

Chef d'Oeuvre M2 IGAI

Encadrant : M. PAULIN

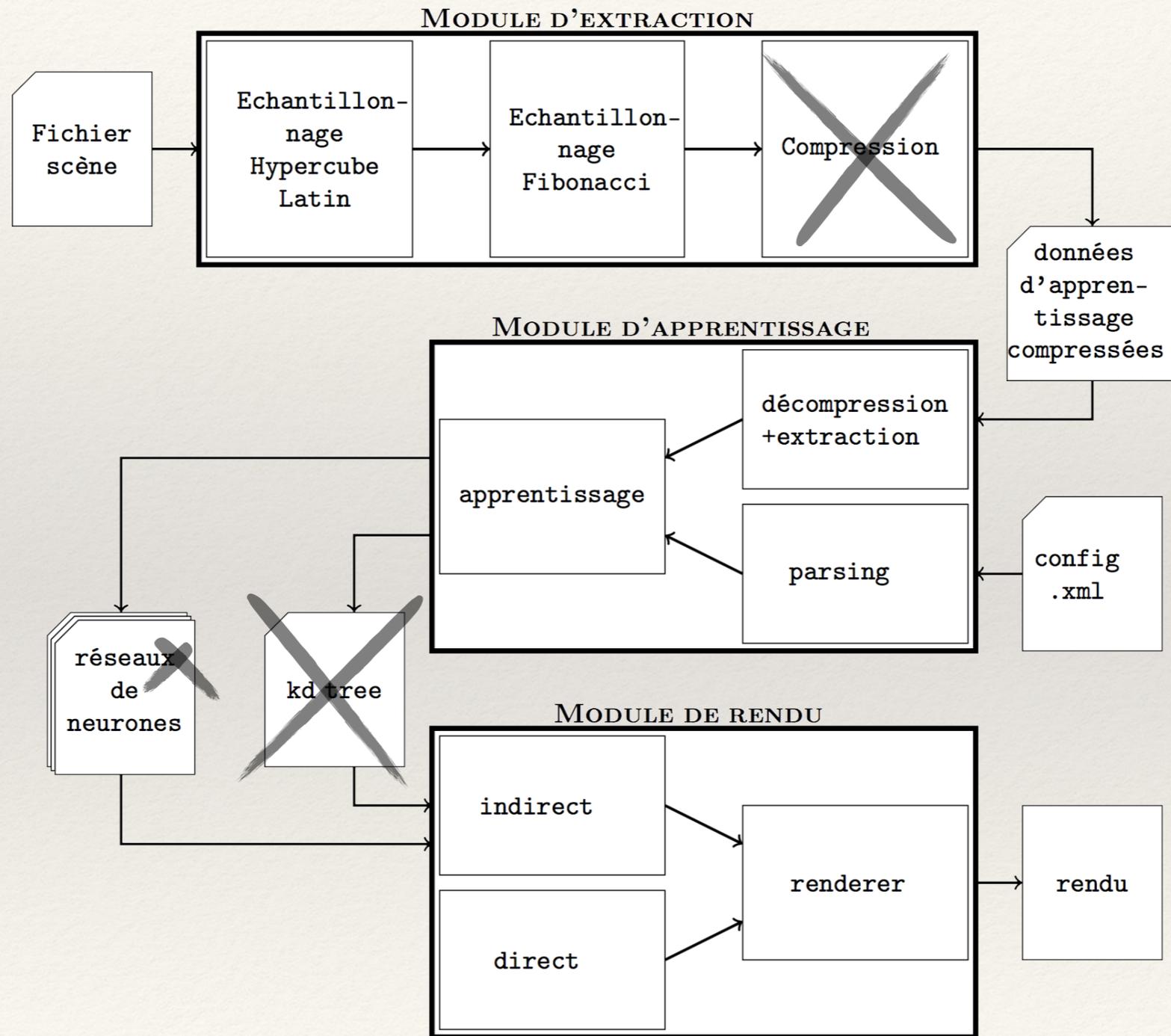
Global Illumination with Radiance Regression Functions

Yannick BERNARD
Pierre GUERINEAU
Kevin MENIEL
Romain MOUTRILLE
Matthias ROVES

Introduction



Structure



Présentation du logiciel

- ❖ **Module d'extraction**
 - ❖ Entrée : N positions de caméra, M positions de lumière ponctuelle.
 - ❖ Echantillonnage par hypercube latin :
 - Génération de $N * M$ fichiers scènes PBRT

Présentation du logiciel

❖ Module d'extraction

❖ Hypercube latin

Integrator "CO_path" "integer maxdepth" [65]

Transform [1 -0 -0 -0 -0 1 -0 -0 -0 -0 -1 -0 **-0 -1 6.8** 1]

Sampler "sobol" "integer pixelsamples" [64]

PixelFilter "triangle" "float xwidth" [1.000000] "float ywidth" [1.000000]

Film "image" "integer xresolution" [1000] "integer yresolution" [1] "string filename" ["cornell-box.png"]

Camera "fibonacci" "float fov" [19.500000]

WorldBegin

MakeNamedMaterial "LeftWall" "string type" ["matte"] "rgb Kd" [0.630000 0.065000 0.050000]

[...]

NamedMaterial "Floor"

Shape "trianglemesh" "integer indices" [0 1 2 0 2 3] "point P" [-10 0 -10 -10 0 10 10 -0 10 10 -0 -10] "normal N" [0 1 0 0 1 0 0 1 0 0 1 0] "float uv" [0 0 1 0 1 1 0 1]

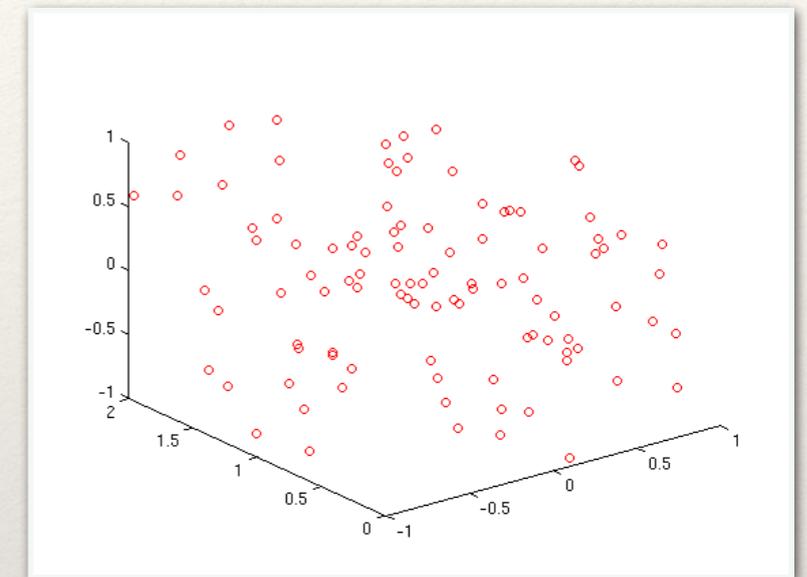
[...]

AttributeBegin

LightSource "point" "rgb I" [17 12 4] "point from" [**0 18 0**]

AttributeEnd

WorldEnd



Présentation du logiciel

- ❖ Module d'extraction
 - ❖ Modification de PBRT
 - ❖ Intégrateur CO_path -> calcul éclairage indirect
 - ❖ Caméra Fibonacci -> caméra sphérique
 - ❖ Écriture des paramètres de rendu dans un fichier .data

Présentation du logiciel

❖ Module d'extraction

❖ Fibonacci

Integrator "CO_path" "integer maxdepth" [65]

Transform [1 -0 -0 -0 -0 1 -0 -0 -0 -0 -1 -0 -0 -1 6.8 1]

Sampler "sobol" "integer pixelsamples" [64]

PixelFilter "triangle" "float xwidth" [1.000000] "float ywidth" [1.000000]

Film "image" "integer xresolution" [**1000**] "integer yresolution" [1] "string filename" ["cornell-box.png"]

Camera "fibonacci"

WorldBegin

MakeNamedMaterial "LeftWall" "string type" ["matte"] "rgb Kd" [0.630000 0.065000 0.050000]

[...]

NamedMaterial "Floor"

Shape "trianglemesh" "integer indices" [0 1 2 0 2 3] "point P" [-10 0 -10 -10 0 10 10 -0 10 10 -0 -10] "normal N" [0 1 0 0 1 0 0 1 0 0 1 0] "float uv" [0 0 1 0 1 1 0 1]

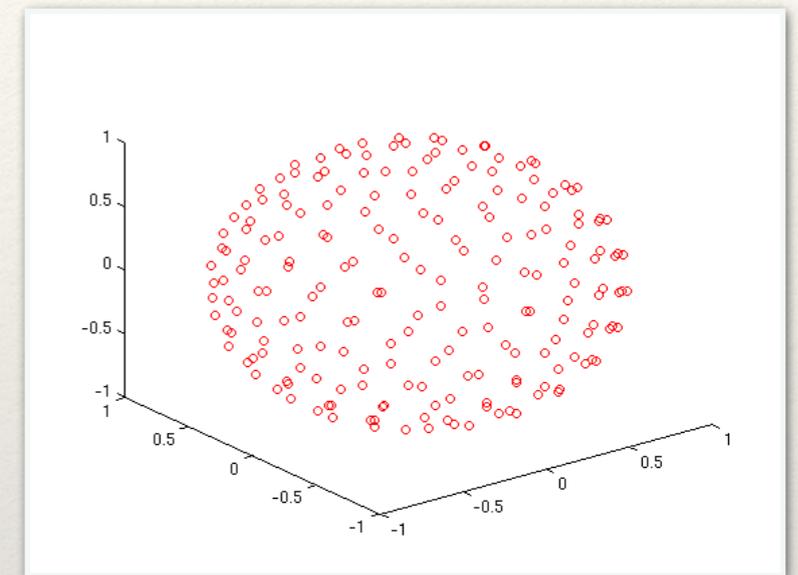
[...]

AttributeBegin

LightSource "point" "rgb I" [17 12 4] "point from" [0 18 0]

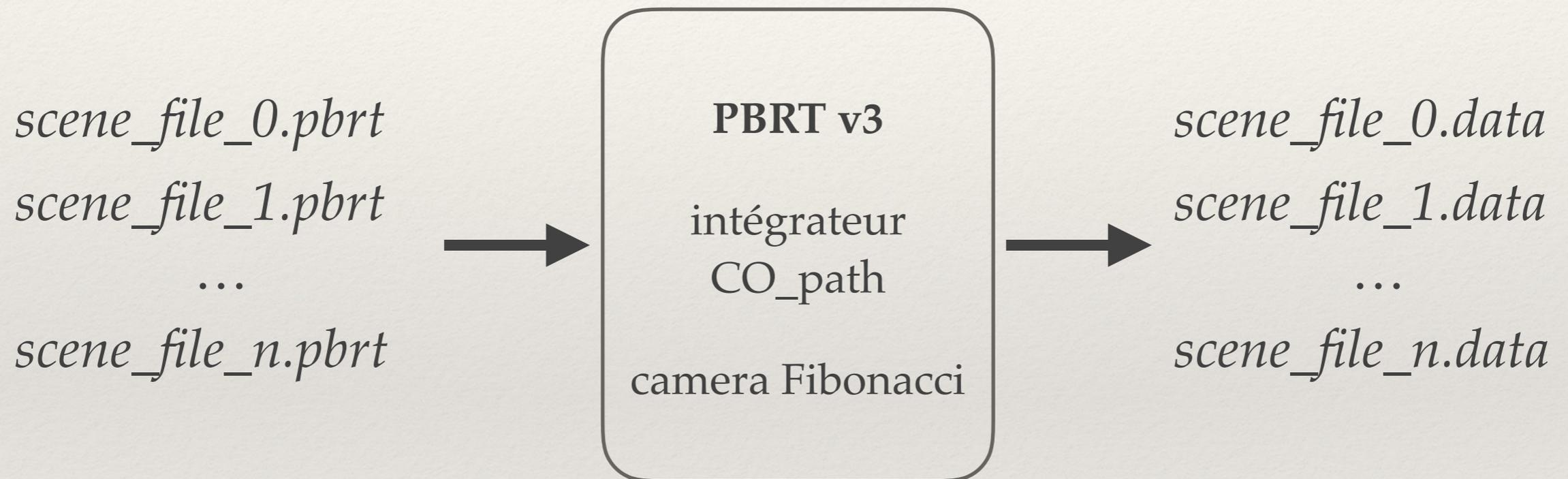
AttributeEnd

WorldEnd



Présentation du logiciel

❖ Module d'extraction

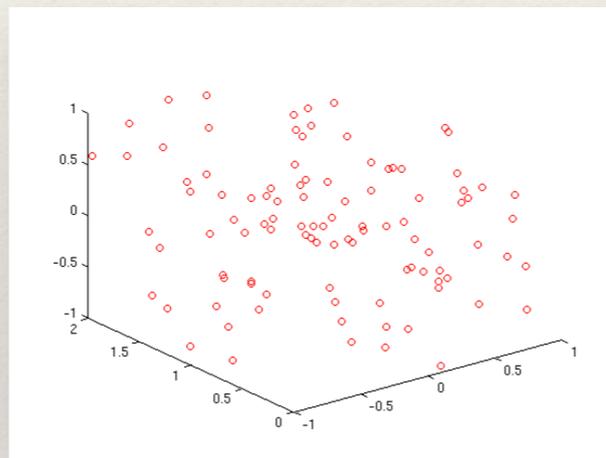


Paramètres : - Taille d'image
- Nombre d'échantillons
pour Fibonacci

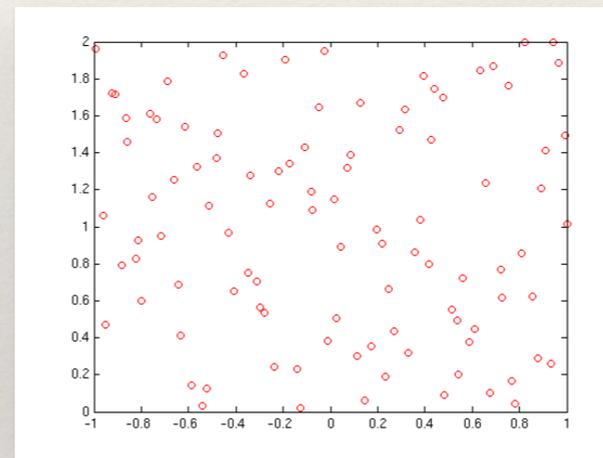
Tests

❖ Extraction

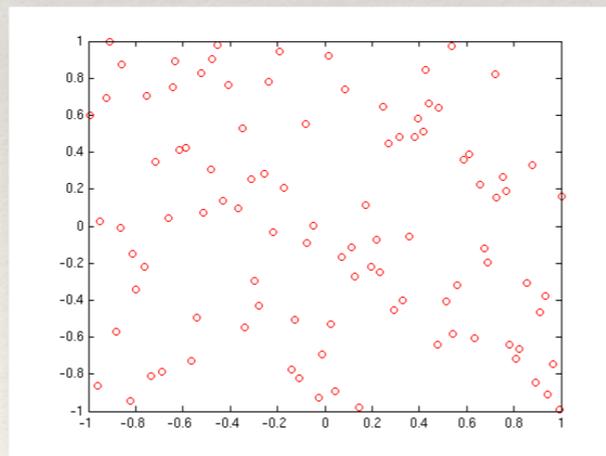
❖ Hypercube latin



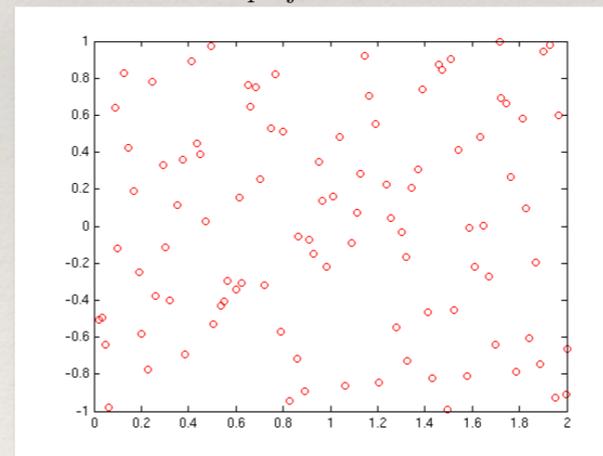
3D



projection x



projection y

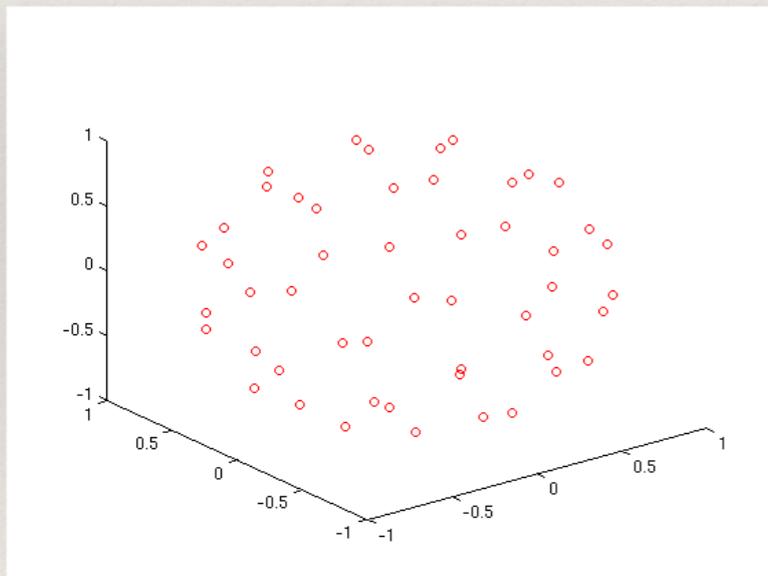


projection z

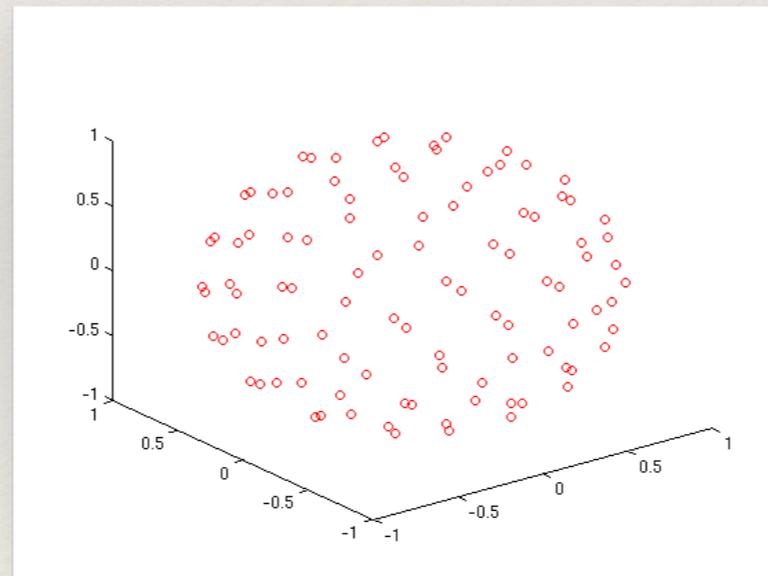
Tests

❖ Extraction

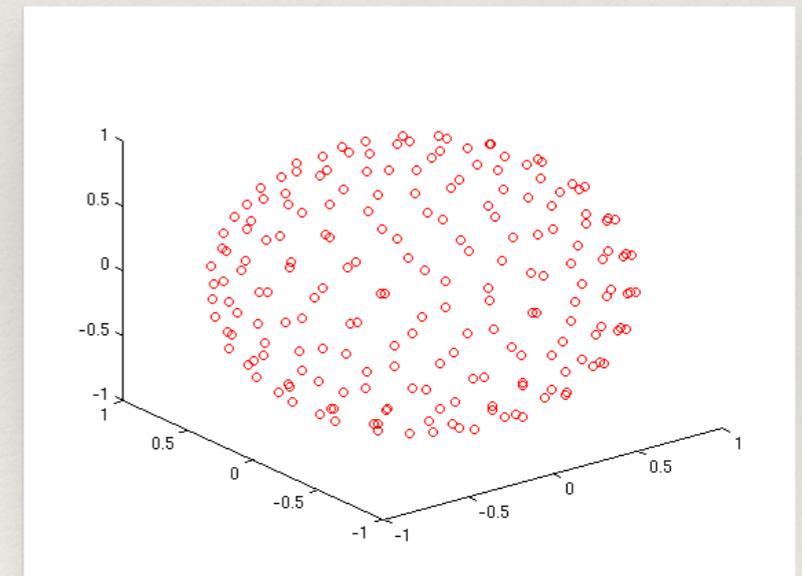
❖ Fibonacci



50 échantillons



100 échantillons

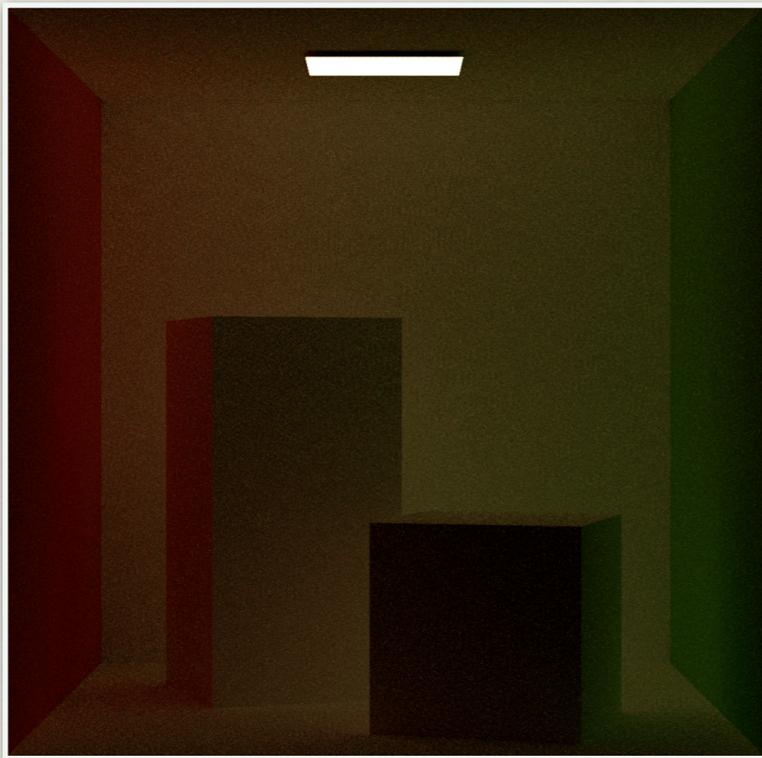


200 échantillons

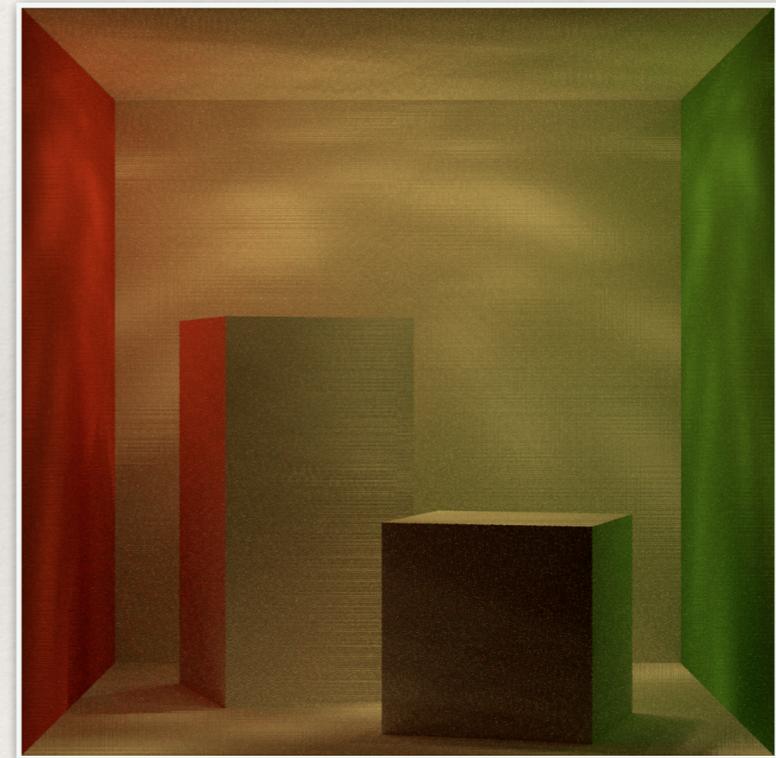
Tests

- ❖ Extraction

- ❖ Intégrateur CO_path



Éclairage indirect de
l'intégrateur path



Éclairage indirect calculé
par notre intégrateur

Description des fonctionnalités

- ❖ Module d'apprentissage
 - ❖ Fichier de configuration des réseaux : config.xml
 - ❖ Nombre de perceptrons par couche
 - ❖ Fonction d'activation pour chaque couche
 - ❖ Méthode des scaling et unscaling layers
 - ❖ Infos pour le splitting des données
 - ❖ Infos pour l'apprentissage (algorithme, erreur à atteindre)

Description des fonctionnalités

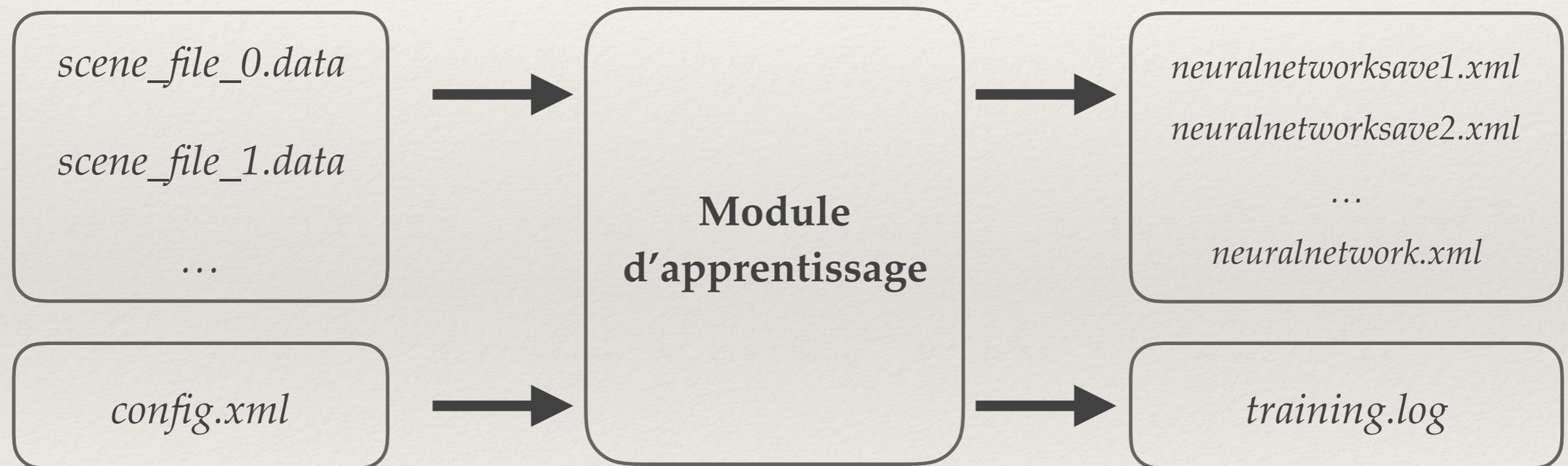
- ❖ Module d'apprentissage
 - ❖ Bouclage sur les données
 - ❖ Boucle sur les fichiers .data pour apprentissage
 - ❖ Ecriture de l'avancement dans un fichier log pour reprise

Description des fonctionnalités

- ❖ Module d'apprentissage
 - ❖ Modifications d'OpenNN
 - ❖ Compatibilité salle 112
 - ❖ Création DataSet à partir d'une matrice

Description des fonctionnalités

❖ Module d'apprentissage



Tests

- ❖ Apprentissage

- ❖ Fichier de configuration

- ❖ Création et parse du fichier avec toutes les valeurs possibles et vérification de la cohérence
 - ❖ Vérification de la prise en compte des erreurs lorsque l'on met des valeurs erronées

Tests

- ❖ Apprentissage

- ❖ OpenNN

- ❖ Bibliothèque très utilisée. Nous la considérons vérifiée et fonctionnant correctement.

Description des fonctionnalités

- ❖ Module de rendu
 - ❖ Modifications du moteur
 - ❖ Chargement de scènes PBRT
 - ❖ Ajout du support des shadowmaps
 - ❖ Mise en place de process et connexions pour le rendu

Description des fonctionnalités

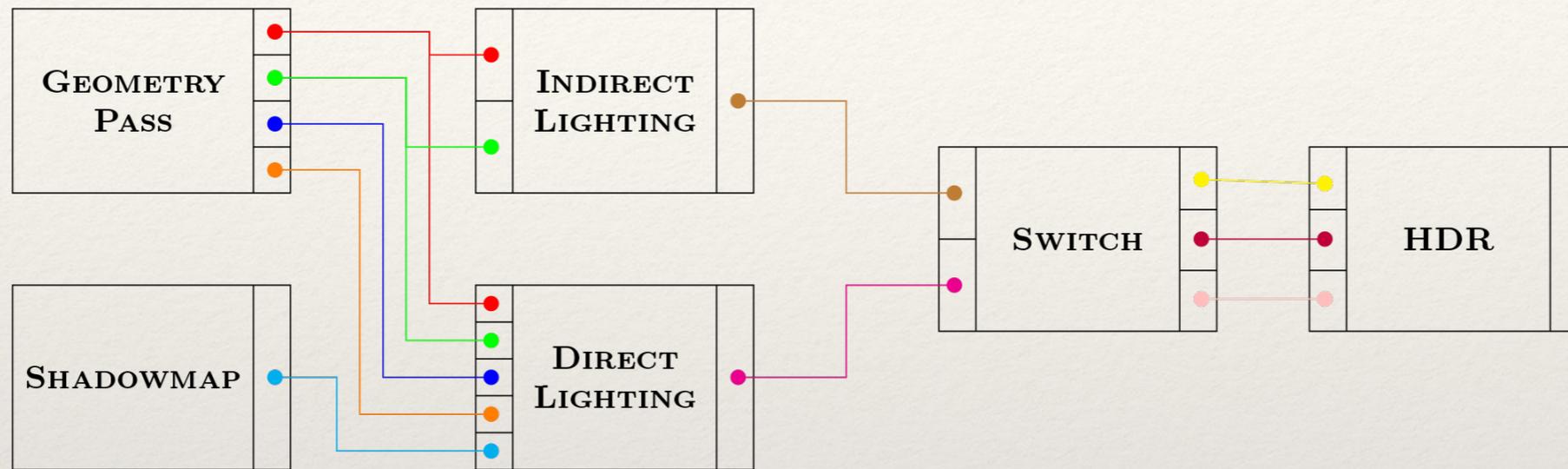


FIGURE 1 – Schéma du pipeline des *RenderProcesses* de notre moteur

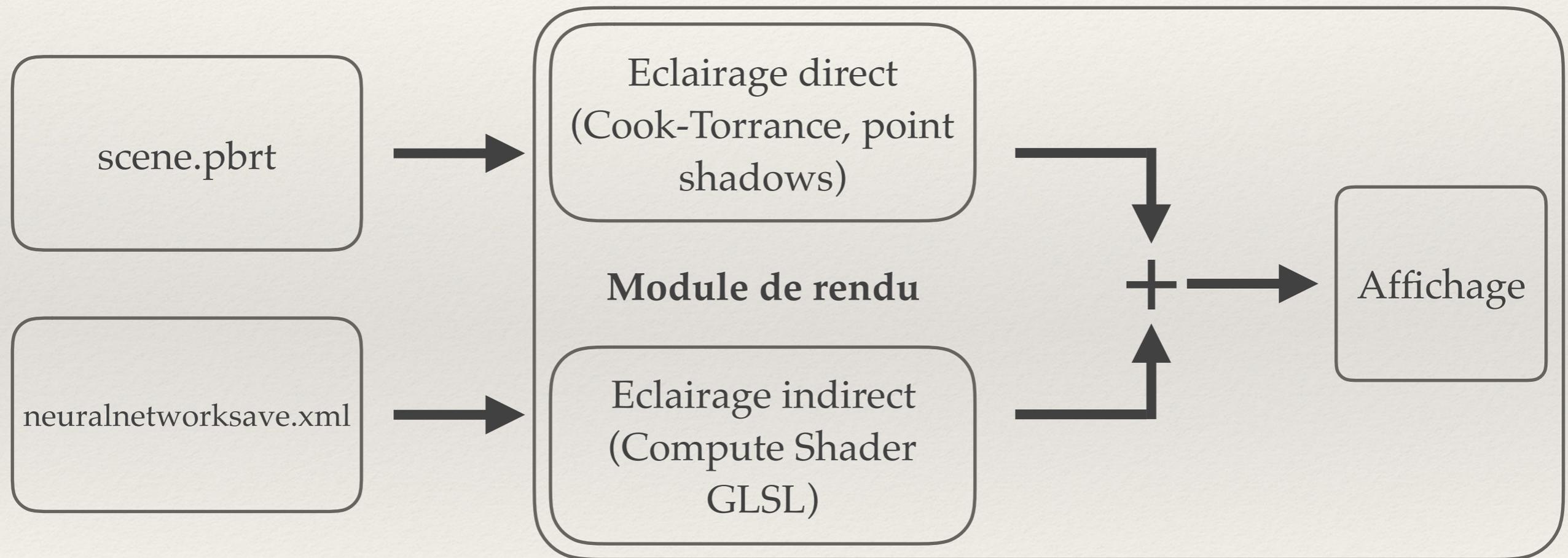
- RGBA - position + profondeur
- RGB - normal
- RGBA - albedo + spéculaire
- RGB - matériau (*roughness + refraction + metalness*)
- DEPTH CUBEMAP - carte des profondeurs
- RGBA - éclairage indirect
- RGB - éclairage direct
- RGB - mix des éclairages direct et indirect
- RGB - résultat seuillé par la luminosité
- RED - luminosité
- **Sortie** : RGB - résultat avec HDR appliqué

Description des fonctionnalités

- ❖ Module de rendu
 - ❖ Eclairage indirect
 - ❖ Evaluation du réseau de neurones (compute shaders GLSL) :
 - ❖ Entrées
 - ❖ frag_position, camera_position, point_light_position, frag_normal, neural_network
 - ❖ Sortie
 - ❖ indirect_lighting

Description des fonctionnalités

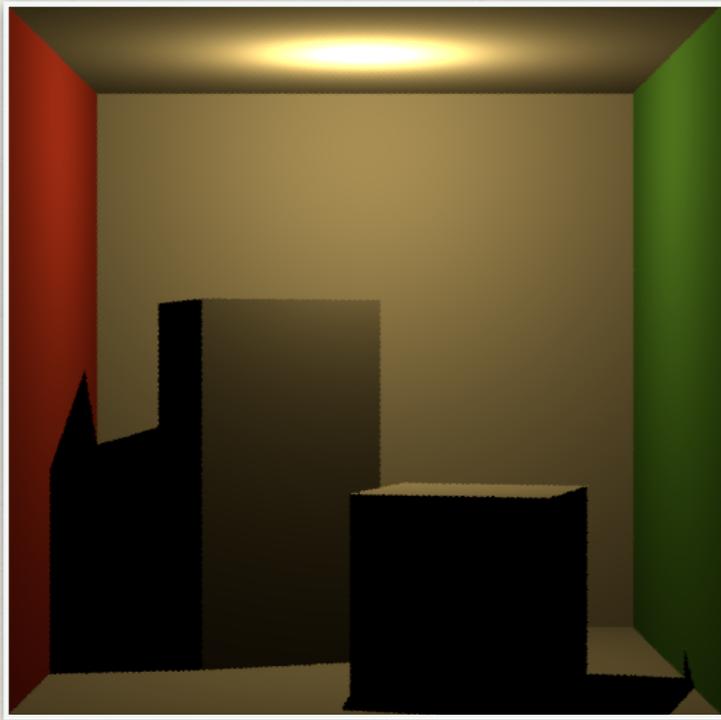
❖ Module de rendu



Tests

- ❖ Rendu

- ❖ Chargement de fichier PBRT



Scène de référence : PBRT v3



Scène chargée dans notre moteur 3D

Tests

- ❖ Rendu

- ❖ Evaluation du réseau de neurones

- ❖ Envoi d'inputs au réseau dans openNN, envoi des mêmes inputs à l'évaluation dans le moteur et comparaison

Matrice de couverture des exigences

EXIGENCES	COUVERTURE	PRIORITÉ
Extraction		
La fonction de radiance va être apprise par morceaux, les données seront générées pour des sous-régions de la scène.	●	●
Une version modifiée de PBRT calculant uniquement la composante indirecte devra être réalisée.	●	●
Des formats de stockage interne et externe devront être définis.	●	●
Apprentissage		
Le réseau de neurone doit être créé grâce à la bibliothèque OpenNN.	●	●
Le réseau de neurone devra calculer l'éclairage indirect d'un point à partir de la position de ce point, de sa normale, de la direction de la lumière et de la direction de la caméra.	●	●
Le réseau de neurone devra calculer l'éclairage indirect d'un point à partir des paramètres de la BRDF.	●	●
Le réseau de neurone doit pouvoir calculer l'éclairage indirect d'un point avec une erreur quadratique inférieure à un seuil défini par l'utilisateur.	●	●
Rendu		
Le rendu de l'éclairage indirect doit se faire en temps réel grâce l'évaluation des données par un réseau de neurones.	●	●
Le rendu de l'éclairage indirect de la scène doit être similaire à celui d'un moteur de rendu hors ligne (PBRT, Mitsuba).	●	●
Le rendu de l'éclairage direct doit être fait en temps réel.	●	●
Le rendu de l'éclairage direct doit utiliser le même type de BRDF que l'éclairage indirect.	●	●
Le rendu de l'éclairage direct doit utiliser le même type de source que l'éclairage indirect.	●	●

Améliorations possibles

❖ Extraction

- ❖ Amélioration de la caméra de Fibonacci (rayons différentiels)
- ❖ Trier les données en amont de l'apprentissage pour gagner du temps.
- ❖ Implémentation du module de compression/décompression pour des données de grandes tailles

Améliorations possibles

- ❖ **Apprentissage**

- ❖ Implémentation des KD-Trees et utilisation de plusieurs réseaux de neurones.

Améliorations possibles

❖ Rendu

- ❖ Amélioration du chargement de scène PBRT (support de maillages quad, différents types d'éclairage, roughness, metalness...)
- ❖ Découper l'évaluation en plusieurs compute shaders (contrainte de mémoire)

Conclusion

- ❖ Sujet complexe mais motivant

Démo

Questions ?