

Samenvatting Computer grafieken

Gemaakt door Bob Vergauwen (Januari 2016)

Les 1 (Hoofdstuk 1-3)

I. LEG HET WERKINGSPRINCIPE VAN RAY TRACING UIT.

Ray tracing is een methode om beelden te genereren door middel van stralen te schieten.

Voor een raytracing algoritme zijn er een aantal zaken nodig die allemaal worden samengevat in de wereld. De onderdelen van zo'n wereld zijn

- een camera
- een view scherm
- objecten
- een achtergrond
- lichten

Element die zich bevinden in de wereld worden allemaal voorgesteld aan de hand van wereld coordinaten(=absolute coordinaten).

Het scherm (view plane) wordt in een eerste stap opgedeeld in een aantal discrete pixels.

Vervolgens worden er per pixel (mogelijk) een aantal stralen geschoten.

Voor elke straal wordt er nadien getest of er ergens een kruising is met een object uit de scene. Is dit niet het geval dan wordt de kleur van de pixel bepaald door middel van een achtergrond. Wanneer er wel een doorsnede is wordt de kleur van de pixel bepaald aan de hand van het object waarop de straal invalt.

Door genoeg pixels te gebruiken kan een raytracer op deze manier een beeld van de scène opbouwen.

Buiten de wereld zijn de stralen de tweede categorie van belangrijke objecten in een raytracer. Er zijn een aantal verschillende stralen

- primaire stralen
 - Stralen die vertrekken vanuit de camera en door een pixel op het scherm gaan.
- secundaire stralen
 - Gereflecteerde of uitgestraalde stralen vanop objecten uit de scène
- Schaduw stralen
 - Schaduw stralen worden gebruikt voor shaders en vertrekken vanop het oppervlak van een object
- licht stralen
 - Licht stralen zijn stralen die vertrekken vanuit een lichtbron en worden gebruikt voor de belichting van de scène.

Het basis idee achter ray tracing

```
define some objects
specify a material for each object
define some light sources
define a window whose surface is covered with pixels

for each pixel
    shoot a ray towards the objects from the center
        of the pixel
    compute the nearest hit point of the ray with the
        objects (if any)

    if the ray hits an object
        use the object's material and the lights to
            compute the pixel color
    else
        set the pixel color to black
```

- Mogelijke problemen
- Aliasing
- Rekening houden met de positie van de objecten, liggen ze voor of achter het scherm.

ELEMENTEN EN ONTWERP VAN EEN RAYTRACER.

De beste methode voor het bouwen van een raytracer is aan de hand van een OO structuur. De scene wordt meestal opgeslagen in een aparte data file, the scene description file.

II. BESPREEK BARYCENTRISCHE COÖRDINATEN. IN WELKE ASPECTEN VAN DE VOLLEDIGE RAY TRACING PIPELINE (BEELDVORMING, SHADING, ...) SPELEN BARYCENTRISCHE COÖRDINATEN EEN ROL?

In een raytracer wordt er een veelheid aan coördinaten stelsels gebruikt, een kleine opsomming:

- wereld coördinaten
- View coördinaten
- object coördinaten
- Shading coördinaten
- Texture coördinaten
- Scherm coördinaten
- ...

Een speciale soort vaar coördinaten systeem is het barycentrische coördinaten stelsel. Barycentrische coördinaten vormen een coördinatenstelsel waarmee een punt vastgelegd wordt ten opzichte van de hoekpunten van een simplex.

Hierbij kan dus elk punt in het simplex (o.a. een driehoek) worden voorgesteld met behulp van een lineaire combinatie van de coördinaten van de hoekpunten.

Wanneer we als voorbeeld een driehoek nemen in een 2D ruimte dan is er duidelijke een redundantie, de punten in deze driehoek kunnen dan ook worden voorgesteld met maar twee basis vectoren.

De redundantie wordt opgelost door te eisen dat de gewichten van de lineaire combinatie tussen 0 en 1 liggen.

Wanneer zo een punt kan worden gevonden is men zeker dat dit punt in het simplex is gelegen. Neem nu de driehoek met hoekpunten abc dan kunnen de barycentrische coördinaten worden gevonden door de driehoek te translaten met 1 van de hoekpunten naar de oorsprong.

$$\begin{aligned} p &= a + \beta(b - a) + \gamma(c - a) \\ p &= (1 - \beta - \gamma)a + \beta b + \gamma c \\ p &= \alpha a + \beta b + \gamma c \quad (\alpha + \beta + \gamma = 1) \end{aligned}$$

Elk punt p kan dan door middel van de bovenstaande vergelijking worden geschreven t.o.v zijn barycentrische coördinaten.

Wanneer het punt p in de driehoek ligt geldt er dat \alpha, \beta en \gamma in het interval [0,1] liggen. Een andere manier om barycentrische coördinaten te begrijpen is door middel van de analogie te maken met het massa centrum van de simplex. Voor een simplex van massa 1 met massa centrum p zijn de gewichten van de hoekpunten gelijk aan hun massa.

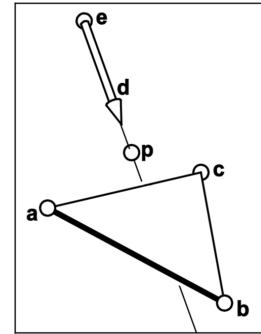
STRAAL INTERSECTIES BEPALEN MET BARYCENTRISCHE COORDINATEN

Voor het vinden van een doorsneden tussen een driehoek en een straal maken we gebruik van de parametrische voorstelling van de straal

$$p(t) = e + td = a + \beta(b - a) + \gamma(c - a)$$

In deze vergelijking zijn 3 onbekende, t, \beta en \gamma. Het stelsel dat moet worden opgelost ziet er als volgt uit:

$$\begin{bmatrix} x_a - x_b & x_a - x_c & x_d \\ y_a - y_b & y_a - y_c & y_d \\ z_a - z_b & z_a - z_c & z_d \end{bmatrix} \begin{bmatrix} \beta \\ \gamma \\ t \end{bmatrix} = \begin{bmatrix} x_a - x_e \\ y_a - y_e \\ z_a - z_e \end{bmatrix}$$



In de rechter figuur staat de verduidelijking van de verschillende parameters die gebruikt zijn in het stelsel.

Voor een complex figuur dat is opgebouwd uit een groot aantal polygonen moeten er vaak veel intersecties worden bepaald. Deze operatie is vrij duur dus is het aangewezen om enkel de intersecties te berekenen waarbij een grote kans is dat er effectief een intersectie gaat zijn.

Dit kan men doen aan de hand van een acceleratie structuur.

Soms is het handiger om voorwerpen te kunnen voorstellen met andere vormen dan polygonen, bijvoorbeeld met bollen of cilinders.

De technieken om intersecties te berekenen met deze voorwerpen zijn iets anders maar komen steeds op hetzelfde idee neer.

INTERSECTIE MET EEN BOL

Voor de intersectie met een bol te bepalen starten we opnieuw van de parametrische voorstelling van de straal. Vervolgens zoeken we een parameter p zodanig dat de straal binnen de afstand R van het centrum is.

Er zijn nu 3 mogelijkheden;

- Er zijn twee oplossingen, de straal snijdt de bol in twee punten
- Er is geen oplossing te vinden = De straal snijdt niet
- Er is maar 1 oplossing = de straal raakt de bol net.

De berekeningen voor dit probleem zijn als volgt:

$$\|p - c\|^2 = R^2$$

$$(p - c) \cdot (p - c) - R^2 = 0$$

$$(e + td - c) \cdot (e + td - c) - R^2 = 0$$

$$(d \cdot d)t^2 + 2d \cdot (e - c)t + (e - c) \cdot (e - c) - R^2 = 0$$

Het komt er dus op neer om voor elke straal en voor elke bol de nulpunten van een veelterm te bepalen. (Kan met een gesloten formule = efficiënt en goedkoop.)

III. HOE WORDEN KLEUREN VOORGESTELD

Het voorstellen van kleuren gebeurd steeds in een kleurruimte (color space). In de meeste implementaties wordt er 1 bit (265 mogelijkheden) per kleur kanaal gebruikt.

Er zijn een aantal gangbare kleurruimtes die worden gebruikt

RGB COLOR SPACE

De RGB voorstelling is een additieve kleuren voorstelling. In een RGB color space wordt elke kleur voorgesteld met een drietal bites. De waarden van de bites stellen de hoeveelheid kleur die voorkomt van elke component. (1,0,0) is bijvoorbeeld rood (0,0,1) is blauw etc.

VOORSTELLEN VAN KLEUREN OP HET SCHERM

Na het bepalen van de kleur van een pixel moet er nog een transformatie van de kleur naar een kleur op het scherm worden doorgevoerd. Deze correctie bestaat uit drie dingen; *Color mapping*, *gamma correction* en *integer mapping*.

- Integer mapping

Na het bepalen van de kleur kan het zijn dat dat kleur waarden geen integer meer is.

Omdat een scherm maar een eindig aantal kleuren kan voorstellen moeten de berekende kleuren nog worden omgezet naar integers.

- Tone mapping

Na het bepalen van de kleur kan het zijn dat de waarden buiten het interval 0,1 liggen. Dit kan niet worden voorgesteld op een scherm en er moet dus een transformatie worden doorgevoerd van de originele kleur naar een voorstelbare kleur. Dit is een ingewikkeld proces en wordt niet verder toegelicht.

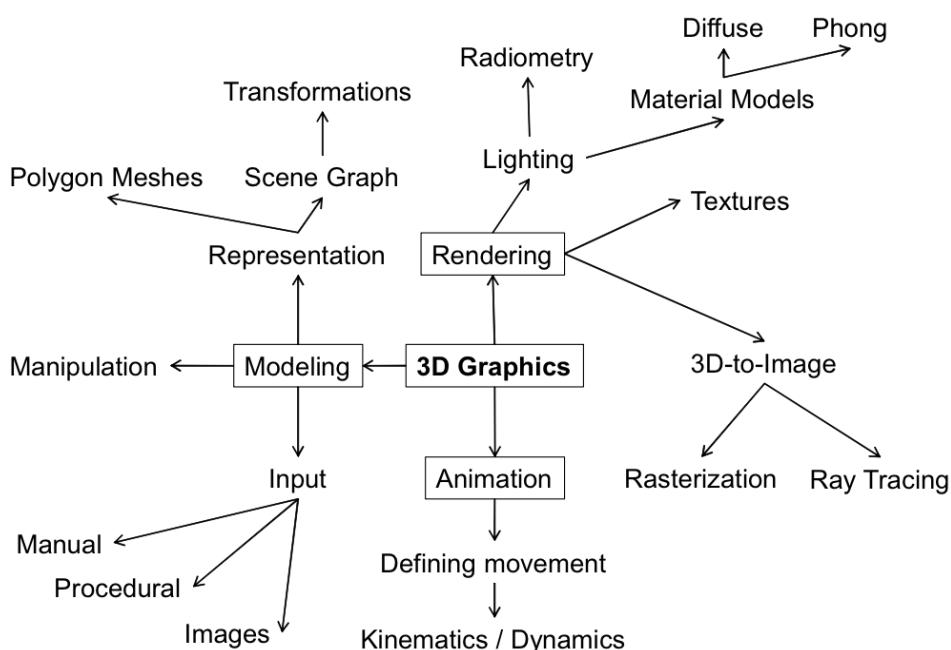
- Gamma mapping

Gamma mapping is nodig omdat de helderheid van een scherm niet evenredig is met de grote van de kleur componenten. Zo zal (255,255,255) meer dan dubbel zo licht lijken dan (128,128,128). Bij gamma mapping worden de berekende kleuren tot de macht 1/gamma gedaan. Hierbij is gamma een specifieke constante van het scherm.

BEPALEN VAN GAMMA

Gamma kan worden gevonden door de helderheid van het scherm te meten. Dit doen we bijvoorbeeld door de output van licht te meten van het scherm wanneer het de helft zwart en de helft wit heeft (dambord). Nadien meten we de helderheid van het scherm wanneer het volledig de kleur $(255,255,255)^{(\text{\gamma})}(1/2)$ weergeeft. Beide helderden moeten overeen komen met elkaar. Op deze manier kan \gamma worden bepaald.

IV. TAXONOMIE COMPUTER GRAPHICS



V. WAT ZIJN KLEURMETAMEREN? WAT IS HET BELANG ERVAN IN COMPUTER GRAPHICS?

VI. VERSCHILLENDEN VORMEN VAN PROJECTIES?

- Orthographic projection

Bij deze projectie staan alle stralen loodrecht op het view plane. Dit is een handige voorstelling voor het ontwerpen van een mesh (in ontwerp software). Bij een orthographische is er geen vervorming door het perspectief.

- Perspectivistische projectie

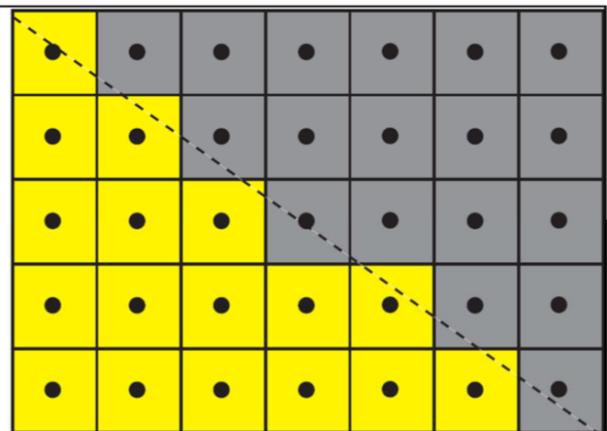
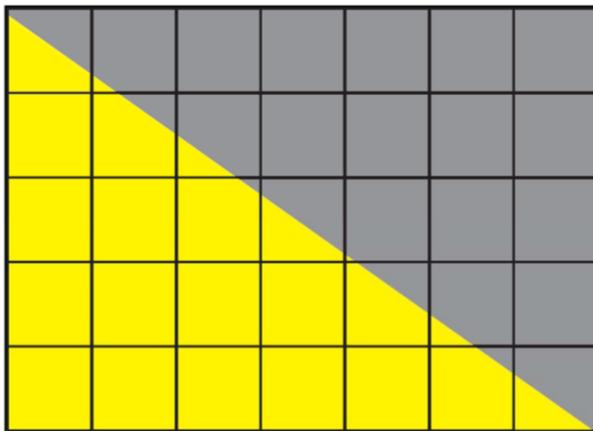
Uitleg volgt in het volgende hoofdstuk.

VII. LEG HET XYZ KLEURMODEL UIT

Les2 (Hoofdstuk 4,8-10)

I. BESCHRIJF HET PROBLEEM VAN ALIASING EN GEEF EEN AANTAL OPLOSSINGEN.

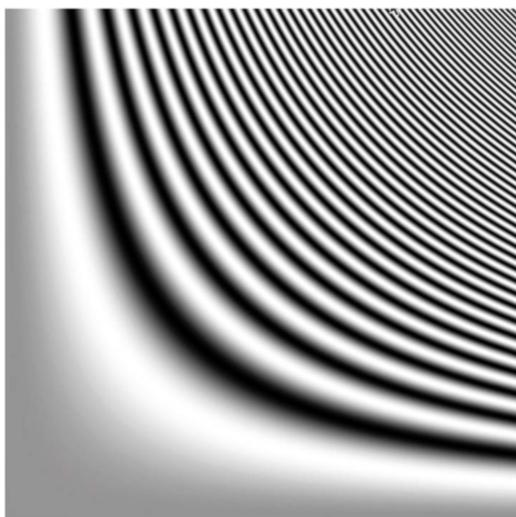
Bij het construeren van beelden met het raytracing algoritme kan het voorkomen dat er meerdere voorwerpen voorkomen in 1 pixel (of door details van het gebruikte texture).



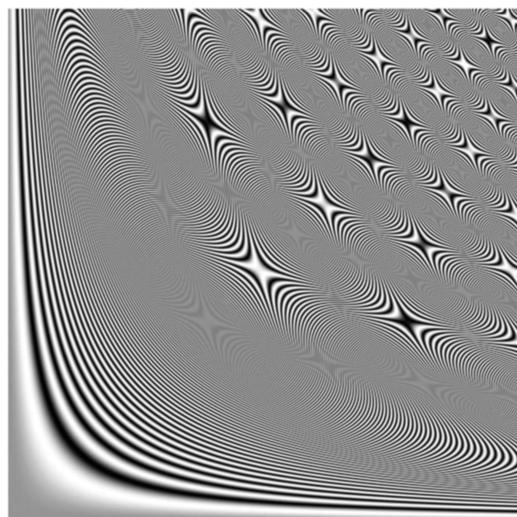
Wanneer bovenstaande scene (links) gerenderd word door 1 staal door het midden van elke pixel bekomt men de rechter afbeelding. Hierop is te zien dat de zacht verlopende lijn nu sterk discontinu is.

Een tweede voorbeeld waarbij aliasing een probleem oplevert is gegeven in de figuur hier onder.

(images are 512 x 512 pixels, function evaluation at centre of pixel)



$$(x, y) \in [0, 3.79]^2$$



$$(x, y) \in [0, 10.83]^2$$

In deze figuur is de volgende functie geplot,

$$f(x, y) = \frac{1}{2}(1 + \sin(x^2 y^2))$$

Door het steeds sterker oscillerend gedrag van de functie kunnen niet all details meer worden weergegeven in 1 pixel en ontstaan er ongewenste patronen.

ANTI-ALIASING TECHNIEKEN

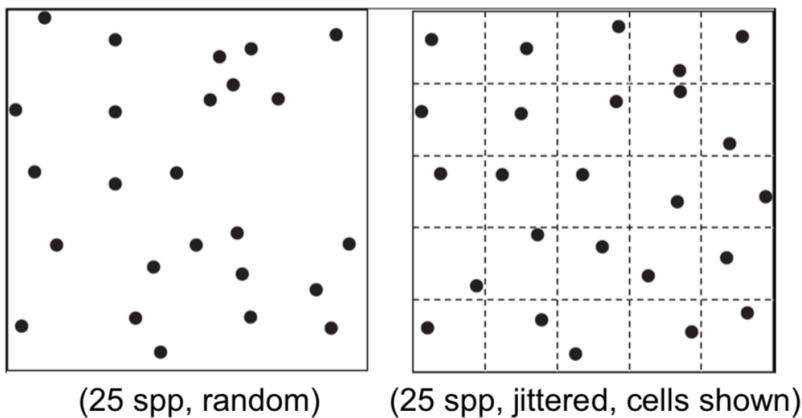
- Verhogen van de resolutie

De meeste eenvoudige techniek is door de resolutie te verhogen waarmee de beelden worden weergegeven. Dit zorgt dat alle lijnen er beter uitzien maar zal nog steeds geen verschil in kleuren geven voor verschillende waarden op de rand.

- Over sampling

Hierbij is het idee om meerdere stralen per pixel te gebruiken om op deze manier een gemiddelde waarde per pixel te kunnen vinden.

Het bepalen van de locaties van de verschillende stralen kan op een aantal manieren, bijvoorbeeld *deterministisch*, *stochastisch* of *jittered* (een combinatie van beide). De twee laatste technieken zijn geïllustreerd in de onderstaande afbeelding.



Het gebruik van een deterministische oversampling is af te raden, dit leidt tot aliasing effecten op hogere frequenties.

Wanneer er volledig random wordt gesampled wordt er een ruis effect gecreëerd.

- Filtering

Een derde manier voor aliasing te vermijden is door middel van een filter te gebruiken.

Oversampling is eigenlijk ook al een filter, namelijk de moving average filter die per pixel wordt gebruikt. Er zijn echter ook filters mogelijk die de informatie van een aantal naburige pixels gebruiken. Twee populaire filters zijn de box filter en de gaussian blur. Voor het box filter wordt de uiteindelijke kleur van de pixel bepaald door het gemiddelde van alle waarden. Voor de gaussische filter worden het gewogen gemiddelde gebruikt met gewichten volgens een gaus curve.

II. ENKELE DEFINITIES VOOR PERSPECTIEF

- Een projectie

Een projectie is een affiene transformatie die een 3-D ruimte transformeert op een 2-D vlak. Dit vlak noemt men meestal het view plane. Alle punten worden getransformeerd via rechte lijnen, deze noemt men projectoren. Bij een perspectief projectie convergeren al deze lijnen in 1 enkel punt, dit heet het center of the projection (brandpunt).

- Orthographic projection

Bij een orthografische projectie ligt het brandpunt op oneindig, alle projectoren vallen loodrecht in op het view plane.

- View volume

Dit is het volume van de ruimte dat kan worden bereikt door een projector. Het View volume hangt af van het view plane en de camera positie.

Merk op dat niet enkel de zichtbare stukken behoren tot het view plane maar alles dat in de piramide ligt met als top het brandpunt. (Het view volume ligt zowel voor als achter het view plane.)

- Field of view.

De ruimte hoek die wordt beschreven door het brandpunt en het view plane is de field of view.

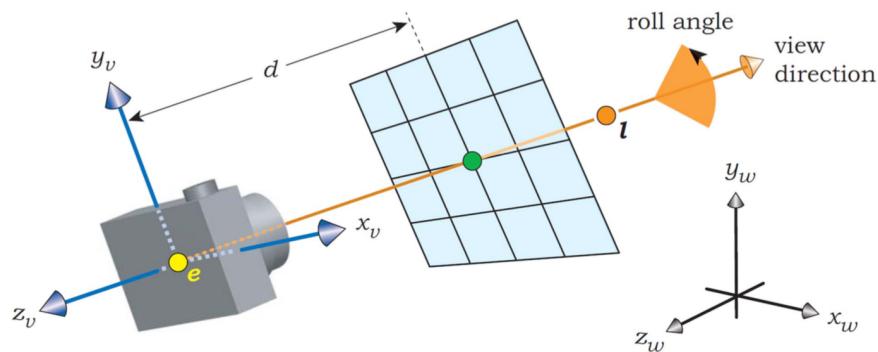
III. EIGENSCHAPPEN VAN PERSPECTIEF (ZOALS VERMELD IN HET BOEK)

- De perspectief projectie word kleiner wanneer het voorwerp zich verder van het brand punt bevindt.
- Wanneer een geprojecteerd voorwerp wordt geroteerd word zijn breedte op het view plane kleiner. Dit effect is gekend als *foreshortening*. (Neem als object bijvoorbeeld een lange smalle balk).
- Een ‘perspectivistische’ projectie behoud rechte lijnen.
- Een set van lijnen die parallel zijn aan het view plane blijven parallel na een projectie op het view plane.
- Lijnen die niet parallel lopen met het view-plane convergeren in 1 enkel punt.
- Een driehoek blijft een driehoek na een projectie.
- Een bol blijft in het algemeen geen bol na een projectie maar wordt vervormd tot een ellips.

IV. BESCHRIJF HET VIRTUAL PINHOLE CAMERA MODEL

Aantal belangrijke features van de pinhole camera:

- Het is mogelijk om een willekeurig view point te nemen (e)
- Mogelijkheid voor willekeurige view richting (l)
- Een willekeurige oriëntatie (Kiezen van boven en onder) (up)
- Willekeurige afstand tot het view plane (d)



De camera wordt gedefinieerd in de wereld coordinaten. Dit is belangrijk om de juiste stalen te kunnen berekenen voor de intersectie met andere objecten. De camera zelf heeft ook nog een stel van eigen coordinaten, dit zijn de *camera coordinaten*.

Het oogpunt e en het richtingspunt / bepalen de kijkrichting doormiddel van de genormaliseerde vector die de twee verbind.

Voor de camera coordinaten te bepalen wordt er een orthogonale basis opgesteld aan de hand van de gegevens van de camera. Deze basis is gegeven door (u, v, w) en wordt berekend door, De vectoren u en v zijn evenwijdig met het viewing plane. De vector w staat loodrecht op het viewing plane. De pixels van het viewing plane worden ook gericht via de vectoren u en v . Nog een laatste opmerking is dat w in de tegenovergestelde richting van de view direction wordt gekozen.

De keuze hiervoor is gemotiveerd doordat we een rechtshandig assenstelsel willen.

Voor de richting van een straal te bepalen in de coordinaten van de camera gebruiken we volgende formule:

$$direction = x \cdot u + y \cdot v - d \cdot w$$

$$x = s(c - h_{res} / 2 + 0.5)$$

$$y = s(r - v_{res} / 2 + 0.5)$$

$$w = \frac{(e - l)}{\|e - l\|} \quad u = \frac{up \times w}{\|up \times w\|} \quad v = w \times u$$

Hierbij is c de kolom nummer van de pixel, r de rij nummer.

Het aantal vrijheidsgraden voor het bepalen van een pinhole camera zijn er

- Het kijkpunt (3)
 - Kijk richting (3)
 - Roll hoek (1)
- = 7

V. DEPTH OF FIELD

- Definitie

De Depth of field van een camera is de verzameling van afstanden waarop voorwerpen in focus zijn op de licht gevoelige film. Voor de pinhole camera is de depth of field oneindig. Doordat de lens oneindig klein is zijn alle voorwerpen scherp. In een real life camera hebben de lenzen een eindige lengte waardoor een diepte onscherpte optreedt. Een lens met een eindige afmeting heeft slechts een enkel punt waarop alles scherp is, deze afstand noemt men de brandpunt afstand. (*Focal distance*)

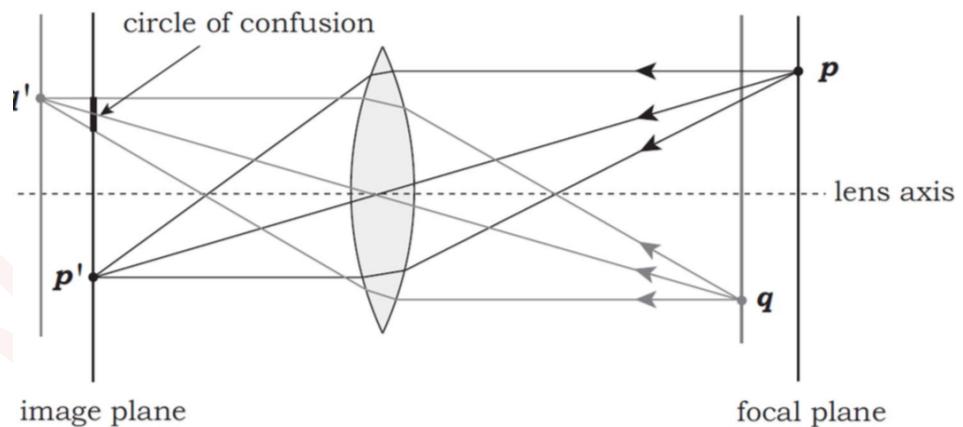
Door gebruik te maken van meer gesofisticeerde camera modellen kan diepte onscherpte worden gesimuleerd. Eén zo'n camera model wordt besproken in het volgend deel.

VI. WAT IS DIEPTE ONSCHERPTE EN HOE KAN DIT EFFECT WORDEN BEKOMEN AAN DE HAND VAN HET RAYTRACING ALGORITME?

Wat? Zie hier boven

DUNNE LENS THEORIE.

Voor we kunnen beginnen met een mogelijke implementatie van een diepte onscherpte te introduceren in onze raytracer is het nuttig de theory van de dunne lenzen nog even te herhalen.



Een dunne lens is ideale lens waarbij de breedte van de lens verwaarloosbaar is t.o.v. de diameter. In dit geval zijn er een aantal nuttige en eenvoudige eigenschappen.

De belangrijkste eigenschap die we hier gaan gebruiken is geïllustreerd in bovenstaande figuur. In deze figuur zijn er 4 vlakken aangeduid, twee vlakken voor de lens en twee er achter. Elk punt dat op het fokaal vlak ligt zal scherp worden weergegeven wanneer het invalt op het image vlak. Punten die er achter of er voor zijn gelegen worden scherp afgebeeld op een ander vlak.

Doordat punten die niet op het fokaal vlak liggen op andere afstanden scherp worden gesteld ontstaat er op deze manier een *circle of confusion*. Dit is een zone waar stralen voorkomen die van 1zelfde punt afkomstig zijn maar niet in 1 punt focussen.

Door de afstand van het image vlak te wijzigen kan de focal afstand aangepast worden.

SIMULATIE VAN DUNNE LENS

Nu het effect van de dunne lens kort is toegelicht is het tijd om het over de simulaties te hebben.

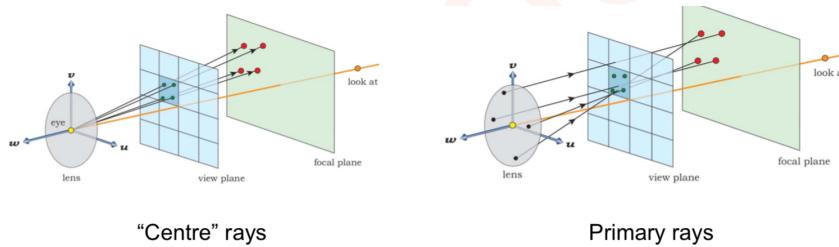
Voor de volledige correcte simulatie van de effect zouden we alle eigenschappen van de lens moeten kennen en berekenen. In een eerste eenvoudig model gaan we dit niet doen. De lens wordt hier vervangen door een cirkel met een bepaalde straal r . Deze cirkel is parallel met het view-plane. In dit model bereken we nooit de echte optische breken maar werken we met een benadering die een aanvaardbaar effect oplevert.

Het idee is eenvoudig:

- 1) Bereken het focus punt p , dit doen we aan de hand van de gegeven focus afstand. Het focus punt kan gevonden worden door een lijn te verbinden door het center van de lens en de pixel. Het focus punt ligt dan op de snijding van het focus vlak en deze primaire straal.
- 2) Een tweede stap is om willekeurige punten te sampelen op de cirkel.
- 3) De lijnen die de willekeurige punten op de cirkel en p met elkaar verbinden zijn dan de primaire stralen die worden gebruikt voor de raytracing.

Het is mogelijk om dit effect te combineren met een AA(anti-aliasing) techniek. Het enige wat we hier nog moeten doen is dan een aantal punten sampelen in de pixel.

Voor elk van dit gesampled punt moet dan terug een focus punt worden bepaald. De tweede stap is dan door opnieuw de focus punten te verbinden met random punten op de schijf. Op deze manier ontstaan er een aantal stralen die dan gebruikt worden om de waarden van de pixel te bepalen. Het is nog nuttig om een kleine opmerking te maken. Door het kiezen van willekeurige punten op de cirkel en willekeurige punten in de pixel kan het zijn dat de stralen niet meer door de originele pixel lopen, dit is geen probleem. Er is geen enkele geometrische beperking die dit tegenhoudt. (Geïllustreerd in onderstaande afbeelding).



Het gebruik van deze extra AA techniek geeft enkel een verbetering bij het onderdrukken van aliasing effecten in de buurt van het focus vlak. Punten op een andere afstand zullen zonder de AA ook geblured worden.

In dit eenvoudig model zijn er twee parameters die extra kunnen worden gekozen, de eerste is de afstand d . De invloed van deze afstand spreekt voor zich. De tweede parameter die vrij kan worden gekozen is de grote van de cirkel. Een vuist regel die we kunnen volgen is; ‘Hoe groter de diameter van de cirkel, hoe sterker de blur zal zijn’.

Het random sampelen van de punten gebeurd best met een uniforme verdeling, wanneer er een gaussische verdeling wordt gebruikt kan men andere effecten bekomen. Merk ook op dat hoe groter men de lens neemt hoe meer punten er moeten worden gesampled om een goed beeld te bekomen. Dit is een beetje contra intuïtief. Door een grotere lens verliest men scherpte (informatie) maar er moet toch meer worden gesampled (gemeten). Voor bepaalde effecten te bekomen is een vorm van post processing dus misschien een betere optie.

Les 3 (Hoofdstuk 20)

Het modelleren van de meeste objecten uit de scène wordt gedaan aan de hand van driehoeken. Twee belangrijke voorbeelden zijn de Stanford bunny en de Utah teapot. Meer moet hier nog niet over geweten zijn. In het algemeen kan er elk soort van objecten in de ruimte worden gebruikt. Er zijn maar twee voorwaarden;

- Het moet mogelijk zijn om op een efficiënte manier te kunnen bepalen of een straal met het voorwerp snijt of niet.
- De normaal vector op elk punt van het object moet kunnen worden berekend.

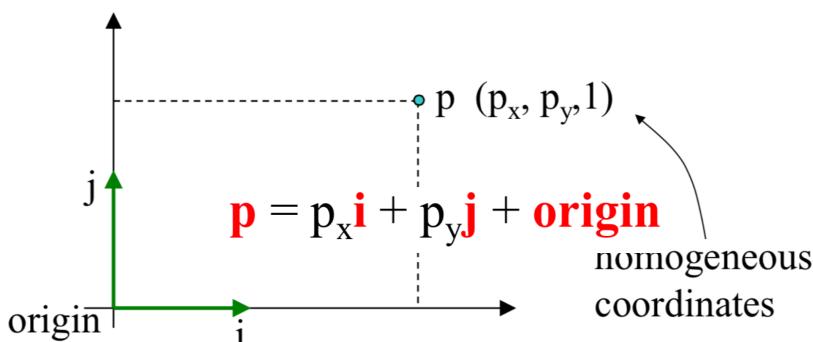
Het idee achter het modelleren van complexe scenes is om een aantal unieke voorwerpen 1 keer te definiëren en nadien getransformeerde kopieën van dit ene voorwerp te gebruiken. Op deze manier moet het voorwerp maar 1 keer worden opgeslagen en voor alle kopieën moeten enkel de gepaste transformaties worden opgeslagen.

Het gebruik van deze methode geeft aanleiding tot de constructie van een hiërarchische boom structuur.

Transformaties zijn dus handig voor het aantal voorwerpen in het geheugen te beperken maar er zijn nog een aantal andere toepassing. Een eerste toepassing is bijvoorbeeld om van camera standpunt te veranderen. In plaats van de camera te herdefiniëren kan de ruimte op een gepaste manier worden getransformeerd.

Een tweede toepassing die wordt besproken in de cursus is voor het voorzien van animaties.

I. LEG HET BEGRIP VAN HOMOGENE COORDINATEN UIT



Elk punt in de ruimte kan worden voorgesteld ten opzichte van een oorsprong en een basis.

Dit is voorgesteld in figuur op vorige pagina. De extra vrijheidsgraad in deze voorstelling heeft te maken met het feit dat we ook rekening gaan houden met verschuivingen. Dit is de categorie van de affiene meetkunde.

De som van twee punten is nog steeds een punt, het verschil van twee punten is een richtingsvector. We gaan nu kort over een aantal verschillende transformaties.

1. Skalering + Inverse

$$\begin{pmatrix} Q_x \\ Q_y \\ 1 \end{pmatrix} = \begin{pmatrix} aP_x \\ bP_y \\ 1 \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ 1 \end{pmatrix} \quad S^{-1} = \begin{pmatrix} 1/S_x & 0 & 0 \\ 0 & 1/S_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

2. Reflectie

Een reflectie komt overeen met een skalering met negatieve gewichten.

3. Rotatie + inverse

$$\begin{pmatrix} Q_x \\ Q_y \\ 1 \end{pmatrix} = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ 1 \end{pmatrix} \quad R^{-1} = \begin{pmatrix} \cos(-\theta) & -\sin(-\theta) & 0 \\ \sin(-\theta) & \cos(-\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

4. Shear + inverse

$$\begin{pmatrix} Q_x \\ Q_y \\ 1 \end{pmatrix} = \begin{pmatrix} P_x + hP_y \\ P_y \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ 1 \end{pmatrix} \quad S_h^{-1} = \begin{pmatrix} 1 & -h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

5. Translatie

$$\begin{pmatrix} Q_x \\ Q_y \\ 1 \end{pmatrix} = \begin{pmatrix} P_x + d_x \\ P_y + d_y \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ 1 \end{pmatrix}$$

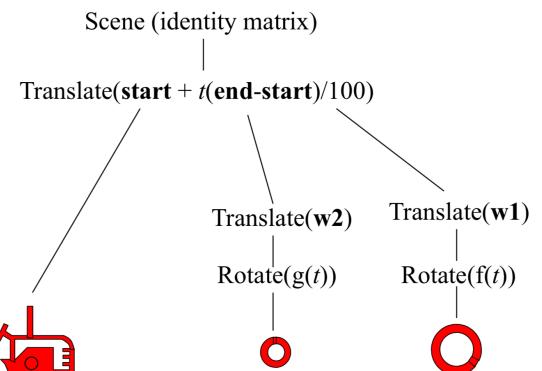
Het samenstellen van transformaties gebeurd gewoon door matrix vermenigvuldigingen in de juiste volgorde te maken.

II. LEG UIT HOE EEN SCÈNE-GRAFE KAN GEBRUIKT WORDEN OM EEN 3D SCÈNE TE MODELEREN.

Denk aan het voorbeeld met de robot. Alle onderdelen van de robot bestonden uit cirkels en een bal. De onderdelen werden onderling gegroepeerd zodat ze armen en benen etc vormde. Op elke collectie van voorwerpen waren op hun beurt een aantal transformaties gedefinieerd.

III. STEL DE VOLLEDIGE SCÈNEGRAFE OP VOOR EEN FIETS (OF EEN AUTO, OF EEN ZONNESTELSEL, ...)

Voorbeeld vanuit de slides:



Les 4(Hoofdstuk 13-15)

Definitie van Shading volgens Wikipedia:

Shading verwijst naar het aanbrengen van een diepte in 3D modellen door het variëren van de belichting van een voorwerp. Shading kan zowel de diepte perceptie aanpassen als de materiaal perceptie.

Het fundamenteel probleem dat we trachten op te lossen bij shading is de vraag hoeveel licht er door elke pixel gaat. Hoe reflecteert het licht op verschillende oppervlakken en wat zijn de relevante lichtpaden die we moeten nagaan?

Bij shading zijn er een aantal materiaal eigenschappen die belangrijk zijn

- Het materiaal kan sterk reflecterend zijn.
- Is het materiaal kan transparant of deels transparant zijn.
- Het materiaal heeft mogelijks een ruwe textuur.
- Er kan subsurface scattering optreden(SSS). (= Proces waarbij het licht het materiaal voor een deel penetreert en er op een andere plaats weer uitkomt.)

Om te bepalen hoe licht een punt op een voorwerp in de scène is moet men de reflectie in dat punt kunnen bepalen. Hiervoor zijn een aantal technieken en modellen beschikbaar. Niet elke model is geschikt voor elke situatie en ze leveren allemaal andere specifieke effecten op. In het volgende deel worden twee belangrijke reflectie modellen besproken. Nadien zal het ook duidelijk worden dat beide modellen kunnen gecombineerd worden.

DIFFUUS REFLECTIE MODEL (LAMBERTIAN REFLECTIE)

De eenvoudigste vorm van shading is door gebruik te maken van de *diffuse reflectie*. De belangrijkste aannamen bij diffuse reflectie is dat

'Het inkomen licht wordt gelijkmatig verstrooid in elke richting'.

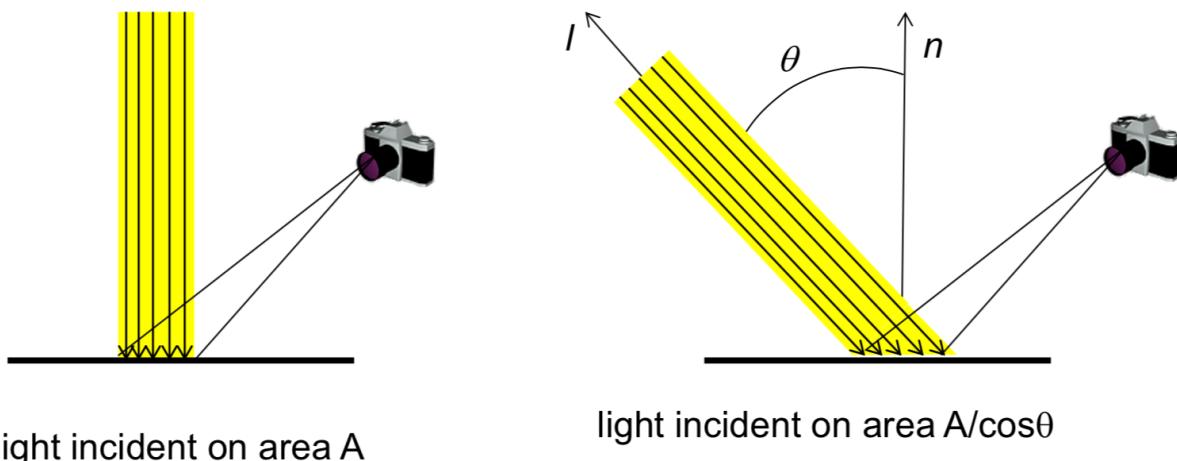
Dit aannamen is van toepassing op materialen die een ruw microscopisch oppervlak hebben, o.a. verf, papier,... In dit reflectie model zijn twee opmerkingen zeer belangrijk:

- De shading hangt niet af van de plaats van de camera!
- De shading is afhankelijk van de positie van de lichtbron!

Door gebruik te maken van deze twee regels is het meestal makkelijk om het gebruikte shading model te achterhalen.

Hoe kan men de diffusieve reflectie nu begrijpen? Het eerste ingrediënt hebben we reeds aangehaald, de reflectie is enkel afhankelijk van de hoeveelheid licht die invalt op het punt en niet van de uitgaande richting. Er ontbreekt dus enkel nog de berekening van de hoeveelheid invallend licht.

Door gebruik te maken van onderstaande afbeelding ziet men dan de hoeveelheid invallend licht afhangt van de hoek waarmee het licht invalt.



We zien dat voor een bepaalde intensiteit van de lichtstraal de energie over een groter oppervlak wordt verdeeld wanneer de lichtstraal onder een hoek invalt op het lichaam. De oppervlakte afhankelijkheid wordt beschreven is evenredig met de cosinus van de invallende richting en de normaal van het lichaam. Samen met een aantal materiaal en licht afhankelijke coëfficiënten wordt de intensiteit van het licht beschreven door volgende formule:

$$L = \frac{k_d}{\pi} c_r L_{light} c_{light} \cos \theta$$

Hierbij hebben we de volgende verklaring voor de constante:

- k_d = reflectie constante van materiaal (in $[0,1]$ 0 betekend niet reflecterend, 1 betekend perfecte reflectie) De deling door pi is voor een normalisatie.
- c_r = RGB waarden voor het oppervlak.(tip: zet nooit een waarde op nul, dit zorgt dat er geen reflectie voor die kleur mogelijk is. Dit geeft vaak aanleiding tot onrealistische effecten.)
- L_{light} = Intensiteit van de lichtstraal
- c_{light} = Kleur van het licht.
- θ = Hoek waarmee het licht invalt.

Merk op dat L een vector is met 3 componenten, elke van de componenten stelt een waarde van het RGB drietal voor. Bemerk dat de cosinus factor kan vervangen worden door het vectorieel product van de lichtstraal en de normaal vector te nemen en nadien te delen door de beide lengtes, op deze manier generaliseert de definitie makkelijk naar hogere dimensies.

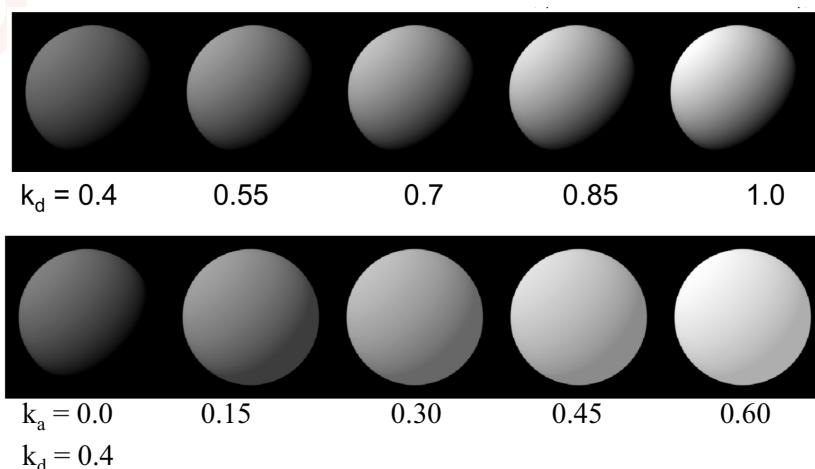
Het uitrekenen van een diffuse schaduw is niet zo moeilijk.

- Traceer een straal door de gewenste pixel.
- Vind de dichtste intersectie met een voorwerp in de scène.
- Bereken de shading in het intersectie punt
 - Hierbij moet er een lichtstraal worden berekend van elke lichtbron die aanwezig is.
 - De som van het genormaliseerde inproduct van alle lichtstralen is dan de bijdrage van de shading in dat punt.

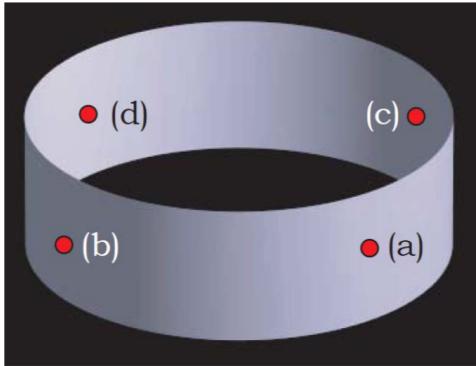
Wanneer enkel een diffuse schaduw wordt gebruikt zijn de waarden waarvoor de cosinus 0 of negatief is steeds volledig zwart. Dit geeft meestal aanleiding tot onrealistische beelden. De oplossing voor dit probleem is door het toevoegen van een *Ambient* term. Dit is een constante die bij het diffuse model wordt opgeteld. Deze constante stelt een vaste achtergrond belichting voor. De formule voor de shader word dan,

$$L = k_a c_r L_{ambient} c_{ambient} + \frac{k_d}{\pi} c_r L_{light} c_{light} \cos \theta$$

Om de invloed van deze term beter te begrijpen zijn hieronder twee voorbeelden van de verschillende shaders gegeven:



Met het diffuse schader model kunnen een aantal problemen optreden. Om te beginnen moet de normaal vector goed bepaald worden. Een aantal veel voorkomende fouten is door de richting van de normaal vector naar de foute kant te nemen. Dit zorgt voor een foute reflectie. Een tweed probleem is bij de bepaling van de lichtstralen. Zo moet er voor elke lichtbron telkens getest worden dat de lichtstraal het oppervlak bereikt. Dit laatste probleem is voorgesteld in volgend figuur:



Op punt d zou geen belichting mogen vallen.

Een ander probleem dat kan optreden is color overflow. Dit wordt opgelost door een goede mapping van de kleur naar het scherm te maken.(Besproken in les 1)

I. DRIE TYPES LICHT

In de slides staan drie types van lichtbronnen vermeld:

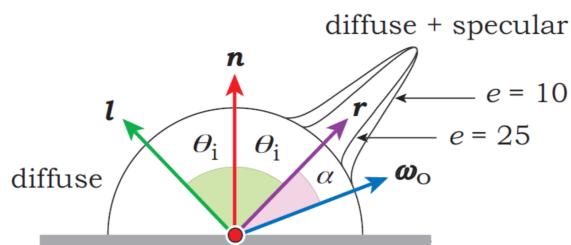
- punt lichtbron, dit is een lichtbron die in elke richting evenveel licht uitstraalt. Let op dat de intensiteit van het licht daalt met een factor $1/r^2$. Dit komt volgt uit het behoud van energie. Door elke bol schil die rond de lichtbron wordt getrokken moet een even grote flux gaan.
- Een directionele lichtbron, dit is een lichtbron die licht in 1 enkel richting stuurt. De intensiteit neemt meestal niet af naarmate de afstand groter wordt. Dit licht is het best te vergelijken met een laser.
- Ambient licht, dit is een constante achtergrond belichting die niet van ergens vandaan komt.

De bovenstaande voorbeelden zijn enkel voorbeelden van punt lichtbronnen. Meer geavanceerde lichtbronnen bestaan ook.

II. HET PHONG MODEL (VRAAG OPGELOST ZOAL AANGEGEVEN OP TOLEDO)

- I. Uiteraard starten met verantwoording: waar past het in, waarom nodig?

Het diffuse model geeft resultaten die onafhankelijk zijn van de positie van de camera. Dit is niet voldoende om realistische beelden te genereren. Hiervoor is een extra model nodig. Het meeste gebruikte model voor dit soort effecten te kunnen genereren is het *phong shading* model. Deze shader wordt gebruikt om highlights in op de figuren aan te brengen.



II. Wiskundige formulering

Bij een phong shading willen we het effect bekomen dat er bepaalde punten een glossy reflectie vertonen. Dit doen we door te kijken naar hoeveel de kijkhoek afwijkt van de reflectie hoek. Wanneer de kijk hoek dicht bij de reflectie hoek licht wensen we een sterke bijdrage te hebben van de shader. Wanneer de hoek tussen de reflectie en de kijk richting groot is dan verwachten we geen reflectie waar te nemen.

De manier om dit effect wiskundig te formuleren is door de intensiteit te moduleren met een functie die sterkt afneemt wanneer de hoek groter wordt.

In het geval van het phong shading model wordt er gebruik gemaakt van een $\cos^e(\theta)$. Door de waarde van e aan te passen kan de breedte van de reflectieve zone worden gewijzigd.

$$L = k_s (\cos \alpha)^e c_s L_{light} c_{light}$$

III. Voorbeeld van hoe een bol bv. eruit zou kunnen zien belicht met Phong shading (maak een schets) + invloed verschillende parameters.

Het volledige model voor de shading dat we tot hier toe hebben besproken kan berekend worden met volgende formule:

$$L = k_s \cos^e(\alpha) c_s L_{light} c_{light} + k_a c_r L_{ambient} c_{ambient} + \frac{k_d}{\pi} L_{light} c_{light} \cos(\theta)$$

Met parameters:

- k_s : een reflectie waarden voor phong model \in [0,1]
- e : parameter dat de breedte van de reflectie kegel bepaald (Hoe groter hoe smaller, e.g. $e=25$)
- c_s : Kleur van phong reflectie. (3\times 1 vector voor elke rgb waarde.)
- L_{light} : Intensiteit van het invallende licht.
- c_{light} : Kleur van het licht (3\times 1 vector voor elke rgb waarde.)
- k_a : een reflectie waarden voor ambient model \in [0,1]
- c_r : Kleur van ambient reflectie
- $L_{ambient}$: Sterkte van ambient licht.
- $c_{ambient}$: Kleur van ambient licht (3\times 1 vector voor elke rgb waarde.)
- k_d : een reflectie waarden voor diffuse model \in [0,1]
alpha is de hoek tussen de reflectie en de kijk richting en theta is de hoek tussen de normaal en de belichting.

IV. Hoe past het in Ray Tracing?

Zelfde als bij diffuse model.

V. Kwaliteit? Het 'ziet er goed uit', maar beantwoordt niet aan fysische werkelijkheid ... verklaring hiervoor.

Geeft een goed resultaat maar is niet conform de behoudswetten van de fysica.

VI. Bespreking eventuele computationale problemen met model: negatieve cosinus factoren; grote machten van n; meerdere lichtbronnen ...

SHADING VAN EEN POLYGON

Nu we het volledige shading model hebben uitgewerkt rest er nog 1 onbekende die we niet hebben uitgeklaard. *Hoe bereken we de normaal van een polygoon mesh?*

De normaal vector van een driehoek bepalen is niet moeilijk. Dit komt overeen met de normaal vector bepalen van het vlak waar de driehoek is in gelegen. Wanneer men echter gewoon gebruik gaat maken van de normaal van elke driehoek kan er een hoekig effect optreden door dat het oppervlak niet continue is voorgesteld door de polygonen mesh.

Om een zachter verloop van de shading te bereiken zijn er een aantal technieken mogelijk.

De eerste techniek is door de normaal van de polygon te bepalen samen met de normaal van de omliggende polygonen. In elk punt wordt dan de normaal bepaald door een interpolatie van de omliggende polygonen. Deze methode heeft *phong* interpolatie.

Een tweede techniek is door de lichtsterke te bepalen op de hoekpunten van de polygonen. Om nadien de lichtsterkte over heel de polyonen te bepalen worden de waarden op de hoeken geïnterpoleerd. Deze methode heet *Gouraud* interpolatie.

Merk op dat er bij een Gouraud interpolatie impliciet gebruik wordt gemaakt van barycentrische coördinaten. Een derde toepassing van barycentrische coördinaten dat later besproken wordt is de interpolatie van een lookup shader.

VII. LEG HET VOLLEDIGE BELICHTINGSMODEL UIT DAT GEBRUIK MAAKT VAN EEN DIFFUSE TERM EN DE ZOGENAAMDE PHONG-TERM. BESPREEK ZOWEL MATHEMATISCHE FORMULERINGEN, VERONDERSTELLINGEN, ILLUSTREER MET VOORBEELDEN.

Les 5(Hoofdstuk 13)

I. RADIOMETRISCHE GROOTHEDEN

- Energie (Q)

De eenheid van energie is Joule. Het geeft aan hoeveel fotonen er gedetecteerd zijn over een bepaald oppervlak en een bepaalde tijd.

- Vermogen (P)

De afgeleide van de energie naar de tijd geeft het vermogen. In radiometrie wordt de vermogen ook vaak verwisseld met de radiant flux. De eenheid is Watt = joule/seconde.

- Irradiance (E)

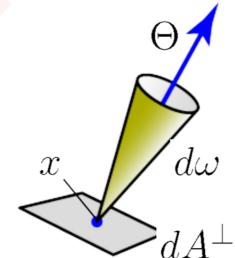
Dit is het vermogen per oppervlakte eenheid. De eenheid is dus W/m².

Enkele eigenschappen van radiantie zijn:

- De radiantie op het pad van een straal die door empty space gaat is constant.
- Radiantie kan eerder waar worden gedefinieerd niet enkel op het oppervlak van een lichaam.
- Op een punt op een oppervlak is de radiantie niet beïnvloed door de richting van de flux. Het maakt niet uit of de stralen toekomen op het oppervlak of worden gereflecteerd.

De radiantie is de belangrijkste grootheid in de rendering vergelijking.

Het begrip radiantie kan op een zeer inzichtelijk manier worden voorgesteld aan de hand van bijgevoegde afbeelding. Op elk punt in de ruimte en voor elke richting is de radiantie gedefinieerd als het aantal fotonen (energie) dat door een infinitesimale cirkel gaat die loodrecht op de richting staat. Dit verduidelijkt de definitie van de radiantie, het is de derde afgeleide van het aantal fotonen naar de tijd, naar de ruimte hoek en naar het oppervlak A.



Wanneer er een lichtstraal invalt onder een hoek op een bepaald oppervlak dan is het inproduct van de richting van de straal en de normaal vector van belang in de berekening van de radiantie. De radiantie in dat punt op het oppervlak wordt dan gegeven door

$$L(x \rightarrow \Theta) = \frac{d^2 P}{dA \cos \theta d\omega_\Theta}$$

Het feit dat radiantie invariant is bij een straal die in free space beweegt maakt dat deze grootheid zo belangrijk is bij raytracing algoritmes. Wanneer een primaire straal invalt op een punt op een lichaam zal de waarden van die straal gelijk zijn aan de radiantie in dat punt in de richting van de primaire straal. In een raytracing algoritme komt het er dus op neer om op elk punt in de scène de radiantie zo efficiënt mogelijk te kunnen berekenen. Nadien worden de waarden van de radiantie dan geprojecteerd op het view plane door middel van de primaire stralen (En mogelijk nog wat AA stralen).

BIDIRECTIONAL REFLECTANCE DISTRIBUTION FUNCTION (BRDF)

In computer graphics worden de reflecterende eigenschappen van objecten vaak voorgesteld aan de hand van *materialen*. Voor elk materiaal wordt er dan de relatie gegeven tussen de invalende en de uitvallende stralen. Deze functie die volledig de reflecterende eigenschappen van het materiaal vast leggen heet de Bidirectional Reflectance Distribution Function.

- Bidirectional: De stralengang is omkeerbaar in klassieke optica.
- Reflectance: De relatie beschrijft de verhouding tussen in en uit komende stralen.
- Distribution: De functie is een verdeling (genormaliseerd), dit wijst erop dat er behoud van energie is. Er wordt evenveel stralen uitgezonden dan er op het voorwerp vallen. (Sommige

materialen absorberen in werkelijkheid licht, dit kan worden gemodelleerd door stralen in het materiaal te laten gaan.)

Nog een aantal eigenschappen van de BRDF zijn:

- Reciprocity: De stralengang is omkeerbaar.

$$f_r(p, \omega_i, \omega_o) = f_r(p, \omega_o, \omega_i)$$

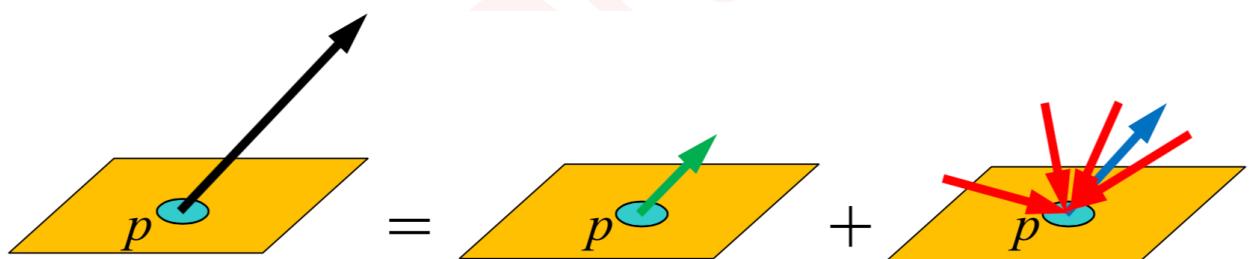
- Lineariteit. Wanneer meer dan 1 straal invalt op het oppervlak zal de reflectie de bijdrage van de twee aparte reflecties zijn.
- Behoud van energie.

$$\forall \omega_o : \int_{\Omega_i} f_r(p, \omega_i, \omega_o) \cos \theta_i d\omega_i \leq 1$$

De interpretatie van deze vergelijking is dat er niet meer energie kan worden uitgezonden in 1 richting dan wat er invalt. Merk op dat hierboven geclaimd wordt dat de integraal op 1 uitkomt. Dit is ook het geval. Men kan echter ook marginaliseren over \omega. (bivariate kans verdeling).

DE RENDERING VERGELIJKING

Nu we hebben gedefinieerd wat we bedoelen met een materiaal is de volgende stap om te beschrijven hoe de materiaal eigenschappen ons beeld in een raytracing algoritme beïnvloeden. Bij raytracing wensen voor elke punt waar een primaire straal invalt op een lichaam de radiantie in dat punt kunnen bepalen. Dit doen we door de bijdrage van de reflecties van alle invallende stralen op te tellen bij een component die het lichaam mogelijk zelf uitstraalt. Dit hebben we schematisch voorgesteld in onderstaande figuur.



Someren over alle rode pijlen komt overeen met een integraal te nemen over de ruimte hoek van mogelijk invallende richtingen. De blauwe pijl is dan gevonden door de integraal in volgende vergelijking. De groene pijl is de constant bijdrage L_e.

$$L_o(p, \omega_o) = L_e(p, \omega_o) + \int_{\text{hemisphere}} f_r(p, \omega_i, \omega_o) L_i(p, \omega_i) \cos \theta_i d\omega_i$$

In bovenstaande vergelijking is $f_r(p, \omega_i, \omega_o)$ de BRDF, $L_i(p, \omega_i)$ is de irradiatie die invalt onder de hoek ω_i .

De uitdaging bij het maken van efficiënte raytracing algoritmes bestaat grotendeels in het vinden van manieren om de rendering vergelijking snel en accuraat te kunnen oplossen. Een methode die we hiervoor zullen bespreken is een monte carlo simulatie. Een tweede methode is door middel van vereenvoudigingen door te voeren.

De twee shadres die voorheen werden besproken passen ook perfect in dit kader. Voor de diffusieve reflectie (*Lambertiaan*) word $f(p, \omega_i, \omega_o)$ constant genomen.

Voor de phong reflectie is er gekozen om als BRDF functie een scherp gepiekte functie te nemen rond de invallende hoek w_i . Ik ben niet zeker of de phong reflectie reciprook is en of ze genormaliseerd is.

Voor een perfecte spiegel word er een dirac functie gekozen als RBDF.

MONTE CARLO INTEGRATIE VOOR HET OPLOSSSEN VAN DE INTEGRAAL.

Een monte carlo integratie is een manier om de waarden van een integraal te schatten door een aantal discrete punten uit te rekenen en daarna de gewogen som van de evaluatie waarden te nemen.

In deze samenvatting worden enkel een paar korte resultaten aangehaald die van belang zijn:

- De variantie (of de fout) op het resultaat is evenredig met de wortel van het aantal samples.
- Importance sampling is een methode om de variantie met een constant factor naar beneden te halen (convergentie orde blijft gelijk). Het idee is om niet volledig random te sampelen maar om een sample distributie te kiezen die goed aansluit bij de vorm van de functie in de integraal. (Voor een phong model is het dus best om meer te samplen rond de reflectie van de invallende straal). Om dit statistische te kaderen wordt er dus een prior distributie gebruikt om zo de efficiëntie van de schatter te verhogen

SIMPLE STOCHASTIC RAY TRACING

Nu we de materiaal eigenschappen hebben gedefinieerd aan de hand van de BRDF en we weten hoe de integraal moet worden opgelost rest ons enkel nog om de invallende stralen te kunnen berekenen, deze bijdrage was gegeven door de term $L_i(p, w_i)$. In onderstaande formule staat de benadering van de integraal aan de hand van een monte carlo simulatie.

$$\approx \frac{1}{N} \sum_{j=1}^N \frac{f_r(p, \omega_j, \omega_o) L_i(p, \omega_j) \cos \theta_j}{p(\theta_j)}$$

Voor elke random gekozen hoek(Invallende straal) is $p(\theta_j)$ de kans op deze straal volgens onze kans verdeling en (voor variantie reductie) en f_r is gegeven. Het is dus duidelijk dat de waarde $L_i(p, w_j)$ de enige onbekende nog is in dit proces.

Deze $L_i(\theta)$ is op zijn beurt een oplossing van de rendering vergelijking in een ander punt in de scene. Dit brengt ons bij het *recursief* oplossen van de rendering vergelijking.

Elke niveau van recursie voegt een nieuwe orde van de reflectie toe. Wanneer we bijvoorbeeld geen recursie toe laten zal er enkel een bijdrage van de lichtbronnen worden gebruikt. Voor een eerste orde recursie zal er ook een bijdrage zijn van indirecte verlichting (een typische schijn op een muur of zo iets). Wanneer er nog meer recursie wordt gebruikt kunnen er nog meer effecten worden toegevoegd.

In het laagste level van de recursie word enkel de bijdrage van de lichtbronnen berekend.

Bij de recursieve evaluatie zijn er een aantal vrijheidsgraden die de kwaliteit en de rekentijd van het algoritme sterk beïnvloeden; hoeveel samples per monte carlo integraal, hoeveel AA samples, hoeveel lagen van recursie?

De meeste gebruikte methode om te antwoorden op de vraag van hoe diep de recursie moet zijn is door gebruik te maken van de binomiale verdeling. Elke keer wanneer er een nieuwe recursie moet worden uitgevoerd word er een binomaal experiment gedaan met een bepaalde kans op succes. Wanneer het antwoord 1 is gaat de recursie verder. Is het antwoord 0 stopt ze. Een tweede stop conditie is wanneer een lichtstraal invalt op een lichtbron. Er wordt dan aangenomen dat de bijdrage uitsluitend door het licht zelf is en niet door reflecties op het licht.

II. BESPREEK DE RENDERING-VERGELIJKING.

- III. WELKE VEREENVOUDIGINGEN WORDEN DOORGEVOERD IN DE RENDERING-VERGELIJKING OM TOT EEN STANDAARD RAY-TRACING ALGORITME, DAT GEBRUIK MAAKT VAN PHONG-SHADING, TE KOMEN?**
- IV. BESPREEK HET VERSCHIL TUSSEN DE HEMISFERISCHE- EN DE OPPERVLAKTE-FORMULERING VAN DE RENDERING VERGELIJKING.**

Les 6(Hoofdstuk 16-18)

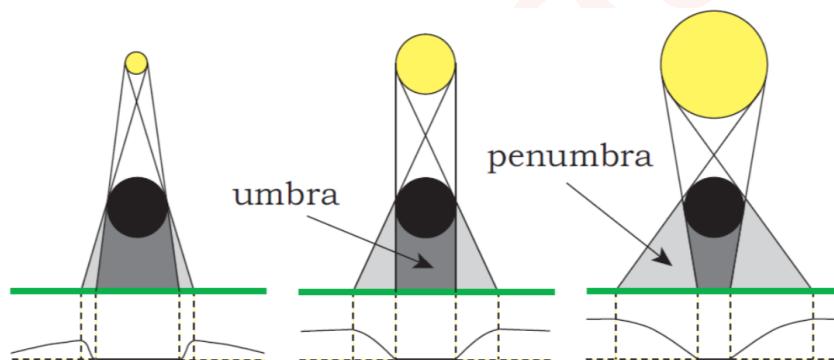
BELANG VAN SCHADUWEN

Er zijn een aantal redenen waarom schaduwen van belang zijn bij het renderen van 3D beelden. Ze werpen bijvoorbeeld licht op onderstaande vragen.

- Hoe ver zijn de object van de grond verwijderd?
- Wat is de relatieve afstand van de objecten tot de camera?
- Hoeveel lichtbronnen zijn er aanwezig in de scène?
- Welke soort van lichtbronnen?

ENKELE DEFINITIES ROND SCHADUWEN

Een schaduw is een plaats waar het licht van een bepaalde licht bron niet kan komen. Vanuit een schaduw kan men de licht bron dus niet zien. Een schaduw van een puntbron zijn schaduwen met een harde rand. Voor lichtbronnen met een eindige afmeting is het iets moeilijker om te zeggen wat de schaduw is en wat niet, de lichtbron kan dan deels bedekt zijn. De schaduw veroorzaakt door een lichtbron met een eindige afmeting heet een zachte schaduw. Hierbij zijn er twee delen waartussen onderscheid wordt gemaakt, de *umberella* en de *penumbra*. In de zone van de umberella is de lichtbron volledig verduisterd, in de penumbra is ze deels verduisterd. De twee zones zijn voorgesteld in onderstaand figuur.



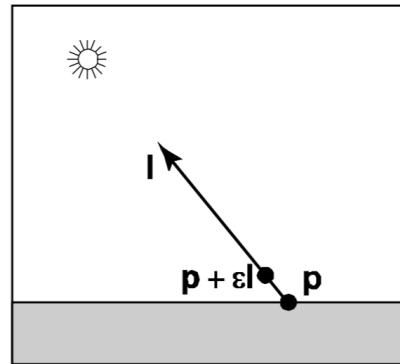
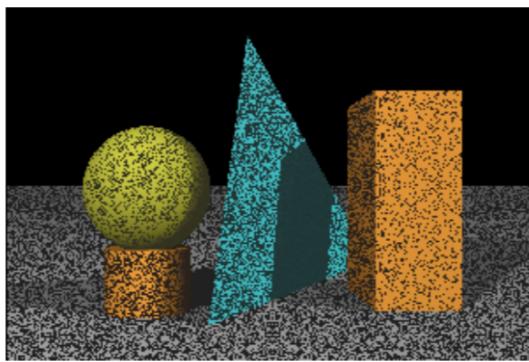
In sommige gevallen kan er nog een derde geval zone ontstaan, de *antumbra*, dit is de zone waar terug licht kan komen dat rond het object heen is gegaan.

IMPLEMENTATIE VAN SCHADUW

De meest eenvoudige implementatie voor het bepalen van een schaduw is aan de hand van *schaduw stralen*. Dit zijn stralen die vertrekken vanop het voorwerp en lopen in de richting van de lichtbron. Wanneer er een intersectie is met de schaduw straal en een ander object dan valt het voorwerp in de schaduw. Met test met andere woorden of de lichtbron zichtbaar is van op het punt of niet.

Deze eenvoudige implementatie heeft nog een probleem, er moet ook nog gecontroleerd worden of het object tussen de lichtbron en het punt in kwestie licht. Als dit niet wordt gedaan kunnen er vreemde effecten optreden.

Er kan ook nog een tweede probleem optreden bij deze naïeve implementatie, dit is gekend als *self intersection*. Hierbij word het object waarop het punt ligt gezien als een object dat in de weg staat. Het eigen object voldoet namelijk aan alle criteria om een schaduw te kunnen voorzien. Dit probleem kan worden opgelost door een bepaalde waarde \epsilonpsilon te definiëren waarmee we het originele punt richting de lichtbron verschuiven. Op deze manier is er geen zelf doorsnijding meer. Deze methode is geïllustreerd in de afbeelding hier onder.

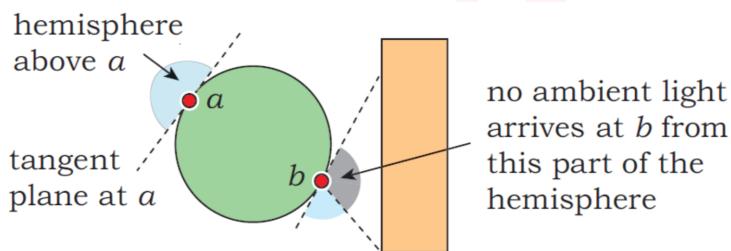


Links, voorbeeld van zelfsnijding. Rechts, voorstelling van de epsilon methode

AMBIENT OCCLUSION

Ambient occlusion is een techniek om snel schaduwen te berekenen in een scène. Het idee is dat de achtergrond een constante licht intensiteit uitstuurt. Punten waarin de achtergrond meer zichtbaar is zullen op deze manier meer licht opvangen.

Punten die afgeschermd liggen door andere voorwerpen zullen dan grotendeels in de schaduw liggen. Dit idee is verder geïllustreerd in onderstaande afbeelding.



Voor de wiskundige formulering achter ambient occlusie vertrekken we van de rendering vergelijking.

$$L_o(p, \omega_o) = L_e(p, \omega_o) + \int_{\text{hemisphere}} f_r(p, \omega_i, \omega_o) L_i(p, \omega_i) \cos \theta_i d\omega_i$$

In deze vergelijking worden twee vereenvoudigingen doorgevoerd. De eerste vereenvoudiging heeft betrekking tot $L_i(p, w_i)$ deze functie is L_i wanneer de achtergrond zichtbaar is in de richting w_i en 0 wanneer dit niet het geval is.

De tweede vereenvoudiging die wordt doorgevoerd is f_r constant te nemen. Dit komt overeen met het diffuse shader model (*Lambertiaan*). De uiteindelijke vorm van de vereenvoudigde rendering vergelijking wordt dan,

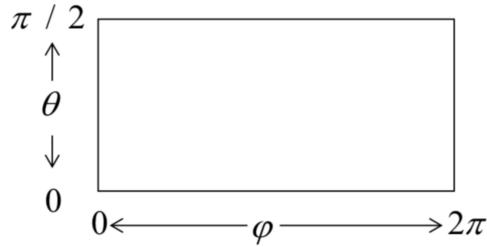
$$L_o(p, \omega_o) = f_r \cdot L_i \int_{\text{hemisphere}} V(p, \omega_i) \cos \theta_i d\omega_i$$

De enige term die nog moet worden uitgewerkt in de integraal is de functie $V(p, w_i)$. Dit is de term die aan geeft of de achtergrond zichtbaar is in de richting w_i .

BEPALING VAN DE OCCLUSION TERM

Voor de occlusion term te bepalen moet de integraal over de hele hemisfeer worden opgelost. Zoals in het vorige hoofdstuk aangehaald kan dit met behulp van een monte carlo simulatie.

Hier voor moet er een sampling over de hemisfeer worden uitgevoerd. Bij deze sampling moet er voorzichtig worden omgesprongen met de keuze van de ruimte hoeken. Wanneer beide hoeken uniform gesampled zouden worden is de verdeling over de hemisfeer niet uniform (Vervorming door projectie van het vlak op de sfeer). Dit kan worden opgelost door de hoek φ uniform te samplen en nadien een tweede variabele x uniform te samplen en te transformeren met $\theta = \text{sqrt}(x)$. Op deze manier wordt er minder gesampled rond theta 0.



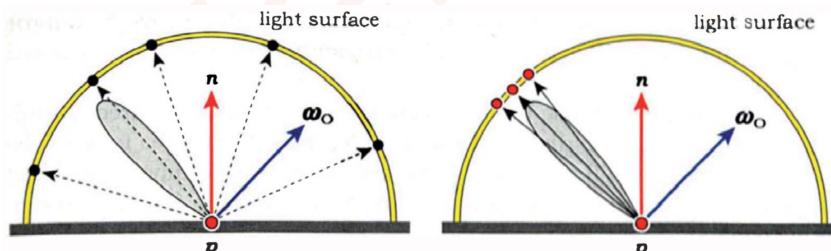
Er kan op een reguliere manier worden gesampled maar dit geeft zeer makkelijk aanleiding tot ongewenste effecten in het beeld.

Een groot nadeel van ambient occlusion is dat de methode niet lokaal is. Wanneer er in de verte iets veranderd aan de scène kan de hele belichting van de scène veranderen. Dit kan mogelijk worden opgelost door enkel te kijken binnen een beperkte straal naar objecten die mogelijk in de weg staan.

OPPERVLAKTE LICHTEN

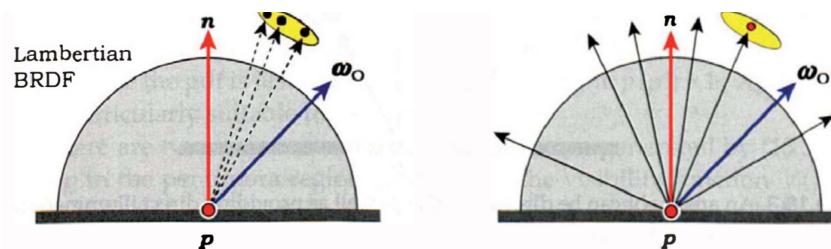
Voorlopig hebben we ons enkel toegespitst tot twee soorten lichten, de puntbron en ambient occlusion. In de echte wereld zijn lichte nooit (of zelden) te plaatsen in 1 van de vorige categorieën, maar hebben lichten een eindige afmeting. Dit geeft ook aanleiding tot zachte schaduwen.

Voor de beschrijving van oppervlakte lichten vertrekken we terug van de rendering vergelijking.



Voor het oplossen van de rendering vergelijking moet er op een bepaalde manier gesampled worden. In beide figuren is ω_o de kijk richting. De grijze zone is de aanduiding voor de BRDF, de gele halve cirkel is de lichtbron.

In de linker figuur wordt er uniform over de lichtbron gesampled. Echter zijn er maar een beperkt aantal lichtstralen die in de grijze zones liggen (die dus een finale bijdragen hebben). In de rechter figuur wordt er uniform in de richting van de BRDF gesampled, hier zullen alle stralen een bijdrage leveren. In dit voorbeeld is de rechter methode de meest efficiënte.



Een tweede mogelijke situatie is geïllustreerd in de afbeelding hierboven. De BRDF is zeer breed maar de lichtbron is nu beperkt in afmetingen. Nu wordt de linker sampling methode (uniform over de lichtbron) meer efficiënt. Beide voorbeelden zijn en vorm van variantie reductie door middel van *importance sampling*. Wanneer welke optie van sampling het best is hangt af van de situatie.

DIRECTE BELICHTING DOOR OPPERVLAKTELICHTEN

De directe belichting is de belichting die rechtstreeks wordt gegenereerd door de oppervlaktelichten. De integratie over de hele hemisfeer kan dus worden vervangen door een integratie over het oppervlak van de lichtbron.

$$= L_e(p, \omega_o) + \int_A f_r(p, \omega_i, \omega_o) L_o(p', -\omega_i) V(p, p') G(p, p') dA_{p'}$$

In bovenstaande integraal is A het oppervlak van alle lichtbronnen, V(p,p') is de binaire functie die aangeeft of punt p' zichtbaar is vanuit punt p. ω_i is de genormaliseerde vector die p en p' verbinden met elkaar. G(p,p') is een term die aangeeft hoeveel licht de bron uitzendt in de richting tussen p en p'. Deze factor is evenredig met $(p-p')^2$ en het product van de cos van beide hoeken tussen ω_i en de normaal van de lichtbron en de normaal van het lichaam.

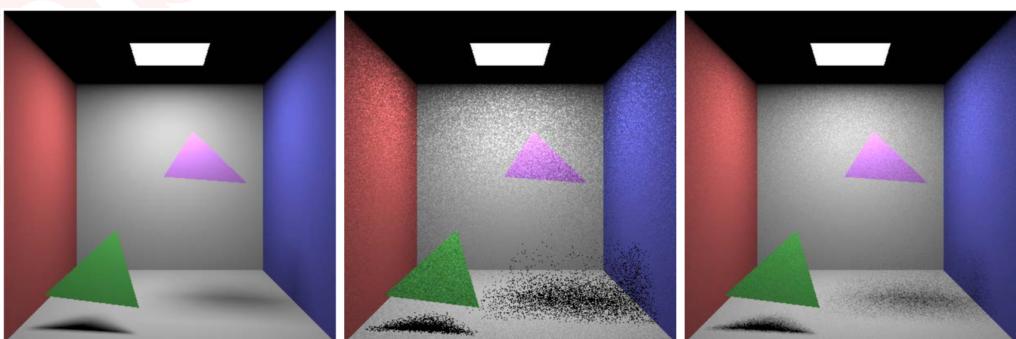
Door verder aan te nemen dat de objecten diffuus oppervlak hebben (constante f_r) en dat de lichtbron een uniforme intensiteit heeft (L_e constante) bekomen we onderstaande vergelijking.

$$= f_r \cdot L_e \int_{A_{lightsource}} V(p, p') \frac{\cos \theta_p \cos \theta_{p'}}{r_{pp'}^2} dA_{p'}$$

Door het gebruik te maken van een monte carlo simulatie word dit benaderd door,

$$\approx \frac{f_r \cdot L_e}{N} \sum_{j=1}^N \frac{V(p, p') \cos \theta_p \cos \theta_{p'}}{pdf(p') r_{pp'}^2}$$

Hierbij is $pdf(p')$ een random verdeling over de lichtbron. Dit kunnen we nu nog even terug koppelen aan de vorige discussie over de sampling techniek. In de eerste afbeelding in de rechter figuur wordt er dus een pdf gekozen met de meeste kans massa rond de BRDF. In de tweede afbeelding in de rechter methode niet conform de afleiding en zal dus nooit worden toegepast. In de onderstaande figuur is een voorbeeld gegeven van een rendering met de bovenstaande techniek. Links in het perfect gerenderd beeld, in het midden werd er 1 schaduw straal (p naar p') gebruikt, in de rechter figuur werden er 9 schaduwstralen gebruikt.



Het is duidelijk dat er twee zones zijn met veel ruis. De eerste zones is diegene waar men een zachte schaduw verwacht (onder de driehoeken). Dit effect treedt op door de beperkte steek proef die wordt uitgevoerd. De tweede plaats met veel ruis is bovenaan de muren. De oorzaak hier is dat

de afstand tussen de lichtbron en de muur sterk afhangt van het gekozen punt op de bron. Een tweede grootheid die sterk varieert is de cosinus van de hoek waarmee het licht de bron verlaat. Als afsluiter voor dit deel bespreken we twee keuzes die de gebruiker kan maken.

- 1) Veel schaduw stralen maar weinig AA stralen.
- 2) Een beperk aantal schaduw stralen maar een groot aantal AA.

In de slides staan twee voorbeeld renderingen uitgewerkt. Wanneer er geen AA wordt gebruikt en 100 schaduw stralen per pixel worden gebruikt komt het totaal aantal stralen per pixel neer op 101. Wanneer er slechts 1 schaduw straal wordt gebruikt maar 100 AA secundaire stralen dan zijn er in totaal 200 stralen per pixel gebruikt. Uit de slides blijkt dat beide beelden dezelfde kwaliteit hebben. Het is in dit geval dus beter om meer schaduw stralen te gebruiken. Het is wel zo dat de aliasing effecten niet worden verholpen.

V. LEG HET BEGRIP AMBIENT OCCLUSION UIT. WELKE BENADERINGEN WORDEN GEMAAKT IN VERGELIJKING MET HET EXACT BEREKENINGEN VAN SCHADUWEN? OP WELKE MANIEREN KUNNEN DEZE BANDAERINGEN ZICHTBAAR WORDEN IN HET BEELD?

VI. OP WELKE MANIER WORDT DE BELICHTING A.G.V. EEN OPPERVLAKTE-LICHTBRON BEHANDELT IN RAY TRACING?

VII. (GEGEVEN EEN SCÈNE SETUP MET EEN OPPERVLAKTELICHTBRON). WAAR IN HET BEELD ZAL JE MEER OF MINDER OF RUIS KUNNEN VERWACHTEN? DOOR WELKE FACTOREN WORDT DEZE RUIS VEROORZAAKT?

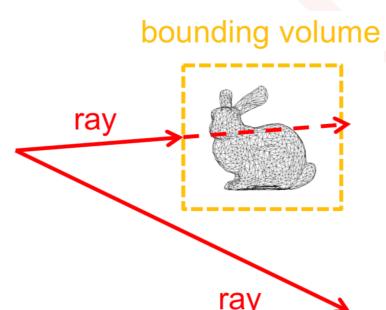
Les 7

KOST VAN RAYTRACING

Bij een naïeve implementatie van een raytracing algoritme moet men elke straal testen op een intersectie met elk voorwerp in de scène. Dit kan snel heel duur worden. Een beter aanpak om is om een manier te verzinnen om het aantal intersectie berekeningen te minimaliseren. Dit wordt is het doel van een acceleratie structuur voor raytracers. Er zijn veel mogelijke manieren om dit te bereiken.

BOUNDING VOLUMES ALS ACCELERATIE STRUCTUUR

Het idee achter bounding volumes is om delen van de scène op te splitsen en in een apart volume te plaatsen waarmee eenvoudig een intersectie kan worden gevonden. Wanneer een straal het bounding volume niet raakt dan moet er niet getest worden of er een mogelijke intersectie is met de inhoud van de bounding box en de straal. Een eenvoudige versie van een acceleratie algoritme op basis van BV is als volgt:



Test de straal voor een intersectie met de BV

wanneer er een intersectie is:

 Test op intersectie met voorwerp in BV

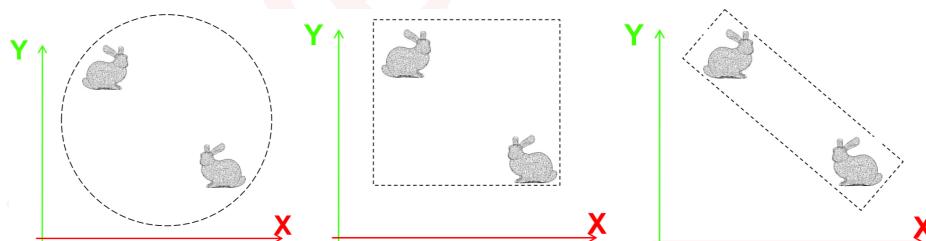
Geen intersectie

 Test op intersecties met andere BV(niet gelegen in de originele BV)

De kost voor een intersectie te berekenen met een BV moet klein genoeg zijn voor een efficiënte acceleratie structuur. Een minimale (logische) voorwaarden moet zijn dat,

$$\text{cost_BV} + \text{cost_group} * \text{probability_ray_hits_BV} < \text{cost_group}$$

Een klassiek probleem bij het ontwerpen van deze acceleratie structuur is de afweging tussen een goedkope of een meer aansluitende BV te construeren. Enkele voorbeelden zijn

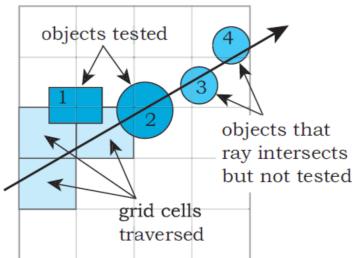


(1) Bol BV, (2) AABB axis aligned bounding box,(3) OBB Oriented bounding boxes

- Bol BV
 - Makkelijk voor intersecties, niet altijd aansluitend
- AABB
 - Makkelijk voor intersecties meestal beter aansluitend dan bol, de intersectie bepalen kan gewoon op basis van de coördinaten van de staal te bekijken.
- OBB
 - Extra rotatie kost, goed aansluiting voor random voorwerpen.

REGULIERE GRID METHODE

De meest eenvoudige methode voor de constructie van bounding boxen is door gebruik te maken van een regulier grid. Hierbij delen we de scène op in reguliere sectoren. In elke sector wordt een referentie bijgehouden naar de objecten die in die sector gelegen zijn. Met deze methode wordt er geen rekening gehouden met de dichtheid van de objecten. Het is dus niet uitgesloten dat bepaalde objecten in meerdere cellen liggen. Een voorbeeld van een mogelijk resultaat dat bekomen kan worden door gebruik te maken van de regular grid verdeling is gegeven in de figuur rechts van deze paragraaf. In de cursus is een voorbeeld gegeven voor de versnelling die kan bekomen worden met deze methode, de versnelling is voornamelijk van belang bij veel objecten.



Voor dit deel over het regulier grid af te sluiten zijn er nog een paar vuistregels die gebruikt kunnen worden bij de implementatie:

- Neem het aantal onderverdelingen evenredig met het aantal voorwerpen in de scène. In het voorbeeld is er geopteerd om 8 keer minder grids te gebruiken dan dat er voorwerpen zijn.
- Controleer of de gevonden intersectie gelegen is binnen de cell. Wanneer dit niet het geval is dan zal er nog een andere cell zijn waarbinnen de intersectie gelegen is. Dit heet een valse snijding. De straal snijdt dan wel met de het voorwerp in de bounding box maar niet in het BV.
- Vermijd het meervoudige te hoeven controleren van een intersectie. Een object kan bijvoorbeeld tot een aantal BV's behoren. Dit kan worden opgelost door de laatste berekening van de intersectie op te slaan in een het object. Wanneer er dan opnieuw wordt gecontroleerd op een snijding kan dit resultaat gebruikt worden. (Let op, sla maar 1 resultaat op anders wordt het te geheugen intensief). Dit heet het mailbox idee.

Plus en minpunten voor deze methode:

- + Even efficiënt voor elke straal
- - Gebruikt veel geheugen.
- - Werkt niet goed met onregelmatige of zeer grote voorwerpen.
- - Veel extra werk, elke straal moet door veel vakjes worden getraced.

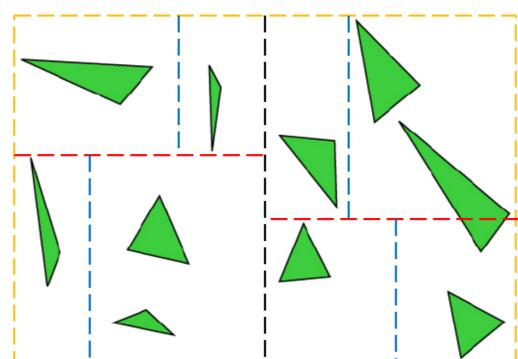
HIËRARCHISCHE BOUNDING VOLUMES

Er zijn twee eenvoudige concepten voor het vinden van goede bounding boxen, de *bottom-up* en de *top-down* methode. Bij de bottom-up methode word er een individuele bounding box rond elk voorwerp in de scène geplaatst nadien worden naburige bounding volumes samen genomen tot groteren bounding boxen. De top-down methode werkt omgekeerd. Er wordt vertrokken van een grote bounding box die heel de scène omvat. Deze word nadien opgesplitst in kleinere volumes. In deze twee methode blijft er nu nog 1 onbekende. *Hoe worden de bounding volumes samen genomen of opgesplitst?*

HIËRARCHISCHE RUIMTE ONDERVERDELING

Een meer geavanceerd idee voor de constructie van een BV is door op een iteratieve wijze de ruimte op te delen in steeds kleinere BV's. De meest weid verspreide methode die dit principes hanteert is die van de *kd-trees*, hierbij wordt de ruimte opgedeeld in as gealineerde vlakken. Andere mogelijke structuren zijn o.a. *BSP-tree*, *Octrees*, *Tetrahedralizations* ...

Een voorbeeld van 4 iteraties met deze methode is gegeven in de figuur rechts van dit alinea. De eerste BV is de gele, daarna zwart, rood en blauw.



De vraag die nog steeds blijft is hoe de optimale subdivisies kunnen worden gevonden. Intuïtief wenste men in elke subdivisie het aantal voorwerpen te halveren. Door naïef de BV steeds in de twee op te delen zal men meestal dit optimum niet bereiken dit is het gevolg van een aantal problemen:

- Objecten zijn meestal niet mooi gelokaliseerd maar kunnen zeer groot zijn.
- Er kan veel lege ruimte zijn.
- Delen we een dichte verzameling van objecten in één BV of maken we verschillende aparte BV voor deze situatie?

Voor een antwoord op deze vragen te geven kan bijvoorbeeld de surface area heuristiek worden gebruikt.

SURFACE AREA HEURISTIEK

voor de surface area heuristiek trachten we de kost van de intersecties te minimaliseren. De kost kan op een recursieve manier worden bepaald aan de hand van de volgende formule,

$$Cost(cell) = C_t + prob(hit_L).Cost(L) + prob(hit_R).Cost(R)$$

De term C_t is een vaste cost voor de cell te intersecteren, L en R zijn twee dochter cellen die we trachten optimaal te kiezen. In bovenstaande vergelijking is de grote onbekende de kans op een botsing met cell L of cell R. De kans dat een 1 van de cellen wordt geraakt is proportioneel met het gemiddelde geprojecteerde oppervlak op de moeder cell. Voor deze kans te schatten kan er een gebruikt men het *Crofton's theorema*, dat stelt dat

“For a convex object, the average projected area equals $\frac{1}{4}$ of the surface area”

Door gebruik te maken van dit theorema bekomen we de volgende benadering voor de kost van het intersecteren van een cell.

$$Cost(cell) = C_t + \frac{S_L}{S_{cell}} \cdot N_L \cdot t_i + \frac{S_R}{S_{cell}} \cdot N_R \cdot t_i = C_t + (S_L \cdot (N_L - N_R) + S_{tot} \cdot N_R) \cdot \frac{t_i}{S_{cell}}$$

Hierin staan $S_{_}$ voor de respectievelijke oppervlaktes van de verschillende cellen, $N_{_}$ het aantal elementen per cell en t_i de kost per intersectie van elk element. De term tussen de haken in het linkse deel van de vergelijking is een monotone functie in de parameter N_r (dit is duidelijk na substitutie $N_l = N - N_r$). Dit houdt in dat het minimum van de cost gegeven wordt door een vlak dat aansluit bij één van de objecten in de scène. Wanneer er n voorwerpen zijn in de cell zijn er dus $2n$ mogelijke kandidaten van snijding. Nu volgen er enkele problemen die volgen uit het bepalen van de optimale snijding.

probleem 1:

De cost voor de deel cell is proportioneel verondersteld met het aantal elementen in die cell. Dit is echter een overschatting, de cellen worden opnieuw opgedeeld.

Alhoewel deze veronderstelling steeds te hoog is geeft het algoritme toch goede resultaten.

probleem 2:

Wat zijn de stop condities van deze heuristiek? Een logische stop voorwaarden is dat we stoppen wanneer de kost na een splitsing groter is dan zonder een splitsing.

Hier is echter een probleem, de kost functie hoeft niet convex te zijn. Wanneer bijvoorbeeld een holle ruimte aanwezig is in het lichaam kan het zijn dat deze niet wordt gevonden door de heuristiek. Ideaal zou men dan moeten doorgaan tot de holle ruimte in een aparte BV zit.

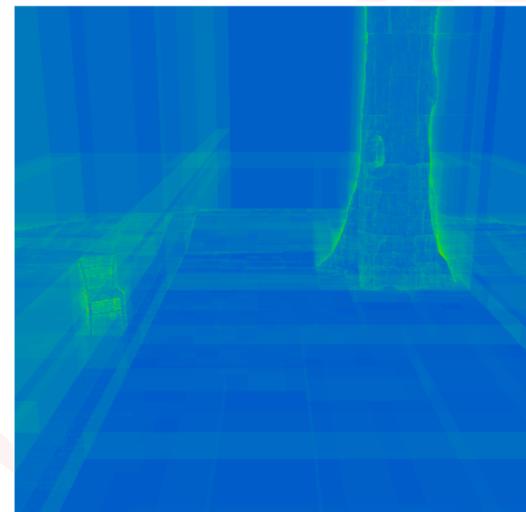
probleem 3:

De KD-boom moet steeds eerst worden opgesteld voor we kunnen beginnen aan de rendering van de afbeeldingen. De optimale kost van het opbouwen van de boom heeft een complexiteit van $N \cdot \log(N)$. Merk op dat er per object in de ruimte 6 as-gealineerde vlakken kunnen worden gevonden. (2 in elke richting).

Over de resultaten van deze heuristiek kunnen we kort zijn. Zelfs wanneer de boom steeds moet worden opgebouwd is de kd-boom methode sneller dan een naïeve rendering. De echte versnelling komt er pas wanneer de objecten in de scène niet bewegen maar wanneer bijvoorbeeld enkel de shaders worden gewijzigd. In zo een scenario moet de boom niet worden herberekend. Voor scenes waar maar kleine lokale animaties moeten worden uitgevoerd is het gebruik van een kd-boom ook aan te raden. In zo een geval kan de boom worden berekend voor de lege (achtergrond) scene. Bij de rendering wordt er dan als eerste na gegaan of de straal in de achtergrond valt of op het nieuw object (bv door BV rond object te plaatsen). Valt de straal op de achtergrond kan men gewoon de kd-boom van de achtergrond gebruiken. In het andere geval gebruikt men een andere techniek voor intersecties met het nieuwe object te bepalen, dit kan dan bijvoorbeeld ook aan de hand van een aparte kd-boom zijn.

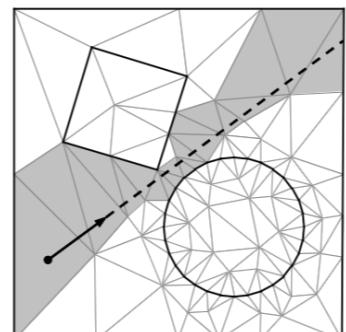
In de figuur hier links is een grafische voorstelling gemaakt voor het aantal intersectie berekeningen in functie van de pixel. Het is duidelijk dat dit aantal het grootste is aan de randen van de objecten. Hier moet er namelijk worden beslist of de straal het voorwerp al dan niet raakt.

Nog een beetje een vreemd resultaat van de kd-boom is het fijn dat het toevoegen van een object soms aanleiding kan geven tot een reductie van het totaal aantal intersectie berekeningen. Dit is het gevolg van het globale karakter van de methode en dat dit een heuristiek is en geen convex optimalisatie probleem. Een absoluut optimum wordt dus niet gevonden door de methode.



ACCELERATING RAY TRACING USING CONSTRAINED TETRAHEDRALIZATIONS (KULEUVEN)

Een methode die aan de kuleuven is ontwikkeld maakt gebruik van tetrahedrons. Het idee is om de lege ruimte te vullen met zo groot mogelijke tetrahedrons. Deze berekeningen gebeuren aan de hand van de *quality Delaunay tetrahedralization*. Een tetrahedron in de ruimte kan twee soorten randen hebben, een vrije en een bezette. Een bezette rand sluit steeds aan bij een voorwerp uit de scène. Wanneer een straal dus zo een rand tegenkomt is er een snijpunt met een object gevonden. In de figuur links van deze paragraaf is een voorbeeld gegeven van deze methode. De straal gaat door de grijze tetrahedrons. Deze methode is iets trager dan kd-boom. De voordelen zijn: Eerste hit is steeds correct, past zich beter aan aan de lokale geometrie van de scène.



- I. LEG DE SURFACE-AREA HEURISTIC UIT. ILLUSTREER MET RELEVANTE VOORBEELDEN.
- II. (GEGEVEN EEN SCÈNE, MET TWEE VERSCHILLENDEN MANIEREN VAN SPATIALE OPDELING IN EEN ACCELERATIESTRUCTUUR). WELKE ACC-STRUCTUUR ZAL BETER PRESTEREN ONDER RAY-TRACING?

Les 8(Hoofdstuk 24-26)

REFLECTIE

Voor het beschrijven van perfecte reflectie starten we terug van de rendering vergelijking.

$$L_o(p, \omega_o) = L_e(p, \omega_o) + \int_{\text{hemisphere}} f_r(p, \omega_i, \omega_o) L_i(p, \omega_i) \cos \theta_i d\omega_i$$

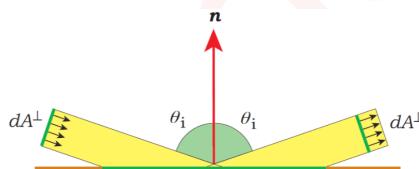
In het geval van een perfecte reflectie kunnen we een gereflecteerde straal definiëren door,

$$r = -\omega_o + 2(n \bullet \omega_o)n$$

hierbij is ω_o de invallende straal, r de reflectie en n de normaal van de spiegel. De rendering vergelijking kan worden vereenvoudigd tot,

$$= f_{\text{specular}}(p, \omega_i, \omega_o) L_i(p, \omega_i)$$

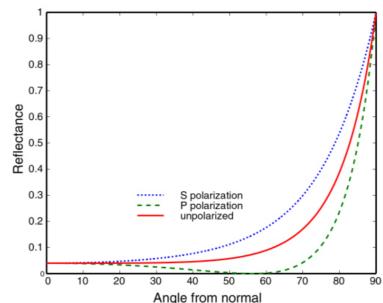
Hierbij is de integraal en de cosinus term weggevallen. Het weg vallen van de integraal komt doordat er enkel licht vanuit 1 richting (r) wordt gereflecteerd. Het weg vallen van de cosinus term komt doordat de doorsnede van de invallende straal en de uitgaande straal nog steeds de zelfde zijn. Dit is geïllustreerd met onderstaande afbeelding.



Met het wegvalen van de integraal is de factor f_{specular} nog wel blijven staan. Dit is de waarden die weergeeft hoeveel procent van het licht wordt gereflecteerd. Door deze functie te laten variëren in functie van de reflectie hoek kunnen mooie effecten worden bekomen. Dit is ook conform de werkelijk wereld. In computer graphics worden twee modellen veel gebruikt, de eerste heeft een constante reflectie percentage over de verschillende hoeken (benadering van werkelijkheid maar goed genoeg) en tweede model heet de *Fresnel* reflectie. Bij Fresnel reflectie worden lichtstralen die onder een lage hoek invallen sterk gereflecteerd. Licht dat loodrecht op het oppervlak invalt wordt bijna niet gereflecteerd. Dit reflectie model komt sterk overeen met licht dat af ketst van een water oppervlak en heeft te maken met de brekingsindex. De hoek afhankelijk van het reflectie percentage wordt benaderd met de volgende formule.

$$R(\theta) = R_0 + (1 - R_0)(1 - \cos \theta)^5 \quad R_0 = \left(\frac{n_t - 1}{n_t + 1} \right)^2$$

De grafiek van deze curve is weergegeven naast de formule.



RECURSIEVE REFLECTIE

Wanneer we de gereflecteerde straal r traceren zijn er 4 mogelijkheden:

1. De straal r raakt geen ander object uit de scène en raakt de achtergrond in punt p , in dit geval wordt de p weergegeven in de reflectie
2. De straal r raakt een lichtbron, in dit geval wordt L_e gereflecteerd.

3. De straal r raakt een niet reflecterend punt p op een lichtaam in de scène. In dit geval wordt de directe belichting berekend in punt p en dit wordt gereflecteerd (Merk op: de indirecte belichting kan ook gebruikt worden maar dan moet er rekening worden gehouden dat dit mogelijks afhangt van punten op een spiegel oppervlak.=extra moeilijkheid)
4. De straal r raakt een andere spiegel, in dit geval wordt er een tweede reflectie straal berekend en start het proces opnieuw.

Wanneer we in situatie 4 zitten moet er opnieuw een reflectie worden bepaald. In principe is dit algoritme niet eindig. Om hier aan te voldoen wordt er een maximale recursie diepte gesteld. Valt de laatste recursie terug op een spiegel wordt er een zwarte kleur terug gegeven. Dit is conform de realiteit, na een aantal reflecties gaat in de echte wereld de straal ook deels geabsorbeerd zijn.

GLOSSY REFLECTIE

Over glossy reflectie kunnen we kort zijn. De BRDF voor dit geval komt overeen met de phong shader,

$$f_r(p, \omega_i, \omega_o) = C \cdot \cos(\theta_r)^e$$

Hoe groter de mach e wordt gekozen hoe meer gepiektd de reflectie zal zijn rond de gereflecteerde richting. Dit houd dus in dat de perfecte reflectie een limiet geval is van dit model. Een nadeel t.o.v de perfecte reflectie is dat de integraal niet weg valt. De sampling gebeurd in dit geval dus volgens een PDE die lijkt op de f_r functie. (importance sampling).

GLOBAL ILLUMINATION

-PATH TRACING

Path tracing is een conceptuele eenvoudige techniek voor het berekenen van de globale belichting. Het idee is een straal te traceren doorheen de scène, elke keer dat de straal een voorwerp raakt wordt de BRDF opgelost aan de hand van een monte carlo integratie. Deze integratie creëert op zijn beurt een aantal stralen die verder recursief worden getraceerd. Wanneer de achtergrond van de scène zwart is kan er enkel een bijdrage van de stralen komen wanneer ze op een lichtbron vallen. Doordat enkel de stalen die eindigen op een lichtbron een bijdrage leveren is de convergentie van deze methode voor het bepalen van de globale belichten zeer traag. Een beter methode bestaat er in de stalen te zoeken die een bijdragen leveren in elk punt.

De rekentijd van deze methode hangt sterk af van de gebruikte materialen. Wanneer er bijvoorbeeld enkel perfecte spiegels aanwezig zijn kan hun BRDF met een straal worden gesampled. Wanneer de achtergrond ook een achtergrondlicht is gaat de rendering sneller. De meeste stralen eindigen dan op een lichtbron wat de recursie eindigt.

- SAMPLING VAN DE LICHTBRON

De trage convergentie van de path tracing voor de globale belichting kan opgelost worden door een hybride model te gebruiken. Hierbij wordt de directe belichting bepaald door middel van een rechtstreekse sampling van de lichtbronnen. De indirecte belichting wordt bepaald door de path tracing methode.

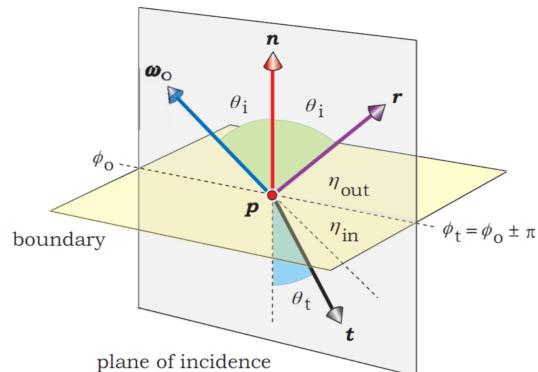
Bij de implementatie van deze methode moet er rekening worden gehouden dat men de directe belichting niet twee keer meetelt. Dit kan worden vermeden door een recursie diepte bij te houden. Wanneer de diepte op 1 staat wordt een licht contributie niet meegeteld.

Eenvoudige transparantie

Bij transparante materialen is er vaak een verandering van de dielectrische eigenschappen van de materialen. Deze verandering geeft aanleiding tot een breking van de stralen. De brekingshoek van een lichtstraal wordt gegeven met door de onderstaande formule.

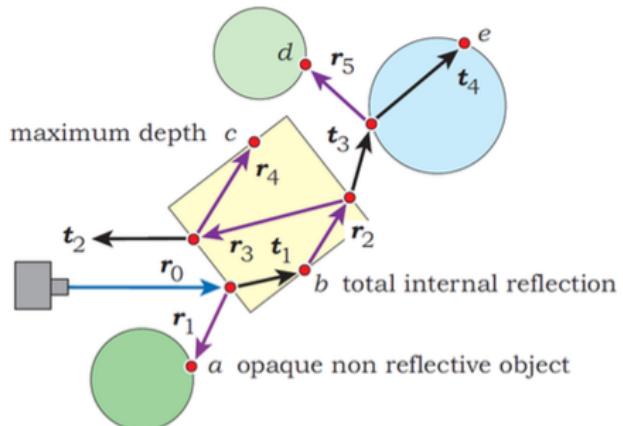
$$\frac{\sin \theta_i}{\sin \theta_t} = \frac{\eta_{in}}{\eta_{out}}$$

Hierbij zijn η_{in} en η_{out} de dialectische coëfficiënten van beide media waarin de straal beweegt. Licht breekt steeds weg van de normaal vector bij een overgang van een optisch ijl naar optisch dicht medium, het omgekeerde geld ook. Bij een overgang van een optisch dicht naar een optisch ijl materiaal (overgang glas naar water) breekt het licht naar de normaal toe. Dit heeft als gevolg dat er soms een reflectie in plaats van een breking optreedt (werkingsprincipe van optische kabels), dit effect heet totale interne reflectie.



SHADING MODEL VOOR REFRACTIE

Doordat licht dat bij de overgang van verschillende media het licht zowel kan breken als reflecteren kunnen er meerdere paden ontstaan. Dit is geïllustreerd in de afbeelding rechts van deze alinea. Dit kan worden opgelost door middel van een recursief schema. Merk op dat er soms geen nieuwe stralen worden gecreëerd. Dit is bijvoorbeeld het geval in punt b en a. Het shader model voor transparante materialen is een uitbreiding voor dat van ondoorzichtige materialen. Voor licht dat door een transparant materiaal gaat moet de recursie minimaal twee zijn. Er zijn steeds minstens twee stralen nodig voor het licht het materiaal weer verlaat. De rendering vergelijking voor een transparant materiaal ziet er exact hetzelfde uit al die voor een ondoorzichtig materiaal.



$$L_{indirect}(p, \omega_o) = \int_{hemisphere} f_r(p, \omega_i, \omega_o) L_i(p, \omega_i) \cos \theta_i d\omega_i$$

Voor een perfecte breking kan de integraal weer worden vervangen door gebruik te maken van een delta functie. Dit geeft de volgende uitdrukking voor de rendering vergelijking,

$$= \frac{\eta_t^2}{\eta_i^2} \cdot cst \cdot L_i(p, \omega_i)$$

REALISTISCHE REFLECTIE

Voor realistische reflecties te berekenen hebben we de principes van behoud van energie nodig. De energie van de twee stralen moet dus optellen tot de energie van de invallende straal. Om hier aan te voldoen gebruiken we een resultaat dat voorheen werd gebruikt voor een zuiver reflectie model af te leiden, namelijk de *Fresnel* reflectie. Het idee is dat stralen die recht invallen op het materiaal meer in het materiaal gaan dan stralen die onder een vlakke hoek op het materiaal invallen. Voor een plot van de coëfficiënten verwijzen we terug naar het begin van dit hoofdstuk over reflectie.

Een tweede extra element dat we wensen te simuleren is dat het licht verzwakt wanneer het doorheen een materiaal gaat. De manier waarop we dit doen is door gebruik te maken van de Beer's law. Deze zegt dat de intensiteit van het licht beschreven wordt door,

$$L(s) = L(0)e^{s \cdot \ln(a)}$$

$L(s)$ is de intensiteit in functie met van de afstand s dat het licht aflegt in het materiaal. $L(0)$ is dan de intensiteit van de originele invallende straal. Door deze factor kleur afhankelijk te maken kunnen er ook verschillende tinten van transparant materiaal worden gecreëerd.

Een derde effect dat we wensen toe te voegen aan het brekingsmodel is de een golflengte afhankelijkheid van de brekingsindex(Staat niet in boek). De logische implementatie hiervoor is om de brekingsindex te laten afhangen van de kleur van het licht. Wanneer er bijvoorbeeld wit licht invalt op een transparant materiaal moet de straal worden opgesplitst in een gewenst aantal stalen, elk met een andere hoek en kleur. De invoering van een kleur afhankelijke breking zal de renderingtijd weer doen toenemen.

- I. **LEG RECURSIEVE RAY TRACING UIT. WELKE VISUELE EFFECTEN KAN MEN MET RECURSIEVE RAY TRACING VOORSTELLEN?**
- II. **BESPREEK ENKELE COMPUTATIONELE PROBLEMEN DIE MEN KAN ONTMOETEN BIJ HET BEREKENEN VAN REFRACTIES IN HET RAY TRACING ALGORITME.**
- III. **ELK RECURSIEF ALGORITME MOET STOPCONDITIES HEBBEN. WELKE STOPCONDITIES HEBBEN ZIN BIJ RECURSIEVE RAY TRACING?**
- IV. **BESPREEK DE PROBLEMEN DIE KUNNEN OPTREDEN BIJ HET BEREKENEN EN VISUALISEREN VAN REFLECTIES EN REFRACTIES VOOR POLYGON-MESSES.**

Les 9(Hoofdstuk 29-30t)

In het vorige deel van de cursus zijn er een aantal constante besproken die het uiterlijk van de scene kunnen beïnvloeden. In dit deel bespreken we een methode om deze constante te moduleren met een gegeven functie F . Zo een functie heet in computer graphics een *texture*. Doorgaans zijn er twee mogelijke voorstellingen voor een texture, de eerste is in de vorm van een functie beschrijving. De tweede is in de vorm van een lookup table in een data structuur (bv een foto ed.)

MAPPING EEN 2D TEXTURE OP EEN 3D LICHAAM

Bij het mappen van een vlakke 2D map op een 3D lichaam zal er steeds een vervorming optreden wanneer het 3D lichaam niet vlak is. We bespreken verdere enkele soorten mappings die kunnen worden gebruikt.

- Cilindrische map

Bij een cilindrische map word een vlak omheen een cilinder gewikkeld. De y -as van de originele afbeelding wordt getransformeerd zodat deze dezelfde lengte heeft als de hoogte van de cilinder. De x -as van de afbeelding wordt gemapt tussen 0 en 2π zodat de x -coördinaat naar de hoek kan worden gemapt.

- Sferische map

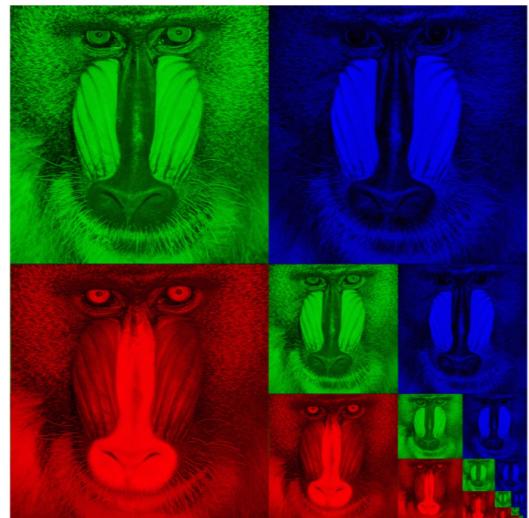
Bij een sferische mapping worden de x en y coördinaten gemapt op de ruimte hoeken het lichaam. Hierbij loopt 1 hoek van 0 tot π radialen en de andere hoek van 0 tot 2π . Bij een sferische mapping zijn de vervormingen het sterkst in de noord en zuidpool.

- Planaire map

Dit komt overeen met een orthogonale projectie van de afbeelding op het oppervlak.

MIP MAPPING

Bij een mapping van een texture op een voorwerp kunnen er ongewenste effecten optreden wanneer de resolutie van beide objecten te sterk verschillen van elkaar. Wanneer het texture een te lage resolutie heeft kan het zijn dat de individuele pixels zichtbaar worden. Wanneer de resolutie van het texture te hoog is kan het zijn dat er aliasing effecten optreden door de kleine details. *MIP Mapping* (multum in parvo) kan hier een oplossing bieden. In een MIP papping wordt de originele afbeelding opgeslagen op verschillende resoluties. Bij de downsampling van elke resolutie worden lineaire filter operaties gebruikt om de aliasing effect uit het beeld te halen. Wanneer de mapping vervolgens wordt uitgevoerd wordt het optimale texture gekozen. De lineaire filtering van de afbeelding is veel goedkoper dan nadien een AA filter te moeten gebruiken. Doordat de resolutie van de afbeelding steeds halveren per stap in de down sampling is de uiteindelijke opslag ruimte maar $4/3$ keer meer dan de grote van de originele afbeelding. Dit is geïllustreerd in de afbeelding rechts van deze alinea.



MAPPING VAN COMPLEXE MESH

Voor de mapping van een complexe driehoek mesh wordt er met elke vertex van de driehoek een coördinaat u,v geassocieerd. De kleur van de hoekpunten van de driehoeken liggen dan vast met de kleur u,v . Voor de kleur van een punt in de driehoek wordt er een interpolatie van de coördinaten gemaakt zodat er een punt u',v' is bepaald voor elk punt in de driehoek. Met dit punt komt er dan op zijn beurt een kleur overeen in de texture map. De ontvouwing van een complexe

mesh is een moeilijke taak. Men wenst de vervorming minimaal te maken. Twee technieken staan besproken in de slides. De eerste map de mesh op een complex samenhangende vorm. De tweede mapt de mesh op verschillende individuele kleine afbeeldingen. Deze laatste techniek heet een atlas texture map.

BUMP MAP

In Het vorig deel hebben we ons steeds beperkt tot de mapping van een kleur c_k op een lichaam in de scene. Een tweede populaire techniek is door de normaal vector te laten variëren aan de hand van een mapping functie, dit kan aan de hand van de volgende formule,

$$n' = \frac{n + \Delta x \cdot a + \Delta y \cdot b}{\|n + \Delta x \cdot a + \Delta y \cdot b\|}$$

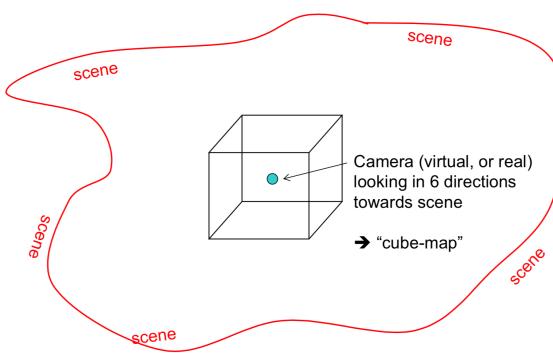
Hierbij haalt men de informatie Dx en Dy uit de mapping. de vectoren a en b zijn twee orthogonale vectoren die een basis vormen van het vlak dat loodrecht staat op n. Door middel van een bump map kan er een groot deel aan details worden toegevoegd aan een beeld zonder het aantal polygonen te moeten verhogen. Dit idee kan men ook gebruiken in de andere richting. Neem bijvoorbeeld een voorwerp met een groot aantal polygonen. We kunnen nu de normaal van dit voorwerp berekenen en opslaan in een map. Nadien kan dan het aantal polygonen in het originele voorwerp worden gereduceerd en de bump gebruikt worden voor de shader te berekenen. Dit geeft goede resultaten.

ENVIRONMENT MAP

Het idee is om de reflecties op voorhand te berekenen en deze nadien te mappen op het voorwerp.

ENVIRONMENT MAP (SKY BOX)

Nog een toepassing van omgeving mapping is geïllustreerd in onderstaande afbeelding.



Deze map kan dan gebruikt worden voor de reflecties van de omgeving te berekenen. De constructie van deze map kan door middel van 6 foto's te nemen in het camera punt of door een aantal fotos te nemen van een spiegelende bol. Let wel op dat je camera apparatuur dan op de bol staat maar dit kan opgelost worden door 4 foto's 90 graden t.o.v elkaar te nemen en aan elkaar te plaatsen. Een omgevingsmap geeft het beste resultaat voor voorwerpen die dicht in de buurt van het camera punt zijn geplaatst.

OMGEVINGSMAP ALS LICHTBRON

De omgevingsmap kan ook worden gebruikt als lichtbron. Dit doen we door schaduw stralen vanop de reflecterende voorwerpen te laten invallen op de omgevingsmap.

PROCEDURAAL TEXTUUR

In het vorige deel werden de texture maps allemaal beschreven door een lookup in een afbeelding. Men kan echter ook functies definiëren die overal kunnen worden geëvalueerd. Op deze manier is een texture niet enkel op de rand maar ook in het lichaam gedefinieerd. Op deze manier krijgt men het effect dat een voorwerp uit een massieve blok materiaal is gehouwd.

Voor een procedurale textuur die discontinuïteiten bevat kunnen er problemen optreden wanneer een discontinuïteit samenvalt met een rand van het texturen. Door numerieke afronding wordt er dan soms deel A en soms deel B van de texture gebruikt. Dit kan niet worden opgelost met AA technieken, in dit geval is de texture gewoon niet goed gedefinieerd. Dit valt wel te voorkomen door de niet met integer waarden in texturen functies te werken.

TRANSFORMATIES VAN TEXTURES

Wanneer we een texture op een voorwerp mappen is de volgende vraag hoe we de map gaan wijzigen wanneer het voorwerp veranderd. De mogelijkheden hier zijn best triviaal en worden niet besproken. Er is geen juist of fout, het is wat de gebruiker zelf wenst te creëren van effect.

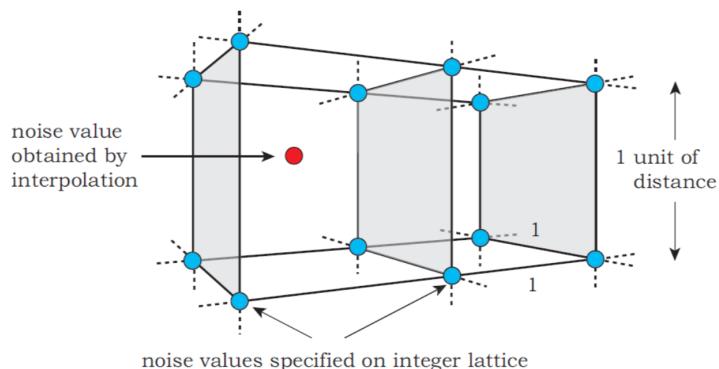
- I. BESPREEK HOE JE DE TRANSFORMATIE VAN TEXTUREN ZOU IMPLEMENTEREN IN EEN RAY TRACER. WELKE PROBLEMEN KUNNEN EVENTUEEL OPDUIKEN?**
- II. INDEN EEN TEXTUUR EEN RESOLUTIE HEEFT DIE VELE MALEN HOGER IS DAN DIEGENEN VAN HET FINALE BEELD IN PIXELS, WAT ZIJN DAN MOGELIJKE PROBLEMEN DIE KUNNEN OPDUIKEN BIJ HET GEBRUIK VAN DERGELIJKE TEXTUUR? WELKE OPLOSSINGEN STEL JE VOOR?**
- III. BESPREEK 2D TEXTURE MAPPING.**

Les 10 (Hoofdstuk 31)

In dit hoofdstuk bespreken we de toepassingen waarbij random noise wordt gebruikt voor het genereren van texture maps.

LATICE NOISE

Voor de generatie van 3D ruis gegevens in de ruimte maken we gebruik van rooster ruis. Het idee is de ruimte op te delen in een 3D rooster. Op elk van de rooster punten bepalen we dan een willekeurige ruis waarden. Om de ruis waarden in de kubus te kennen maken we gebruik van een 3D interpolatie. Dit concept is voorgesteld in onderstaande afbeelding.



De lineaire 3D (bi-lineaire) interpolatie gaat als volgt te werk; bereken 4 geïnterpoleerde punten op de rechtstaande ribben met als z coördinaat diegene van het rode punt. Interpolate de 4 punten 2 aan 2 met als x coördinaat dat geen van het rode punt. De twee resterende punten worden dan opnieuw geïnterpoleerd in de y coördinaat van het rode punt. M.a.w bereken steeds het gewogen gemiddelde van paren van punten met telkens 1 extra coördinaat gefixeerd.

Voor een zachter verlopend resultaat kan er ook een hogere orde interpolatie worden gebruikt.

SOM VAN RUIS FUNCTIES

Om verschillende ruis patronen te verkrijgen word er vaak gewerkt met de som van ruis functies. Men vertrekt dan van een laag frequente ruis functie en telt hier steeds frequentie verschoven kopieën bij op.

$$\text{fractal_sum}(p) = \sum_{j=0}^{n-1} \frac{\text{noise}(2^j p)}{2^j}$$

Op deze manier worden er fractaal structuren bekomen. Let wel op dat wanneer de factor j te hoog is gekozen er aliasing effecten kunnen optreden.

Een variant van bovenstaande formule is om gebruik te maken van de abs functie bij het optellen van de noise. Op deze manier ontstaan er een patroon dat *turbulentie* beschrijft. De parameters die in dit noise texture kunnen worden gewijzigd zijn o.a de kleur van de ruis, de gewichten in de som, de diepte, ...

Een geheugen efficiënte manier om de ruis voor te stellen is aan de hand van een soort hash functie in 3 integer parameters (de rooster coordinaten). Op deze manier moet er enkel de hash functie worden opgeslagen en niet alle ruis elementen.

Een laatste toepassing van ruis in voor de generatie van een random terrein. Dit spreekt voor zich maar is wel een voorbeeld dat besproken is in de les.

Examenwiki Vtk

Januari 2015

- I. VERSCHILLENDEN MANIEREN BESPREKEN OM SCHADUWEN IN 3D SCENE WEER TE GEVEN
- II. BESPREEK DE VERSCHILLENDEN PROBLEMEN DIE KUNNEN OPTREDEN BIJ EEN COMBINATIE VAN TEXTUUR EN TRANSFORMATIES.
- III. BESPREEK DE VERSCHILLENDEN CONSTANTEN IN HET DIFFUSE + PHONG SHADING MODEL. SUBVRAAG OVER EEN TORUS MET EEN PUNTLIGHT EN EEN CAMERA, WAAR KOMEN DE HIGHLIGHTS TEVOORSCIJN?
- IV. EXACT DEZELFDE ALS EXAMENVRAAG 66 UIT ZIJN BESTAND.

- I. Gegeven een scene met een driehoek die bepaald wordt door hoekpunten op coördinaten $(1,0,0)$, $(0,1,0)$ en $(0,0,1)$. Geef de transformatiematrix wanneer achtereenvolgens een schaling in de x - en y -richting wordt uitgevoerd met een factor $\sqrt{2}$, een rotatie van 45 graden volgens de z -as en een verplaatsing volgens de richting $(1,-1,1)$. Bereken ook de coördinaten van de getransformeerde hoekpunten. De camera bevindt zich op positie $(5,0,0)$ en kijkt in de richting van de oorsprong. De up-vector loopt volgens de positieve y -as. Geef de cameratransformatie en bereken de getransformeerde hoekpunten.
- II. Wat is een environment map (ook wel illumination map of reflection map genoemd)? Welke visuele effecten zijn er met behulp van een environment map mogelijk?
- III. Wat is perspectief-correcte interpolatie?
- IV. Gegeven een statische scène met een aantal objecten. We beschikken over twee spatiale versnellingsstructuren met mogelijk verschillende eigenschappen (structuur, opdeling in cellen, aantal objecten per cel, ...) om deze scène te ray traceren. Welke strategie gebruik je om te bepalen welke van beide versnellingsstructuren het beste is? Heeft de positie van de camera een invloed op deze strategie? Stel dat we een animatie van de scène willen maken. Heeft dit een effect op de keuze voor één van beide versnellingsstructuren? Speelt het een rol of het pad van de camera al dan niet op voorhand gekend is?

Voorbeelden van volledige antwoorden

Bespreek het Phong-belichtingsmodel

- I. Uiteraard starten met verantwoording: waar past het in, waarom nodig?
- II. Wiskundige formulering
- III. Voorbeeld van hoe een bol bvb. eruit zou kunnen zien belicht met Phong shading (maak een schets) + invloed verschillende parameters.
- IV. Hoe past het in Ray Tracing?
- V. Kwaliteit? Het 'ziet er goed uit', maar beantwoordt niet aan fysische werkelijkheid ... verklaring hiervoor.
- VI. Bespreking eventuele computationale problemen met model: negatieve cosinus factoren; grote machten van n ; meerdere lichtbronnen ...

Geef en verklaar de Surface Area Heuristic

- I. Verantwoording: waar past het in (acc-structuren), waarvoor dient het (optimale opsplitsing)?
- II. Mathematische verantwoording (kans dat een straal een subvolume raakt ...)
- III. Specifiek in context van kd-boom: cel splitsen via vlak in 2 kindercellen
- IV. waar zit dan de optimale positie van het vlak?
- V. Voorbeeld geven
- VI. Eventueel toepassing op andere situaties. Bvb. BVH?

Veel gemaakte fouten op het examen

- I. Uitspraak namen: bvb. Pong-shading ipv. Phong ('Fong') shading. Trek ik geen punten voor af, maar irritant.
- II. Verschil tussen Phong-shading en Phong-interpolatie.
- III. Geen inzicht in gebruik barycentrische coordinaten in 2D of 3D.
- IV. Zichtbaarheidbepaling staat los van shading en kunnen ontkoppeld worden. In ray tracing wordt zichtstraal voor beide gebruikt.
- V. Formules niet kennen: transformatiematrices, shading, e.d. Formules ook kunnen interpreteren! Er is meestal meer dan 1 manier om de dingen wiskundig neer te schrijven.
- VI. Door elkaar halen van acceleratiestructuren voor ray tracing.
- VII. Surface Area Heuristic niet correct uitrekenen.

Aandachtspunten voor studeren

- I. Ik hecht vrij veel belang aan inzicht in de grafische algoritmes zoals we die behandeld hebben in de les. Niet alleen het hoe, maar ook voordelen, nadelen, kritische bedenkingen etc.
- II. De uitleg die je geeft mag gebaseerd zijn op andere bronnen dan mijn slides of het cursusboek. Indien je dus wil studeren uit een ander boek of gebaseerd op informatie gevonden via het web, is dit voor mij even goed als de uitleg die in het cursusboek staat. Zolang het maar juist is ...
- III. Notaties of wiskundige formuleringen hoeven ook geen exacte reproductie zijn met identieke symbolen e.d. Het is belangrijker om te weten wat een formule voorstelt en wat de verschillende termen zijn, dan wel de notatie juist te hebben en niet te weten wat ze betekent. Eenvoudig voorbeeld: de Phong-exponent wordt meestal als 'n' aangeduid in de meeste teksten (maar niet in huidig boek :-)), maar als je nu een p, s of delta gebruikt is mij om het even. Maar wat ik wel belangrijk vindt is dat je weet dat als de Phong-exponent groter wordt, wat dan het resultaat is op het resulterende beeld. M.a.w. denk na over de wiskunde, reproduceer niet a