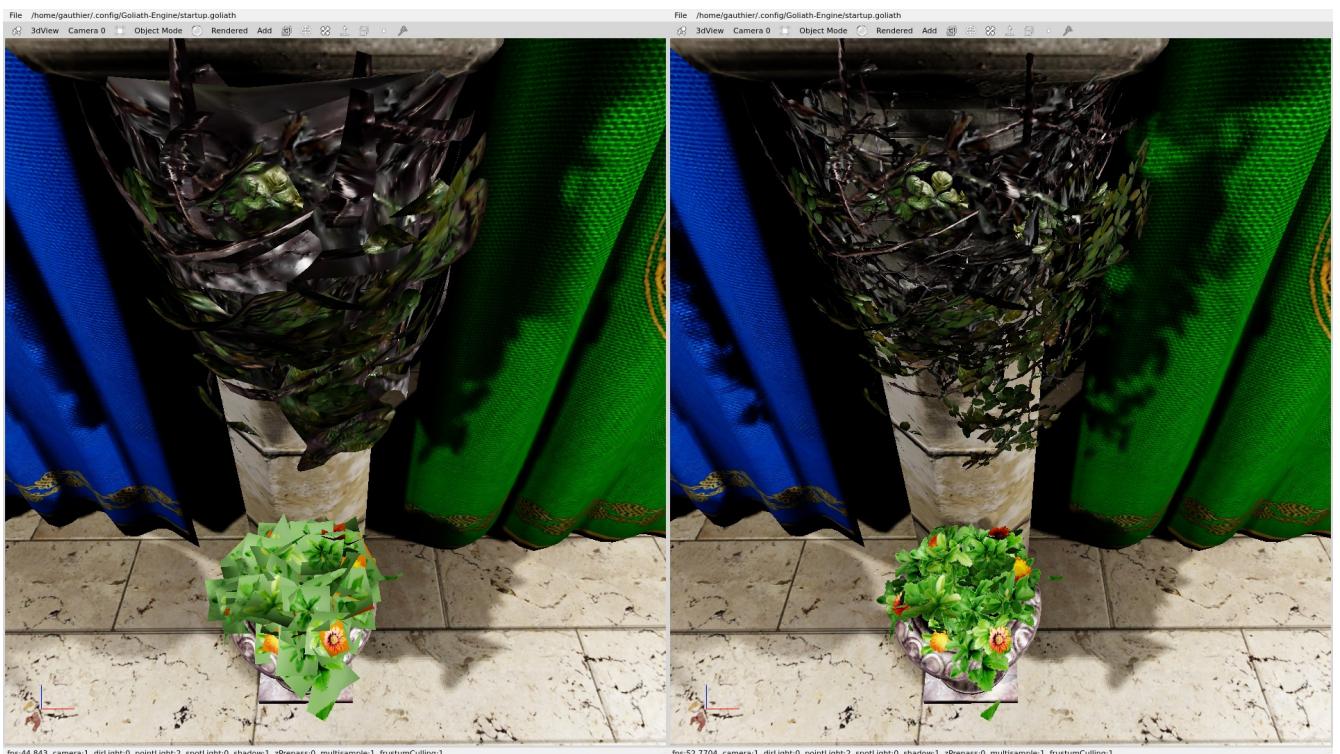


Figure 13: gauche: sans texture d'opacité, droite: avec texture d'opacité
remarquez la meilleure définition de l'ombre portée sur le rideau

3. Shader program et vues multiples

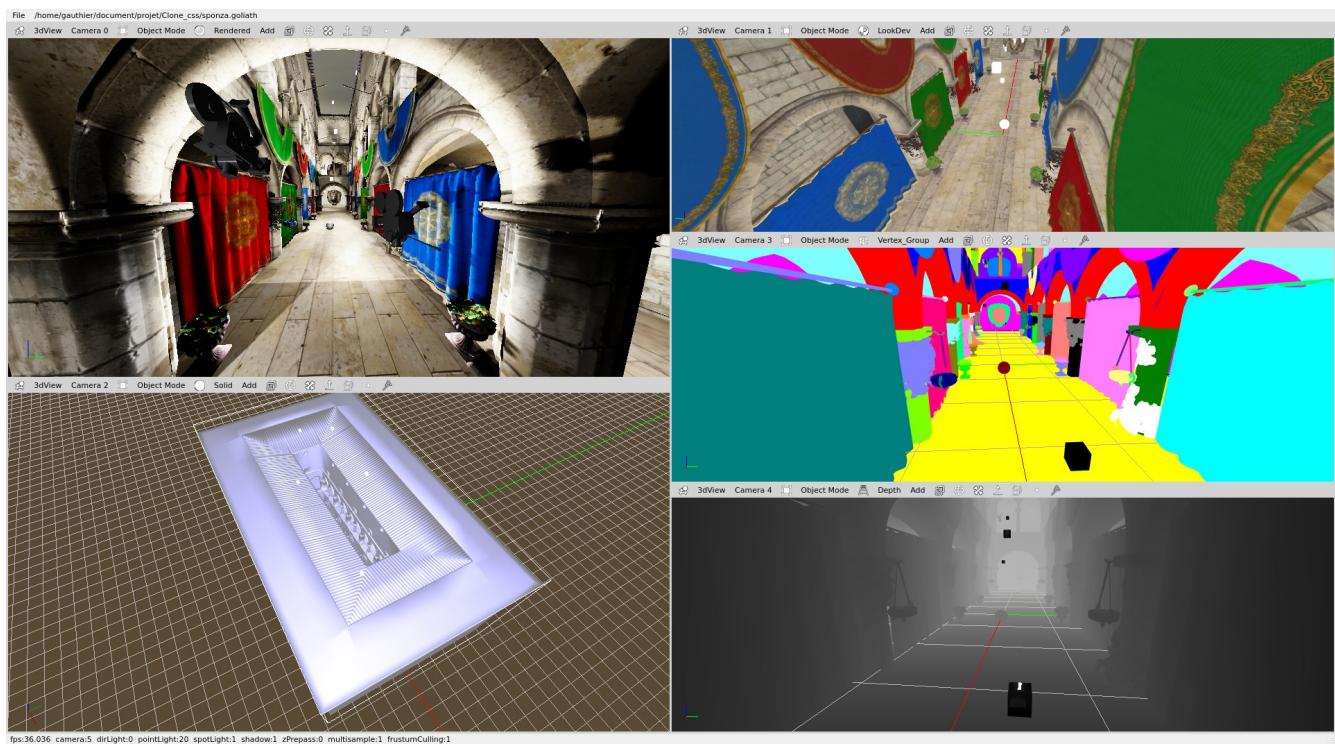


Figure 14: respectivement de haut en bas et de gauche à droite les modes de rendu (shader program) : Rendered, Solid, LookDev, visualisation des ensembles de mesh par couleur, Depth

4. Lumière

a) Édition

Gestion/Édition des différents types de lumière (directionnelle, point light, spot).

Création d'un éditeur de lumière pouvant changer les caractéristiques des différentes lumières

Vidéo de démo : <https://youtu.be/gDdghUDYpok>

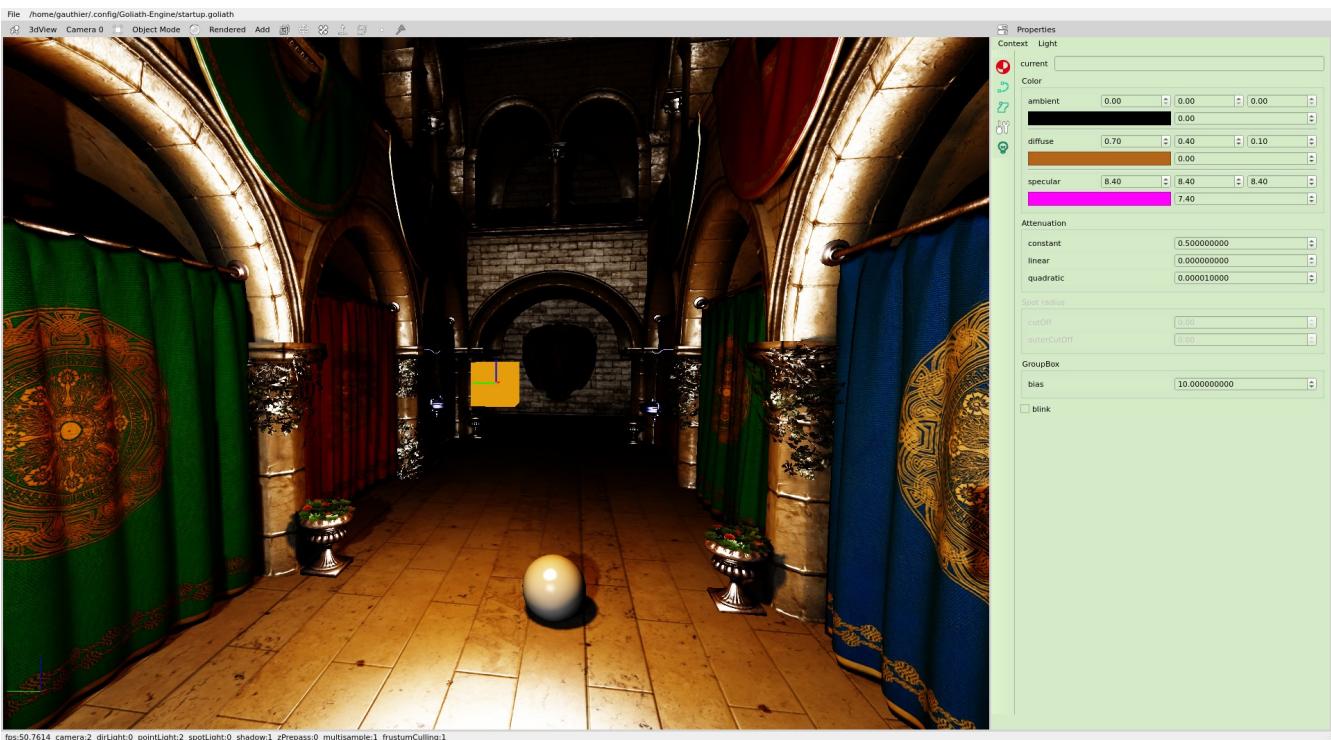


Figure 15: gauche: Sponza en mode Rendered, droite: éditeur de la lumière dernière sélectionnée

b) Soleil

Visualisation de la réelle position du soleil selon sa direction et changeant d'intensité selon l'orientation (lever/coucher de soleil).

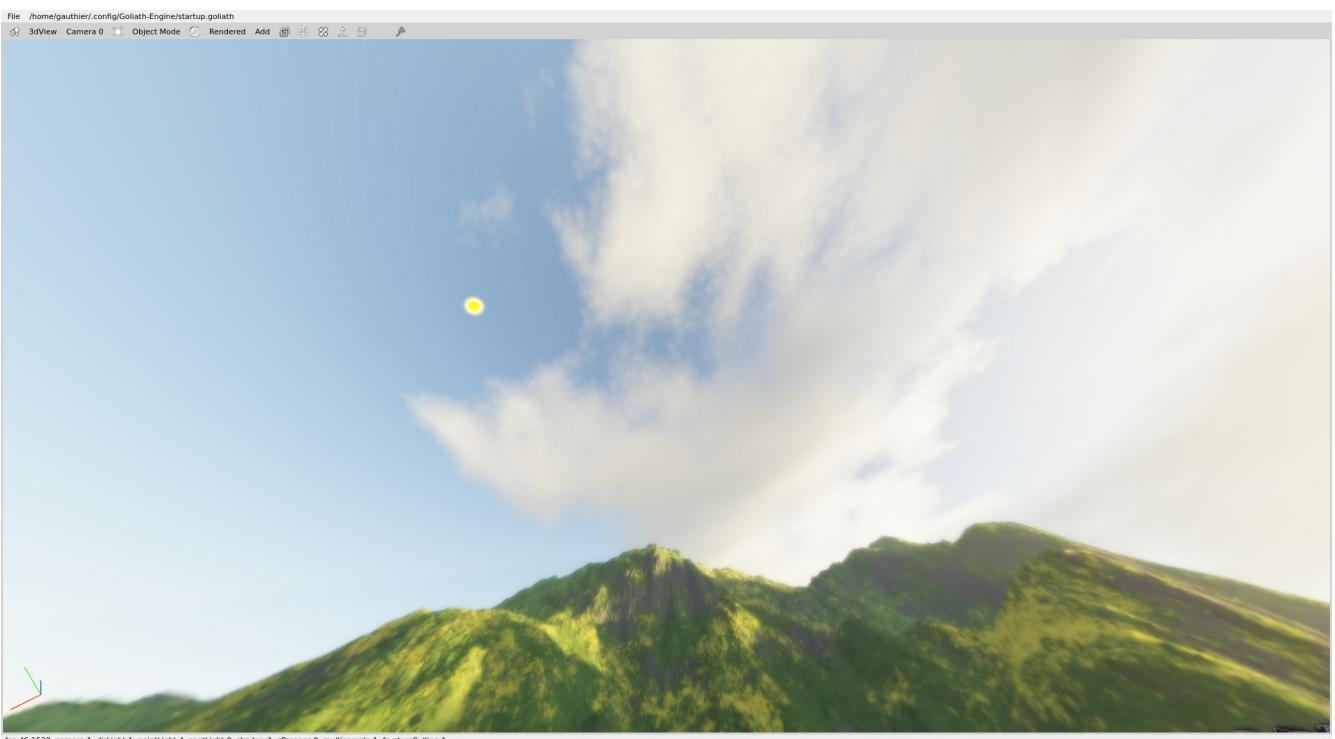


Figure 16: position du soleil calculée directement avec sa direction due à la propriété de source infinie

5. Matériaux

Référence : <https://learnopengl.com/Lighting/Basic-Lighting>

Gestion des matériaux sans texture.

Utilisation des couleurs ambiantes, diffuses, spéculaires ainsi que de la rugosité du modèle importé via Assimp, permettant l'utilisation d'une BRDF classique.

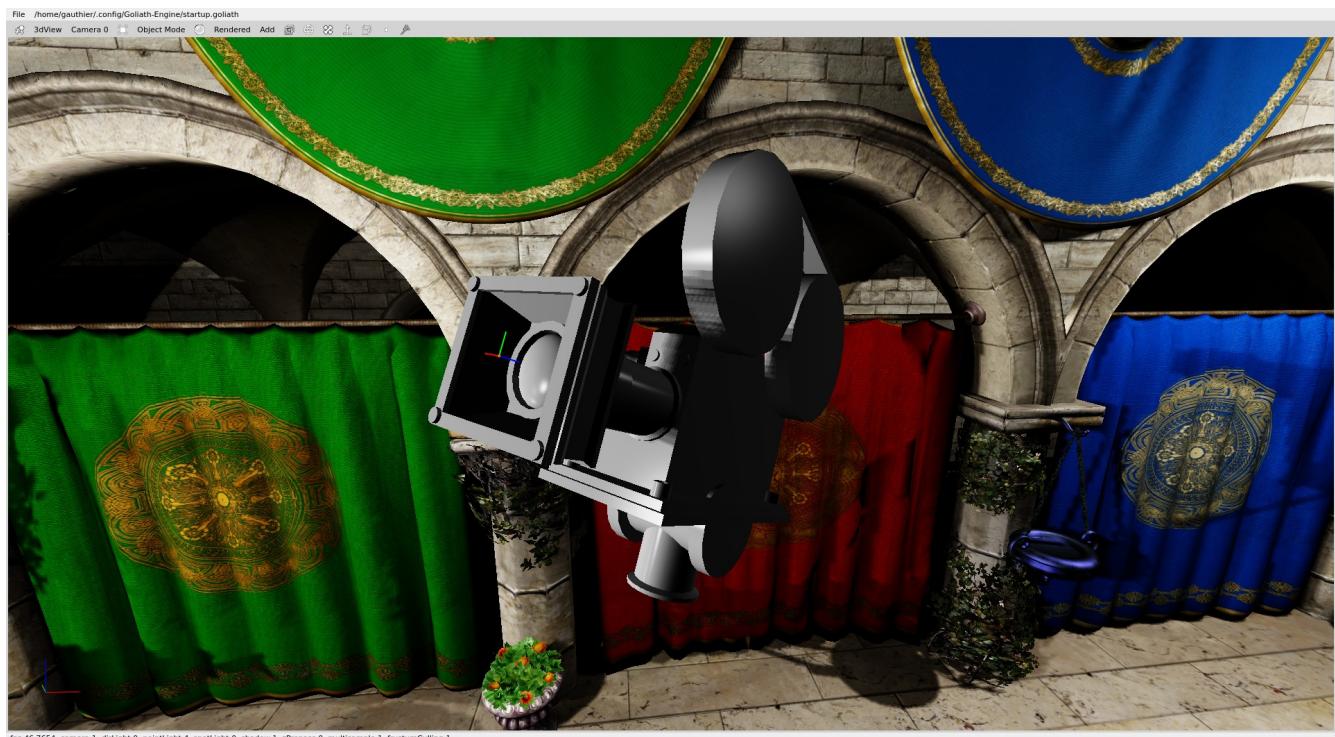


Figure 17: exemple d'une brdf sur un matériau sans texture

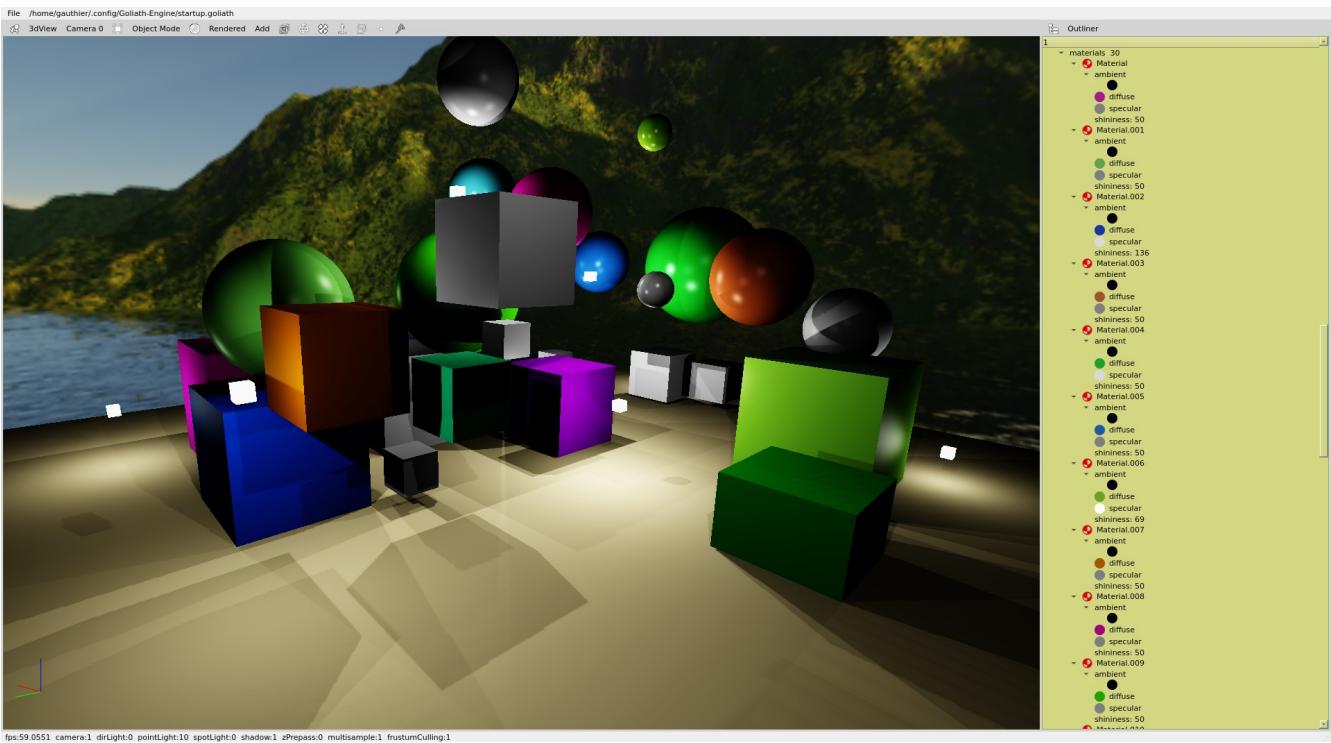


Figure 18: gauche: mode Rendered de la scène cubes_shperes, droite: liste des matériaux utilisés avec leurs composantes couleurs ambiante, diffuse et spéculaire



Figure 19: Remarquez la différence de réflexion spéculaire entre le cuir à gauche et le matériau mat à droite

Optimisation

1. World Space Frustum culling

Source code : src/opengl/Frustum.* src/opengl/BoundingBox.*

Vidéo démo : <https://youtu.be/xsooSpulDy8>

2. Real time point light map calculation

Calcul d'ombrages de plusieurs (20) éclairages ponctuels en temps réel.

Sélection des mises à jour des cartes d'ombrages selon la distance à la caméra.

Vidéo démo : <https://youtu.be/IASGQYvVZYA>

Article intéressant (évolution possible) : Efficient Virtual Shadow Maps for Many Lights