M2 IGI - Université Lyon 1

Rapport de stage

Rendu réalise et temps réel pour la réalité augmentée

Auteur :

Hadrien Croubois
M2 IGI – UCBL - ENS de Lyon

Encadrants :

Jean-Philippe Farrugia
Maître de conférences, LIRIS / IUTa, université Lyon 1

Jean-Claude Iehl

Remerciements

Table des matières

Périphérique d'acquisition

Partie 1.1

Modèle

Partie 1.2

Calibration

Partie 1.3

Détails d'implémentation

Repérage dans l'espace

Partie 2.1

Hiérarchie de référentiels

Afin de mettre en place les différents mécanismes de repérage, il à été nécessaire de hiérarchiser les repères relatifs aux différents référentiels considères.

Le référentiel principal est le référentiel du monde. Il est par définition fixe au cours du temps et est définit par rapport à la mire utilisé pour l'acquisition. Ce repère est centré au centre géométrique de la mire et servira aussi bien à positionner l'objet à afficher qu'à définir servir de référentiel de base pour l'EnvMap.

Les différentes faces de la mire sont elle même fixes dans le repère du monde car lié a l'objet physique (la mire) qui le définit. Les transformation entre le repère de chacune des faces de la mire et le repère du monde sont codées dans les marqueur présents sur les faces (matrice *model*).

Ainsi, ayant identifier une des faces à l'aide du marqueur présent sur cette dernière, il est possible, en appliquant la transformation codé par ce marqueur, de reconstruire le repère du monde.

L'interface d'acquisition est elle aussi repéré vis a vis du repère principal. La transformation entre ces deux repère est codé par la matrice *view* et changera au cours du temps, l'utilisateur se déplaçant par rapport à la mire.

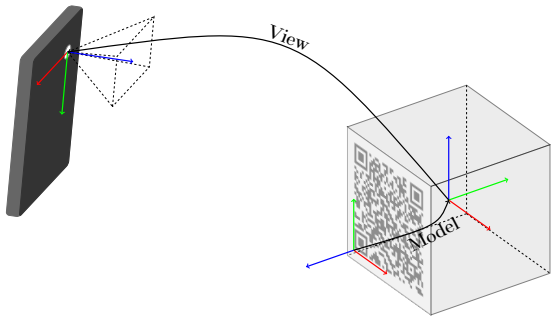


FIGURE 2.1 – Les différents référentiels

Les différentes cameras présentes sur l'interface d'acquisition seront à leur tour positionnées relativement à l'interface d'acquisition. Cette dernière transformation est disponible dans les fichiers de configurations de camera chargés au démarrage.

TABLE 2.1 Hiérarchie des matrices de transformations

	QRcode	
	⇅	
	Face	
$model^{-1}$ ↱		↳ $model$
	Monde	
$view^{-1}$ ↱		↳ $view$
	Smartphone	
$orientation^{-1}$ ↱		↳ $orientation$
	Camera	
$cvToGl^{-1}$ ↱		↳ $cvToGl$
	Vue OpenGL	
$projection^{-1}$ ↱		↳ $projection$
	Image rendu	

Partie 2.2

Évaluation des transformations

Compte tenu de la hiérarchie de référentiel décrite dans la section précédente, il est nécessaire, pour décrire complètement le système, de décrire chacune des matrices de transformation.

2.2.1

Calibration de la camera

2.2.2

Identification des marqueurs

2.2.3

Models OpenCV / OpenGL

Les cartes d'environnement

Partie 3.1

Principe

Partie 3.2

Reconstruction dynamique

Partie 3.3

Détails d'implémentation

Rendu

Partie 4.1

État de l'art

Partie 4.2

Éclairage ambiant

4.2.1

De la physique aux mathématiques

Le calcul de l'énergie reçu en un point de l'objet revient à intégrer le flux lumineux sur l'ensemble des direction visibles.

$$\mathcal{E}(p,\vec{n}) = \frac{1}{\pi}\int_{\mathcal{H}^2(\vec{n})} \mathcal{L}(p,\vec{\omega}) \times \vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega} \qquad (4.2.1.1)$$

L'intégrale selon $\int_{\mathcal{H}(\vec{n})}\mathrm{d}\omega$ correspond à une intégrale suivant l'hémisphère visible et pondéré par l'angle solide en supposant qu'il n'y a pas d'occultation. Afin de modéliser les phénomènes d'auto-occultation on devrai utiliser la formule suivante

$$\mathcal{E}(p,\vec{n}) = \frac{1}{\pi}\int_{\mathcal{V}(p,\vec{n})} \mathcal{L}(p,\vec{\omega}) \times \vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega} \qquad (4.2.1.2)$$

Ou $\mathcal{V}(p,\vec{n})$ est la restriction de l'hémisphère suivant le vecteur $\vec{n}$ a l'espace effectivement visible depuis le point p (ce qui renvient à considérer $\mathcal{H}(\vec{n})$ privée des direction correspondants à de l'auto-occultation).

Ici on fait d'abord la supposition que l'éclairement selon une direction $\vec{\omega}$ est indépendant du point considéré. Cette approximation, est nécessaire pour utiliser, sans reconstruction 3D complexe, l'envmap reconstruite dynamiquement.

On obtient donc l'équation

$$\mathcal{E}(p,\vec{n}) = \frac{1}{\pi}\int_{\mathcal{V}(p,\vec{n})} \mathcal{L}_{glob}(\vec{\omega}) \times \vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega} \qquad (4.2.1.3)$$

On fait alors la supposition que les phénomènes d'auto-occultation influencent de manière moyenne et globale l'énergie reçu en un point. Cela nous permet de revenir à une intégration sur tout le demi espace $\mathcal{H}^2(\vec{n})$ en ajoutant simplement un coefficient de $\mathcal{P}_{\mathcal{V}}(p)$ valant 1 en l'absence d'auto-occultation et 0 pour une auto-occultation totale.

Ce coefficient définit sur la surface de l'objet pourra être pré-calculé et fournit sous forme de texture (voir section 5.1, page 11).

$$\begin{aligned}\mathcal{P}_{\mathcal{V}}(p) &= \frac{\int_{\mathcal{V}(p,\vec{n}(p))}\vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega}}{\int_{\mathcal{H}(\vec{n}(p))}\vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega}}\\ &= \frac{1}{\pi}\int_{\mathcal{V}(p,\vec{n}(p))}\vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega}\end{aligned} \qquad (4.2.1.4)$$

d'ou

$$\mathcal{E}(p,\vec{n}) = \frac{\mathcal{P}_{\mathcal{V}}(p)}{\pi}\int_{\mathcal{H}(p,\vec{n})}\mathcal{L}_{glob}(\vec{\omega})\times\vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega} \qquad (4.2.1.5)$$

Enfin, on fera une dernière approximation suivant la méthode proposé dans [Mcguire et al., 2013] : on considère la fonction $\mathcal{L}_{glob}$ (décrite par l'envmap) comme constante sur chaque face de l'envmap. Le niveau de mipmap le moins détaillé nous donne en effet une valeur moyenne pour la face considérée.

Si l'on oublie un instants les problèmes d'auto-occultation et que l'on s'intéresse a l'angle solide décrit par une face du cube centrée au point de vue, un rapide calcul permet d'évaluer l'angle solide décrit par une face complètement visible comme étant

$$\Omega_F = \iint_F \frac{1}{\|\vec{\omega}\|^3}\,\mathrm{d}\omega = \frac{2\pi}{3} \qquad (4.2.1.6)$$

L'angle solide sur une face partiellement visible (selon une demi sphère caractérisée par l'hyperplan de vecteur $\vec{n}$)est alors défini par

$$\Omega_F(\vec{n}) = \iint_F \frac{(\vec{\omega}.\vec{n})}{\|\vec{\omega}\|^3}\,\mathrm{d}\omega \qquad (4.2.1.7)$$

Avec la fonction de HeavySide definie par :

$$(x) = \begin{cases}0 & \text{si } x<0\\ 1 & \text{si } x\geq 0\end{cases}$$

La figure 4.1 retrace l'évolution de $\Omega_F(\vec{n})$ selon l'angle θ pour des orientation de $\phi = 0$ et $\phi = \frac{\pi}{2}$

Les poids affectés aux valeurs de chaque face de l'envmap sont donc l'intégration du produit scalaire normalisé $\frac{\vec{\omega}.\vec{n}}{\|\vec{\omega}\|}$ par rapport à l'angle solide sur la face considéré

$$\begin{aligned}W_F(\vec{n}) &= \iint_F \frac{\vec{\omega}.\vec{n}}{\|\vec{\omega}\|}\times\frac{(\vec{\omega}.\vec{n})}{\|\vec{\omega}\|^3}\,\mathrm{d}\omega\\ &= \iint_F \frac{\vec{\omega}.\vec{n}\times(\vec{\omega}.\vec{n})}{\|\vec{\omega}\|^4}\,\mathrm{d}\omega\end{aligned} \qquad (4.2.1.8)$$

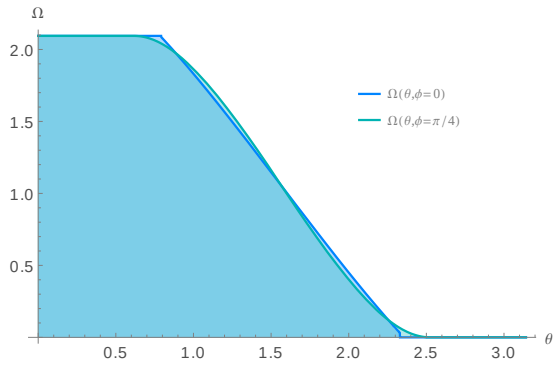


FIGURE 4.1 – Évolution de $\Omega_F(\vec{n})$ selon l'orientation de $\vec{n}$

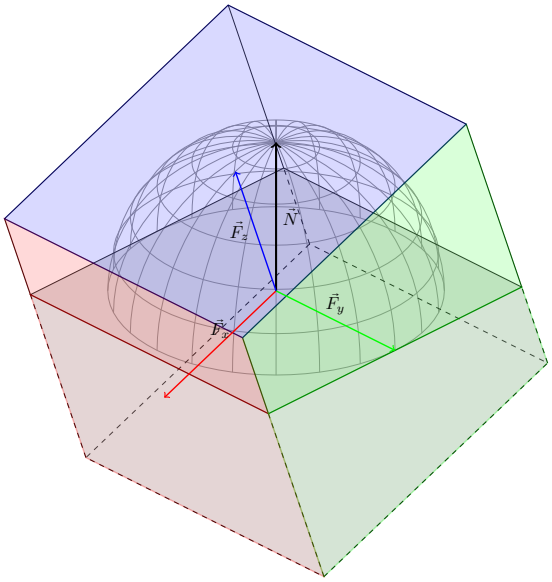


FIGURE 4.2 – Intégration de l'envmap par rapport à une facette

On notera qu'on a bien

$$\sum_{F\in Faces} \Omega(F) = 4\pi \tag{4.2.1.9a}$$

$$\forall \vec{n} \sum_{F\in Faces} \Omega(F,\vec{n}) = 2\pi \tag{4.2.1.9b}$$

$$\forall \vec{n} \sum_{F\in Faces} W_F(\vec{n}) = \pi \tag{4.2.1.9c}$$

Dès lors, l'approximation selon laquelle $\mathcal{L}$ est constante sur chaque face nous donne

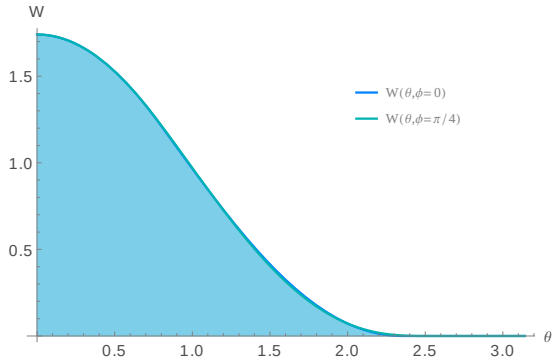
$$\begin{aligned}\mathcal{E}(p,\vec{n}) &= \frac{\mathcal{P}_\mathcal{V}(p)}{\pi} \sum_{F\in Faces} \mathcal{L}_{glob}(\vec{F}) W_F(\vec{n}) \\ &= \frac{\mathcal{P}_\mathcal{V}(p)}{\pi} \sum_{F\in Faces} \mathcal{L}_{glob}(\vec{F}) \iint_F \frac{\vec{\omega}.\vec{n} \times (\vec{\omega}.\vec{n})}{\|\vec{\omega}\|^4} \,\mathrm{d}\omega\end{aligned} \tag{4.2.1.10}$$

4.2.2 Approximation numérique

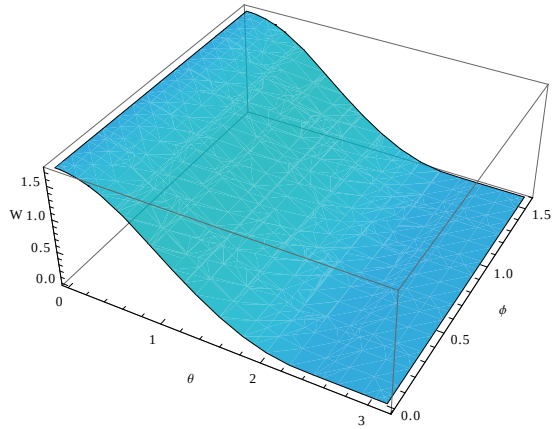
En considérant la face F_{Z^+} on obtient (sans perte de généralité)

$$W_{F_{Z^+}}(\vec{n}) = \iint_{[-1;1]^2} \left[\frac{(x.\vec{n}_x + y.\vec{n}_y + \vec{n}_z)}{(x^2+y^2+1)^2} (x.\vec{n}_x + y.\vec{n}_y + \vec{n}_z)\,\mathrm{d}x\,\mathrm{d}y \right] \tag{4.2.2.1}$$

Le problème est alors de trouver un moyen de calculer, ou au moins d'approcher, la fonction $W_F(\vec{n})$. Ce calcul étant par ailleurs fait en chaque nœud du maillage, il est primordial de le faire en utilisant un minimum de ressource, quitte à évaluer une valeur approchée qui affectera le résultat de manière faible comparativement avec l'approximation faite précédemment et selon laquelle $\mathcal{L}$ est constante sur chacune faces.



(a) $W(\theta)$



(b) $W(\theta,\phi)$

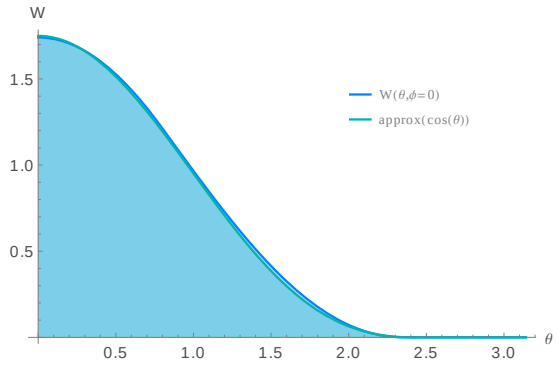
FIGURE 4.3 – Évolution de $W_F(\vec{n})$ selon l'orientation de $\vec{n}$

Comme le montre la figure 4.3, la fonction W_F dépendant principalement de θ on tentera de l'approximer par une fonction de $\vec{n}.\vec{F} = \cos(\theta)$

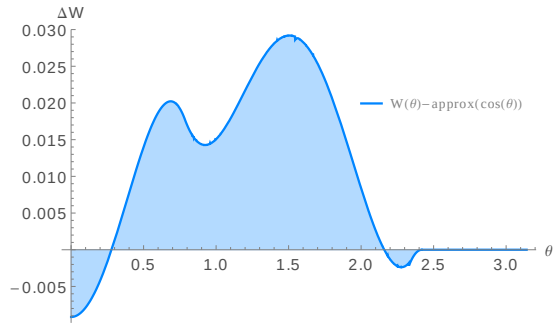
Une approximation simple est la fonction

$$approx : \cos(\theta) \mapsto \frac{\left[\max(.75+\cos(\theta),0)\right]^2}{1.75} \tag{4.2.2.2}$$

Comme le montre la figure 4.4, cette fonction est, malgré sa grande simplicité proche de la fonction $W_F(\vec{n})$.



(a) $W(\theta)$ et $approx(\cos(\theta))$



(b) ΔW

FIGURE 4.4 – Comparaison entre $W_F(\vec{n})$ et $approx(\cos(\theta))$

4.2.3 Discussion

La méthode développée ici repose sur l'approximation de la formule (4.2.1.10) et plus particulièrement de l'intégrale présenté dans la formule (4.2.1.8). La complexité de cette intégrale réside dans le domaine effectif d'intégration, conditionné a la fois par la forme carré des faces de l'envmap et la présence d'un terme caractéristique de l'hémisphère visible.

Plusieurs formules ont été étudiée pour des domaines d'intégration hémisphériques ou partiellement hémisphérique ainsi que pour des polygones entièrement visibles [Snyder and Report, 1996] (ce qui fait disparaître le terme de).

Il aurait été possible d'adapter un modèle polygonale évoqué dans [Snyder and Report, 1996] mais les polygones sur lesquels il aurait fallu intégrer variant selon les conditions de visibilités il aurait été nécessaire d'effectuer de coûteux calculs de géométrie.

L'objectif principal étant ici une grande vitesse d'exécution, l'inexactitude des résultats étant de toute façon largement oubliée au regard des approximations faites précédemment on préféra utiliser une fonction grossièrement approché mais simple à calculer.

Partie 4.3 Ombrage

4.3.1 Modèle

Un indice visuel primordiale à la vraisemblance visuel des images produites est la présence d'ombres portées provoquées par l'ajout de l'objet . referen

La modélisation de l'impact d'un tel ajout peut théoriquement être calculé qu'en connaissant la géométrie de la scène et la nature des matériaux qui la compose.

On fera ici plusieurs hypothèses dans le but d'obtenir un modèle qui soit calculable en temps réels tout en donnant des résultats vraisemblables.

Le calcul d'ombre douces est alors fait en décomposant l'objet en une hiérarchie de sphère comme présenté dans [Iwanicki, 2013] et en sommant les contribution des différentes sphères.

Pour évaluer l'ombre douce projeté par une sphère il suffit alors d'évaluer le manque d'illumination. On rappel les formules suivantes

$$\mathcal{E}_{ambiant}(p,\vec{n}) = \frac{1}{\pi}\int_{\mathcal{H}^2(\vec{n})} \mathcal{L}_{glob}(\vec{\omega}) \times \vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega}$$

$$\mathcal{E}_{ombre}(p,\vec{n}) = \mathcal{E}_{ambiant}(p,\vec{n}) - \frac{1}{\pi}\int_{\mathcal{S}(p)} \mathcal{L}_{glob}(\vec{\omega}) \times \vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega}$$

avec $\mathcal{S}(p)$ la portion de sphère visible depuis la point considéré.

L'ombre est rendu en assombrissant le pixel associé ce point d'un facteur

$$\mathcal{F}(p) = \frac{\mathcal{E}_{ombre}(p,\vec{n})}{\mathcal{E}_{ambiant}(p,\vec{n})} \tag{4.3.1.1}$$

$$= 1 - \frac{\int_{\mathcal{S}(p)} \mathcal{L}_{glob}(\vec{\omega}) \times \vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega}}{\int_{\mathcal{H}^2(\vec{n})} \mathcal{L}_{glob}(\vec{\omega}) \times \vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega}} \tag{4.3.1.2}$$

Les différents niveaux de détail de l'envmap nous permettant d'obtenir des approximations de $L(p,\vec{\omega})$ sur $\mathcal{S}(p)$ et sur $\mathcal{H}^2(\vec{n})$ on peut simplifier la formule en :

$$\mathcal{F}(p) = 1 - \frac{\mathcal{L}_{glob}(\mathcal{S}(p))}{\mathcal{L}_{glob}(\mathcal{H}^2(\vec{n}))}\left(\frac{1}{\pi}\int_{\mathcal{S}(p)} \vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega}\right) \tag{4.3.1.3}$$

Le calcul de l'angle solide formé par la sphère, et pondéré par un cosinus est détaillé dans [Snyder and Report, 1996]. On retiendra que dans notre cas ou la sphère est supposée au dessus du plan sur lequel se projettent les ombres :

$$\int_{\mathcal{S}(p)} \vec{\omega}.\vec{n}\,\mathrm{d}\vec{\omega} = \cos(\omega)\sin^2(\alpha) \tag{4.3.1.4}$$

avec α le demi angle sous lequel est vu la sphère et ω l'angle entre la vertical (normale à la surface sur laquelle se projettent les ombres) et la direction de la sphère.

On obtient ainsi :

$$\mathcal{F}(p) = 1 - \cos(\omega)\sin^2(\alpha)\frac{\mathcal{L}_{glob}(\mathcal{S}(p))}{\mathcal{L}_{glob}(\mathcal{H}^2(\vec{n}))} \tag{4.3.1.5}$$

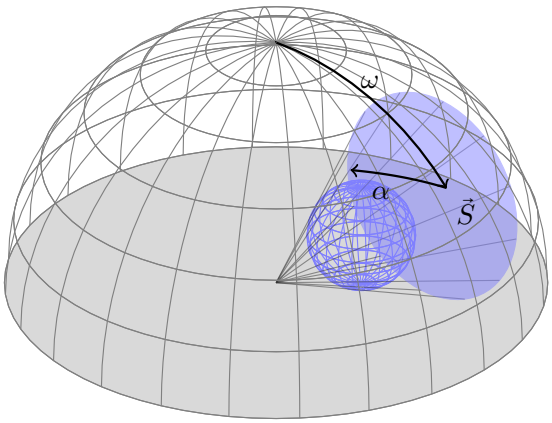


FIGURE 4.5 – Obstruction par une sphere

4.3.2 Discussion

La ou il est habituel de segmenter l'environnement pour en extraire une hiérarchie de sources lumineuses, la méthode proposée ici permet d'évaluer des ombres douces à partir de données d'environnement sans étape de segmentation ni calcul de visibilité.

Un développement intéressant serait de construire une décomposition hiérarchique, potentiellement intégrée dans un octree, et de l'évaluer plus ou moins profondément selon la distance considérée.

Partie 4.4 Reflets spéculaires

Le calcul de l'éclairage ambiant revient à considérer une BRDF purement lambertienne. Afin d'améliorer le réalisme du rendu il convient d'adopter un modèle de Phong en ajoutant une part d'éclairage spéculaire.

Cet éclairage spéculaire permet de rendre de manière intéressante des surfaces métallique, dans la limite d'un seul reflet. Le modèle adopté ici est par ailleurs isotrope, ce qui ne permet par le rendu de matières comme du métal brossé pour lesquels on retrouve des directions privilégiés.

Dans notre cas l'évaluation du reflet se fait naturellement en calculant la réflexion du rayon incident, donné par la position du point considéré dans l'espace de la camera, relativement à la normale de l'objet dans ce même espace. On accédera ensuite à la valeur d'éclairage directement dans l'envmap.

Une fois de plus on pourra utiliser les différents niveaux de mipmap à notre avantage, la considération du niveau de mipmap revenant la considérer l'angle d'un cône autour de l'axe du reflet. Un tel cône permet ainsi de caractériser le caractère spéculaire de l'objet, cette dernière pouvant varier d'un reflet parfait –miroir– à un reflet plus diffus –plastique–. On utilisera également l'ombre projeté calculé précédemment afin de moduler les reflets.

Partie 4.5 Vers un modèles à micro-facettes

Les modèles employés ici permettent un rendu rapide mais présentes des limitations en terme de qualité. Il serait intéressant, dans l'optique améliorer encore la qualité du rendu, de considérer un modèle à micro-facette. Le principes des modèles à micro-facette est de considérer la surface de l'objet comme un ensemble de petites faces orientés selon des normales propres a chacune (micro-normales). La distribution statistique des micro-normales autour de la normale géométrique du maillage (macro-normale) permet de déterminer les mécanismes de réflexion. Parmi les nombreux avantages d'un telle méthode il y a la possibilité de caractériser des surfaces anisotropes par le biais de directions privilégiés dans la distribution de facettes.

L'auto-occultation entre facettes, variable selon le point de vue, permet par ailleurs de calculer une normal intermédiaire entre la normale géométrique et les micro-normales des facette (mézo-normal) qui correspond à l'intégration des micro-normales sur l'ensemble des facette visible [Bruneton et al., 2010] [Heitz et al., 2013b] . Cette figure mézo-normale, utilisée à la place de normale géométrique, permet de corriger les reflets sur des surfaces vue sous un angle important.

L'intégration complète d'un tel modèle à été envisagé de la manière suivante :

1. Choix d'une distribution de normale (gaussienne) et étude du modèle associé ;
2. Ajout aux objets 3D d'une texture (optionnelle) caractérisant localement la direction privilégiée et la force de l'anisotropie associée ;
3. Évaluer, dans le fragment shader, la distribution de direction reflétées en fonction de l'angle de vue et des caractéristiques stockées dans la texture ;
4. Évaluer la lumière incidente selon la distribution calculée précédemment.

Au delà des calcul complexes, mais heureusement déjà documentées, du premier point [Heitz et al., 2013a], l'évaluation de l'envmap selon des distributions anisotrope nécessite de lourds calculs. Il serait possible de les réduire fortement via un pré-calcule (convolution) mais dans notre cas le caractère dynamique de l'environnement ne permet pas un telle approche.

Dans notre cas, il serait nécessaire d'évaluer, sans pré-calcul, l'intégration anisotrope de l'envmap. Les outils de filtrages anisotropes ne sont ici pas exploitable en l'état car une filtre anisotrope est définit entre autre par le facteur d'anisotropie qui le caractérise. Le nombre d'unité de texture étant grandement limité, il n'est pas possible de charger plusieurs fois l'envmap avec des filtrages différents qui couvriraient toutes nos attentes.

Les outils de lecture de texture par gradient [1] mis en place dans openGL ne conviennent pas non plus car même si les vecteurs fournis (`dX` et `dY`) sont caractéristique d'une forte anisotropie, seul celui de norme maximal compte dans l'évaluation de l'accès au niveau de texture.

1. `textureGrad()`

Il conviendrait donc, pour permettre en place les mécanismes d'évaluation anisotrope voulus, d'approximer l'intégration vis de multiples accès à la texture, le long de la direction privilégie d'anisotropie pour procéder à une intégration implicite.

Partie 4.6

Pipeline

Compte tenu des méthodes décrites précédemment, le pipeline de rendu se décompose en différentes parties (voir figure 4.6)

1. L'étape de pré-calcul permet l'évaluation de données propres au modèle. Ces données n'étant pas influencé par la localisation dans l'espace ni par les caractéristiques d'environnement lumineux, il n'est pas nécessaire de les recalculer en temps réelle et on préféra donc stocker les résultats pré-calculés.

   Ambiant : information d'auto-occultation ($\mathcal{P}_{\mathcal{V}}(p)$), stocké dans une `texture2D` ;

   Sphères : décomposition de l'objet comme union de sphères, stocké sous forme de `vec4[]`.

2. L'étape de calcul temps réel, qui évalue des résultats temporaires nécessaires à la réalisation du rendu final. Ces résultats doivent êtres réévaluer dynamiquement car ils dépendent de paramètres dynamique tel que les données d'environnement.

   Ombre : ombre douce projeté par l'objet, elle dépend de l'environnement lumineux décrit par l'envmap.

3. L'étape de rendu qui produit l'image telle qu'elle est vue par l'utilisateur.

   Rendu objet : affichage de l'objet, en tenant compte de l'éclairage ambiant, et des reflets spéculaires ;

   Rendu ombre : affichage des ombres en surimpression afin d'intégrer l'objet ne manière plus réaliste.

Partie 4.7

Résultats

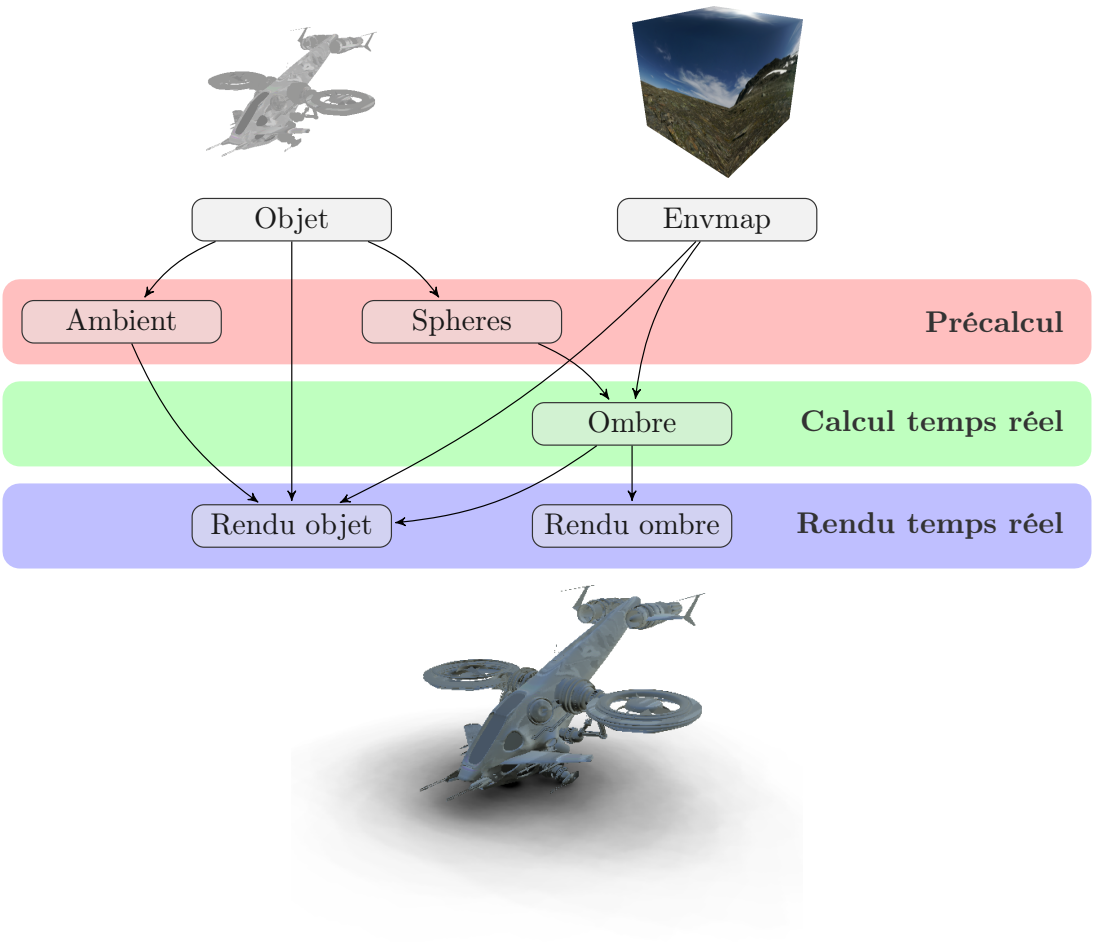


FIGURE 4.6 – Pipeline développé

Precalcul

Nous avons détaillés dans le chapitre précédent les différentes étapes de rendu pour l'intégration d'un objet 3D dans une scène dynamique acquise en temps réel. Certaines des données nécessaire à un tel rendu étant indépendantes de la scène nous nous sommes proposés de les pré-calculer.

Partie 5.1

Éclairage ambiant sous forme de texture

Nous avons vu dans la section 4.2, page 5, le calcul de l'éclairage ambiant. Dans ce calcul intervenait un terme d'auto-occultation propre à l'objet 3D considéré.

$$\mathcal{P}_{\mathcal{V}}(p) = \frac{1}{\pi} \int_{\mathcal{V}(p,\vec{n}(p))} \vec{\omega}.\vec{n}\, \mathrm{d}\vec{\omega} \tag{5.1.0.1}$$

5.1.1

Intégration statistique

Si s'agit donc d'évaluer, en tout point de l'objet, l'intégrale du produit scalaire entre le vecteur d'intégration et la normale à l'objet, pour un vecteur d'intégration parcourant l'espace visible.

On peut alors reformuler le calcul comme suit.

$$\begin{aligned}\mathcal{P}_{\mathcal{V}}(p) &= \frac{1}{\pi} \int_{\mathcal{V}(p,\vec{n}(p))} \vec{\omega}.\vec{n}\, \mathrm{d}\vec{\omega} \\ &= \frac{1}{\pi} \int_{\mathcal{H}(\vec{n}(p))} \vec{\omega}.\vec{n} \times visible(p,\vec{\omega})\, \mathrm{d}\vec{\omega} \\ &= \frac{1}{\pi} \int_{\mathcal{S}} \vec{\omega}.\vec{n} \times (\vec{\omega}.\vec{n}) \times visible(p,\vec{n})\, \mathrm{d}\vec{\omega}\end{aligned} \tag{5.1.1.1}$$

Pour évaluer une elle intégrale, on procède par une intégration statistique (Monte Carlo). Pour cela on simule le domaine d'intégration sur l'espace de définition en tirant n points uniformément sur une sphère. Ces points représentes les vecteurs $\vec{\omega}$.

Pour chacun de ces points in s'agit d'évaluer :

$$\vec{\omega}.\vec{n} \times (\vec{\omega}.\vec{n}) \times visible(p,\vec{n})$$

En réalité le terme $(\vec{\omega}.\vec{n})$ car tout point pour lequel $visible(p,\vec{n})$ sera non nul sera assuré d'être dans l'hémisphère décrit par $(\vec{\omega}.\vec{n})$. La fonction à intégrer peut donc se résumer à :

$$\vec{\omega}.\vec{n} \times visible(p,\vec{n})$$

5.1.2

Évaluation de la visibilité

L'évaluation la visibilité est faite à l'aide d'une shadowmap. Étant donnée un point de vue (vecteur $\vec{\omega}$ aléatoire) il s'agit d'effectuer une première étape de rendu, avec une projection orthographique pour en conserver que les informations de profondeurs retenus dans le Z-buffer (voir figure 5.1).

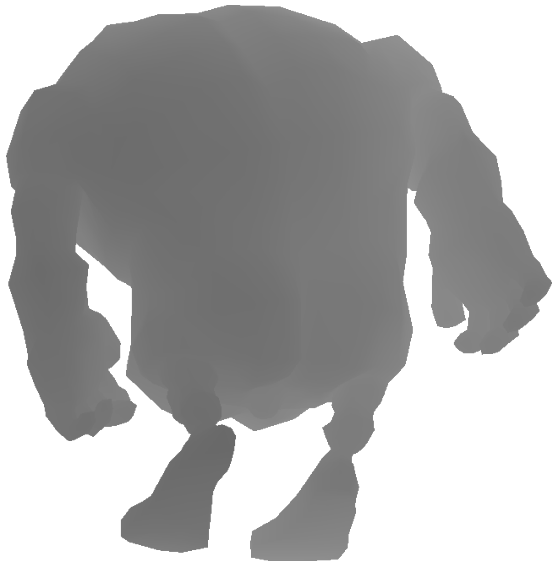


FIGURE 5.1 – Z-buffer issue de la première étape de rendu (shadowmap)

Ces données de profondeurs permettent ensuite d'évaluer si un texel est visible ou non, simplement en comparant la profondeur du texel considéré et celui retenu dans la texture et qui caractérise le texel le plus proche selon le rayon associé.

La seconde étape de rendu consiste alors à écrire, dans l'espace texture associé a l'objet, les informations de visibilités qui auront été calculé (voir figure 5.2). A cette étape, il est important de s'assurer que la texture est bien écrite, y comprit au niveau des jointures entre les différents éléments.

référen

5.1.3

Post traitement

Une fois évalué pour une source, les résultats sont ajoutés à une texture qui somme les contributions des différentes sources. Cette texture contient ainsi le résultat de

FIGURE 5.2 – Rendu en espace texture pour une source

l'intégration considère.

Une fois toutes les sources évaluées, il reste quelques étapes de post traitement :

— L'intégration étant normalisé, les valeurs calculées sont théoriquement entre 0.0 et 1.0. Cependant du fait de la distribution aléatoire des sources utilisées pour l'intégration, il peut arriver qu'en certains points ne pressentant pas d'auto-occultation la valeur dépasse 1.0. Les valeurs sont alors seuillés afin de ne pas poser de problème au moment du stockage au format `.png`.
Les données étant quantifiées entre 0 et 255, un dépassement de la valeur flottante à stoker peut provoquer une traduction en un entier supérieur à 255. Ces nombres étant stockés sous forme de caractère ASCII, un dépassement à 256 ou 257 peut alors provoquer une évaluation modulo 256 soit un stockage sous forme de 0 ou de 1.

— Les parties de l'image ne correspondant pas a des coordonnées de textures valides étant jusque la vides. L'évaluation des niveaux de mipmap pour la texture considéré risque de prendre en compte des éléments qui ne sont par représentatives de la réalité géométrique de l'objet. Afin de limiter ce biais dans l'évaluation des niveaux de mipmap, on remplit les parties vides et non représentative avec la valeur moyenne calculée sur les parties représentatives. Ainsi on limite l'incohérence des données considère en bordure de patch pour les niveaux de mipmap les plus élevés.

A l'issue de ces différents post-traitement, il ne reste qu'a exporter la textures sous forme d'image (voir figure 5.3) qui sera chargée le moment voulu.

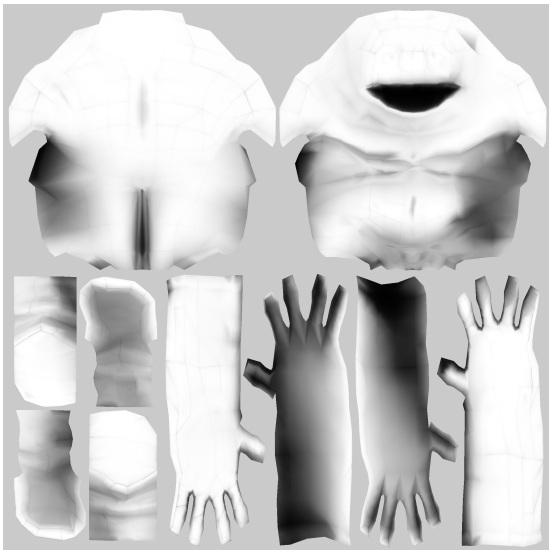


FIGURE 5.3 – Données pré-calculées en espace texture après post-traitement (1000 sources)

Partie 5.2

Décomposition en sphères

L'application

Partie 6.1

Structure

Partie 6.2

Configuration

Partie 6.3

Résultats

Perspectives d'évolution

Partie 7.1

Acquisition HDR

Partie 7.2

BRDF anisotropes

Partie 7.3

Modèles dynamiques

Annexes

Bibliographie

[Aila, 2005] Aila, T. (2005). Conservative and Tiled Rasterization Using a Modified Triangle Setup. 10(3) :1–7.

[Akerlund et al., 2007] Akerlund, O., Unger, M., and Wang, R. (2007). Precomputed Visibility Cuts for Interactive Relighting with Dynamic BRDFs. *15th Pacific Conference on Computer Graphics and Applications (PG'07)*, pages 161–170.

[ATI,] ATI. ATI Technologies Inc. Diffuse Cube Mapping. pages 4–6.

[Blinn, 1977] Blinn, J. F. (1977). Models of light reflection for computer synthesized pictures. pages 192–198.

[Brennan, 2002] Brennan, C. (2002). Accurate Environment Mapped Reflections and Refractions by Adjusting for Object Distance. pages 1–6.

[Bruneton et al., 2010] Bruneton, E., Neyret, F., and Holzschuch, N. (2010). Real-time Realistic Ocean Lighting using Seamless Transitions from Geometry to BRDF. *Computer Graphics Forum*, 29(2) :487–496.

[Dupuy et al., 2012] Dupuy, J., Heitz, E., Iehl, J.-c., Poulin, P., Neyret, F., and Ostromoukhov, V. (2012). Linear Efficient Antialiased Displacement and Reflectance Mapping. 32(6) :1–11.

[Gibson et al., 2003] Gibson, S., Cook, J., Howard, T., and Hubbold, R. (2003). Rapid Shadow Generation in Real-World Lighting Environments.

[Gibson et al., 2001] Gibson, S., Howard, T., and Hubbold, R. (2001). Flexible Image-Based Photometric Reconstruction using Virtual Light Sources. *Computer Graphics Forum*, 20(3) :203–214.

[Gross, 2003] Gross, M. (2003). Algebraic Point Set Surfaces.

[Havran et al., 2005] Havran, V., Smyk, M., Krawczyk, G., and Myszkowski, K. (2005). Interactive System for Dynamic Scene Lighting using Captured Video Environment Maps.

[Havran et al., 2003] Havran, V., Smyk, M., Krawczyk, G., Myszkowsky, K., and Seidel, H.-P. (2003). Importance Sampling for Video Environment Maps. (Eurographics) :2003.

[Heitz et al., 2013a] Heitz, E., Nowrouzezahrai, D., Poulin, P., and Neyret, F. (2013a). Filtering color mapped textures and surfaces. *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games - I3D '13*, page 129.

[Heitz et al., 2013b] Heitz, E., Nowrouzezahrai, D., Poulin, P., and Neyret, F. (2013b). Filtering Non-Linear Transfer Functions on Surfaces. *IEEE transactions on visualization and computer graphics*, XX(X) :1–14.

[Heymann et al., 2005] Heymann, S., Smolic, A., and Froehlich, B. (2005). Illumination reconstruction from real-time video for interactive augmented reality. pages 1–4.

[Igehy, 1999] Igehy, H. (1999). Tracing ray differentials. *Proceedings of the 26th annual conference on Computer graphics and interactive techniques - SIGGRAPH '99*, pages 179–186.

[Isidoro, 2005] Isidoro, J. R. (2005). Filtering Cubemaps, Angular Extent Filtering and Edge Seam Fixup Methods.

[Iwanicki, 2013] Iwanicki, M. (2013). Lighting technology of The Last of Us.

[Khan et al., 2002] Khan, E. A., Reinhard, E., and Fleming, R. W. (2002). Image-Based Material Editing 1.

[Křivánek and Colbert, 2008] Křivánek, J. and Colbert, M. (2008). Real-time Shading with Filtered Importance Sampling. *Computer Graphics Forum*, 27(4) :1147–1154.

[Laffont et al., 2012] Laffont, P.-Y., Bousseau, A., and Drettakis, G. (2012). Rich Intrinsic Image Decomposition of Outdoor Scenes from Multiple Views. *IEEE transactions on visualization and computer graphics*, pages 1–16.

[Lopez-Moreno et al., 2010] Lopez-Moreno, J., Hadap, S., Reinhard, E., and Gutierrez, D. (2010). Compositing images through light source detection. *Computers & Graphics*, 34(6) :698–707.

[Madsen and Laursen, 2003] Madsen, C. B. and Laursen, R. (2003). A scalable gpu-based approach to shading and shadowing for photorealistic real-time augmented reality.

[Manson and Schaefer, 2012] Manson, J. and Schaefer, S. (2012). Parameterization-Aware MIP-Mapping. *Computer Graphics Forum*, 31(4) :1455–1463.

[Marshall and Martin, 1997] Marshall, A. D. and Martin, R. R. (1997). Geometric least-squares fitting of spheres, cylinders, cones and tori. pages 1–20.

[Mcguire et al., 2013] Mcguire, M., Evangelakos, D., Wilcox, J., Donow, S., and Mara, M. (2013). Plausible Blinn-Phong Reflection of Standard Cube MIP-Maps. pages 1–8.

[Mercier et al., 2006] Mercier, B., Meneveaux, D., and Fournier, A. (2006). A Framework for Automatically Recovering Object Shape, Reflectance and Light

Sources from Calibrated Images. *International Journal of Computer Vision*, 73(1) :77–93.

[Oat and Sander, 2006] Oat, C. and Sander, P. (2006). Ambient aperture lighting. *ACM SIGGRAPH 2006 Courses on - SIGGRAPH '06*, page 143.

[P. Debevec, 2005] P. Debevec (2005). A Median Cut Algorithm for Light Probe Sampling.

[Parker et al., 1999] Parker, S., Martin, W., Sloan, P.-P. J., Shirley, P., Smits, B., and Hansen, C. (1999). Interactive ray tracing. *Proceedings of the 1999 symposium on Interactive 3D graphics - SI3D '99*, pages 119–126.

[Rempel et al., 2006] Rempel, A. G., Trentacoste, M., Seetzen, H., Young, H. D., Heidrich, W., Whitehead, L., and Ward, G. (2006). Ldr2Hdr : On-the-fly Reverse Tone Mapping of Legacy Video and Photographs. pages 2–7.

[Sloan and Corporation,] Sloan, P.-p. and Corporation, M. Stupid Spherical Harmonics (SH) Tricks.

[Snyder and Report, 1996] Snyder, J. M. and Report, T. (1996). Area Light Sources for Real-Time Graphics.

[Unger et al., 2008] Unger, J., Gustavson, S., Larsson, P., and Ynnerman, a. (2008). Free Form Incident Light Fields. *Computer Graphics Forum*, 27(4) :1293–1301.

[Wang and Å kerlund, 2009] Wang, R. and Å kerlund, O. (2009). Bidirectional Importance Sampling for Unstructured Direct Illumination. *Computer Graphics Forum*, 28(2) :269–278.

Liste des tableaux

Liste des exemples de code

Liste des figures