Dankwoord

Real programmers don't comment their code. It was hard to write, it should be hard to understand.

- Onbekend

ItWasHardToWrite, ItShouldBeHardToUnderstand.

- Olivier Dumont over het gebruik van spaties

Met blijdschap kijken we terug naar de vele leuke dingen die we hebben meegemaakt aan deze universiteit. Eén van deze leuke dingen waren de lessen Computergrafieken en de bijhorende practica. Dit verwondert misschien de doorsnee lezer maar de berekening van mooi beeld of de implementatie van een snel algoritme kan veel voldoening geven. Voor het aanreiken van dit boeiend onderzoeksdomein en het promoten van onze thesis willen we professor Dutré bedanken.

Dit eindwerk zou er helemaal anders hebben uitgezien indien we niet de deskundige begeleiding hadden gehad van Olivier Dumont. We hadden het geluk een begeleider te hebben die actief ons werk opvolgde en bijstuurde. We willen deze knuffelbeer daarom zeer hartelijk bedanken.

De goede sfeer in het het computerlokaal in gebouw 200B heeft ons door de zwaardere momenten van het eindwerk geholpen. Daarom willen iedereen bedanken die daar vaak vertoefde. Speciale dankbetuiging gaat uit naar Evy Theunis die na het ingeven van haar eindwerk nog een nacht lang onze thesis heeft nagelezen.

Tenslotte bedanken we onze ouders die ons de mogelijkheid hebben geboden deze studies te ondernemen en ons door dik en dun hebben gesteund.

Inhoudsopgave

1 Inleiding **1**

1.1 Historisch overzicht 2

1.1.1 Pre-computertijdperk 2

1.1.2 Geboorte van computergrafieken 2

1.1.3 Ray tracing 3

1.1.4 Rendering vergelijking 3

1.2 Doelstelling 3

1.3 Overzicht 4

2 Globale belichting **5**

2.1 Lichtmodellen 6

2.1.1 Beschrijving voor lichtgedrag 6

2.1.2 Radiometrische grootheden 8

2.2 Materiaalmodellen 9

2.2.1 BSSRDF 9

2.2.2 BRDF 10

2.2.3 Voorbeelden 11

2.3 Cameramodellen 12

2.4 Rendering vergelijking 13

2.5 Algoritmes 15

2.5.1 Path tracing 15
2.5.2 Radiositeit 18
2.5.3 Photon mapping 19
2.5.4 Andere algoritmes 23

3 Lichttransportcomponenten **27**

3.1 Probleemstelling 27
3.2 Algemene aanpak 28
3.3 Rendercomponenten 29
3.3.1 Lichtpadtypes 30
3.3.2 Geometrische segmentatie 33
3.4 Photon mapping met lichtpadtypes 33
3.5 Benadering van componenten 34
3.6 Analyse: volgorde van de componenten 35
3.6.1 Rechthoekige segmentatie 35
3.6.2 Prioriteit van een component 36
3.7 Renderstrategie 38
3.7.1 Incrementeel renderen 38
3.7.2 Tijdsgebonden renderen 39
3.8 Mogelijke verbeteringen 40
3.9 Bleed 42
3.9.1 Visualisatie per gebied 42
3.9.2 Visualisatie per lichtpadtype 42

4 Eigen aanpak **43**

5 Fotonsegmentatie **45**

5.1 Schatten bijdragen lichtpadtypes 46

5.2 Versneld schatten bijdragen lichtpadtypes 47
5.2.1 Motivatie 47
5.2.2 Methode 47
5.2.3 Toepassing 50
5.3 Aanpak segmentatie bijdragen lichtpadtypes 51
5.4 Segmentatie via randdetectie 54
5.5 Mean shift segmentatie 57
5.5.1 Principe 58
5.5.2 Toepassing op beelden 59
5.6 Aanpassingen mean shift segmentatie 63
5.6.1 Keuze van bandbreedteparameters 63
5.6.2 Behoud van minder belangrijke grote gebieden 67
5.6.3 Adaptieve minimale regiogrootte 67
5.7 Afronden van de segmentatie 70
5.8 Evaluatie 74

6 Toepassing van de fotonsegmentatie 75
6.1 Segmentatie per lichtpadtype 76
6.2 Berekening van thresholdmap en prioriteiten met fotonsegmentatie 78
6.2.1 Thresholdmap 78
6.2.2 Prioriteiten 79
6.2.3 Evaluatie 79
6.3 Verfijning van de gevisualiseerde benadering met fotonsegmentatie 80
6.3.1 Scherpe randen 82
6.3.2 Vorm van de randen 84
6.3.3 Fotonbenadering 85
6.3.4 Evaluatie 88

6.4 Verfijning van de rendercomponenten met fotonsegmentatie 88
6.4.1 Prioriteit 89
6.4.2 Nieuwe randen 91
6.5 Vergelijking werking methodes 94
6.5.1 Oorspronkelijke methode 94
6.5.2 Aangepaste thresholdmap en prioriteiten 96
6.5.3 Verfijnde benadering 96
6.5.4 Verfijnde componenten 96

7 Resultaten **97**

7.1 Vergelijking met path tracing 97
7.2 Vergelijking van de verschillende methodes 98
7.3 Scène met glazen draak 101
7.3.1 Resultaten 101
7.3.2 Parameters 101
7.3.3 Preprocessing 102
7.3.4 Rendering 105
7.4 Scène met metalen ring 106
7.4.1 Resultaten 107
7.4.2 Preprocessing 107
7.4.3 Rendering 107
7.5 Scène met glazen ei 109
7.5.1 Resultaten 109
7.5.2 Preprocessing 109
7.5.3 Rendering 110
7.6 Scène met kamers 110
7.6.1 Preprocessing 111

7.6.2 Rendering 111

8 Besluit **113**

8.1 Overzicht eigen werk en bekomen resultaten 113

8.2 Verfijningen en uitbreidingen 114

8.2.1 Verbetering van gemaskeerde gepartioneerde vervaging 114

8.2.2 Meer randen afleiden uit de geometrische informatie 115

8.2.3 Voorspellen van de rendertijd 115

8.2.4 Verbeterde fotonbenadering 115

8.2.5 Fout in perceptuele metriek 116

8.2.6 Compensatie hoeken bij adaptieve bemonstering 116

A Monte Carlo integratie **117**

A.1 Kansrekenen 117

A.1.1 Discrete variabelen 117

A.2 Numerieke integratie 119

B Illustraties bij het hoofdstuk resultaten **121**

B.1 Glazen draak 121

B.2 Metalen ring 127

B.3 Glazen ei 131

B.4 Kamers 135

Hoofdstuk 1

Inleiding

Computergrafieken is een bloeiend deelgebied van de computerwetenschappen. Hierbij wordt de focus gelegd op de creatie, bewaring en manipulatie van computergegenereerde beelden. Computergrafieken is een toegepaste wetenschap in zijn volste betekenis. Concepten uit de natuurkunde, wiskunde, biologie, ... worden veelvuldig toegepast.

De resultaten van computergrafieken zijn nauwelijks weg te denken uit het dagelijks leven. Er volgen enkele voorbeelden. Beelden in kranten of televisie worden eerst bewerkt voordat ze getoond worden. Computergegenereerde animatiefilms zoals "The Incredibles" of "Shrek" lokken volle filmzalen. Nieuwe producten kunnen, nog voor een prototype bestaat, realistisch op een computerscherm getoond worden. Recent worden computergrafieken ook in de medische wereld gebruikt, bijvoorbeeld bij operaties.

Een tweedimensionaal beeld bestaat uit pixels waarbij elke pixel visuele informatie bevat. Deze beelden kunnen bekomen worden op verschillende manieren, bijvoorbeeld via een camera of een tekenprogramma. Indien alle pixels bekend zijn, kunnen deze bewerkt worden. Tweedimensionale beelden hebben als nadeel dat er geen informatie is over de diepte in het beeld. Het is ook niet mogelijk om van gezichtspunt te veranderen. Uitzonderingen hierop vormen beeldgebaseerde visualisering en computervisie.

Driedimensionale grafieken hebben deze problemen niet. Eerst wordt er een scène gemodelleerd. Deze bevat driedimensionale voorwerpen met bepaalde materiaaleigenschappen, een aantal lichtbronnen en een virtuele camera. Het modelleren kan op verschillende manieren gebeuren. Een voorbeeld hiervan is het verkrijgen van een driedimensionale voorstelling door scanning.

Een scène kan *gerenderd* worden. Dit wil zeggen dat uitgaande van de scène en de camera een tweedimensionaal beeld wordt bekomen. Er kan getracht worden om op een zo waarheidsgetrouw mogelijke wijze te renderen. Hierbij wordt het fysisch gedrag van licht in de scène gesimuleerd. Recent is er ook onderzoek geleverd naar niet-realistische rendering. De bedoeling hiervan is het maken van artistieke beelden.

De voorwerpen in een scène kunnen ook geanimeerd worden. Dit resulteert in bewegende beelden. In films en computerspellen is een realistische animatie bijvoorbeeld zeer belangrijk.

Dit eindwerk gaat over één bepaald aspect van de computergrafieken, namelijk het fotorealistisch renderen, ook wel *globale belichting* genoemd. In deel 1.1 wordt eerst een beknopt historisch overzicht van fotorealistische rendermethodes gegeven. Vervolgens wordt de doelstelling van de thesis uitgelegd in deel 1.2. Tenslotte wordt in deel 1.3 een kort overzicht gegeven van de verschillende hoofdstukken.

1.1 Historisch overzicht

1.1.1 Pre-computertijdperk

Reeds in de prehistorie trachtte de mens de wereld rond zichzelf voor te stellen. Getuige zijn de grotschilderingen die men op het Europese continent aantreft. Naarmate de mens meer evolueerde, werden deze voorstellingen steeds waarheidsgetrouwer. Tijdens de renaissance achterhaalde men de principes van de perspectieftekening. Accurate schaduwen en lichtreflecties werden later toegevoegd. Een toppunt van realisme werd bereikt in de schilderkunst.

Tijdens de 15^e eeuw werd de camera obscura ("donkere kamer") ontwikkeld. Dit apparaat projecteerde de werkelijkheid zoals die te zien was door de opening van het apparaat. Het geprojecteerde beeld kon echter niet vastgelegd worden. In 1825 slaagde de Franse uitvinder Nicéphore Niépce hier wel in. De fotografie was geboren. Deze ontdekking zou voorgoed een andere bestemming geven aan de schilderkunst. De nadruk werd niet meer gelegd op het realisme. Om realistische beelden te maken werd een fototoestel gebruikt, wat veel eenvoudiger is. Met een druk op de knop kon men de realiteit vastleggen, zij het in twee dimensies.

Men wilde echter niet altijd een waarheidsgetrouw beeld van de werkelijkheid, maar van een *plausibele* werkelijkheid. Door middel van speciale bewerkingstechnieken en wat knip- en plakwerk was het mogelijk nieuwe beelden samen te stellen uit oude beelden, beelden in te kleuren, ... Deze technieken moesten echter manueel toegepast worden en waren bijgevolg zeer tijdsintensief.

1.1.2 Geboorte van computergrafieken

De geboorte van computergrafieken wordt veelal toegeschreven aan *Sketchpad*, een doctoraatsthesis van Ivan Sutherland aan het M.I.T. in 1963. Het is de voorloper van moderne tekenprogramma's.

Door de komst van het rasterscherm was men niet meer beperkt tot lijntekeningen maar was ook het gebruik van ingekleurde veelhoeken mogelijk. Door veelhoeken met het juiste perspectief te tekenen kunnen complexe modellen verkregen worden. Deze veelhoeken worden ingekleurd. Primitieve inkleuringsalgoritmes berekenden meestal slechts één kleur per veelhoek. Hoewel er rekening gehouden werd met de hoek van het invallende licht, was het realisme zeer ver te zoeken.

De inkleuringsalgortimes van Gouraud en Phong zorgden voor een grote verbetering. Zij gebruikten interpolatie om realisme te verkrijgen. Bij Gouraud-interpolatie worden de kleurwaarden op de hoekpunten van de veelhoek berekend en vervolgens geïnterpoleerd over de veelhoek. Bij Phong-interpolatie worden de normalen geïnterpoleerd en wordt over de veelhoek kleurenwaarden berekend.

Er werden steeds meer en betere technieken ontwikkeld om het realisme te verhogen. Zo werden bijvoorbeeld schaduwen en texturen ingevoerd.

1.1.3 Ray tracing

In 1980 ontwikkelde Turner Whitted een zeer populair renderingalgoritme, namelijk ray tracing. In plaats van veelhoek per veelhoek te tekenen, wordt pixel per pixel overlopen. Vanuit het oog wordt door de pixel een straal geschoten die in de scène botst. Zo kunnen sommige materialen zoals spiegels op een natuurlijke manier voorgesteld worden.

1.1.4 Rendering vergelijking

Een van de meeste belangrijke concepten voor globale belichtingsalgoritmes, de rendering vergelijking, werd voorgesteld door Kajiya in 1986 [Kaj86]. In deze paper werd voor de eerste keer de volledige transportvergelijking beschreven die het licht in een scène beschreef en dit in de context van computergrafieken. Het beschrijft alle lichttransportverschijnselen als een recursieve integraal.

Deze voorstelling van globale belichting laat een algemene aanpak van het probleem toe. Er kan gekozen worden tussen verschillende soorten integratietechnieken om de rendering vergelijking numeriek op te lossen. Een van deze technieken die zeer geschikt bleek, was Monte Carlo integratie. Zo onstond de algemeen toepasbare Monte Carlo path tracing.

1.2 Doelstelling

Met de tot heden ontwikkelde methodes is er sterke vooruitgang geboekt in globale belichting. De vraag is niet meer hoe waarheidsgetrouwe beelden kunnen bekomen worden, maar hoe deze *snel* kunnen gegenereerd worden. Zelfs met de steeds verhogende process-

orkracht kan een naieve implementatie van Monte Carlo path tracing geen aanvaardbaar beeld genereren in redelijke tijd.

Er zijn veel methodes ontwikkeld om dit proces te versnellen. Meestal behandelen ze het beeld echter op een uniforme manier, of zijn ze vooral geschikt voor een bepaald type scène.

Een adaptieve aanpak van het visualisatieprobleem wordt voorgesteld in "Light Transport Component Analysis & Rendering" [DD05]. Met een initiële schatting van het resultaat wordt een analyse uitgevoerd zodat de juiste methode in het juiste deel van het beeld kan toegepast worden. Er wordt vooral gefocust op het genereren van beelden die misschien niet volledig juist zijn, maar dat wel lijken. Daarmee tracht men snelheidswinst te bekomen. Beelden worden incrementeel aangemaakt zodat er niet moet gewacht worden totdat het algoritme eindigt.

Dit eindwerk zal trachten dit raamwerk voor optimale globale belichting verder uit te werken zodat er sneller een beter beeld wordt bekomen. De analyse wordt uitgebreid en de rendering wordt verfijnd zodat er meer gericht te werk kan gegaan worden.

1.3 Overzicht

Hierna komt een korte samenvatting van elk hoofdstuk.

In hoofdstuk 2 wordt globale belichting uit de doeken gedaan. Er wordt nagegaan op welke manier de fysische principes voor beeldvorming gemodelleerd kunnen worden in een computer. Recente algoritmes maken gebruik van Monte Carlo integratie, wat besproken wordt in bijlage A. Ook worden een paar succesvolle methodes aangehaald om het basisprincipe te versnellen.

Vervolgens wordt in hoofdstuk 3 de paper "Light Transport Component Analysis & Rendering" [DD05] toegelicht. Het biedt een raamwerk aan om het beeld op te splitsen in verschillende gebieden voor een meer adaptieve aanpak van het visualisatieproces.

Hoofdstuk 4 vat samen welke verbeteringen we trachten toe te brengen aan de voorgestelde methode. In hoofdstuk 5 ontwikkelen we een segmentatiemethode die in hoofdstuk 6 wordt toegepast om [DD05] te verfijnen.

De bekomen resultaten worden vervolgens besproken in hoofdstuk 7. We gaan na wat de sterke punten en de tekortkomingen zijn. Uitgebreide illustraties bij dit hoofdstuk vindt men terug in bijlage B.

Tenslotte geeft hoofdstuk 8 een algemeen besluit over het onderzoek verricht in het kader van deze thesis.

Hoofdstuk 2

Globale belichting

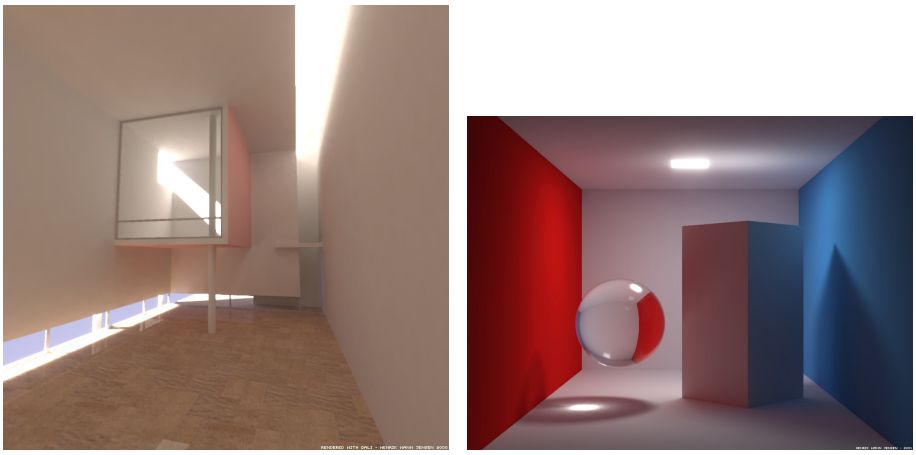
Het doel van globale belichtingsalgoritmes is een waarheidsgetrouw beeld te maken. Dit gebeurt aan de hand van een beschrijving van een driedimensionale scène. Hierin worden lichtbronnen en de plaatsing en grootte van voorwerpen met hun materiaaleigenschappen beschreven. Aan de hand van een virtuele camera die wordt geplaatst in de scène wordt het beeld berekend of *gerenderd*. Deze camera wordt ook lichtsensor of oog genoemd. Indien deze scène in het echt zou worden nagebouwd en een echte camera zou worden gebruikt die dezelfde eigenschappen heeft als de virtuele camera, zou een menselijke waarnemer geen verschil mogen merken tussen het computergegeneerd beeld en het echte beeld. Effecten die niet waargenomen kunnen worden, mogen dus onder bepaalde omstandigheden verwaarloosd worden.

De term *globale belichting* slaat op het feit dat de belichting op een fysische correcte manier in rekening wordt gebracht en dat dus alle lichteffecten berekend kunnen worden.

In figuur 2.1 zijn twee voorbeelden te zien van computergegeneerde beelden. In het eerste beeld is de lichtbron niet te zien. De muren worden verlicht door de weerkaatsingen van het licht. Er is veel *indirecte belichting*. De term indirect komt van het feit dat het licht niet rechtstreeks afkomstig is van een lichtbron. Het tweede beeld toont ook indirecte belichting. Zo is het plafond niet donker en vertoont het op sommige plaatsen een rode en blauwe schijn. Dit wordt aangeduid met de term *color bleeding*. De bijdrages van de directe belichting hebben een veel grotere invloed dan in het eerste beeld.

Reflectie of refractie van licht kan zorgen voor een bundeling van het licht. Een voorbeeld hiervan is te zien op de bodem van een zwembad. De watergolven zorgen voor breking en bundeling zodat heldere vlekken te zien zijn op de bodem van het zwembad. Deze lichtvlekken worden *caustics* genoemd. Dit is ook merkbaar in het tweede beeld van figuur 2.1. Het licht van de lichtbron wordt gebundeld door de glazen bol zodat er een caustic op de grond te zien is.

In deel 2.1 wordt fysisch gebaseerde voorstellingswijzes voor licht besproken. Hieruit



(a) (b)

Figuur 2.1: Realistische computergegeneerde beelden (*eigendom van Henrik Wann Jensen*): (a) toont een beeld waar de indirecte belichting zeer belangrijk is. In (b) is er omwille van de bundeling van de licht doorheen de glazen bol een *caustic* te zien. Dit is de lichte lichtvlek onder de bol.

kan een cameramodel ontwikkeld worden. Dit wordt besproken in deel 2.3. Realistische materiaalmodellen zijn een noodzaak om waarheidsgetrouwe beeldsynthese uit te voeren. Een overzicht van de meest gebruikte modellen wordt gegeven in deel 2.2.

De vergelijking die het mogelijk maakt om het volledige lichttransport te kunnen beschrijven is de *rendering vergelijking*. Een bespreking hiervan is te vinden in deel 2.4. Meestal zal deze vergelijking niet analytische kunnen worden opgelost. Met behulp van numerieke integratietechnieken of eindige elementenmethodes kan er toch een beeld gerenderd worden. In deel 2.5 wordt een overzicht gegeven van de meest gebruikte algoritmes die de rendering vergelijking proberen op te lossen.

2.1 Lichtmodellen

2.1.1 Beschrijving voor lichtgedrag

Berekeningen met licht vereisen een voorstellingswijze of model dat het gedrag van licht (onder bepaalde omstandigheden) kan verklaren. Licht heeft een duaal karakter: het kan beschreven worden als een golf of als een deeltje. Er wordt een overzicht gegeven van de belangrijkste lichtmodellen.

Quantummodel

Het quantummodel biedt het meest gedetailleerde en correcte model. Het beschrijft nauwkeurig het duale karakter van licht tot op submicroscopisch niveau. Het is gebaseerd op de quantummechanica. Licht bestaat uit kleine pakketjes met elk een gequantificeerde hoeveelheid energie. Deze worden fotonen genoemd. Het quantummodel is te complex om te gebruiken bij realistische beeldsynthese.

Golfmodel

Het golfmodel is gebaseerd op de wetten van Maxwell. Licht wordt gezien als een hoogfrequente elektromagnetische golf. Het model is een vereenvoudiging van het quantummodel. Hierdoor kunnen niet alle lichteffecten verklaard worden. Een voorbeeld hiervan is het foto-elektrisch effect. Het materiaal zendt elektronen uit indien er licht op het materiaal invalt.

Diffractie en interferentie zijn effecten die optreden indien licht interageert met voorwerpen waarvan de grootte vergelijkbaar is met de golflengte van licht. Het golfmodel beschrijft op een accurate manier deze en andere effecten, zoals polarisatie. Hoewel het een vereenvoudiging is van het quantummodel, is het een complex model, dat in vele gevallen te complex is om te gebruiken om realistische beelden te genereren.

Geometrische optica

Geometrische optica is een verdere vereenvoudiging. Dit model is bruikbaar indien de voorwerpen waarmee het licht interageert veel groter zijn dan de golflengte van zichtbaar licht. In het geometrisch opticamodel worden de volgende veronderstellingen gemaakt:

- Licht reist in vacuüm volgens rechte lijnen.
- Licht kan onmiddellijk door een medium reizen. Met andere woorden: licht reist met een oneindig hoge snelheid. Het licht zal zich dus steeds in een evenwichtstoestand bevinden.
- Licht verandert niet door externe factoren zoals zwaartekracht of magnetische velden.

Dit model is niet het meest fysisch correcte model. Het beschrijft echter wel de meest voorkomende lichteffecten zoals reflectie, refractie, absorptie en emissie zodat de kleur en de intensiteit van objecten wel kunnen berekend worden. Het gebruik van een *participerend medium* in een driedimensionale scène is ook mogelijk. Participerende media zijn media waarin het licht verstrooid, uitgestraald of geabsorbeerd wordt. Een voorbeeld het

Symbool	**Quatiteit**	**Eenheid**
Q	Energie	J
Φ	Flux	W
E	Irradiantie	W/m^2
B	Radiositeit	W/m^2
L	Radiantie	$W/(sr\ m^2)$

Tabel 2.1: Belangrijkste radiometrische grootheden

effect van koplampen in de mist. Door de mist zijn de lichtstralen uit de koplampen van overal zichtbaar.

Het geometrische opticamodel is het meest gebruikt lichtmodel in computergrafieken.

2.1.2 Radiometrische grootheden

Om op een fysisch correcte manier licht te beschrijven zijn grootheden nodig. De radiometrie levert deze grootheden. Het is het onderdeel van de fysica dat zich bezighoudt met het meten van licht. De belangrijkste radiometrische grootheden worden besproken.

- **Energie**, Q, wordt uitgedrukt in Joule (J).
- **Flux**, Φ, beschrijft hoeveel energie er per tijdseenheid uit, naar of door een oppervlak stroomt. Het wordt uitgedrukt in Watt (*W* of $\frac{\text{Joule}}{s}$).

$$\Phi = \frac{dQ}{dt}$$

- **Irradiantie**, E, is de *invallende* flux per eenheid oppervlakte. Irradiantie is dus gelijk aan

$$E = \frac{d\Phi}{dA} = \frac{d^2Q}{dtdA}$$

  Het wordt uitgedrukt in Watt per m^2 ($\frac{W}{m^2}$). De *uitgaande* flux per eenheid oppervlak wordt de **radiositeit**, B, of **exitantie**, M, genoemd.

- **Radiantie**, L, duidt aan hoeveel flux op het oppervlak toekomt of het oppervlak verlaat op een bepaald punt in het oppervlak, per ruimtehoek en per eenheid *geprojecteerde* oppervlak:

$$L = \frac{d^2\Phi}{d\omega dA^{\perp}} = \frac{d^2\Phi}{d\omega dA\cos\theta}$$

  Een ruimtehoek wordt gedefinieerd als het oppervlak op een hemisfeer gedeeld door het kwadraat van de straal van de hemisfeer:

$$\Omega = \frac{A}{r^2}$$

Deze wordt uitgedrukt in sterradialen (sr). De ruimtehoek die gevormd wordt door de hele hemisfeer is gelijk 2πsr. Radiantie wordt uitgedrukt in $W/(\text{sr } m^2)$. Dit is de belangrijkste grootheid voor beeldsynthese. Deze zorgt voor het *uitzicht* van voorwerp. Sensoren zoals het menselijk oog en camera's zijn gevoelig voor radiantie. De respons van sensoren is evenredig met de hoeveelheid radiantie die ontvangen wordt. De evenredigheidsconstante is afhankelijk van de grootte en vorm van de sensor. Een andere eigenschap van radiantie is dat deze volgens een rechte lijn in vacuüm onveranderlijk is. Dit vergemakkelijkt de berekening van radiantie. De radiantiewaarden van alle punten in een scène drukken de evenwichtstoestand uit.

In tabel 2.1 worden deze grootheden samengevat. Ze kunnen afhankelijk zijn van de golflengte. Een voorbeeld hiervan is de spectrale radiantie L_λ. De radiantie wordt verkregen door de spectrale radiantie te integreren over alle mogelijk golflengtes:

$$L = \int_0^\infty L_\lambda d\lambda$$

2.2 Materiaalmodellen

Om realistische beelden te maken is een fysisch gebaseerde materiaalbeschrijving nodig. Dit laatste gebeurt in een materiaalmodel. Deze beschrijft wat de relatie is tussen invallend licht en het licht dat uitgestuurd wordt aan het oppervlak van het materiaal.

Er wordt verondersteld dat licht onmiddellijk het oppervlak verlaat en dit met dezelfde golflengte. Effecten zoals fosforentie en fluorescentie kunnen hierdoor niet gevisualiseerd worden. Deze kunnen echter veelal verwaarloosd worden. Een visualisatie die hier wel rekening mee houdt wordt besproken in [WTP01].

2.2.1 BSSRDF

Licht bereikt een oppervlak binnen in een punt x' onder een bepaalde hoek Φ. Het wordt vervolgens verstrooid in het materiaal en verlaat het oppervlak in een (ander) punt x onder een hoek Θ. Dit verstrooiingseffect is zeer goed merkbaar bij doorschijnende materialen als marmer en menselijke huid.

Een functie die dit soort gedrag beschrijft is de BSSRDF of *bidirectional scattering surface reflectance distribution function*. Deze is het verband tussen de differentiële gereflecteerde radiantie dL in het punt x en richting Θ, en de differentiële irradiantie in het punt y en richting Φ:

$$S(x' \leftarrow \Phi, x \rightarrow \Theta) = \frac{dL(x \rightarrow \Theta)}{dE(y \leftarrow \Phi)}$$

Wegens de grote complexiteit, gebruikt men de BSSRDF in renderingalgoritme echter weinig.

2.2.2 BRDF

Bij vele materialen kan echter de bijkomende veronderstelling gemaakt worden dat het punt waar het licht invalt en het punt dat het licht het oppervlak verlaat, hetzelfde is. Een functie die dit gedrag beschrijft is de bidirectionele reflectantie distributie functie (BRDF):

$$f_r(x, \Psi \rightarrow \Theta) = \frac{dL(x \rightarrow \Theta)}{dE(x \leftarrow \Psi)}$$

De irradiantie kan geschreven worden in functie van de radiantie:

$$E = \int_{\Psi} L(x \leftarrow \Psi) \cos(N_x, \Psi) d\omega_{\Psi}$$

zodat de volgende veelgebruikt uitdrukking voor de BRDF verkregen wordt:

$$f_r(x, \Psi \rightarrow \Theta) = \frac{dL(x \rightarrow \Theta)}{L(x \leftarrow \Psi) \cos(N_x, \Psi) d\omega_{\Psi}}$$

De BRDF heeft de volgende eigenschappen:

Bereik De BRDF kan elke positieve waarde aannemen.

$$f_r \epsilon [0, \infty [$$

Dimensies De BRDF is een vierdimensionale functie die gedefinieerd is elk punt van het oppervlak. Vier dimensies komen overeen met de invallende richting en de andere twee dimensies komen overeen met de uitgaande richting. Veel materialen zijn echter isotroop. Dit wil zeggen dat waarde van de BRDF niet afhankelijk van de oriëntatie van het oppervlak.

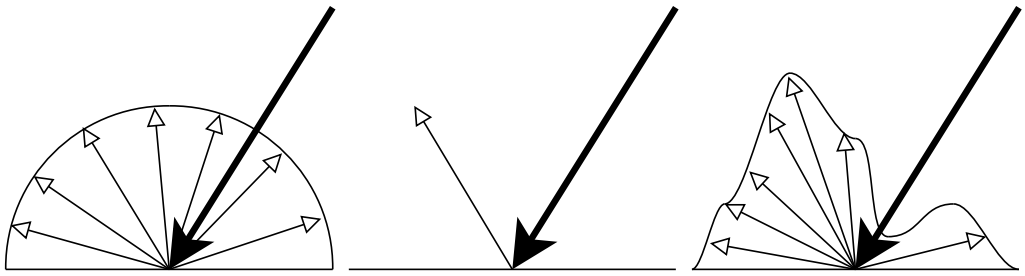
Reciprociteit Indien de inkomende en uitgaande richtingen verwisseld worden, zal de BRDF dezelfde waarde hebben. Dit wordt de *Helmhotz reciprociteit* genoemd.

$$f_r(x, \Theta \leftarrow \Psi) = f_r(x, \Theta \rightarrow \Psi) = f_r(x, \Theta \leftrightarrow \Psi)$$

Behoud van energie De BRDF voldoet aan de wet van behoud van energie. Er geldt dus voor alle richtingen Ψ :

$$\int_{\Omega_x} f_r(x, \Theta \leftrightarrow \Psi) \cos(N_x, \Theta) d\omega_{\Theta} \leq 1$$

In figuur 2.2 zijn enkele voorbeelden van een BRDF te zien.



(a) Ideaal diffuus (b) Ideaal spiegelend (c) Reeële BRDF

Figuur 2.2: Voorstelling van de BRDF van isotrope materialen

2.2.3 Voorbeelden

De meest gebruikte materiaalmodellen worden besproken.

Diffuus materiaal

Diffuse materialen zijn materialen met op microscopisch niveau een zeer ruw oppervlak. Het licht wordt in alle richtingen verstrooid. Een ideaal diffuus materiaal wordt ook een Lambertiaans materiaal genoemd. Er wordt verondersteld dat de richting van een gereflecteerde lichtstraal perfect willekeurig is. De uitgestraalde radiantie is dus constant. Hierdoor is de BRDF gelijk aan:

$$f_r(x) = \frac{\rho_d(x)}{\pi} = k_d$$

De diffuse reflectantie $\rho_d(x)$ duidt aan hoeveel procent van het licht gereflecteerd wordt.

Spiegelend materiaal

Spiegelende materialen zijn zeer gladde materialen. Typische voorbeelden zijn metaaloppervlakken of gladde diëlektrische opppervlakken zoals glas en water. Doordat de meeste materialen oneffenheden bevatten zal niet al het licht gespiegeld worden in de richting van de perfect spiegelende richting, maar in een soort kegel rond deze richting.

Voor ideale spiegelende materialen geldt dat

$$L(x \rightarrow \Theta_s) = \rho_s(x)L(x \leftarrow \Phi)$$

De spiegelende reflectantie $\rho_s(x)$ duidt aan hoeveel procent van het licht gespiegeld wordt. Voor de perfecte spiegelende richting Θ_s geldt

$$\Theta_s = 2\cos(\Phi, N_x)N_x - \Phi$$

N_x is de normaal op het oppervlak in het punt x.

Voor gladde homogene metalen en diëlectrica kan uit de Fresnelvergelijkingen gehaald worden hoeveel licht er gereflecteerd wordt.

Materialen die niet diffuus of spiegelend zijn worden *glossy* materialen genoemd. De BRDF hiervan is meestal moeilijk analytisch te beschrijven. De fysisch gebaseerde modellen en de parametrische modellen proberen hier een benadering voor op te stellen.

Fysisch gebaseerde modellen

Een voorbeeld van een fysisch gebaseerd model is het Cook-Torrance model [CT82]. Hierbij wordt verondersteld dat het materiaal bestaat uit kleine gladde microfacetten. Dit zijn kleine vlakjes die een willekeurige richting hebben. Een inkomende lichtstraal zal op een willekeurige microfacet landen. De BRDF ziet er als volgt uit:

$$f_r(x, \Theta \leftrightarrow \Phi) = \frac{F(\beta)}{\pi}\frac{D(\theta_h)G}{(N \cdot \Phi)(N \cdot \theta)} + k_d$$

$F(\beta)$ is de Fresnel-term. Deze beschrijft de speculaire reflectie op een microfacet. De distributie van de microfacetten wordt voorgesteld door $D(\theta_h)$. G is de factor die rekening houdt dat sommige vlakjes andere kunnen maskeren. De factoren in de noemer dienen ter normalisatie. Tenslotte is er k_d voor het diffuse deel van het materiaal.

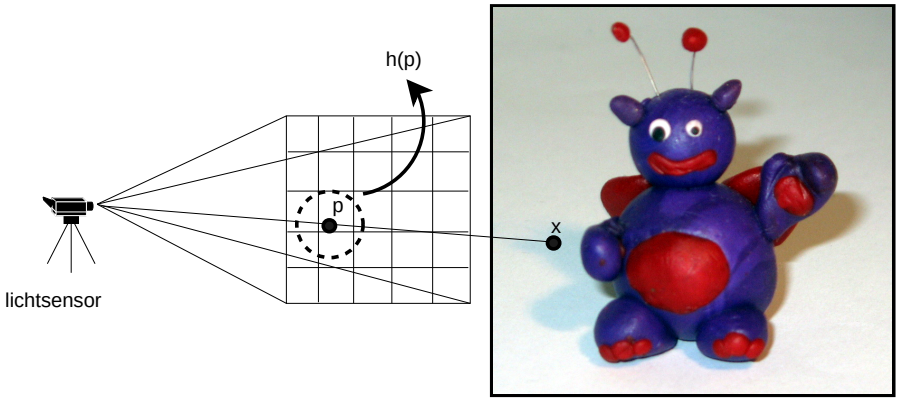
Andere voorbeelden van fysisch gebaseerde modellen zijn het Oren-Nayar model [ON95] dat ruwe diffuse materialen modelleert en het He model [HTSG91].

Parametrische modellen

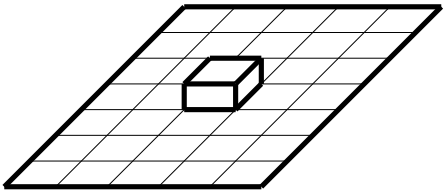
Een voorbeeld van een parametrisch model is het Lafortune model [LFTG97]. Dit model fit gewijzigde Phong-lobben aan de hand van opgemeten gegevens. Het Phong model is een veelgebruikt eenvoudig materiaalmodel maar heeft als nadeel dat het niet fysisch correct is. Het Lafortune model lost dit probleem op. Zo kan er met behulp van een relatief eenvoudig model toch realistische materialen gevisualiseerd worden.

2.3 Cameramodellen

Een cameramodel duidt aan wat de relatie is tussen de radiantiewaarden in de scène en het berekende beeld. Aangezien lichtsensoren gevoelig zijn voor radiantie, moet er voor



Figuur 2.3:



Figuur 2.4: 2D-doosfilter

elke pixel van het te berekenen beeld een radiantiewaarde L_{pixel} worden gevonden.

Deze waarde is een gewogen gemiddelde van alle radiantiewaarden in het beeldvlak. Dit wordt schematisch voorgesteld in figuur 2.3. Er geldt dat

$$L_{pixel} = \int_{beeldvlak} L(p \rightarrow \text{lichtsensor}) h(p) dp = \int_{beeldvlak} L(x \rightarrow \text{lichtsensor}) h(p) dp$$

p is een punt op het beeldvlak en x is het punt dat zichtbaar is vanuit de lichtsensor door het punt p. *h(p)* is een wegings- of filterfunctie.

In figuur 2.4 is een eenvoudige doosfunctie te zien. Dit is een zeer eenvoudige en veelgebruikte filterfunctie die het gemiddelde van alle radiantiewaarden door één pixel berekent. Een realistischer cameramodel is te vinden in [KMH95]. Hierbij wordt de camera gemodelleerd met een reeële lens zodat er een gevoel van diepte ontstaat doordat voorwerpen op verdere afstand er vager uitzien.

2.4 Rendering vergelijking

De rendering vergelijking werd voor het eerst beschreven door Kajiya [Kaj86]. Deze geeft aan voor elk punt van de scène en voor elke uitgaande richting wat de waarde van de

radiantie is. Hieruit worden de radiantiewaarden van het beeldvlak berekend en wordt voor elke pixel de bijhorende radiantie gevonden. De rendering vergelijking wordt in deze paragraaf afgeleid.

De radiantie in een punt x in de richting Θ is gelijk aan

$$L(x \rightarrow \Theta) = L_e(x \rightarrow \Theta) + L_r(x \rightarrow \Theta)$$

Hierbij is L_e gelijk aan de zelf-uitgestraalde radiantie en L_r de radiantie die gereflecteerd wordt. Objecten die geen lichtbron zijn, hebben geen zelf-uitgestraalde radiantie. De BRDF in het punt x is gelijk aan

$$f_r(x, \Psi \leftrightarrow \Theta) = \frac{dL_r(x \rightarrow \Theta)}{L(x \leftarrow \Psi)\cos(N_x, \Psi)d\omega_\Psi}$$

Hieruit kan de gereflecteerde radiantie gehaald worden:

$$L_r(x \rightarrow \Theta) = \int_{\Omega_x} f_r(x, \Psi \leftrightarrow \Theta)L(x \leftarrow \Psi)\cos(N_x, \Psi)d\omega_\Psi$$

Uiteindelijk wordt de rendering vergelijking bekomen:

$$L(x \rightarrow \Theta) = L_e(x \rightarrow \Theta) + \int_{\Omega_x} f_r(x, \Psi \leftrightarrow \Theta)L(x \leftarrow \Psi)\cos(N_x, \Psi)d\omega_\Psi$$

Deze vergelijking bestaat ook in andere vormen. Zo is het mogelijk om de radiantie in een punt x te schrijven als de integraal over alle oppervlakken van de scène. Er wordt verondersteld dat er geen participerend medium is zodat geschreven kan worden dat de invallende radiantie in een punt x gelijk is aan de uitgaande radiantie in een punt y:

$$L(x \rightarrow \Theta) = L(y \rightarrow -\Theta)$$

De rendering vergelijking wordt dan gelijk aan:

$$L(x \rightarrow \Theta) = L_e(x \rightarrow \Theta) + \int_A f_r(x, \Psi \leftrightarrow \Theta)L(y \leftarrow -\Psi)V(x, y)G(x, y)dA_y$$

$V(x, y)$ is de visibiliteitsfunctie. Deze duidt aan ofdat het punt x *zichtbaar* is vanuit y en omgekeerd. $G(x, y)$ wordt de geometrische term genoemd:

$$G(x, y) = \frac{\cos(N_x, \Phi)\cos(N_y, -\Phi)}{r_{xy}^2}$$

r_{xy} is de afstand tussen de punten x en y.

2.5 Algoritmes

In dit deel geven we een beknopt overzicht over de belangrijkste algoritmes om het globale belichtingsprobleem op te lossen. Voor meer informatie over globale belichting verwijzen we naar [DBB03].

Er zijn twee grote klassen van algoritmes te onderscheiden die de rendering vergelijking proberen op te lossen. Een eerste klasse gebruikt een eindige elementenmethode en heeft een scène-gebaseerde aanpak. Deze aanpak laat toe om op voorhand voor heel de scéne radiantiewaarden te berekenen en vervolgens te visualiseren. Bij een beeldgebaseerde aanpak moet voor elk camerastandpunt de radiantiewaarden opnieuw berekend worden. De laatste jaren zijn de eerste soort algoritmes steeds minder aantrekkelijk geworden. De belangrijkste nadelen zijn:

- Een driedimensionale opsplitsing van de scéne is nodig.
- Er is een beperking op de materiaalmodellen die gebruikt kunnen worden.
- Reflecties zijn moeilijk te berekenen.
- De nauwkeurigheid is niet op pixelniveau.

In deel 2.5.2 wordt deze aanpak kort besproken.

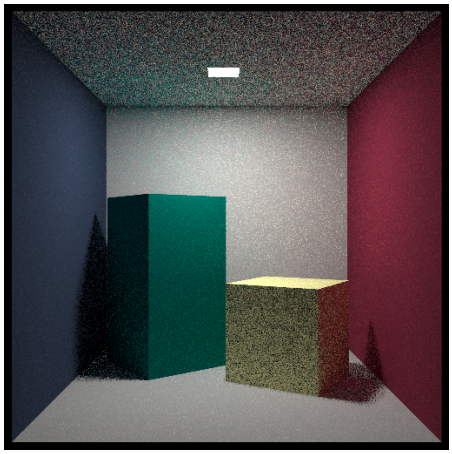
De algoritmes uit de tweede klasse lossen de rendering vergelijking op via numerieke integratie en zijn beeldgebaseerd. Zij kennen de voorgaande problemen niet. Een veel gebruikte numerieke integratietechniek is Monte Carlo integratie. Meer informatie hierover is te vinden in bijlage A. Deze algoritmes worden besproken in deel 2.5.1.

Tenslotte worden ook de belangrijkste hybride algoritmes besproken. Deze bouwen verder uit concepten beide klassen. Een zeer effectieve en robuste techniek is het photon mapping algoritme dat in deel 2.5.3 besproken wordt. Het concept *photon map* is zeer belangrijk voor dit eindwerk. Andere globale belichtingsalgoritmes worden kort besproken in deel 2.5.4.

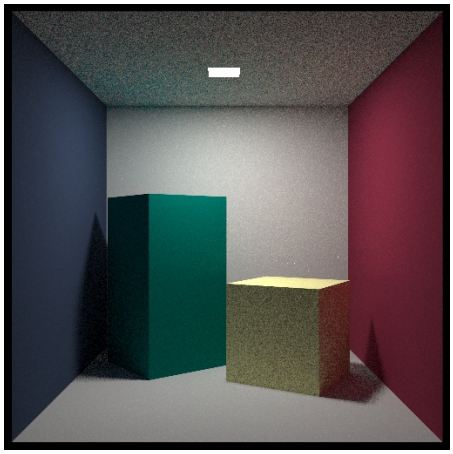
2.5.1 Path tracing

De principes van Monte Carlo integratie kunnen gebruikt worden bij het berekenen van de rendering vergelijking.

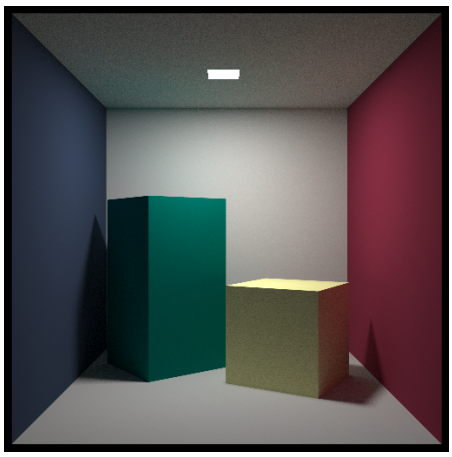
Om de radiantie $L(x \rightarrow \Theta)$ in het punt x te berekenen is het nodig dat de hemisfeer wordt bemonsterd. Dit gebeurt door een richting Ψ te kiezen aan de hand van een bekende kansdichtheidsfunctie. Vanuit het punt x wordt er een straal geschoten in de richting Ψ. Deze straal botst in het punt y. Er moet nu worden gezocht naar de radiantie



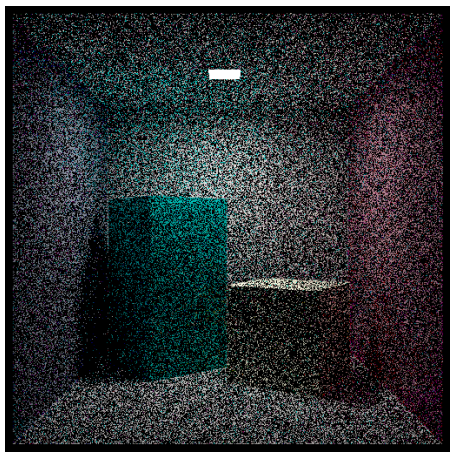
(a) 4 bemonsteringen



(b) 16 bemonsteringen



(c) 128 bemonsteringen



(d) 128 bemonsteringen, zonder afzonderlijk bemonstering van lichtbronnen

Figuur 2.5: Resultaten van het path tracing algoritme met verschillend aantal bemonsteringen per pixel.

in het punt y, die uitgestraald wordt naar het punt x. Dit kan gevonden op een analoge manier, door weer stralen te schieten in willekeurige richtingen en de waarden van de radiantie. De recursie stopt indien de straal geen voorwerp raakt.

Indien dit om een naieve manier wordt uitgevoerd, zal het door de recursie zeer lang duren eer er convergentie naar een oplossing optreedt. Er zijn twee mogelijkheden om dit te versnellen:

- Er kan gewerkt worden met een maximale recusiediepte. De invloed van zeer lange paden is in de meeste gevallen verwaarloosbaar. Deze oplossing is echter niet onvervalst omdat niet alle gevallen de lange paden verwaarloosd mogen worden.
- Een mogelijkheid die wel nog steeds een onvervalste schatter oplevert is het gebruik van russische roulette. Dit wordt geïllustreerd aan de hand van het volgende voorbeeld. Stel I de gezocht integraal:

  $$I = \int_0^1 f(x)$$

  Indien de functie $f(x)$ verticaal gescalleerd wordt met een factor $1/P$ en horizontaal met een factor P wordt de integraal I gelijk aan

  $$I_{rr} = \int_0^P \frac{1}{P} f\left(\frac{x}{P}\right) dx$$

  met P kleiner of gelijk aan 1. Indien Monte Carlo integratie wordt toegepast en een uniforme kansdichtheidsfunctie wordt gebruikt om deze integraal te berekenen, wordt de volgende schatter verkregen:

  $$\langle I_{rr} \rangle = \begin{cases} \frac{1}{P} f(\frac{x}{P}) & \text{voor } x \leq P \\ 0 & \text{voor } x > P \end{cases}$$

  De verwachtte waarde van deze schatter is gelijk aan de gezochte waarde. Bij het berekenen van recursieve integralen zoals de rendering vergelijking, is er een bepaalde kans $\alpha = 1 - P$ dat de recursie stopt. Voor hogere waarden van α zal de recursie sneller stoppen, maar zal de varantie groter zijn dan indien kleine waarden voor α worden genomen.

Indien er één straal per hemisfeer wordt genomen, bekomt men het path tracing algoritme. Het is echter ook mogelijk om meerdere stralen per hemisfeer te nemen. Dit zorgt echter voor een exponentieel geheugengebruik. Kajiya merkte op dat het belangrijker is om de berekeningen te concentreren op lichtpaden die weinig reflecties ondergaan heb.

Er wordt een functie $r(x, \Psi)$ ingevoerd die een punt y teruggeeft dat wordt bekomen indien een straal vanuit x wordt geschoten in de richting Ψ en dus is

$$\begin{aligned} L(x \leftarrow \Psi) &= L(y \rightarrow -\Psi) \\ &= L(r(x, \Psi) \rightarrow -\Psi) \end{aligned}$$

De rendering vergelijk kan dan worden geschreven als

$$\begin{aligned}
L_r(x \rightarrow \Theta) &= \int_{\Omega_x} f(\Theta \leftrightarrow \Psi) L(x \leftarrow \Psi) \cos(N_x, \Psi) d\omega_\Psi \\
&= \int_{\Omega_x} f(\Theta \leftrightarrow \Psi) L(r(x, \Psi) \rightarrow -\Psi) \cos(N_x, \Psi) d\omega_\Psi \\
&= \int_{\Omega_x} f(\Theta \leftrightarrow \Psi) \left[L_e(r(x, \Psi) \rightarrow -\Psi) + L_r(r(x, \Psi) \rightarrow -\Psi) \right] \cos(N_x, \Psi) d\omega_\Psi \\
&= \int_{\Omega_x} f(\Theta \leftrightarrow \Psi) L_e(r(x, \Psi) \rightarrow -\Psi) \cos(N_x, \Psi) d\omega_\Psi \\
&\quad + \int_{\Omega_x} f(\Theta \leftrightarrow \Psi) L_r(r(x, \Psi) \rightarrow -\Psi) \cos(N_x, \Psi) d\omega_\Psi \\
&= L_{\text{direct}}(x \rightarrow \Theta) + L_{\text{indirect}}(x \leftarrow \Theta)
\end{aligned}$$

Door de bijdrages van de directe en indirecte belichting afzonderlijk te berekenen wordt een snellere convergentie bekomen. Dit is te zien in figuur 2.5.

De berekening van de directe belichting gebeurt door middel van *schaduwstralen* die de lichtbronnen bemonsteren. Door middel van bemonstering volgens belang [1] kan de indirecte belichting sneller berekend worden. Voor verdere optimalisaties wordt er verwezen naar [DBB03].

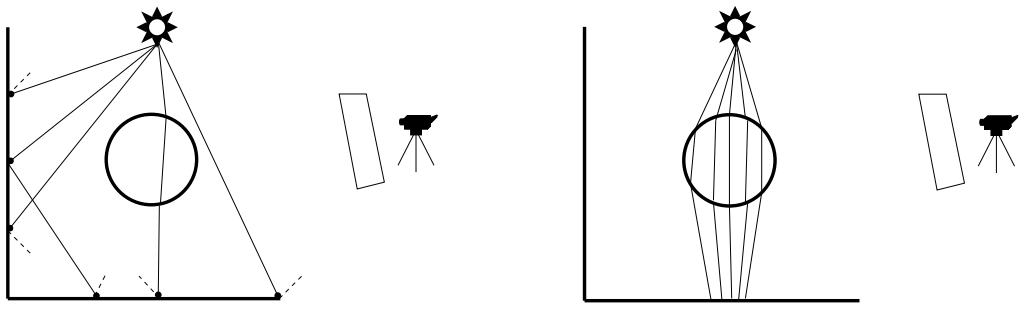
Aangezien dat Monte Carlo integratie een stochastisch proces is, zal er ruis op treden. De fout zal dalen met $1/\sqrt{N}$, indien N het aantal bemonsteringen per pixel is. Voor een ruisvrij beeld zullen er daarom typisch 1.000 tot 10.000 bemonsteringen per pixel nodig zijn, wat zeer veel is. Path tracing is in pure vorm bijgevolg een zeer traag.

2.5.2 Radiositeit

Vroegere radiositeitsalgoritmes berekenden de radiositeit voor elke veelhoek in de scène. Van daar is de term afkomstig. Tegenwoordig wordt de term gebruikt voor een hele klasse van algoritmes die gebaseerd zijn op eindige elementenmethodes. Het oppervlak van de objecten in de scène wordt hiervoor opgedeeld in kleine stukjes, bijvoorbeeld driehoeken. Er worden de volgende veronderstellingen gemaakt:

- Alle lichtbronnen zijn diffuse lichtbronnen. $L_e(x \rightarrow \Theta)$ is onafhankelijk van Θ.
- Elk oppervlak bestaat uit een diffuus materiaal. $f_r(x, \Theta \leftrightarrow \Psi)$ is onafhankelijk van Θ en Ψ.
- De radiantie wordt benaderd door middel van een stuksgewijze functie. In een klein stukje oppervlak wordt de radiantie verondersteld constant te zijn.

[1] De Engelse term hiervoor is *importance sampling*.



Figuur 2.6: Photon mapping maakt gebruik van twee soorten photon maps. In (a) wordt de *globale photon map* getoond. Deze bewaart de bijdrages van alle types belichting. Om caustics snel te renderen wordt er een tweede photon map gebruikt, de *caustic map*. Deze is in (b) te zien. Enkel de fotonen die bijdragen tot de vorming van caustics worden hier in opgeslagen.

Samengevat gaat een radiositeitsalgortime als volgt. Het lichttransport voor de hele scène moet worden berekend. Vervolgens wordt een camerastandpunt gekozen en wordt de scène gevisualiseerd. Deze laatste stap kan zeer snel gebeuren zodat interactieve visualisaties mogelijk zijn.

1. Eerste wordt de scène in kleine stukjes, veelhoeken, verdeeld.
2. Vervolgens worden de vormfactoren berekend. Deze duiden aan hoe dat het lichttransport tussen de verschillende elementen verloopt.
3. De evenwichtstoestand van de scéne kan benaderd worden door een lineair systeem. Hieruit worden de radiositeitswaarden van elke veelhoek berekend.
4. Daarna wordt de scène gevisualiseerd aan de hand van de stand van de virtuele lichtsensor.

Een voordeel ten opzichte van Monte Carlo gebaseerde methodes is dat er geen ruis optreedt, maar aan randen van de veelhoeken kunnen er discontinuïteiten te zien zijn. Dit laatste kan zeer storend zijn. Via interpolatie kan men de discontinuïteiten enigszins wegwerken.

2.5.3 Photon mapping

Photon mapping is een praktische en robuuste twee-stapsmethode. Het maakt gebruik van een speciale structuur, de photon map, om een benadering op te stellen voor de irradiantie. In deze structuur worden *fotonen* opgeslagen. Deze bevatten informatie over

inkomende flux in de scène. Het gebruik van de photon map zorgt ervoor dat caustics en indirecte belichting op een snellere manier gerenderd worden dan (bidirectionele) path tracing.

Net zoals bij bidirectionele path tracing wordt er vertrokken van de camera en vanuit de lichtbronnen. De informatie vanuit de lichtbronnen wordt bewaard in de photon map in de vorm van fotonen. Deze fotonen worden vanuit de lichtbronnen geschoten in de scène en worden op de oppervlakken van de objecten van de scène bewaard. Afhankelijk van de eigenschappen van het materiaal is er een bepaalde kans dat het foton verder verstrooid zal worden doorheen de scène. In de photon map wordt er opgeslagen waar dat het foton is ingeslagen en welk vermogen en richting het foton heeft.

Een belangrijke voordeel van photon mapping is dat caustics hier zeer efficiënt gerenderd mee kunnen worden. Caustics treden op indien licht wordt gefocust. Er zullen dus niet te veel fotonen nodig zijn om een goede schatting te krijgen. Caustics treden op in een beperkt deel van de scène. De caustic map is een photon map die enkel fotonen bevatten die bijdragen tot de visualisatie van caustics. De globale photon map is een photon map die alle soorten fotonen bevat, ook deze die bijdragen tot caustics. Door middel van twee photon maps te gebruiken, wordt het mogelijk om caustics onmiddelijk te visualiseren via de caustic map. In figuur 2.6 worden deze voorgesteld.

Photon mapping is echter geen onvervalst algoritme. Dit wil zeggen dat indien het algoritme meermaals wordt uitgevoerd, er gemiddeld geen juist oplossing wordt bekomen. Photon mapping is gelukkig wel een consistent. Indien er genoeg fotonen in de photon map zitten wordt er toch een correct beeld verkregen.

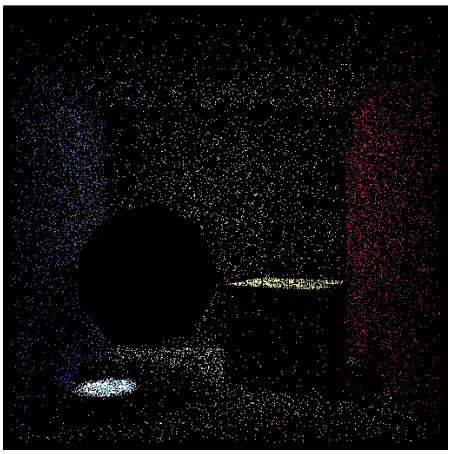
Samengevat verloopt photon mapping als volgt:

1. Er worden fotonen uit de lichtbronnen geschoten. Deze worden bewaard in de photon maps.
2. Er wordt gebruikt gemaakt van de photon maps bij het renderen.

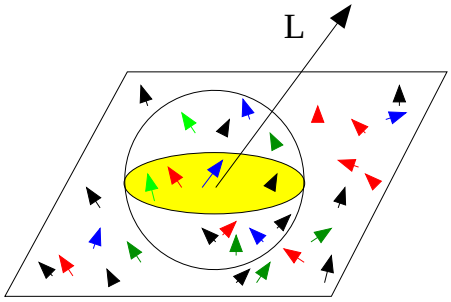
Voorstelling van de photon map

Er moet veel gezocht worden naar fotonen. Een structuur waar efficiënt zoeken naar de k dichtste buren mogelijk is bijgevolg een noodzaak. Indien de fotonen uniform verdeeld zijn over de ruimte is een 3 dimensionaal rooster waarbij de fotonen zich in kubussen bevinden een bijna optimale structuur. De photonen zijn echter niet uniform verspreid aangezien deze meestal op oppervlaktes worden bewaard.

Een betere oplossing is het gebruik van een kd-boom. Dit is een multi-dimensionale binaire zoekboom. In elk van de knopen wordt er 'één van de dimensies opgesplitst. Een eendimensonale kd-boom is een binaire boom. De photon map bestaat uit driedimensionale punten en dus is een driedimensionale boom nodig. In elk van de knoop wordt er



Figuur 2.7: Fotonen in de photon map



Figuur 2.8: Schatting van de radiantie aan de hand van de photon map

een verwijzing naar een foton bijgehouden. In de knopen wordt ook bijgehouden volgens welke as er wordt opgedeeld en verwijzingen naar de linkse en rechtse deelbomen. De fotonen in de links respektievelijk rechtse deelboom bevinden zich boven respektievelijk onder het vlak door de foton en evenwijdig met een bepaald as. Dit zorgt er voor dat er gemiddelde gezocht kan worden in $O(\log n)$ tijd. In het slechtste geval wordt er gezocht in $O(n)$ tijd. Het kan worden aangetoond dat indien er gezocht wordt naar de k dichtste fotonen, deze gemiddeld worden gevonden in $O(k + \log(n))$.

Het balanceren heeft ook het voordeel dat de voorstelling zeer compact is. In figuur 2.5.3 zijn de bewaarde fotonen in de photon map te zien.

Schatting van de radiantie

De fotonen in de photon map stellen een benadering op voor de inkomende flux van de scène. De dichtheid van de fotonen is dus gelijk aan de irradiantie.

De rendering vergelijk stelt dat

$$L_r(x \rightarrow \Theta) = \int_{\Omega_x} f_x(\Theta \leftrightarrow \Psi) \cos(N_x, \Psi) L(x \leftarrow \Psi) d\omega_\Psi$$

De relatie tussen inkomende flux en radiantie is gelijk aan

$$L(x \leftarrow \Psi) = \frac{d^2\Phi}{\cos(N_x, \Psi) d\omega_\Psi dA}$$

De radiantie kan dus herschreven worden als

$$L_r(x \rightarrow \Theta) = \int_{\Omega_x} f_x(\Theta \leftrightarrow \Psi) \frac{d^2\Phi}{dA}$$

De inkomende flux Φ kan benaderd worden door middel van de k dichtste fotonen in de buurt van het punt x. Deze k dichtste fotonen bevinden zich in een bol. Dit is te zien in figuur 2.5.3. Er wordt verondersteld dat het oppervlak rond het punt x waar de radiantie wordt gezocht lokaal vlak is. De bol kan dan geprojecteerd worden op een vlak. De oppervlakte van deze projectie is gelijk aan πr^2. Elk foton heeft een bepaald vermogen Φ_f en een bepaalde richting Ψ_f. De benaderde radiantie is dus gelijk aan

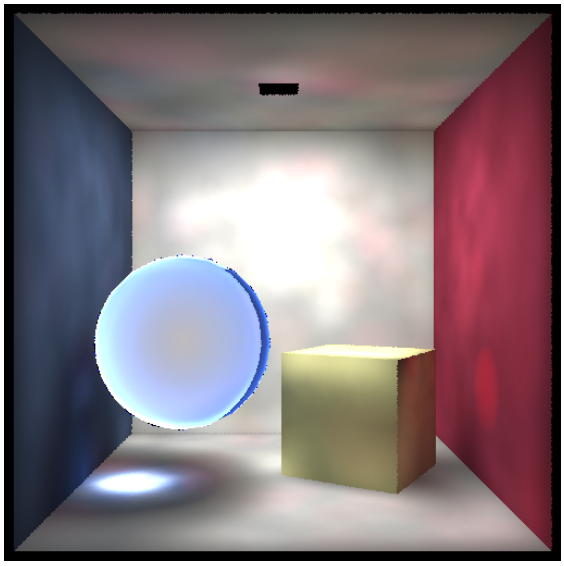
$$L_r(x \rightarrow \Theta) \approx \frac{1}{\pi r^2} \sum_{i=1}^{k} f_x(\Theta \leftrightarrow \Psi_f) \Phi_f$$

Het zoeken naar de dichtste fotonen gebeurt steeds in een bol met een constante straal. Hierdoor kan het gebeuren dat er minder dan k fotonen gevonden worden. Indien er genoeg fotonen in de photon map zijn en er gezocht wordt naar voldoende fotonen zal er een goede schatting gevonden worden de radiantie. Het optimaal aantal fotonen is echter afhankelijk van de scène.

Renderen met de photon map

De eenvoudigste manier om renderen met behulp van de photon map, is het rechtstreeks visualiseren van de photon map. Zoals te zien in figuur 2.9 is de benadering niet onmiddelijk bruikbaar. Door een pixel van het beeldvlak wordt vanuit de camera een straal geschoten. Hierdoor wordt het object dat het zichtbaar is door de camera gevonden. Op deze plaats wordt de radiantie die uitgestraald wordt naar de camera berekend. De radiantie wordt opgesplitst in vier componenten: de directe belichting, de indirecte diffuse belichting, belichting door caustics en spiegelende of glossy belichting. Elk van deze componenten wordt afzonderlijk berekend:

Directe belichting De bijdrage door de directe belichting wordt berekend door het rechtstreeks te bemonsteren van de lichtbronnen, zoals beschreven is in 2.5.1.



Figuur 2.9: Rechtstreekse visualisatie van de photon map.

Glossy en spiegelende reflecties De bijdrages worden gevonden door gebruik te maken van ray tracing.

Caustics Caustics worden gerenderd door directe visualisatie van de caustic map.

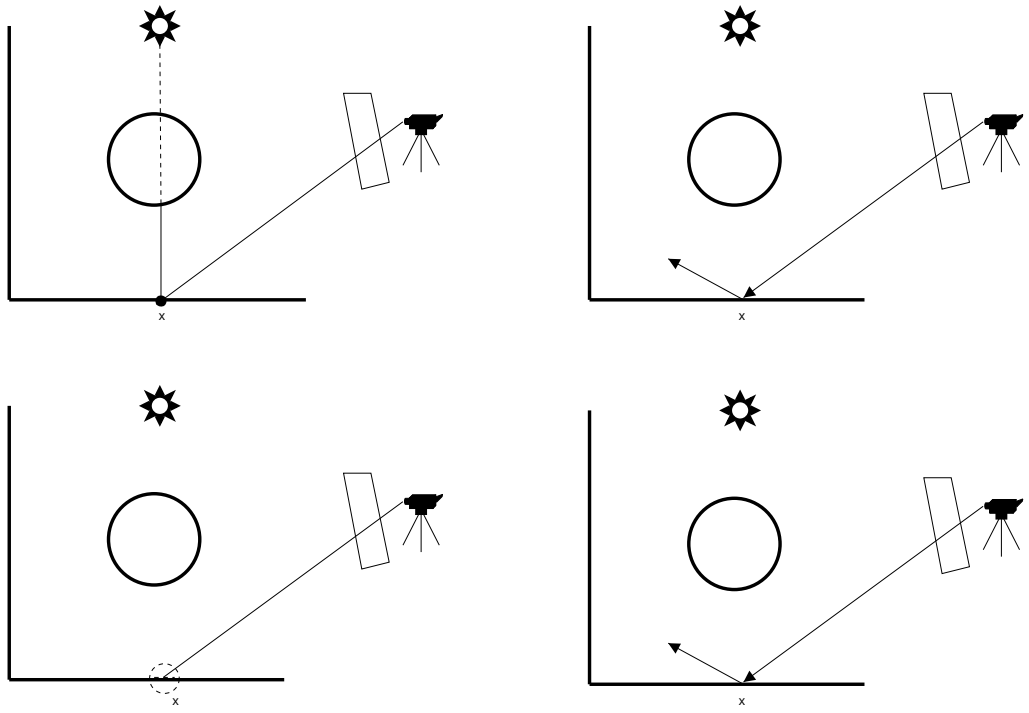
Indirecte diffuse belichting Door de hemisfeer te bemonsteren en stralen te schieten volgens technieken gezien in 2.5.1 wordt deze bijdrage berekend. Indien een diffuus oppervlak geraakt wordt, wordt de globale photon map gebruikt om de radiantie te berekenen.

Deze componenten worden opgeteld om de totale radiantie te verkrijgen. In figuur 2.10 worden de verschillende componenten en hun berekeningswijze met behulp van photon mapping voorgesteld.

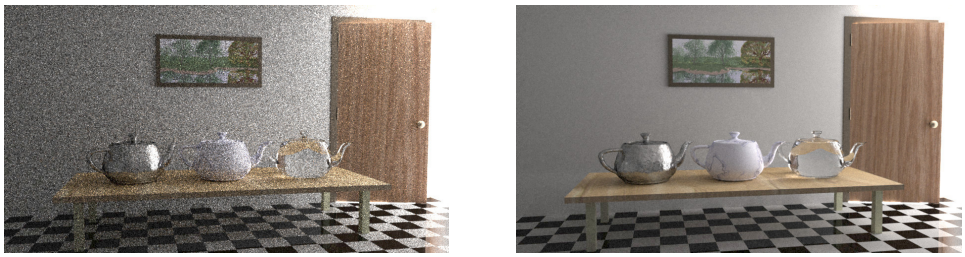
Verder informatie over photon mapping is te vinden in [Jen01].

2.5.4 Andere algoritmes

In deze sectie stellen we kort enkele andere belangrijke globale belichtingsalgoritmes voor.



Figuur 2.10: Renderen van de verschillende belichtingscomponenten via photon mapping. (a) De berekening van de *directe belichting* gebeurt door middel van het bemonsteren van de lichtbronnen. (b) Speculaire en glossy componenten worden berekend met behulp van path tracing. Deze vertonen een piek in hun BRDF, waardoor een te grote hoeveelheid fotonen nodig is om voor een goed benadering te krijgen. (c) Caustics worden gevisualiseerd door middel van de caustic map. (d) Indirecte diffuse belichting wordt berekend door de hemisfeer te bemonsteren en stralen te schieten. Indien een diffuus oppervlak wordt geraakt, wordt de benadering uit de globale photon map gebruikt.



(a) Bidirectionele path tracing

(b) Metropolis lichttransport algoritme

Figuur 2.11: Voorbeeld van een complexe scéne. Beide beelden werden in dezelfde tijd gerenderd.

Bidirectionele path tracing

In het path tracing algoritme wordt er steeds vertrokken vanuit de virtuele camera. Lichteffecten zoals caustics zijn hierdoor zeer moeilijk te berekenen. Bidirectionele path tracing [LW93] zorgt er daarom voor dat er niet enkel vanuit de camera wordt vertrokken, maar ook vanuit de lichtbronnen. Om geldige lichtpaden te vinden, worden de lichtpaden uit de lichtbron verbonden met de lichtpaden uit de camera. Er moet wel rekening gehouden worden dat de Monte Carlo schatter nog steeds juist wordt berekend.

Het algoritme zorgt voor een verbetering van lichtpaden die moeilijk te berekenen zijn met path tracing zoals caustics.

Metropolis lichttransport algoritme

Het Metropolis lichttransport algoritme (MLT) werkt op een andere manier dan de voorgaande algoritmes. Er worden geen bemonsteringen van de te integreren functie genomen, maar er worden bemonsteringen gegeneerd die evenredig zijn met deze functie. Dit zorgt ervoor dat het beeld wordt bemonsterd met een densiteit van stralen die evenredig is met de radiantie. Het werk wordt gefocust op de heldere delen van het beeld.

Het algoritme wordt gestart door middel van een aantal lichtpaden te generen met behulp van (bidirectionele) path tracing. Deze paden worden gekopieerd en gemuteerd. Indien dit nieuwe pad niet geldig is (het pad gaat bijvoorbeeld door een muur), wordt het weggegooid en het huidige pad wordt opnieuw gebruikt om een bijdrage te leveren aan het beeld en als basis voor een nieuwe mutatie. Indien dit nieuwe pad geldig is dan wordt het met een bepaalde kans geaccepteerd.

Het algoritme is zeer goed in situaties die anders zeer moeilijk te berekenen zijn. Dit zijn gevallen waarvoor een klein gebied verantwoordelijk is voor het grootste deel van de belichting. Een voorbeeld is een donkere kamer waarbij het licht afkomstig is vanuit een naburige kamer waarbij de deur tussen de kamers op een kier staat (zie figuur 2.11). Eens dat er een belangrijk lichtpad gevonden is, zal dit lichtpad gemuteerd worden om naburige belangrijke paden te vinden. Voor scènes met een normale belichting zal het algoritme meestal weinig verbetering brengen. Dit is te wijten aan het feit dat de hele ruimte van lichtpaden belangrijk is. Enkel bij de aanwezigheid van caustics kan een eventuele verbetering mogelijk zijn. Voor scénes met weinig coherentie, zoals bijvoorbeeld een kubus met vele gaten, zal zelfs een verslechtering ten op zicht van path tracing merkbaar zijn.

Het algoritme heeft ook vele parameters die afhankelijk zijn van de te renderen scène. Er zijn bijvoorbeeld vele mogelijkheden om paden te muteren. Het is daarom zeer moeilijk op een goede manier te implementeren.

Verdere informatie over het algoritme is te vinden in [VG97].

Meerstapsalgoritmes

Een meerstapsalgoritme gebruikt verschillende algoritmes en combineert deze resultaten van deze methodes met verschillende wegingsfactoren. Meer informatie over de constructie van meerstapsalgoritmes is te vinden in [SW99].

Irradiance Caching

Irradiance caching is een zeer doeltreffende techniek om de berekening van indirecte diffuse belichting te versnellen. Het kan zeer goed gebruikt worden in combinatie met photon mapping. De berekening van indirecte belichting is zeer kostelijk via Monte Carlo integratie. Om een voldoende geconvergeerd beeld te krijgen zijn er vele bemonsteringen nodig. Zelfs indien er gebruik wordt gemaakt van photon mapping zijn er nog relatief veel bemonsteringen nodig. Het is daarom belangrijk om deze berekening te versnellen.

Het basisprincipe achter irradiance caching is dat berekende irradiantiewaarden bewaard worden en hergebruikt worden. Enkel voor diffuse oppervlakken kan dit efficiënt berekend worden. De berekening van de benaderde irradiantie waarden maakt gebruik van rotatie- en translatiegradiënten. Ze duiden aan hoe dat de irradiantiewaarden veranderen met positie en richting.

Een verdere bespreking is te vinden in [WH92].

Hoofdstuk 3

Analyse en rendering van lichttransportcomponenten

Dit hoofdstuk bespreekt de techniek besproken in de paper Light Transport Component Analysis & Rendering, O. Dumont en Ph. Dutré [DD05]. Het biedt de basis voor het eigen werk en werd initieel in parallel met onze methode ontwikkeld.

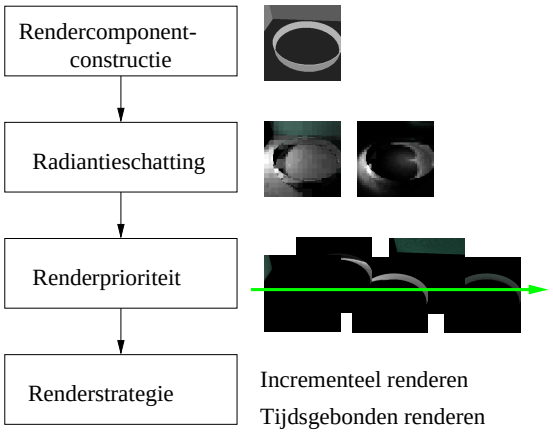
In deel 3.1 worden de problemen van globale belichting (hoofdstuk 2) aangekaart. Een algemene oplossingsstrategie wordt voorgesteld in deel 3.2. De nieuwe aanpak voert een paar nieuwe concepten in zoals rendercomponenten (deel 3.3) en photon mapping met lichtpadtypes (deel 3.4). Verschillende delen van het beeld worden eerst benaderd (deel 3.5) en vervolgens in een bepaald volgorde (deel 3.6) berekend met een bepaalde strategie, besproken in deel 3.7. Vervolgens wordt er in deel 3.8 nagegaan op welk vlak deze paper verbeterd of verfijnd kan worden. Dit vormt het aansluitingspunt met het hoofdstuk 6 dat een paar verfijningen doorvoert.

Om te besluiten wordt er in deel 3.9 nog kort een iets verteld over de implementatie van het raamwerk BLEED waarin deze methode werd uitgewerkt.

3.1 Probleemstelling

Er bestaan veel methodes die pogen globale belichting te versnellen. Vaak zijn ze echter gericht op een bepaald effect in een scène. De methode moet dus als het ware gekozen worden in functie van de te renderen scène. Photon mapping [Jen96][Jen01] werkt bijvoorbeeld het best bij aanwezigheid van caustics, Metropolis licht transport [VG97] bij een grote graad van indirecte belichting, en lokale belichtingsomgevingen [FBG02] bij een groot aantal lichtbronnen.

Een veelvoorkomende eigenschap van deze methodes is dat ze meer rekentijd willen be-



Figuur 3.1: Overzicht van de aanpak. (uit [DD05])

steden aan delen van de scène die veel bijdragen aan het beeld. Dit zorgt voor een vermindering van de variantie in het beeld en het zorgt dus ook voor een snellere convergentie ervan. Deze paper gaat na of dit proces niet adaptief kan gebeuren in functie van de scène.

Een tweede aandachtspunt is het sneller bekomen van beelden, zij het onvolledig. Concreet zouden de tussenresultaten bruikbaar moeten zijn; elk voorwerp moet al zijn globale kleur dragen en er mag geen ruis zichtbaar zijn.

3.2 Algemene aanpak

De aanpak staat geïllustreerd in figuur 3.1

Eerst wordt het werk opgedeeld in verschillende *componenten*. Er wordt zowel op basis van gebied binnen het beeld als op basis van soort belichting (lichtpadtype) opgesplitst. Het totale beeld is dan de som van al de berekende componenten. Het opsplitsen wordt besproken in deel 3.3.

Componenten kunnen al dan niet berekend zijn. Indien ze dat niet zijn, worden ze benaderd door een uniforme radiantie over hun oppervlak. Deel 3.5 bespreekt hoe men tot deze benadering komt. Er kan vertrokken worden van een volledig benaderd beeld en stapsgewijs componenten vervangen door ze te renderen.

Men heeft er baat bij eerst de meest relevante componenten te synthetiseren. Sommige kunnen bijvoorbeeld niet of nauwelijks bijdragen tot het eindbeeld. Deze kunnen tot op het einde uitgesteld worden. Een heuristiek voor de volgorde van de berekeningen wordt naar voren gebracht in deel 3.6.

Tenslotte moet er beslist worden hoe lang men een bepaalde component wil renderen. Dit hangt ondermeer af van welke strategie er gekozen wordt: incrementeel of tijdsgebonden, zie deel 3.7.

3.3 Rendercomponenten

Een eerste stap is dus het opsplitsen van het werk. Een werkeenheid noemt men een *component*. Door componenten te definiëren kan men de volgorde van de berekeningen beïnvloeden. In een klassiek renderproces is men namelijk vrij te beginnen renderen wat of waar men wenst. Het gebruik van componenten probeert deze vrijheidsgraad nuttig in te vullen.

Men kan meerdere manieren bedenken om het beeld in componenten op te delen:

in gebieden Groepen pixels worden toegekend aan componenten. Hiermee kan men beslissen welk deel van het beeld eerst gesynthetiseerd wordt.

in RGB Elk kleurkanaal wordt toegekend aan een component. Kleuren worden afzonderlijk gevisualiseerd. Dit kan ook toegepast worden in andere kleurencoderingen. Zo kan men bijvoorbeeld met YIQ encodering [Pra91] onderscheiden tussen luminantie en kleur.

volgens lichtbron De bijdrages van elke lichtbron wordt toegekend aan een verschillende component. Deze aanpak biedt de mogelijkheid om de invloed van lichten op het beeld op te splitsen.

volgens lichtpadtypes Het criterium om bijdrages hier te onderscheiden is het soort pad dat het licht heeft afgelegd van lichtbron tot sensor. Het concept lichtpadtype wordt in deel 3.3.1 verder toegelicht.

Elke opsplitsing biedt mogelijkheden, maar hier werd geopteerd voor opsplitsing in gebieden en volgens lichtpadtypes.

De resulterende opsplitsing kan opgevat worden als een cartesiaans product van de twee partities. Indien er twee gebieden g_1 en g_2 zijn, en er eveneens twee lichtpadtypes l_1 en l_2 gekozen worden, resulteert dat in vier (2×2) componenten: (g_1, l_1), (g_1, l_2), (g_2, l_1) en (g_2, l_2). Dit is de werkwijze in deze paper.

Een alternatieve manier om opdelingen te combineren is de ene afhankelijk maken van de andere. Bijvoorbeeld: het lichtpadtype l_1 wordt opgedeeld in gebieden g_{1a} en g_{1b}, terwijl het lichtpadtype l_2 slechts een gebied g_2 bevat. Dit levert dus drie componenten op: (l_1, g_{1a}), (l_1, g_{1b}) en (l_2, g_2). Deze manier van combineren zal gebruikt worden in hoofdstuk 6.

Type oppervlak	Notatie
lichtbron	**L**
diffuus oppervlak	**D**
glossy oppervlak	**G**
speculair oppervlak	**S**
lichtsensor	**E**

Tabel 3.1: Notaties voor interacties met oppervlakken in een lichtpad

3.3.1 Lichtpadtypes

Lichtpad

Een *lichtpad* is een pad dat een lichtcontributie aflegt van lichtbron naar sensor. De eigenschappen van zo'n pad bepalen of dit licht bijvoorbeeld directe belichting, color bleeding of caustic is:

- Bij directe belichting interageren de lichtcontributies eenmaal met een oppervlak buiten de lichtbron en de sensor, zoals in figuur 3.2(b).
- Bij color bleeding zijn er typisch twee snel opeenvolgende diffuse botsingen.
- Caustics hebben een opeenvolging van lichtbron, speculaire reflectie of refractie en een diffuus oppervlak zoals in figuur 3.2(c).

Dit zijn effecten die duidelijk te onderscheiden zijn.

In [Hec90] wordt een notatie ingevoerd om lichtpaden te karakteriseren volgens de verschillende interacties met oppervlakken. De mogelijke interacties staan opgesomd in tabel 3.1.

Op deze manier wordt elk pad voorgesteld door **L**, een opeenvolging van **D**, **G** of **S** en uiteindelijk **E**. Voorbeelden:

LE Een rechtstreeks pad van lichtbron naar sensor.

LDE Een pad dat eenmaal diffuus botst.

LSSSE Een pad dat een paar keer wordt gespiegeld of gebroken.

Het kan echter ook interessant zijn onderscheid te maken tussen verschillende lichtbronnen of verschillende voorwerpen. Voor lichtbronnen is de notatie dan $\mathbf{L_i}$ waar i de lichtbron identificeert. Zodoende kan men onderscheid maken tussen lichteffecten van

Symbool	Betekenis
( )	groepering van symbolen
*	een aantal keer het vorige symbool
+	minstens eenmaal het vorige symbool
[ ]	eenmaal een van de symbolen

Tabel 3.2: Reguliere expressies

Omschrijving	Reguliere expressie
Lichtbronnen die direct zichtbaar zijn, eventueel door speculaire reflecties	$\mathbf{LS^*E}$
Directe belichting + spec.	$\mathbf{L[DG]S^*E}$
Caustics + spec.	$\mathbf{LS^+[DG]S^*E}$
Indirect diffuus + spec.	$\mathbf{L[DG][DG]S^*E}$
Hoogdiffuus + spec.	$\mathbf{L[SDG]^+[DG][DG]S^*E}$
Diffuse caustics + spec.	$\mathbf{L[SDG]^*[DG]S^+[DG]S^*E}$

Tabel 3.3: Lichtpadtypes als reguliere expressies

verschillende lichtbronnen. Dit hadden we reeds voorgesteld in het begin van dit deel bij het opsplitsen van componenten volgens lichtbron.

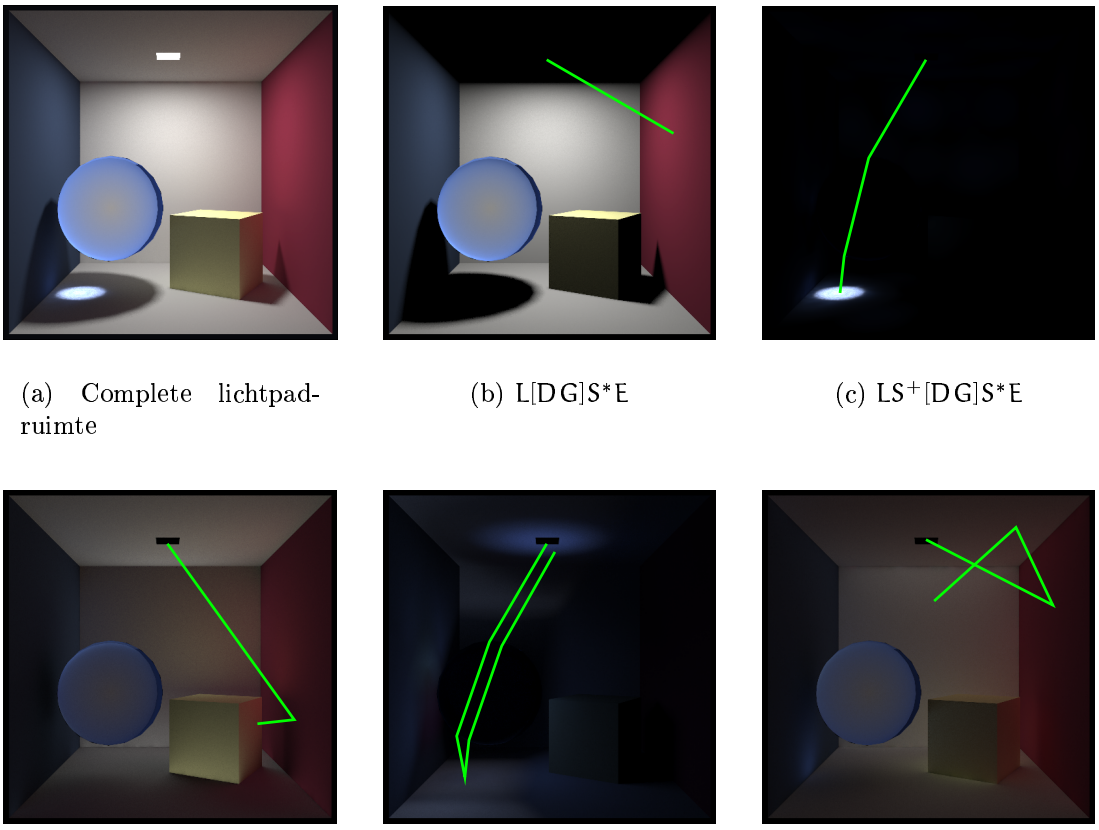
Bijvoorbeeld $\mathbf{L_1D_9E}$: Een pad dat vertrekt van lichtbron 1 en diffuus botst op voorwerp 9 eer op de sensor in te vallen. De **E** krijgt nooit een subscript aangezien er slechts één sensor aanwezig is in een scène.

Lichtpadtype

Een *lichtpadtype* is een uitdrukking dat verschillende lichtpaden karakteriseert. Een methode hiervoor is gebruik maken van reguliere expressies [Sip01]. Een kort overzicht van de syntax vindt men terug in tabel 3.2.

Een paar voorbeelden van lichtpadtypes beschreven met behulp van reguliere expressies worden gegeven in tabel 3.3 en geïllustreerd in figuur 3.2.

Deze expressies kunnen bij aanwezigheid van indices van de interagerende oppervlakken uitgebreid worden om bijvoorbeeld de lichtbronnen te onderscheiden. Als illustratie $\mathbf{L_1D_*^*E}$: deze reguliere expressie karakteriseert alle lichtpaden vertrekkend van lichtbron 1, een willekeurig aantal keren botsend op een willekeurig diffuus voorwerp en de sensor bereikend.



(a) Complete lichtpadruimte

(b) L[DG]S*E

(c) LS$^+$[DG]S*E

(d) L[DG][DG]S*E

(e) L[X]*[DG]S$^+$[DG]S*E

(f) L[X]$^+$[DG][DG]S*E

Figuur 3.2: Lichtpadtypes met voorbeeldpaden in het groen: (a) de volledig gerenderde lichtpadruimte, (b) enkel directe belichting, (c) enkel caustics, (d) enkel indirecte belichting met 2 diffuse botsingen, (e) en (f) meer exotische lichtpadtypes. (uit [DD05])

3.3.2 Geometrische segmentatie

De opsplitsing in gebieden gebeurt door een opsplitsing volgens geometrie. Er wordt gezocht naar gebieden die horen bij zichtbare objecten in een scène. Per pixel wordt er nagegaan welke object er zichtbaar is door deze pixel. Pixels met hetzelfde zichtbare object worden samengevoegd in één gebied. Op de grenzen van gebieden kunnen er fouten optreden. Meerdere objecten kunnen zichtbaar zijn door de dezelfde pixel. De grenzen van de gevonden gebieden zullen daarom met één pixel worden uitgebreid. Dit zorgt voor overlappende gebieden. Bij het renderen wordt er hiermee rekening gehouden.

We hebben dit geïmplementeerd in BLEED (een nieuwe ray tracer, zie deel 3.9). Er zijn twee manieren om zichtbare objecten te vinden. De eerste manier maakt gebruik van stralen. Vanuit de camera wordt er door elke pixel een straal geschoten. Het eerste object dat geraakt wordt door deze straal is het dichtste object.

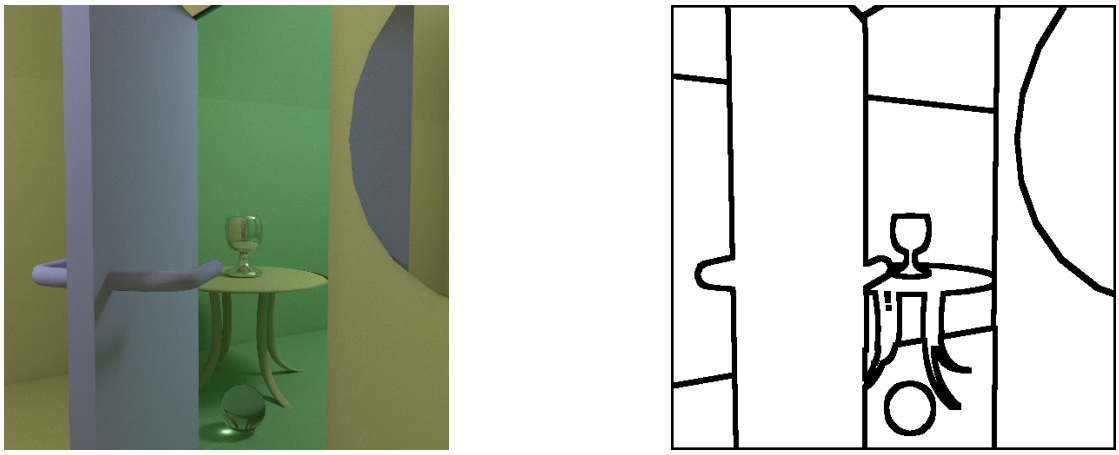
De tweede manier maakt gebruik van *rasterizen*. Hierbij worden de voorwerpen geprojecteerd op een tweedimensionaal vlak. Deze voorwerpen worden achter elkaar getekend in een beeld dat hetzelfde formaat heeft als het te renderen beeld. De voorwerpen worden echter zo getekend zodat telkens het dichtste object door elke pixel te zien is. Meer informatie over deze techniek is te vinden in [Hil00].

Een voordeel van de tweede techniek is dat deze op een efficiënte manier in hardware uitgevoerd kan worden. Een verdere uitwerking hiervan is buiten de doelstelling van de thesis. Aangezien het eenvoudig is om met stralen te werken in BLEED hebben we enkel de eerste techniek uitgewerkt. Het resultaat van een eenvoudige scène kan men zien in figuur 3.3. De segmentatie verloopt zeer snel en ligt in de grootte orde van enkele seconden voor complexe scènes. Deze tijd is te verwaarlozen ten opzichte van de totale tijd om te renderen.

3.4 Photon mapping met lichtpadtypes

Voor meerdere technieken in deze paper is een schatting van de radiantie in het beeld nodig. Deze schatting wordt bekomen met behulp van een photon map (zie deel 2.5.3) waarna de schatting wordt bekomen door per pixel de dichtste fotonen te zoeken (zie deel 2.5.3). Het aantal fotonen in de photon map mag reeds redelijk hoog gekozen worden, aangezien het hoe dan ook kan hergebruikt bij het synthetiseren. Het aantal *gezochte* fotonen bij de eigenlijke schatting wordt best wel laag gehouden. Het kan niet of amper hergebruikt worden en kan bovendien een significante impact hebben op de uiteindelijke snelheid van uitvoering.

Zuivere photon mapping is niet bruikbaar in dit geval. Men wil per pixel de bijdrage van elk lichtpadtype kunnen nagaan. Standaard photon mapping berekent enkel de radiantie in een pixel. Hiervoor bewaart elk foton enkel de radiantie. Er is geen informatie over



(a) De scène (b) De segmentatie

Figuur 3.3: Voorbeeld van geometrische segmentatie

het afgelegd pad zodat men het lichtpadtype niet kan afleiden.

Om dit op te lossen wordt er per foton meer informatie bewaard:

- Het soort van de laatste interactie: **D**, **G** of **S**.
- Een verwijzing naar het foton die deze laatste botsing heeft veroorzaakt.
- Eventueel een identificatie van het oppervlak van de laatste interactie.

Met deze extra informatie kan recursief nagegaan worden wat het deellichtpad van een foton is.

Met behulp van reguliere expressies wordt het foton geklasseerd in een lichtpadtype. Men bekomt zo een radiantie per pixel en lichtpadtype.

3.5 Benadering van componenten

Door het invoeren van componenten kan men een volgorde in de berekeningen opleggen. Indien men echter bijvoorbeeld de opsplitsing in gebieden beschouwt, heeft het niet veel zin een volgorde op te leggen zonder meer. Tot de laatste component gevisualiseerd is, zal het beeld namelijk onvolledig of *biased*[1] zijn, in die mate dat het tussenresultaat onbruikbaar is.

[1]De term bias wordt gebruikt voor methodes die hoe dan ook een verkeerd resultaat geven. In dit geval worden sommige lichteffecten in de tussenresultaten sowieso niet berekend of benaderd.

De gewijzigde photon mapping kan gebruikt worden om dit op te lossen. Voor elke component beschouwt men zijn gebied G en zijn lichtpadtype L. Men accumuleert per component alle bijdragen die ertoe behoren. De bijdragen moeten zich dus bevinden in een pixel van het gebied G en horen tot het lichtpadtype L. Door te delen door het oppervlak van het gebied G bekomt men de gemiddelde radiantie in de component.

Men vertrekt dus van een beeld van alle benaderde componenten. Wanneer een component berekend wordt, wordt de benadering vervangen door de gerenderde versie.

3.6 Analyse: volgorde van de componenten

Eens de componenten bepaald zijn, moet er een volgorde gekozen worden. Per component worden de gemiddelde *lichtpadbijdrages* geschat. Een lichtpadbijdrage is een radiantiewaarde per lichtpadtype. Elke component wordt verder opgesplitst in deelgebieden waar eveneens de radiantie wordt geschat. Wordt er veel variantie opgemerkt bij de fijnere onderverdeling, betekent dit dat de schatting over de hele component niet betrouwbaar is. Deze variantie wordt een maat om de verschillende componenten te ordenen.

3.6.1 Rechthoekige segmentatie

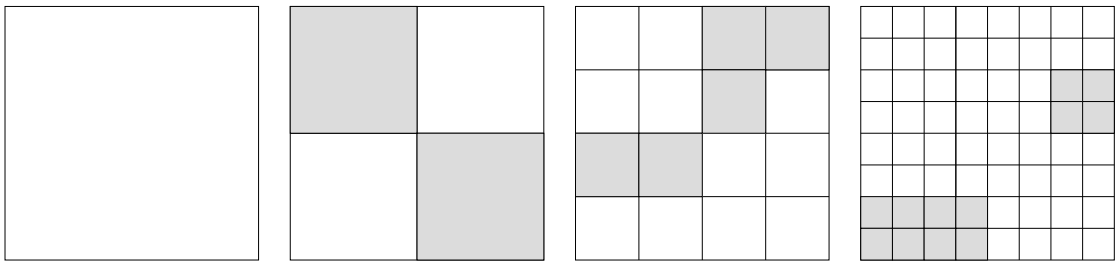
Aangezien er vertrokken is van redelijk ruwe lichtinformatie uit de photon mapping, is de informatie die men bekomt per pixel niet betrouwbaar. Er zijn namelijk weinig fotonen per pixel. Door meerdere pixels samen te nemen bekomt men meer fotonen per gecreëerd gebied zodat de statistische betrouwbaarheid stijgt.

De variantie in een pixel kan echter sterk verschillen over het beeld. Om de groottes van de nieuwe gebieden te bepalen kan men dus best adaptief te werk gaan.

Er wordt een deelbemonsterende piramide opgesteld. Het laagste niveau is het beeld van de per-pixel lichtpadbijdrages. Het hoogste niveau is één gebied met het gemiddelde van alle bijdrages. Aangezien variantie in principe daalt volgens $1/\sqrt{N}$ waarbij N het aantal bemonsteringen is, zal op dat hoogste niveau de variantie een stuk lager moeten liggen. Figuur 3.4 illustreert deze piramide.

Het algoritme voor het bepalen van het lokale niveau werkt recursief. Gegeven een rechthoekig gebied uit de piramide in laag i wordt de mogelijke fout op de waarde van de radiantie berekend. Is deze onder een bepaalde drempel, dan wordt dit gebied toegevoegd aan de segmentatie. Is deze boven die drempel, dan gaat men naar niveau $i+1$ waar de rechthoek in vier delen is opgesplitst. Voor deze vier nieuwe rechthoeken wordt weer hetzelfde principe toegepast.

Uiteindelijk bekomt met een volledige segmentatie van het beeld bestaande uit rechthoekige gebieden. Elk gebied heeft een schatting voor de radiantie binnen dat gebied



Figuur 3.4: Deelbemonsterende piramide in vier niveaus bij rechthoekige segmentatie. Elke rechthoek berekent de gemiddelde lichtpadbijdrages over zijn pixels. Elke rechthoek wordt steeds in vier gelijke delen onderverdeeld. In dit voorbeeld zijn de rechthoeken die effectief gebruikt worden grijs gekleurd. Voor sommige pixels is het gebied klein, voor andere pixels is het groot.

dat betrouwbaar genoeg is. Een resultaat bekomen met de implementatie van de paper vindt men op figuur 3.5.

3.6.2 Prioriteit van een component

Eenmaal er over het hele beeld een betrouwbare schatting van de lichtpadbijdrages is bekomen, kan deze gebruikt worden om de prioriteit van de componenten te berekenen.

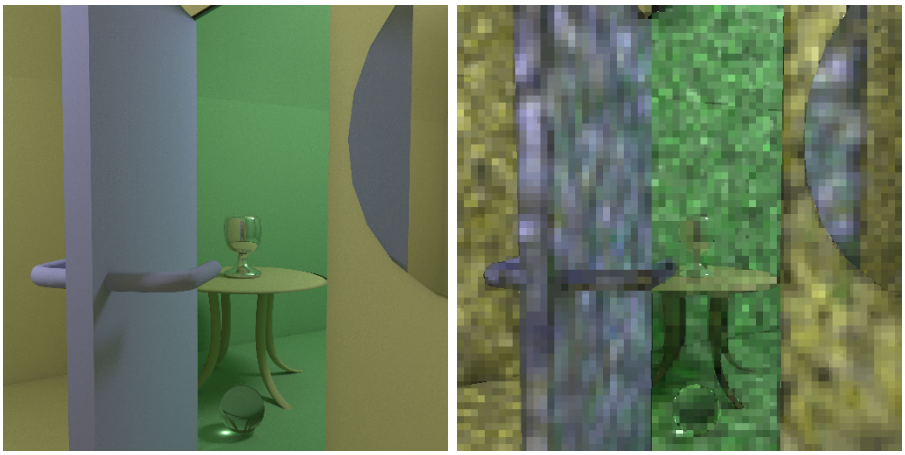
Men gaat uit dat de benadering van de componenten correct is, i.e. de uniforme radiantie van de benadering is de gemiddelde radiantie over de component. Dit wil zeggen; indien een component werkelijk een uniforme radiantie heeft, is deze benadering gelijk aan de gerenderde component. Deze mag dus in het renderproces uitgesteld worden tot in de laatste stap.

Anderzijds, neemt men een component met veel variatie, zal het verschil tussen de benadering en gerenderde component hoog zijn, zodat de prioriteit voor deze hoog moet zijn. Dit wordt geïllustreerd in figuur 3.6.

Met behulp van de rechthoekige segmentatie kan deze variantie berekend worden. Als maat hiervoor neemt men het gewogen gemiddelde van het kwadratisch verschil in luminantie tussen de uniforme radiantie en de radiantie van de rechthoeken.

De bekomen schatting uit de photon map bevat lichtpadbijdrages. Aangezien componenten slechts één lichtpadtype hebben, komt dat neer op een radiantiewaarde per pixel voor elke component.

Voor component C met gebied G_c en luminantie L_c, en S_c is de verzameling van alle rechthoekige gebieden I die overlappen met G_c.

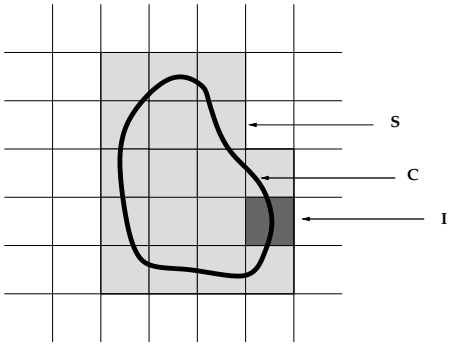


(a) Volledig beeld (b) Rechthoekige benadering

Figuur 3.5: Rechthoekige segmentatie

(a) Gerenderde component (b) Benaderde component (c) Gerenderde component (d) Benaderde component

Figuur 3.6: Prioriteit van een component. (b) is een goede benadering voor (a) terwijl (d) een slechte benadering is voor (d)



Figuur 3.7: Berekening van prioriteiten van componenten

$$\text{Prioriteit}(C) = \frac{\sum_{i \in S_c} \text{Opp}(G_i) \cdot (L_c - L_i)^2}{\sum_{i \in S_c} \text{Opp}(G_i)}$$

Figuur 3.7 illustreert de verschillende termen in de formule.

3.7 Renderstrategie

Er worden twee strategieën besproken. Enerzijds incrementeel met nadruk op bruikbare tussenresultaten, anderzijds tijdsgelimiteerd met nadruk op een zo volledig mogelijk beeld in een vooraf opgegeven tijd.

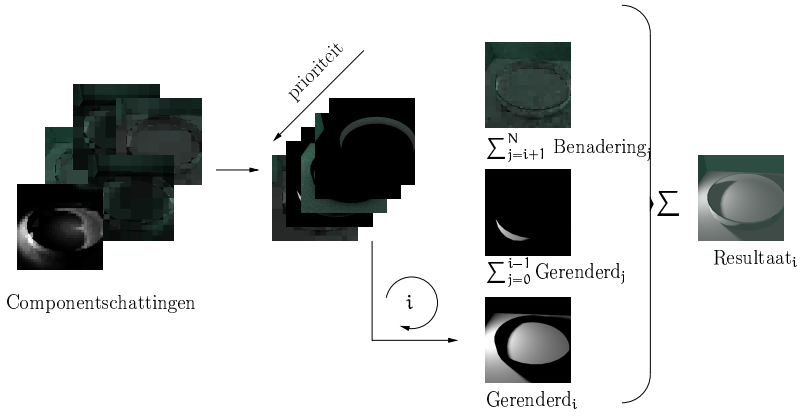
Aangezien de implementaties van de strategieën uiteindelijk zeer gelijkend zijn, wordt vooral aandacht besteed aan de incrementele aanpak.

3.7.1 Incrementeel renderen

Met deze strategie wil men een rendering verkrijgen dat geen ruis bevat en onderbreekbaar is. Een schets van het principe is te vinden op figuur 3.8.

Er wordt vertrokken van een volledig benaderd beeld. De benadering is een uniforme invulling van de component. De componenten worden geordend zoals besproken in de deel 3.6. Daarop begint men te itereren over de componenten. Behandelde componenten worden berekend. Hun benadering wordt vervangen door hun gerenderde versie. Het bekomen beeld wordt bewaard. Hiermee bereikt men de vooropgestelde onderbreekbaarheid.

Het andere doel is dat alle beelden die men bekomt (en dus ook tussenbeelden) helemaal geen ruis bevatten. Er wordt gebruik gemaakt van de techniek besproken in [Dal93] en



Figuur 3.8: Incrementeel renderen: Componentschattingen worden berekend en geordend volgens prioriteit. Vervolgens elke component wordt berekend zodat telkens een tussenresultaat wordt gegenereerd door de benadering en de voorheen berekende componenten te combineren. (uit [DD05])

[BM98]. Het beschrijft een techniek om een beeld te renderen tot het geconvergeerd is voor het menselijk oog. Op basis van het reeds bekomen beeld (benadering + gerenderd) wordt een *thresholdmap* opgebouwd. Een thresholdmap is een tweedimensionale structuur die per pixel een drempel berekent waaronder het menselijk oog geen verschil meer kan merken. Er wordt eveneens per pixel berekend wat de variantie is van de schatting. Op die manier kan men renderen tot er genoeg pixels zijn geconvergeerd.

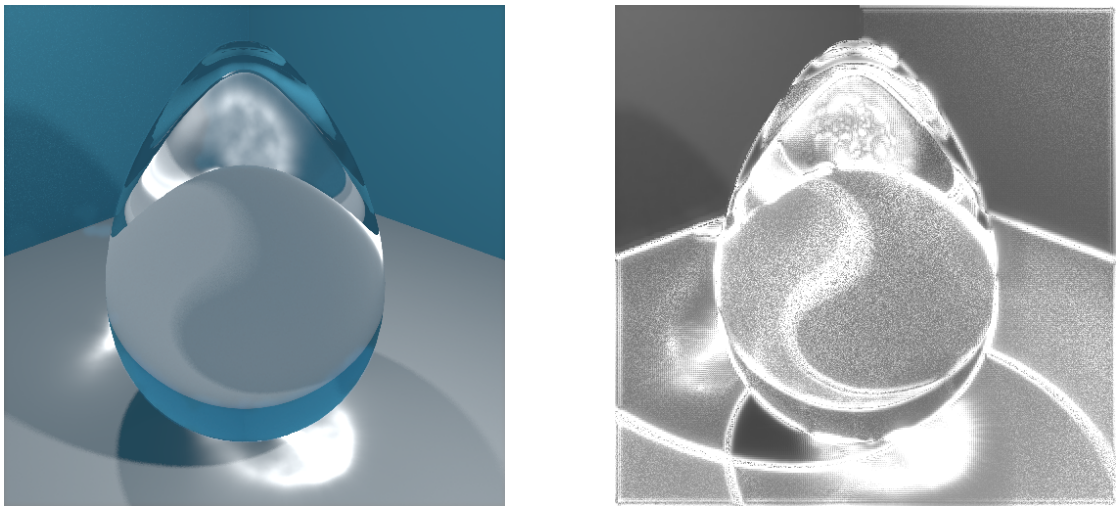
De thresholdmap wordt geïllustreerd op figuur 3.9. Hoe witter een pixel, hoe hoger de toegelaten fout op de waarde in die pixel. Zo hebben effen vlakken een lage tolerantie, terwijl bij een overgang tussen schaduw en direct licht een grote fout wordt toegelaten. De menselijke perceptie is daar minder gevoelig omdat de luminantie sterk varieert.

Toepassingen van deze strategie zijn te vinden in figuren B.1, B.6, B.9 en B.12 in bijlage B.

3.7.2 Tijdsgebonden renderen

Met deze strategie wilt men een beeld zonder bekomen in een voorop vastgelegde tijd. Het principe wordt visueel voorgesteld op figuur 3.10

De beschikbare tijd wordt verdeeld over de verschillende componenten. Er wordt net zoals bij de incrementele strategie component per component gerenderd. Wanneer een bepaalde component zijn beschikbare tijd overschrijdt, moet de ruis verwijderd worden. Dat wordt gedaan door een combinatie te nemen tussen gerenderde en benaderde component. Hoe meer variantie in het gerenderde, hoe belangrijker de bijdrage van de initiële benadering.



(a) Het basisbeeld

(b) De thresholdmap

Figuur 3.9: Voorbeeld van een thresholdmap

3.8 Mogelijke verbeteringen

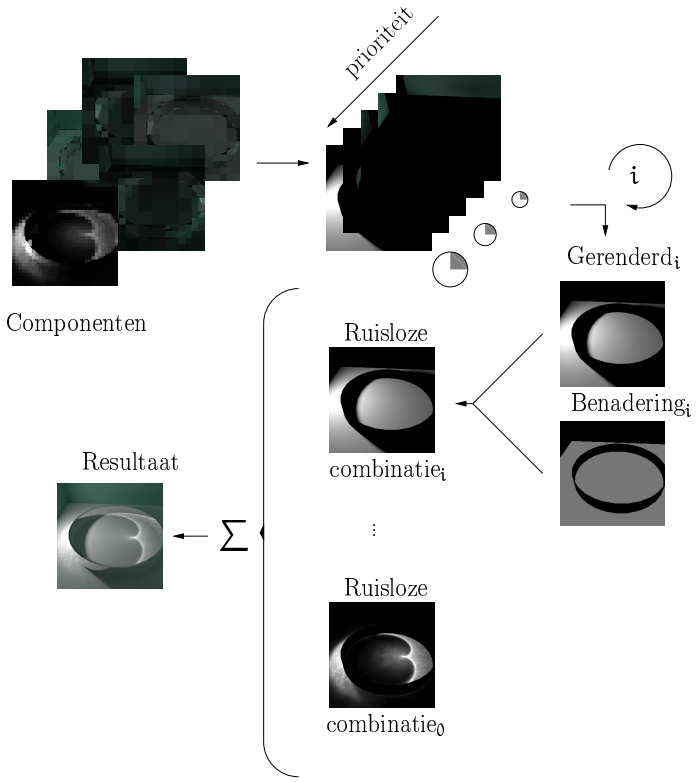
De besproken methode biedt nog mogelijkheid tot verbetering.

Perceptuele prioriteit Men zou kunnen gebruik maken van psycho-perceptuele maten om de prioriteit van elke component te bepalen. De partitie van de lichtpadruimte zou geoptimaliseerd worden gebruik makend van resultaten zoals [SFWG04].

Animaties Het algoritme zou uitgebreid kunnen worden naar animaties. De volgorde van de componenten moet dan wel stabiel blijven om geen temporele effecten te verkrijgen.

Meer geadapteerde componenten De segmentatie in componenten wordt in de beeldruimte bepaald volgens geometrie. Deze componenten zouden vaak best opgesplitst worden omdat ze intern teveel variatie vertonen. Verder zouden veel kleine componenten die nogal elkaar lijken mogen gegroepeerd worden.

Gebruikersgestuurd De gebruiker zou kunnen aanduiden welke delen van het beeld meer aandacht zou moeten krijgen.



Figuur 3.10: Tijdsgebonden renderen: componentschattingen worden berekend en geordend. De totaal toegelaten rendertijd wordt verdeeld over alle componenten in functie van de grootte en prioriteit. Niet-geconvergeerde componenten worden gecombineerd met benaderingen om perceptueel ruisloze resultaten te verkrijgen. (uit [DD05])

3.9 Bleed

Deze methode werd geïmplementeerd in de nieuwe ray tracer genaamd BLEED. Het werd en wordt ontwikkeld door Olivier Dumont.

Tijdens het eerste deel van dit eindwerk werd er in parallel met hem gewerkt. Bijgevolg is er een kruisbestuiving van code. Bijvoorbeeld werd de geometrische segmentatie, wat hier uitgebreid gebruikt wordt, door ons uitgewerkt. Onze invloed bevat ook wat structurele elementen nodig om het raamwerk uit te bouwen. De implementatie ervan was eerder voor de hand liggend en wordt dus niet besproken.

BLEED heeft een paar belangrijke eigenschappen die het uitermate geschikt voor verdere verfijning van de methode.

3.9.1 Visualisatie per gebied

Een eerst belangrijk punt van BLEED is dat het sterke ondersteuning biedt voor het visualiseren per gebied. Er wordt een concept gebied ingevoerd dat pixels bijhoudt die ertoe behoren en eveneens de lichtinformatie kan accumuleren. Zo kan men tegelijk dat concept gebruiken voor de echte visualisatie als voor de benadering.

3.9.2 Visualisatie per lichtpadtype

Daarnaast is er de nog belangrijkere en minder voor de hand liggende ondersteuning voor visualisatie per lichtpadtype.

Een klassieke aanpak met path tracing is niet geschikt. Men laat een straal geschoten uit de sensor botsen en men accumuleert de bijdragen van de botsingen. Het resultaat is radiantie. Informatie over het lichtpad is verloren gegaan.

Er is een gemodificeerd algoritme nodig. Bij elke botsing voegt met het type interactie (zie figuur 3.1) toe in lichtpad dat men bijhoudt. Indien dat lichtpad niet meer past in het lichtpadtype dat men zoekt, moet men de huidige bemonstering annuleren. Het principe wordt uitgebreid met een photon map, waarbij enkel geschikte fotonen worden gebruikt bij de radiantieschatting.

Hoofdstuk 4

Eigen aanpak

Dit hoofdstuk bespreekt onze eigen bijdragen. De paper [DD05] bevatte reeds eigen bijdragen, maar deze limiteerden zich tot meer eenvoudige problemen en medewerking in de opbouw van de structuur in BLEED.

We willen de methode die voorgesteld wordt in deze paper verbeteren. De basis hiervoor is een nieuwe segmentatietechniek.

In [DD05] werd een rechthoekige segmentatie gebruikt om een initiële schatting te verkrijgen van de radiantie over het beeld. Gezien de vorm (rechthoeken) en de opbouw (piramide) is er weinig vrijheid in de resulterende segmentatie. De methode heeft de neiging om teveel te splitsen door de te strakke beperkingen.

In hoofdstuk 5 ontwikkelen we daarom een nieuwe segmentatie die zich baseert op initiële lichtinformatie. Hierbij is de vorm van de nieuwe gebieden volledig vrij. De segmentatie zal zoeken naar gebieden met uniforme radiantie en deze afsplitsen. De onderliggende techniek is mean shift [CM02].

We willen sneller meer volledige beelden kunnen renderen. Na de eerste iteraties is het beeld vaak nog zeer onvolledig. Het is mogelijk dat grote schaduwen of caustics dan nog altijd niet zichtbaar zijn. Dit vermindert de aannemelijkheid van het bekomen beeld. Dat komt door het feit dat de voorlopige benadering uit een uniforme kleur over het hele oppervlak van het voorwerp bestaat.

Indien de initiële benadering beter is kan men vlugger beslissen de iteratie te stoppen en een tussenresultaat gebruiken. In hoofdstuk 6 wordt aan de hand van de vooropgestelde fotonsegmentatie een betere benadering bekomen.

Een laatste punt waar verbetering mogelijk is, is een meer gerichte opdeling in rendercomponenten. Een opdeling volgens geometrie is een zeer ruwe opdeling. Lichteffecten zoals caustics en schaduwen zijn veelal enkel te zien op een deel van het oppervlak van

een object. Het zou daarom nuttig zijn om enkel het relevante deel van het voorwerp te renderen en de rest ervan te benaderen. Dit wordt eveneens in hoofdstuk 6 uitgewerkt.

Hoofdstuk 5

Fotonsegmentatie

In dit hoofdstuk stellen we een nieuwe segmentatiewijze voor die gebaseerd is op lichtinformatie.

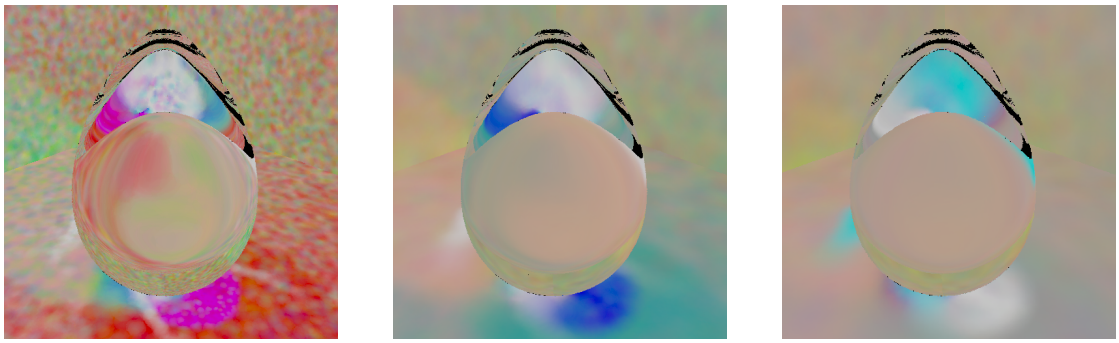
Net zoals in [DD05] halen we een schatting van de lichtinformatie uit de photon map. Onze segmentatie deelt het beeld niet op rechthoekige deelgebieden, maar zorgt ervoor dat de vorm van segmentatie beter aansluit bij bijdragen van de verschillende lichtpadtypes.

De fotonsegmentatie bestaat uit drie stappen. Eerst wordt er voor elke pixel aan de hand van de photon map een schatting van de bijdragen van de verschillende lichtpadentypes in die pixel berekend. Het basisalgoritme wordt besproken in deel 5.1. Een methode om een versnelde schatting te krijgen beschrijven we in deel 5.2.

Vervolgens wordt er gezocht naar gebieden waar de bijdragen uniform zijn in deel 5.3. We hebben hiervoor twee werkwijzen gevolgd. De eerste manier, waarvan een bespreking te vinden is in deel 5.4, maakt gebruikt van randdetectie om randen van belangrijke gebieden te vinden. De tweede manier is een toepassing van het *mean shift algoritme*. In deel 5.5 is een beknopte bespreking van dit algoritme en het gebruik ervan te vinden. Deze methode wordt verder verfijnd in in deel 5.6.

De bekomen opsplitsing bevat soms nog te scherpe randen. Daarom stellen we in deel 5.7 een methode om de segmentatie te verfijnen. Tenslotte is er in deel 5.8 een evaluatie van de fotonsegmentatie te vinden.

Aan de hand van een geometrische segmentatie worden de gevonden gebieden verder opgesplitst. Dit zal de kwaliteit van de segmentatie verder verbeteren. Op grenzen van voorwerpen zijn er meestal discontinuïteiten. Aangezien de geometrische segmentatie zeer eenvoudig uit te voeren is, is dit een snelle manier om de onze segmentatie te verbeteren.



(a) k = 20 (b) k = 160 (c) k = 240

Figuur 5.1: Voorbeeld van visualisatie van schatting lichtpadtypes. De beelden zijn multidimensionale beelden getekend met valse kleuren. Verschillende beelden gebruiken verschillende kleuren voor hetzelfde lichtpadtype. Er werd telkens gezocht naar de k dichtste fotonen.

5.1 Schatten bijdragen lichtpadtypes

Om de bijdrage te kennen van elk lichtpadtype in elke pixel sturen we een straal vanuit de camera. Aangezien een schatting wordt berekend, wordt er abstractie genomen van het feit dat er door dezelfde pixel verschillende objecten kunnen gezien worden. Indien deze straal een speculaire oppervlak raakt, wordt de straal verder gepropageerd totdat deze botst op een oppervlak dat niet speculair is. Er wordt steeds bijgehouden welke weg de straal heeft afgelegd.

Vervolgens wordt er gezocht naar de k dichtste fotonen in een bolvormige omgeving met vaste straal. Elk gevonden foton draagt bij tot één type lichtpad. Dit lichtpadtype wordt gevonden uit het pad dat straal uit de lichtsensor heeft afgelegd en het pad dat de foton heeft afgelegd.

Een bespreking van de photon map is te vinden in deel 2.5.3. Methodes die de kwaliteit van de photon map verbeteren zullen ook de schatting van de bijdragen van de lichtpadtypes verbeteren op voorwaarde dat deze methodes rekening houden met de verschillende lichtpadtypes.

We hebben dit algoritme geïmplementeerd in Bleed. De methode wordt toegepast op alle pixels en de bekomen schattingen zijn bewaard in een beeld. In figuur 5.1 worden beelden getoond die gemaakt zijn door middel van het zoeken naar respectievelijk de 20, 160 en 240 dichtste fotonen.

Zoeken naar meer fotonen zorgt voor minder vlekken in het beeld, wat de segmentatie vergemakkelijkt. Voor alle types lichtpaden in een bepaald punt, moeten er fotonen wor-

den gevonden. Het aantal fotonen moet dus voldoende hoog zijn. Een te grote verhoging van het aantal fotonen heeft echter geen merkbaar effect meer. Dit is te zien in figuur 5.1. Het verschil tussen zoeken naar de 160 of 240 dichtste fotonen is verwaarloosbaar. Een verhoging van het totaal aantal fotonen is dan nodig om een verbetering te verkrijgen. Dit heeft echter als nadeel dat de rekentijd en het geheugengebruik stijgt.

Om te zorgen dat de kwaliteit van de schatting voldoende hoog is, hebben we in onze testen telkens gezocht naar de 160 dichtste fotonen in een photon map bestaande uit ongeveer 160.000 fotonen.

5.2 Versneld schatten bijdragen lichtpadtypes

5.2.1 Motivatie

De bedoeling van de fotonsegmentatie is de methode beschreven in [DD05] te verbeteren. Belangrijk bij het genereren van beelden is dat het snel gebeurt. Door veel fotonen te zoeken per pixel wordt de voorbereidende fase (oftewel preprocessing) te traag om goede resultaten te bekomen. Daarom hebben we onderzocht hoe het schatten van de lichtpadbijdragen over het beeld kan versneld worden.

5.2.2 Methode

Het geleverde werk voor het aantal gezochte fotonen f en het aantal pixels p is proportioneel met $f \cdot p$. Over het hele beeld is uiteraard $p = w \cdot h$. Dus geldt

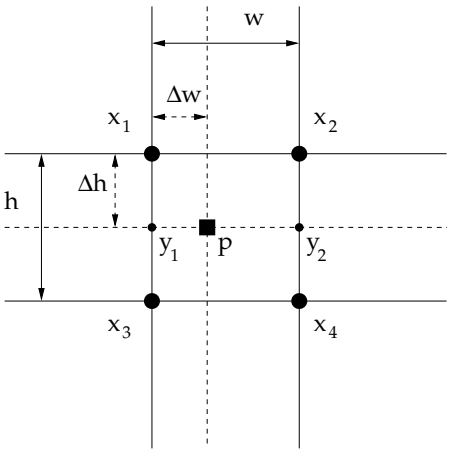
$$\text{werk} = K \cdot f \cdot w \cdot h$$

Het aantal gezocht fotonen per pixel f moet redelijk hoog gehouden worden, anders wordt er teveel ruis geïntroduceerd. De andere manier is het aantal pixels verlagen. De breedte w of hoogte h aanpassen is echter niet mogelijk. Het aantal pixels moet dus losgekoppeld worden van het totaal aantal pixels aanwezig in het beeld. Zo kunnen we een aantal pixels schatten en daartussen interpoleren.

Vast rooster

De eenvoudigste werkwijze is een vast rooster van punten te bemonsteren over het beeld en daartussen tweemaal lineair te interpoleren. Gegeven een rooster R willen we de lichtpadbijdragen in pixel p berekenen, zie figuur 5.2. Daarvoor interpoleren we tussen de punten x_1, x_2, x_3 en x_4. In een eerste stap worden punten y_1 en y_2 berekend.

$$l_{y_1} = \left(1 - \frac{\Delta h}{h}\right) \cdot l_{x_1} + \frac{\Delta h}{h} \cdot l_{x_3}$$



Figuur 5.2: Rechthoekige interpolatie

$$l_{y_2} = \left(1 - \frac{\Delta h}{h}\right) \cdot l_{x_2} + \frac{\Delta h}{h} \cdot l_{x_4}$$

Door deze nieuwe punten te interpoleren bekomt men dan p.

$$l_p = \left(1 - \frac{\Delta w}{w}\right) \cdot l_{y_1} + \frac{\Delta w}{w} \cdot l_{y_2}$$

Eliminatie van l_{y_1} en l_{y_2} levert op:

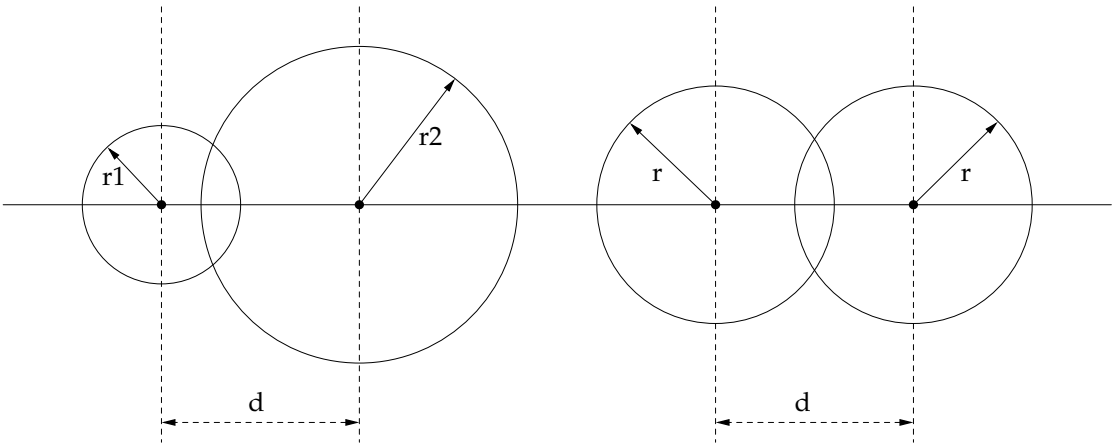
$$l_p = \left(1 - \frac{\Delta w}{w}\right) \cdot \left(1 - \frac{\Delta h}{h}\right) \cdot l_{x_1} + \frac{\Delta w}{w} \cdot \left(1 - \frac{\Delta h}{h}\right) \cdot l_{x_2} + \left(1 - \frac{\Delta w}{w}\right) \cdot \frac{\Delta h}{h} \cdot l_{x_3} + \frac{\Delta w}{w} \cdot \frac{\Delta h}{h} \cdot l_{x_4}$$

De parameters w en h moeten nog bepaald worden. Voor sommige delen van het beeld mogen deze zeer groot zijn, voor andere delen zijn deze best slechts één pixel groot. We kunnen dus moeilijk een goede globale waarde voor w en h vinden.

Adaptief rooster

Uit het vorige blijkt dat een adaptief rooster geschikter zou zijn. Om de lokale w en h te bepalen is een heuristiek nodig. Een paar factoren zijn belangrijk bij het beslissen of w en h geschikt zijn.

- Verschil in radiantie: indien pixels sterk in waarde verschillen, is het waarschijnlijk beter om niet te interpoleren.
- Afstand in pixels: pixels die dichter bij elkaar staan mogen eerder geïnterpoleerd worden.



Figuur 5.3: Intersectie van foton-bollen

- Gemeenschappelijke fotonen: in elke pixel wordt een aantal fotonen gezocht voor de densiteitsschatting. Men kan nagaan of de verschillende pixels veel fotonen gemeenschappelijk hebben. Indien dat er veel zijn, is het zeer waarschijnlijk dat de lichtpadbijdragen quasi lineair zullen variëren tussen de pixels.

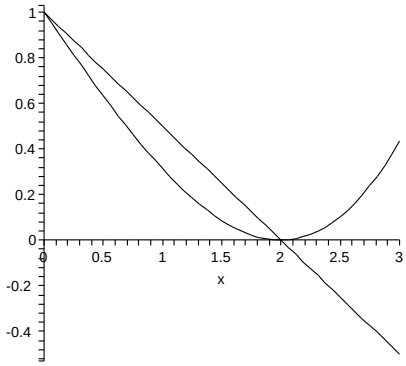
Een combinatie van deze factoren nemen is nogal complex. Daarom hebben we slechts één element gekozen. We overlopen de mogelijkheden. Met enkel verschil in radiantie is de methode verkeerd. Over grote oppervlakken kunnen sommige pixels toevallig gelijkaardig zijn. Afstand in pixels op zich zou een regelmatig rooster opleveren, wat we proberen te vermijden. De beste oplossing lijkt dus nagaan hoeveel fotonen er gemeenschappelijk zijn.

Beschouw twee pixels met respectievelijk r_1 en r_2 als afstand van het verst gevonden foton in de pixel. Deze vormen twee "foton-bollen". De afstand tussen de twee intersecties in de pixels wordt voorgesteld als d, zie figuur 5.3 links.

We veronderstellen dat de fotonen uniform verspreid zijn over het bolvolume. De fractie gemeenschappelijke fotonen herleidt zich dan tot de berekening van het gemeenschappelijk volume. Eveneens veronderstellen we dat we voor de eenvoud mogen kiezen de twee bollen dezelfde straal toe te kennen, met name $r = \frac{r_1+r_2}{2}$, zoals in figuur 5.3 rechts. Gegeven deze veronderstellingen is de fractie gemeenschappelijk volume of *overlappingsgraad*

$$\text{overlappingsgraad} = 1 - \frac{3}{4}\left(\frac{d}{r}\right) + \frac{1}{16}\left(\frac{d}{r}\right)^3$$

Eens men een drempelwaarde voor de overlappingsgraad bepaalt heeft, bekomt men een heuristiek voor de grootte van de rechthoeken waarbinnen er geïnterpoleerd mag worden.



Figuur 5.4: Gemeenschappelijk volume van bollen: vergelijking tussen de correcte formule $1 - \frac{3}{4}x + \frac{1}{16}x^3$ en de lineaire benadering $1 - \frac{1}{2}x$

Het is ook handig een heuristiek te hebben die ook werkt voor niet-overlapppende bollen om zeer ruwe interpolatie te bekomen. De formule voor de overlappingsgraad is echter enkel geldig binnen het interval $[0, 2]$. Na 2 stijgt de functie weer, wat voor deze uitbreiding niet bruikbaar is. Een eenvoudige lineaire benadering lost dit op:

$$\text{overlappingsgraad} = 1 - \frac{1}{2}\left(\frac{d}{r}\right)$$

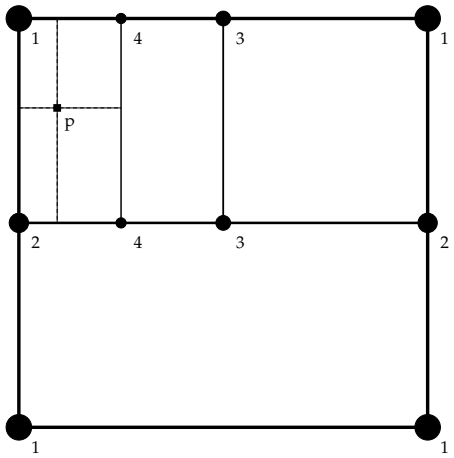
Figuur 5.4 vergelijkt de twee functies.

Met de nieuwe heuristiek kunnen we een adaptief rooster bouwen. We beginnen met de vier hoekpunten van het beeld. Indien $1 - \left(\frac{d}{r}\right) < D$ voor de diagonale hoekpunten, waarbij de D de gekozen drempelwaarde is, overlappen de fotonbollen elkaar niet genoeg en wordt de rechthoek opgesplitst. Er ontstaan nieuwe hoekpunten. Hun waarde wordt geschat uit de photon map. De nieuwe rechthoeken worden op dezelfde manier behandeld. Indien $1 - \left(\frac{d}{r}\right) \geq D$, dan is er genoeg overlapping en wordt die rechthoek geïnterpoleerd zoals eerder beschreven.

5.2.3 Toepassing

Een voorbeeld van deze recursieve opsplitsing staat in figuur 5.5. Elke bol stelt een effectieve schatting uit de photon map voor. Het getal erbij toont in welke volgorde ze bemonsterd zijn. In de linkerbovenrechthoek ziet men de uiteindelijke interpolatie in een rechthoek die voldoet aan de voorwaarden.

Het algoritme wordt gedemonstreerd voor $D = 0.85$, wat betekent dat er enkel wordt geïnterpoleerd tussen pixels die minstens $\pm 85\%$ fotonen gemeenschappelijk hebben, zie



Figuur 5.5: Adaptieve onderverdeling voor snelle lichtschatting

figuur 5.6. Het volledige beeld wordt bekomen in 22 minuten, terwijl dat slechts 9 minuten[1] duurde bij interpolatie. Grotere versnelling kan bekomen worden door D lager te kiezen. Een te lage D zal echter een hoekige segmentatie opleveren.

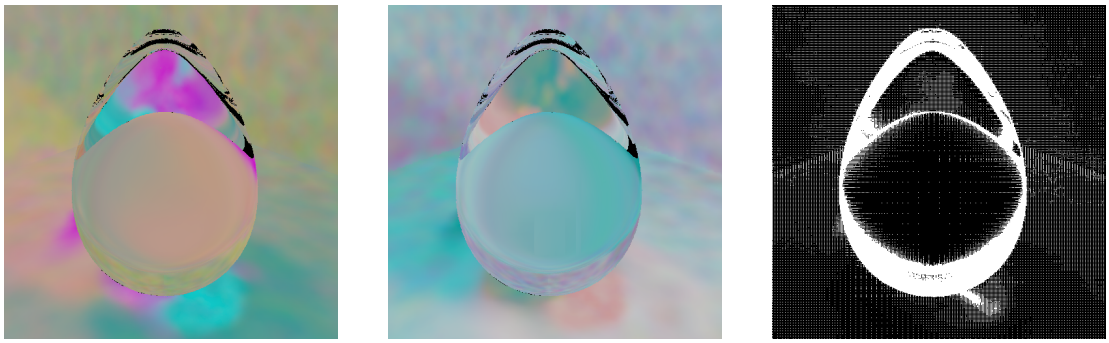
Een voorbeeld is te zien in figuur 5.6. Behalve het gebruik van andere valse kleuren is er amper verschil te merken tussen de eerste twee beelden. Op 5.6(c) is te zien hoe de pixels zijn gekozen. Het midden van het ei werkt als een vergrootglas zodat nabije pixels daar veel gemeenschappelijke fotonen hebben. Dat verklaart waarom er daar zo weinig bemonsteringen worden genomen. Anderzijds zien we dat rechtsonder het ei er veel bemonsterd wordt. Daar is namelijk een caustic aanwezig en bijgevolg veel fotonen. Veel fotonen betekent een kleine r zodat de drempel D moeilijk overschreden wordt.

5.3 Aanpak segmentatie bijdragen lichtpadtypes

Nadat de schattingen zijn berekend, voeren we een algoritme uit dat beelden segmenteert. Dit kan op verschillende manieren gebeuren. Er kan gewerkt worden met alle lichtpadtypes tegelijk of de lichtpadtypes kunnen afzonderlijk gesegmenteerd worden. Dit laatste heeft als voordeel dat er minder geheugen nodig is en verschillende segmentatieparameters of segmentatie-algortitmes gebruikt kunnen worden per lichtpadtype. De segmentatiestap zal echter wel iets rekenintensiever zijn.

De bijdragen voor elk lichtpadtype bestaan uit drie componenten aangezien er gewerkt wordt met het *Luv* kleurenmodel. Voor de segmentatie gebruiken we echter maar één component, de luminantie. Deze duidt de intensiteit aan van de kleur en wordt gehaald uit de *Luv*-waarden. Er wordt immers gezocht naar gebieden waar de intensiteit van de

[1] Resolutie 512x512 op een Pentium IV 2GHz



(a) Schatting per pixel

(b) Schatting met interpolatie

(c) Bemonsterde pixels in de interpolatiemethode

Figuur 5.6: Voorbeeld van versnelde lichtschatting. De beelden zijn multidimensionale beelden getekend met valse kleuren. De verschillende beelden gebruiken verschillende kleuren voor hetzelfde lichtpad.

bijdragen uniform is. Een bijkomend voordeel is dat de segmentatie minder geheugen nodig heeft.

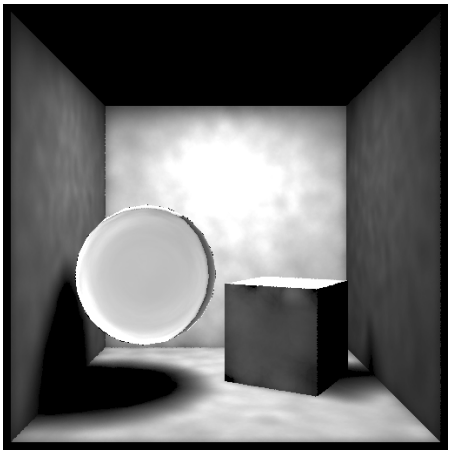
In plaats van te werken met de absolute bijdragen werken we met relatieve bijdragen. Hiervoor wordt iedere bijdrage in elke pixel gedeeld door de som van de afzonderlijke bijdragen in deze pixel. Dit verhoogt de kwaliteit van de segmentatie:

- Het effect van een foute schatting wordt kleiner door de relatieve bijdrage te nemen. Kleine schommelingen zullen hierdoor minder effect hebben.
- Enkel wijzigingen van de verhoudingen tussen bijdragen kunnen gedetecteerd worden. Segmentatie van de absolute bijdragen gaat ook wijzingen aan de som van de bijdragen detecteren. Een gebied met een gradiënt zal hierdoor moeilijk gedetecteerd worden en de kans bestaat dat er oversegmentatie optreedt (trap-effect). De benadering hiervan zorgt voor artefacten in het beeld. Artefacten zijn storende elementen in een beeld die onstaan door het verschil tussen de benaderingen onderling of door het verschil tussen de benaderde en de reeds gerenderde gebieden.
- Indien een lichtpadtype in een bepaald gebied belangrijk is, is het voordelig om dit gebied afzonderlijk te renderen.

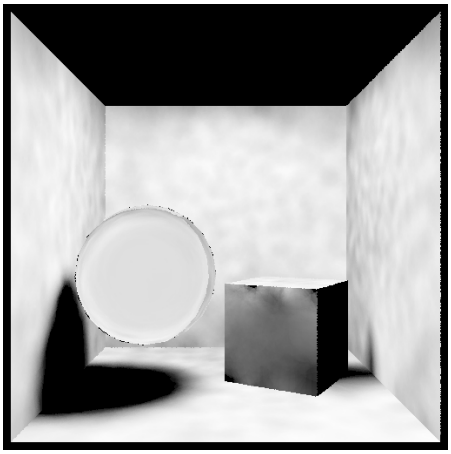
In figuur 5.7 zijn de relatieve en absolute bijdragen van twee lichtpadtypes te zien.

We hebben 2 segmentatie-algoritmes geïmplementeerd in BLEED:

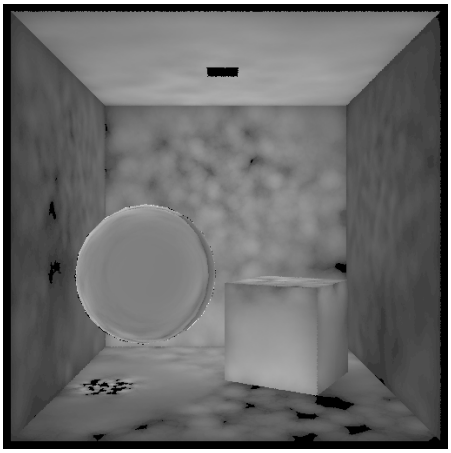
- Segmentatie via randdetectie



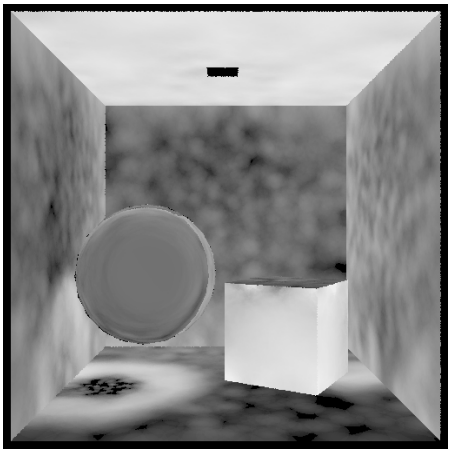
(a) Absolute bijdragen directe belichting



(b) Relatieve bijdragen directe belichting



(c) Absolute bijdragen van een indirect lichtpadtype



(d) Relatieve bijdragen van een indirect lichtpadtype

Figuur 5.7: Relatieve en absolute bijdragen van twee lichtpadtypes

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \qquad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad \begin{bmatrix} -1 & 0 & -1 \\ -2 & 0 & -2 \\ -1 & 0 & -1 \end{bmatrix}$$

(a) Laplace filter (b) Verticale Sobel filter (c) Horizontale Sobel filter

Figuur 5.8: Enkele randdetectiefilters.

- Mean shift segmentatie

5.4 Segmentatie via randdetectie

Indien randen van interessante gebieden kunnen gedetecteerd worden, kan er een vulalgoritme gebruikt worden om gebieden te vinden. Het vulalgoritme vult alle pixels op van een gebied, rekening houdend met de gevonden randen. Er wordt telkens een pixel gekozen die nog niet tot een gebied behoort. Vervolgens wordt dit gebied opgevuld. Dit wordt herhaald totdat alle pixels tot één gebied behoren.

Randen in een beeld kunnen gevonden worden door het beeld te filteren. Dit gebeurt door middel van convolutie met een filtermatrix. Meer informatie over beeldfilters is te vinden in [Pra91].

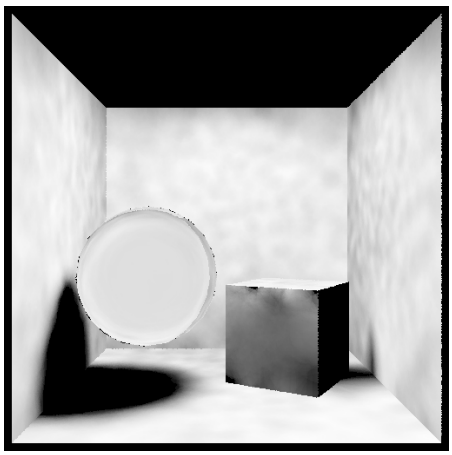
Er bestaan verschillende soorten randdetectiefilters zoals de Laplacefilters en Sobelfilters. Deze filters en bijhorende gefilterde beelden zijn te zien in de figuren figuur 5.8 en 5.9.

Deze filters zijn zeer ruisgevoelig en de gefilterde beelden zijn niet onmiddelijk bruikbaar. Een betere manier bij beelden met ruis is de volgende. Randen kunnen gevonden worden uit de nuldoorgangen van de afgeleide van de gradiënt (de Laplaciaan) van het gefilterd beeld. Dit wordt geïllustreerd in figuur 5.10.

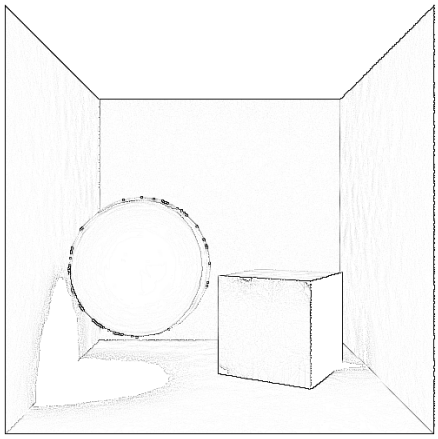
Deze methode wordt in het Engels ook *Laplace of Gaussians* genoemd en bestaat uit drie stappen:

Ruisverwijdering Deze afgeleide is zeer gevoelig voor ruis. Het is daarom noodzakelijk dat de ruis verwijderd wordt. Ruis in beeld is hoogfrequent en een laagdoorlaatfilter zorgt ervoor dat een deel van de ruis gefilterd wordt. We hebben de volgende filter gebruikt:

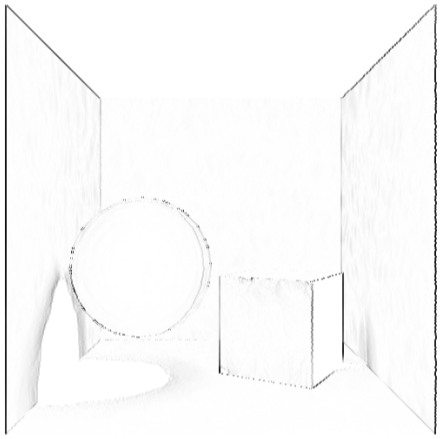
$$H_{\text{blur}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$



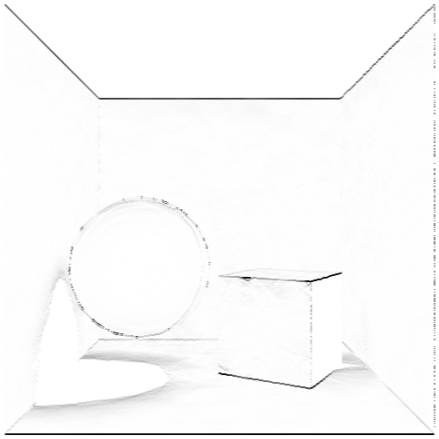
(a) Origineel beeld



(b) Laplacefilter

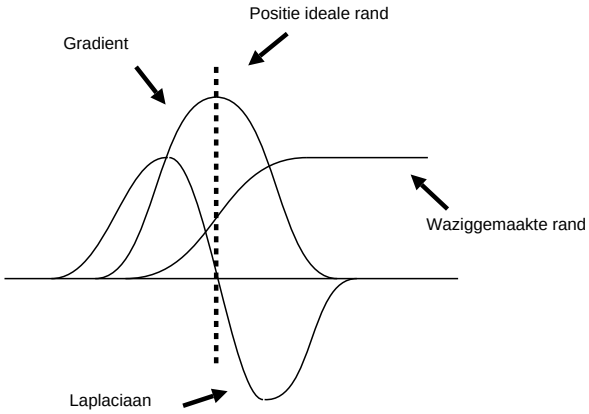


(c) Vertical Sobelfilter



(d) Horizontale Sobelfilter

Figuur 5.9: Randfilters uit figuur 5.8 toegepast op een beeld. De gefilterde beelden zijn geïnverteerd.



Figuur 5.10: Randdetectie

Dit is een 5×5 vervagingsfilter. Het gegeneerde beeld bevat echter weinig hoogfrequente ruis, maar toch zijn er vlekken aanwezig omwille van het schatten. Toepassing van deze vervagingsfilter zorgt ervoor dat de vlekken minder sterk aanwezig zijn.

Filtering Een benadering voor de afgeleide van de gradiënt is de Laplacefilter. Er bestaan verschillende Laplacefilters. In BLEED hebben we gebruik gemaakt van

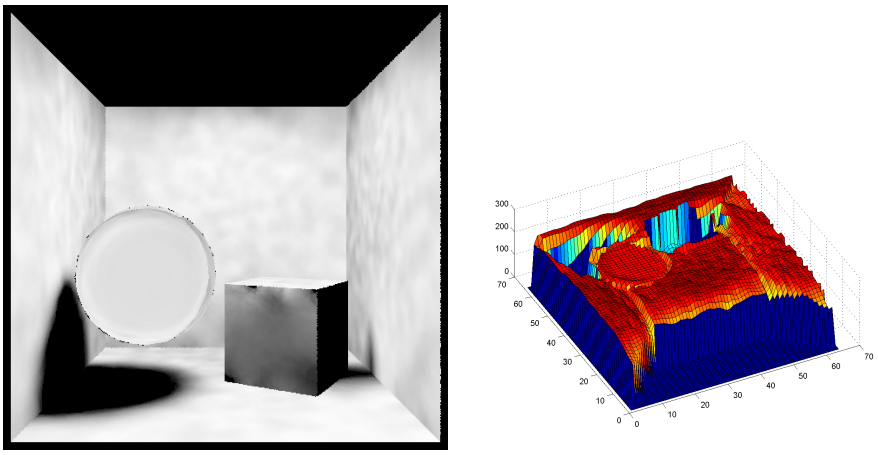
$$H_{Laplace} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Deze filter wordt toegepast op het beeld.

Detectie van randen Randen zijn te vinden in de nuldoorgangen van de afgeleide van de gradiënt. Indien in de omgeving van een gefilterde pixel zich een positieve en negatieve waarde bevindt, dan bevindt deze pixel zich op een rand. Dit wordt geïllustreerd in figuur 5.10. Door middel van een parameter kan de gevoeligheid van de randdetectie worden ingesteld.

Vervolgens wordt er een vulalgoritme uitgevoerd om gebieden te vinden.

In veel gevallen bevatten de randen gaten. Hierdoor stopt het vulalgoritme niet tijdig en zijn de gevonden gebieden te groot. De bekomen resultaten zijn bijgevolg niet bruikbaar. De benadering van de lichtpadtypes bevat te weinig scherpe randen om te kunnen leiden tot een goede segmentatie.



(a) Beeld met luminantiewaarden.

(b) Voorstelling van dit beeld met behulp van een puntenwolk.

Figuur 5.11: Voorstelling van een beeld met één kleurendimensie door middel van een puntenwolk.

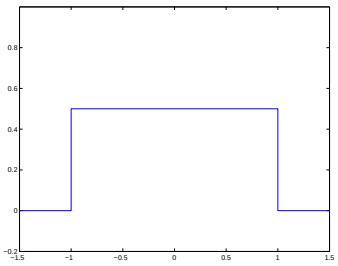
.

5.5 Mean shift segmentatie

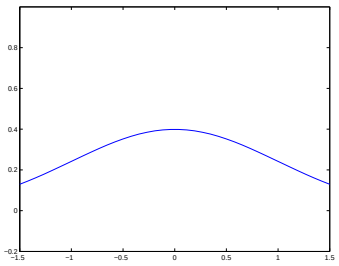
Het doel van het mean shift algoritme is om clusters (segmenten) te vinden in veeldimensionale puntenwolken. Ook een beeld kan worden beschreven door middel van een puntenwolk. Men heeft dan twee ruimtelijke dimensies (namelijk de positie van de pixel). De rest van de dimensies wordt gevormd door de kleurcomponenten van het beeld. Een beeld met Luv-kleuren bestaat dus vijf dimensies. Een voorbeeld hiervan met één kleurendimensie is te zien in figuur 5.11.

Het mean shift algoritme is een niet-parametrische clusteringsmethode. Dit wil zeggen dat het aantal en de vorm van de clusters niet bekend moeten zijn. Bij de segmentatie van lichtpadtypes is dit niet op voorhand bekend en bijgevolg zijn parametrische clusteringsmethodes niet bruikbaar.

Aangezien het algoritme clusters kan vinden in puntenwolken, kan het ook gebruikt worden om fotonen te clusteren. Fotonen beschrijven echter maar een gedeelte van het lichtpadtype en dus is de kostelijke stap van het zoeken per pixel naar fotonen ook hier noodzakelijk. Een volledig gezichtspunt-onafhankelijke segmentatie van lichtpadtypes is bijgevolg niet mogelijk. Segmentatie van de fotonen heeft voorts nog als nadeel dat de segmentatie langer zou duren aangezien de hoeveelheid fotonen meestal veel groter is dan het aantal pixels. De aanpak om fotonen te clusteren hebben we bijgevolg niet verder uitgewerkt.



(a) $K(x) = \frac{1}{2}\, I(|x| \leq 1)$ (Uniform)



(b) $K(x) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2}\, x^2)$ (Gaussiaans)

Figuur 5.12: Enkele veelgebruikte univariate kernelfuncties. De functie I is de indicatorfunctie, dit wil zeggen $I(A) = 1$ indien A waar is, anders 0.

5.5.1 Principe

De invoergegevens bestaan uit punten x_i met $i = 1, \ldots, n$ in een bepaalde d-dimensionale ruimte. Deze punten kunnen beschouwd worden als uitkomsten van een toevalsvariabele X en bijgevolg kan er een bijhorende kansdichtheidsfunctie $f(\bullet)$ gedefinieerd worden. Plaatsen waar de dichtheid van punten groot is zijn dus plaatsen waar de kansdichtheidsfunctie groot is en omgekeerd.

De meest gebruikte methode om deze functie te benaderen is gebruikt te maken van een kernelfunctie $K(\bullet)$. Deze methode wordt kerneldichtheidsschatting genoemd. De kernelfunctie duidt aan of een punt *dicht genoeg* bij het middelpunt van de kernelfunctie ligt. Meer informatie over kernelfuncties is te vinden in [HMSW04]. Enkele voorbeelden van kernelfuncties zijn te zien in figuur 5.12.

Het mean shift algoritme maakt gebruik van een radiaal symmetrische kernel $K(\bullet)$ waarvoor geldt dat

$$K(x) = c_{k,d} k(x^\top x)$$

De functie k wordt het profiel van de kernelfunctie genoemd en $c_{k,d}$ is een normalisatieconstante. De functie $g(x)$ wordt gedefineerd als $-k'(x)$ en de bijhorende kernelfunctie $G(\bullet)$ is gelijk aan

$$G(x) = c_{g,d} g(x^\top x)$$

De volgende schatter voor de kansdichtheid wordt bekomen:

$$\hat{f}_{h,K}(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

De bandbreedteparameter h zorgt voor de zachtheid van het verloop van de kansdichtheidsfunctie. Hoe groter deze is, des te zachter is het verloop van de benaderende kans-

dichtheidsfunctie. Eventueel kunnen voor verschillende dimensies verschillende bandbreedteparameters worden gebruikt.

Deze kansdichtheidsfunctie bevat lokale maxima. Deze worden de *modes* genoemd. Indien voor elk punt gevonden kan worden wat de dichtsbijzijnde mode is, wordt er een segmentatie van punten verkregen. Deze maxima zijn stationaire punten en dus is de gradiënt van de kansdichtheidsfunctie gelijk aan 0:

$$\hat{\nabla} f_{h,K} = 0$$

De mean shift vector is in een punt x is gelijk aan het verschil tussen het gewogen gemiddelde in dit punt en het punt zelf:

$$m_{h,G}(x) = \frac{\sum_{i=1}^{k} x_i g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)}{\sum_{i=1}^{k} g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)} - x$$

In [CM02] is er bewezen dat de mean shift vector evenredig is met de genormaliseerde gradiënt van de kansdichtheidsfunctie. Lokale maxima kunnen bijgevolg gevonden door de mean shift vector te berekenen en de kernel $G(x)$ te transleren in de richting van de mean shift vector. Als dit meerdere keren herhaald wordt zal de mean shift vector naar nul convergeren en zal er een stationair punt gevonden worden. Indien na een kleine perturbatie het convergentiepunt hetzelfde blijft, is een lokaal maximum gevonden. Voor verdere eigenschappen van de mean shift vector verwijzen we naar [CM02].

5.5.2 Toepassing op beelden

De verschillen in ruimtelijke dimensies en kleurcomponenten van een beeld zorgen er voor dat er twee verschillende bandbreedte parameters h_r en h_k gebruikt moeten worden. h_r bepaalt de resolutie op pixel-niveau en h_k bepaalt deze op gebied van de kleurcomponenten. De bijhorende kernelfunctie is gelijk aan

$$K_{h_r,h_k}(x) = \frac{C}{h_r^2 h_k^2} k\left(\left\|\frac{x_r}{h_r}\right\|\right) k\left(\left\|\frac{x_k}{h_k}\right\|\right)$$

met x_r de pixelcoördinaten van x en x_k de kleurcomponenten van x. C is een normalisatieconstante. In de praktijk heeft het type kernelfunctie een kleine invloed. Gaussiaanse of Epanechnikov kernelfuncties geven goede resultaten en dus enkel zullen de bandbreedtes moeten gekozen worden.

Indien dit algoritme wordt toegepast op beelden wordt een gefilterd beeld bekomen, waar de kleurcomponten van elke pixel vervangen zijn door de kleurcomponenten van de bijhorende mode. Er worden geen echte segmenten gevonden aangezien het aantal modes nog veel te groot is. Vervolgens worden pixels waarvan de modes dicht genoeg bij

elkaar liggen, samengevoegd. Zo ontstaat er een gesegmenteerd beeld. De bandbreedteparameters bepalen wat dicht genoeg is. Segmenten die te klein zijn kunnen eventueel samengevoegd worden met grotere gebieden.

Samengevat, bestaat het mean shift segmentatie-algoritme uit 3 stappen:

1. Elke pixel wordt gefilterd. Dit gebeurt aan de hand van de volgende stappen:
   (a) Bereken de mean shift vector in de omgeving van deze pixel.
   (b) Indien geen convergentie, transleer zoekomgeving in de richting van de mean shift vector en ga naar de vorige stap.
   (c) Indien de gevonden mode geen lokaal maximum is, ga terug naar de eerste stap.

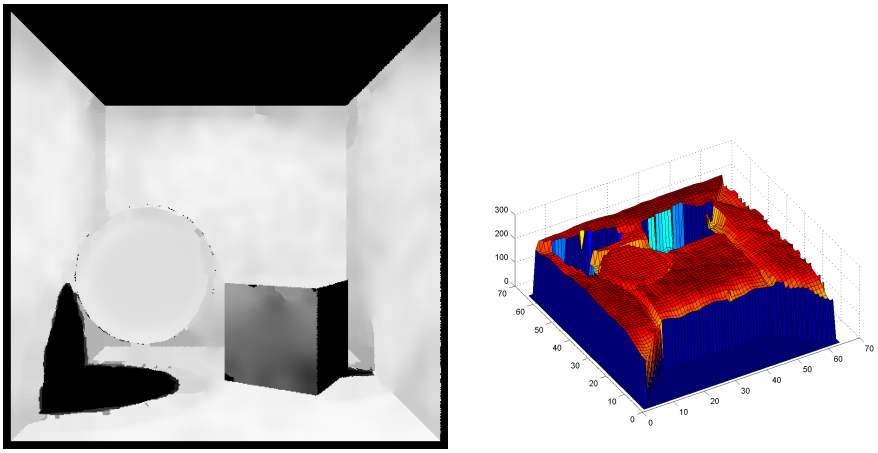
   Een voorbeeld van deze filterstap is te zien in figuur 5.13 waar figuur 5.11 gefilterd wordt. Merk op dat er geen echte segmentatie bekomen wordt.
2. Zoek naar modes die dicht genoeg bij elkaar liggen en voeg de gebieden die bij deze modes horen samen.
3. Eventueel kunnen gebieden die bestaat uit weinig pixels samengevoegd worden met grotere gebieden. Het resultaat van de segmentatie van figuur 5.11 is zien in figuur 5.13.

Een efficiënte implementatie van het algoritme toegepast op beelden wordt beschreven in [CGM02]. Deze implementatie is ingepast in BLEED en gebruikt als basis genomen voor de segmentatie van lichtpadtypes. Als kleurcomponenten worden de relatieve luminanties van de bijdragen voor de verschillende lichtpadtypes genomen. In figuur 5.15 wordt de invloed van de verschillende bandbreedteparameters nagegaan voor de segmentatie van de bijdrage van een caustic lichtpadtype.

Om veel kenmerken te vinden uit de schatting moeten de bandbreedtes laag gekozen worden. Een nadeel hiervan is dat er dan veel gebieden worden gevonden. Dit is niet erg zinvol, zowel om te benaderen als om te renderen. We hebben te maken met de volgende problemen:

- We willen grote grove gebieden, maar niet ten koste van een te sterke vermindering van de kwaliteit van de segmentatie.
- Wazige overgangen zijn moeilijk te segmenteren
- Vlekken die ontstaan door een mindere benadering van de lichtpadtypes verminderen het resultaat sterk. Een verhoging van de bandbreedte brengt geen oplossing aangezien dat er dan minder belangrijke dingen in het beeld gevonden worden.

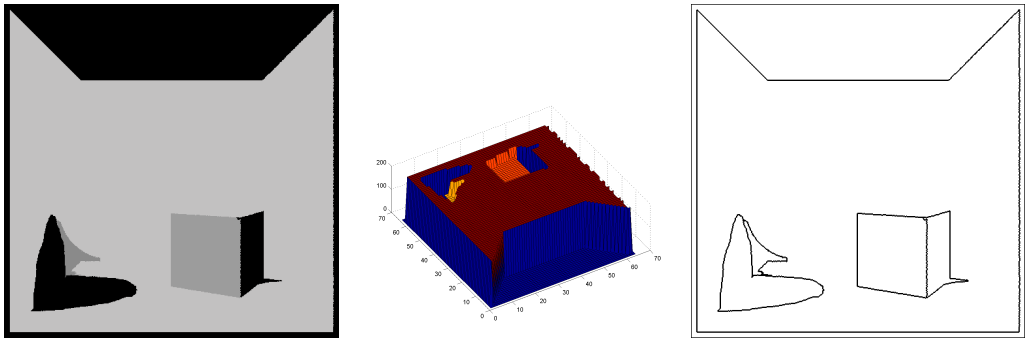
In het volgende deel worden mogelijke oplossingen voorgesteld voor deze problemen.



(a) (b)

Figuur 5.13: Voorbeeld van een beeld gefilterd met het mean shift algoritme. Als invoerbeeld werd figuur 5.11 genomen. Het beeld kan ook worden voorgesteld door een puntenwolk.

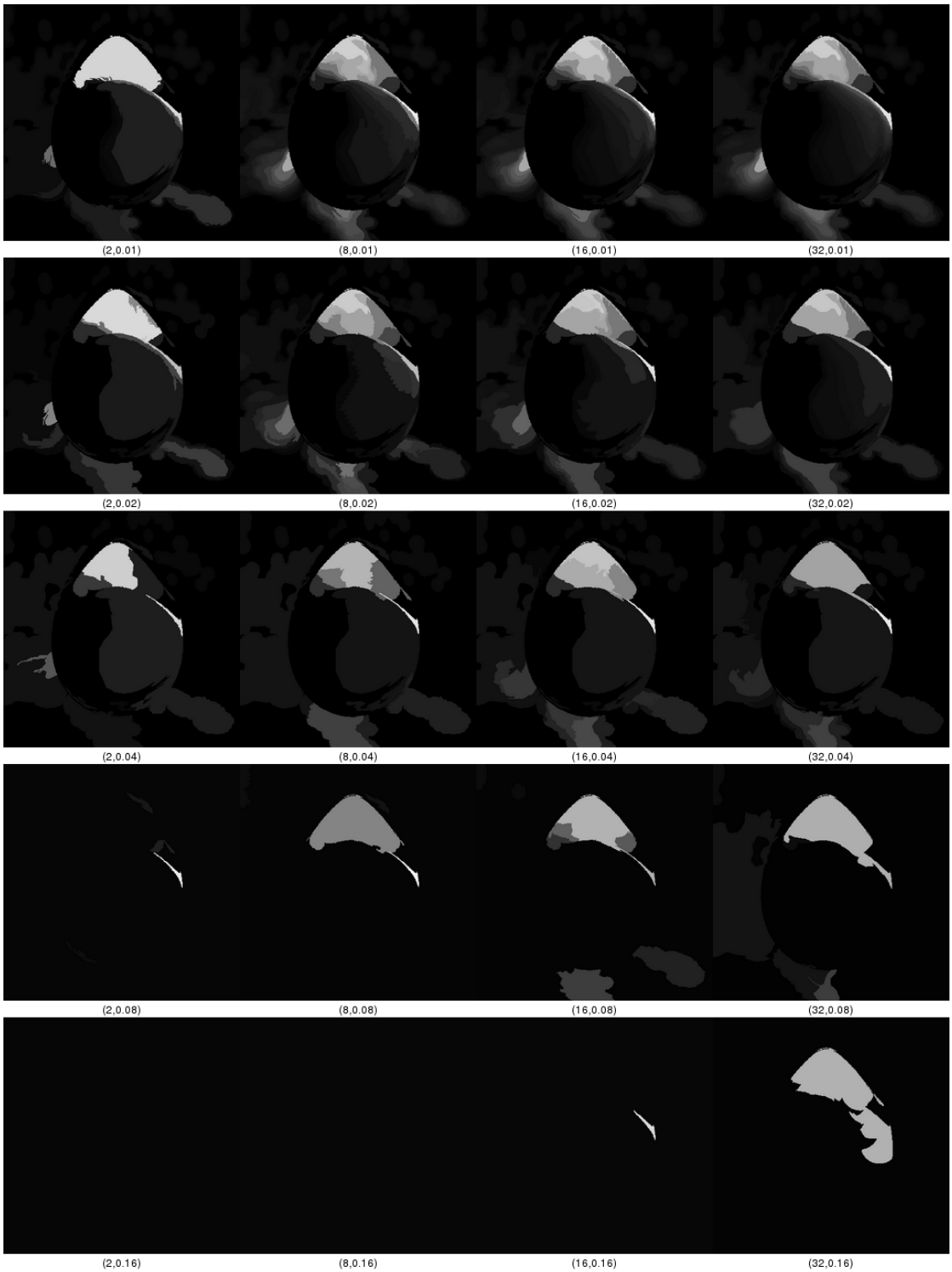
.



(a) (b) (c)

Figuur 5.14: Voorbeeld van een beeld gesegmenteerd met het mean shift algoritme. Als invoerbeeld werd figuur 5.11 genomen. Er is een zeer goede segmentatie bekomen, het algoritme heeft immers de belangrijkste gebieden kunnen vinden.

.



(a) (h_r, h_k)

Figuur 5.15: Invloed van keuze bandbreedteparameters op resultaat mean shift algoritme. De parameters h_r en h_k zijn respectievelijk de ruimtelijke bandbreedte en de kleurencomponentbandbreedte.

5.6 Aanpassingen mean shift segmentatie

In dit deel stellen we enkele oplossingen voor om een betere mean shift segmentatie te verkrijgen. Als invoerbeelden gebruiken we figuur 5.16.

5.6.1 Keuze van bandbreedteparameters

Het blijkt dat variatie van de ruimtelijke bandbreedteparameter minder effect heeft dan de variatie van de kleurcomponentenbandbreedte. In de verdere bespreking wordt de ruimtelijke bandbreedteparameter constant genomen. De bekomen resultaten van het mean shift algoritme zijn telkens gemaakt met $h_r = 5$.

In dit deel wordt de kleurcomponentenbandbreedte geschat. Hierbij krijgt elke kleurendimensie dezelfde bandbreedte.

Eerst wordt de bandbreedte op empirische wijze geschat. Vervolgens wordt uitgaande van informatie in het beeld de bandbreedte geschat. Tenslotte wordt een adaptieve versie van het mean shift algoritme besproken waar voor elke pixel een andere bandbreedte wordt bepaald. De resultaten worden op het einde van deze paragraaf besproken.

De bedoeling van deze schattingen is om steeds gebruikt te worden in samenwerking met de andere verbeteringen.

Empirische schatting

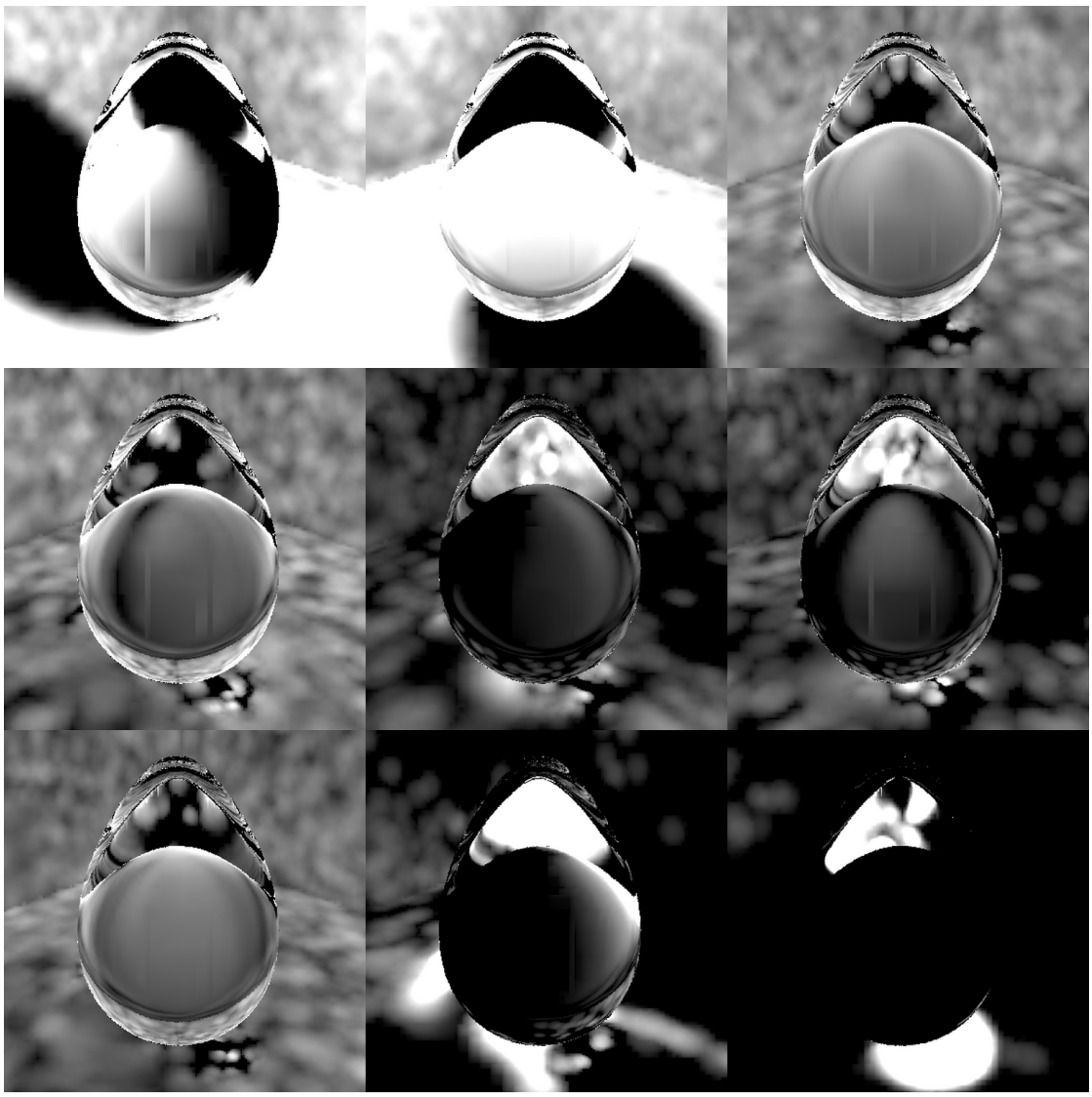
Aangezien dat we werken met relatieve schattingen, zijn de beelden steeds genormaliseerd. Hoewel de benaderingen van de verschillende lichtpadtypes veel variatie vertonen, hebben we toch op empirische wijze een bruikbare bandbreedte kunnen vinden:

$$h_{emp} = 0.02$$

Schatting per lichtpadtype

De optimale bandbreedte wordt gedefinieerd als de bandbreedte die de gemiddelde kwadratische fout tussen de geschatte dichtheid en de reële dichtheid minimaal maakt. Dit is echter niet bruikbaar, aangezien de Laplaciaan van functie die geschat wordt, niet bekend is. In de statistisch literatuur worden een aantal methodes voorgesteld om deze bandbreedte te schatten.

Een zeer eenvoudige methode is Silverman's vuistregel. Men maakt de veronderstelling dat de onbekende dichtheid niet veel verschilt van een normale Gaussiaanse dichtheid.



Figuur 5.16: Schatting bijdrages verschillende lichtpaden

De schatter van de bandbreedte is dan gelijk aan

$$h_{\text{vuist}} = \left[\frac{4\hat{\sigma}^5}{3n}\right]^{1/5} \approx 1.06\hat{\sigma}^{-1/5}$$

We hebben deze methode geïmplementeerd. De berekening ervan kan zeer snel gebeuren. Het lijkt op het eerste gezicht geen goede methode aangezien de dichtheid sterk zal verschillen van de normale dichtheid. Toepassing van deze bandbreedte zal daarom geen geen verbetering opleveren.

Deze bandbreedte moet dan ook niet afzonderlijk gebruikt worden maar in combinatie met de adaptieve schatting per pixel. Het biedt een eerste ruwe schatting.

Schatting per pixel

Indien er voor elke pixel een afzonderlijke bandbreedte gevonden kan worden, kan er potentieel een betere segmentatie bekomen worden. In [CRM01] wordt een versie van het mean shift algoritme met variabele bandbreedte voorgesteld. Dit algoritme bestaat uit vier stappen. Er wordt verondersteld dat de punten x_i met $i = 1, \ldots, n$ gegeven zijn.

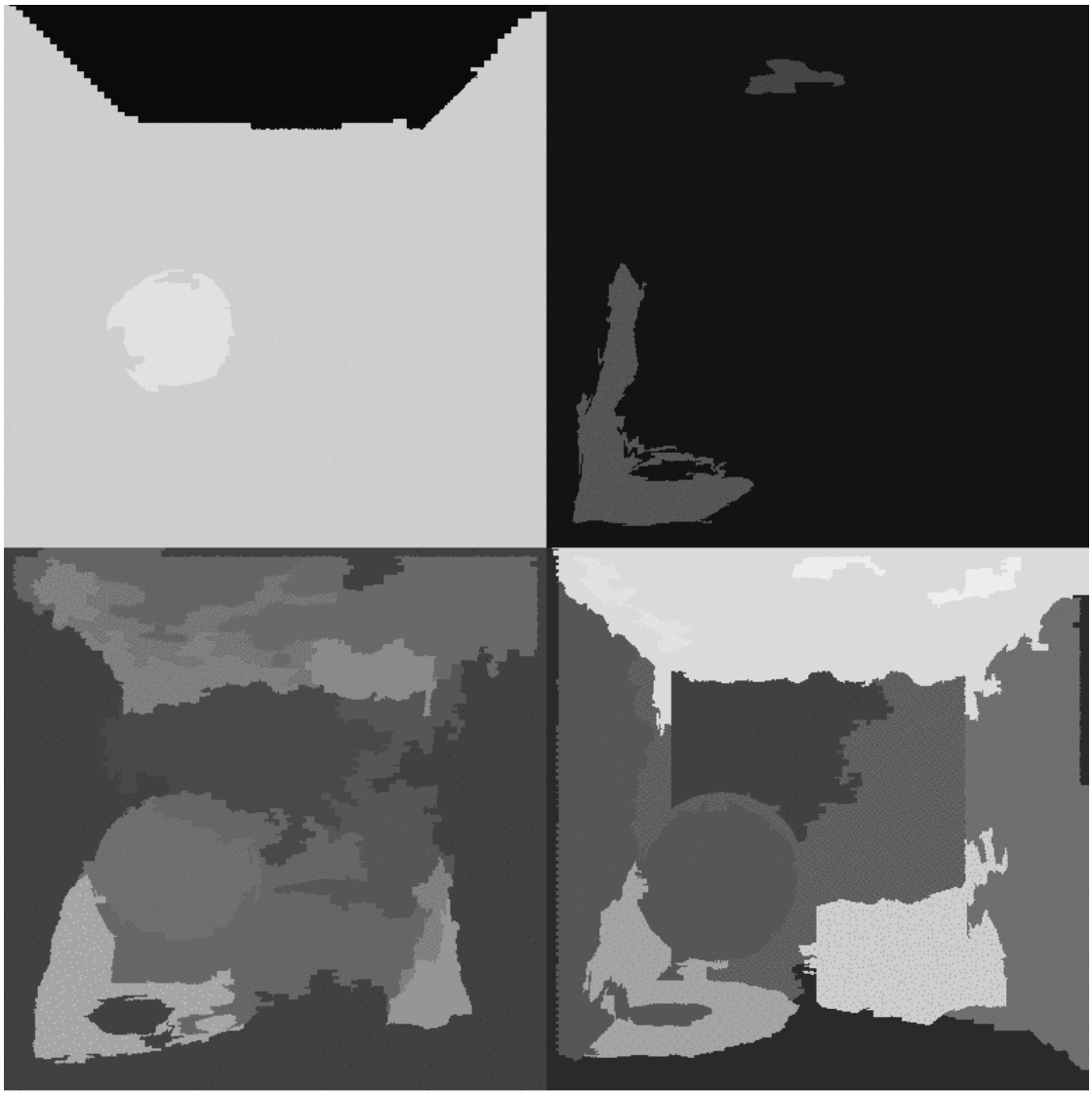
1. Zoek een eerste goede schatting van de bandbreedte h_0 en bereken voor elke pixel een schatting $\hat{f}_{h_0}(\bullet)$ van de dichtheid. Deze bandbreedte kan gevonden worden door Silverman's vuistregel.
2. Bereken $\log \lambda = n^{-1} \sum_{i=1}^{n} \log \hat{f}_{h_0}(x_i)$
3. Voor elke pixel wordt de adaptieve bandbreedte dan gevonden als

$$h(x_i) = h_0 \left[\frac{\lambda}{\hat{f}_{h_0}(x_i)}\right]^{1/2}$$

4. De segmentatie wordt bekomen op analoge manier als bij mean shift segmentatie met vaste bandbreedte. De vaste bandbreedte wordt hier vervangen door een de bandbreedte die bij de pixel hoort.

We hebben dit geïmplementeerd in BLEED. Het is te zien in figuur 5.17. Het blijkt dat de mean shift methode met variabele bandbreedte mindere resultaten geeft dan de mean shift methode met vaste bandbreedte. Dit is enigszins verrassend.

De beste resultaten hebben we bereikt met h_{emp}. Voor verdere resultaten wordt bijgevolg steeds deze bandbreedte voor de kleurenbandbreedte gebruikt.



(a)

Figuur 5.17: Schatting via adaptieve bandbreedte per pixel. De segmentatie van enkele lichtpadtypes van dezelfde scène. Er worden mindere resultaten bekomen dan indien er met een vast bandbreedte wordt gewerkt.

5.6.2 Behoud van minder belangrijke grote gebieden

Nadat er een gefilterd beeld is bekomen, worden de modes die dicht genoeg bij elkaar liggen, samengevoegd. We stellen echter een extra voorwaarde zodat grote vormen van gebieden die minder belangrijk zijn, zoals schaduwen, behouden blijven. Minder belangrijk wordt hier gezien in het kader van de bijdrage voor het lichttransport, niet voor de segmentatie. In veel gevallen zijn de randen van deze gebieden wazig wat voor moeilijkheden zorgt. Indien donkere gebieden minder makkelijk samengevoegd worden, bestaat het probleem dat er kleine gebieden te zien zijn die te wijten zijn aan een slechte benadering. In de volgende paragraaf wordt een methode beschreven om deze bekomen gebieden te verwijderen.

Indien we veronderstellen dat twee modes slecht worden samengevoegd indien

- de kleurcomponenten van beide modes dicht genoeg bij nul ligt *of*
- de kleurcomponenten van beide modes *niet* dicht genoeg bij nul ligt.

Hierdoor leiden wazige overgangen niet meer tot het samenvoegen van belangrijke en minder belangrijke gebieden. Voor een uniforme kernel en de bijdrage van één lichtpadtype ligt de kleurcomponent l_x dicht genoeg bij nul indien

$$\frac{l_x^2}{h_k^2} < \epsilon$$

met ϵ een klein getal.

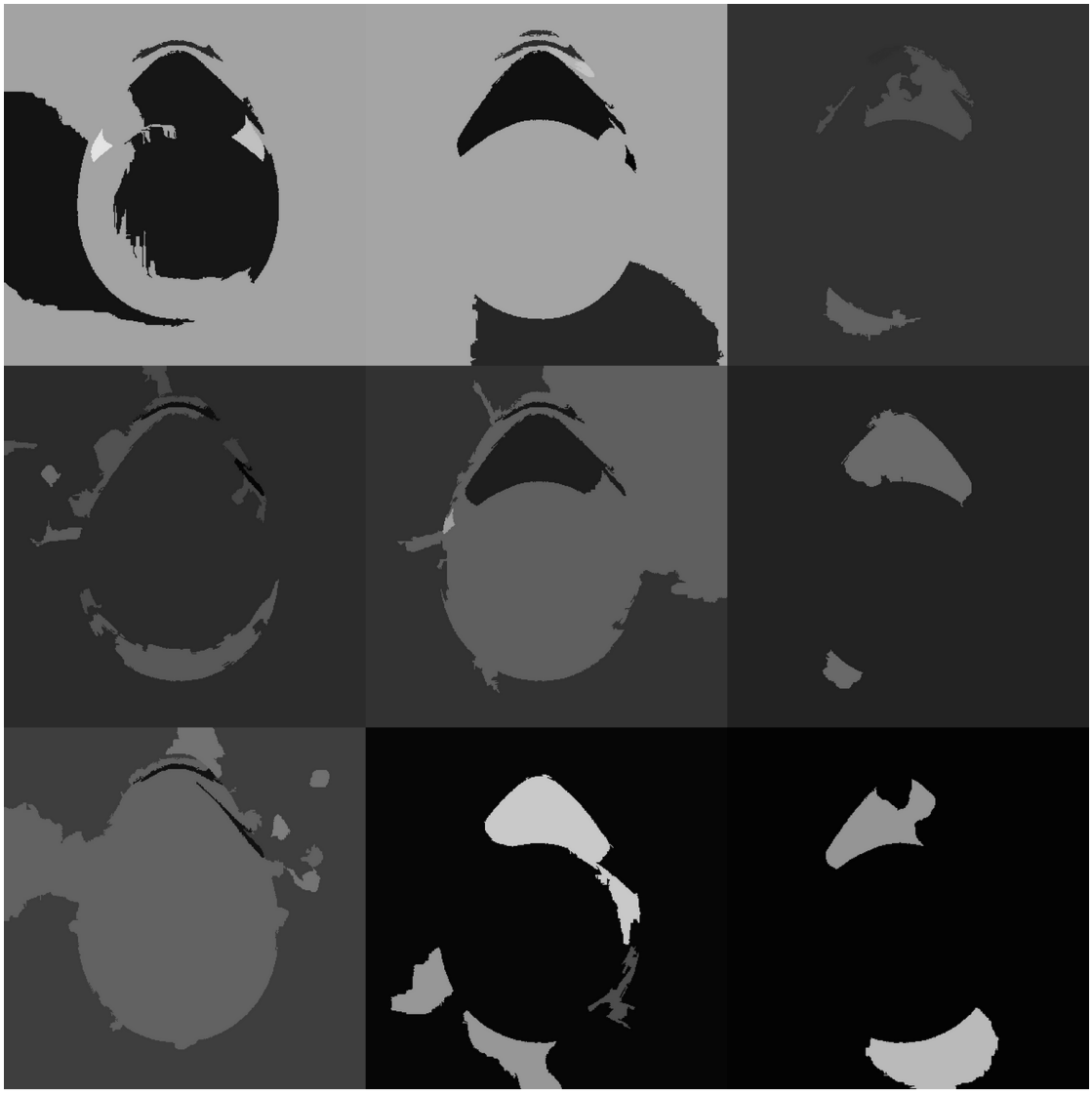
Een resultaat hiervan is te zien in figuur 5.18. Indien deze verfijning niet wordt gebruikt, wordt het resultaat in 5.19 verkregen.

5.6.3 Adaptieve minimale regiogrootte

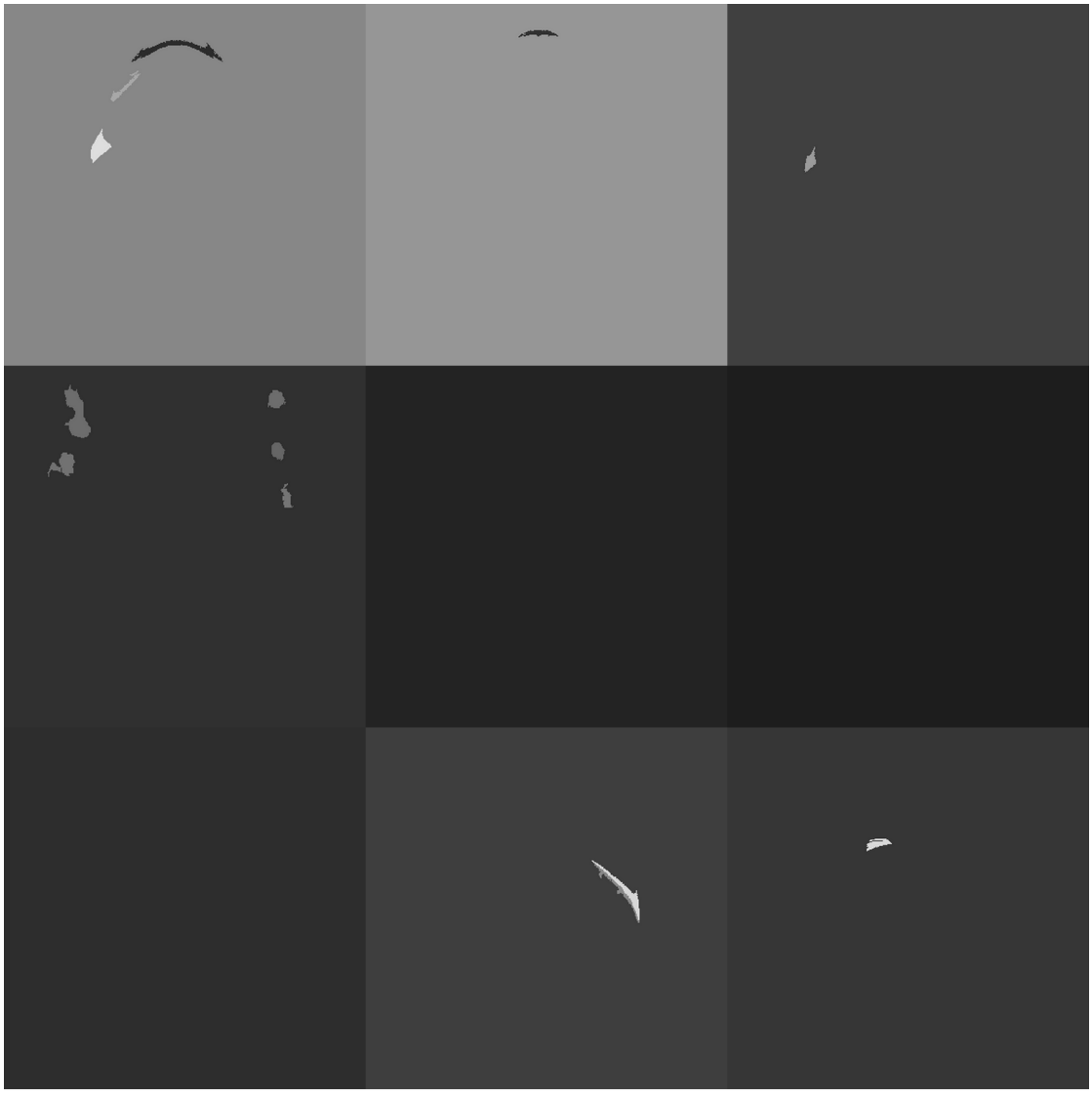
In deze paragraaf wordt de minimale grootte van de segmenten op een adaptieve manier bepaald.

Uit testen met een vaste minimale regiogrootte hebben we gevonden dat een adaptieve minimale regiogrootte moet voldoen aan de volgende eisen:

- Belangrijke gebieden mogen een klein aantal pixels bevatten. Een voorbeeld hiervan is een kleine caustic.
- Voor gebieden die minder belangrijk zijn, volstaat een zeer grove segmentatie. Indirecte lichtpadtypes zijn moeilijk goed te segmenteren omwille van de mindere benadering. Bijgevolg zullen er in veel gevallen kleine vlekken te zien zijn. Indien de minimale regiogrootte groot genoeg is, zullen deze vlekken verdwijnen.



Figuur 5.18: Behoud van minder belangrijke grote gebieden.



Figuur 5.19: Behoud van minder belangrijke grote gebieden, *niet* toegepast.

We doen dit in twee stappen. In de eerste stap wordt een minimale regiogrootte M_{beeld} voor het volledige beeld bepaald. Elk gebied dat minder pixels bevat dan M_{beeld} wordt samengevoegd met het gebied dat zich het dichtste bevindt met het te kleine gebied. Dit wordt zowel gedaan op gebied van kleurcomponenten als op pixelniveau.

Indien we veronderstellen dat

$$M_{beeld} \sim \sqrt{A}$$

met A het aantal pixels van het beeld, dan wordt de minimale regiogrootte gelijk aan

$$M_{beeld}(A) = \frac{\sqrt{A}}{C}$$

In de gebruikte testscènes is er steeds $C = 4$ genomen.

De tweede stap wordt er voor elke regio bepaald wat de bijhorende minimale grootte $M(\text{regio})$ is. We beperken ons hierbij tot beelden bestaande uit één lichtpadtype. Indien de ruimte van de kleurcomponenten een Euclidische ruimte is, kan het begrip lengte ingevoerd worden. Voor de Luv-kleurenruimte is dit mogelijk.

Hoe kleiner dat de vector met de kleurcomponenten is, des te kleiner is het belang van deze kleurenvector en bijgevolg mag de minimale regiogrootte hier groter zijn. We definiëren L_{max} als de maximale gevonden lengte in het beeld en de functie $L(\text{regio})$ geeft de lengte van de kleurcomponent in de bepaalde regio. In functie van $L(\text{regio})/L_{max}$ bepalen we nu voor elke regio de minimale regiogrootte. We nemen hiervoor een monotoon dalende functie. Het maximum stellen we gelijk aan αM_{beeld}. Een tweede parameter β wordt ingevoerd om de vorm van de functie te bepalen.

De adaptieve minimale regiogrootte kan dan gevonden worden als

$$M(\text{regio}) = M_{beeld} \cdot [\alpha - (\alpha - 1)\frac{1 - \frac{e^{\beta(1-\frac{L(\text{regio})}{L_{max}})}}{e^{\beta}}}{1 - \frac{e^{0}}{e^{\beta}}}$$

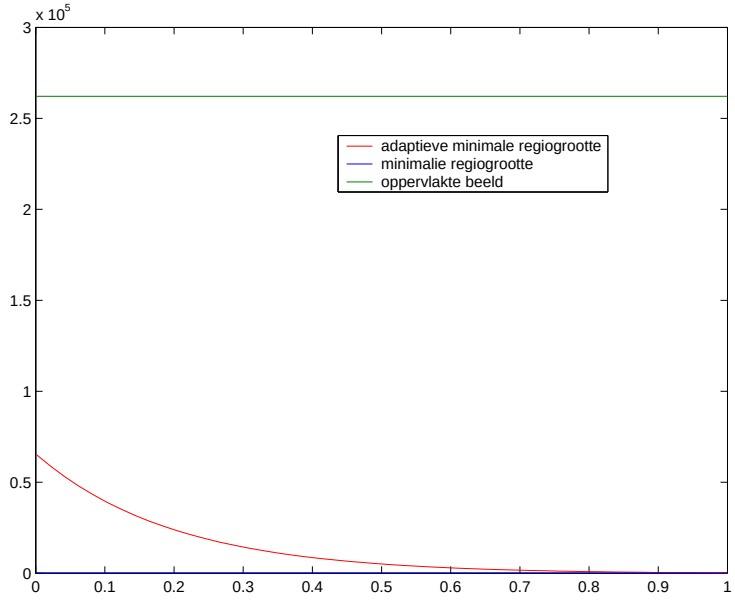
In figuur 5.20 wordt deze functie geschetst. In de gebruikte testscènes is er steeds $\alpha = \sqrt{(A)}$ en $\beta = 5$ genomen.

In figuur 5.21 worden de effecten van de adaptieve minimale regiogrootte getoond. De vorige verbeteringen werden ook toegepast.

5.7 Afronden van de segmentatie

De ontwikkelde segmentatie heeft soms de neiging gekartelde randen te vinden. De kans dat men zo'n rand wenst te bekomen is zeer klein. Daarom hebben we een extra verfijning toegevoegd aan de methode om karteling in randen weg te werken.

Het basisprincipe om een rand af te ronden gaat als volgt:



Figuur 5.20: Bepaling van adaptieve minimale regiogrootte in functie van L(regio)/L_{max}. Gebieden die te klein zijn worden samengevoegd.

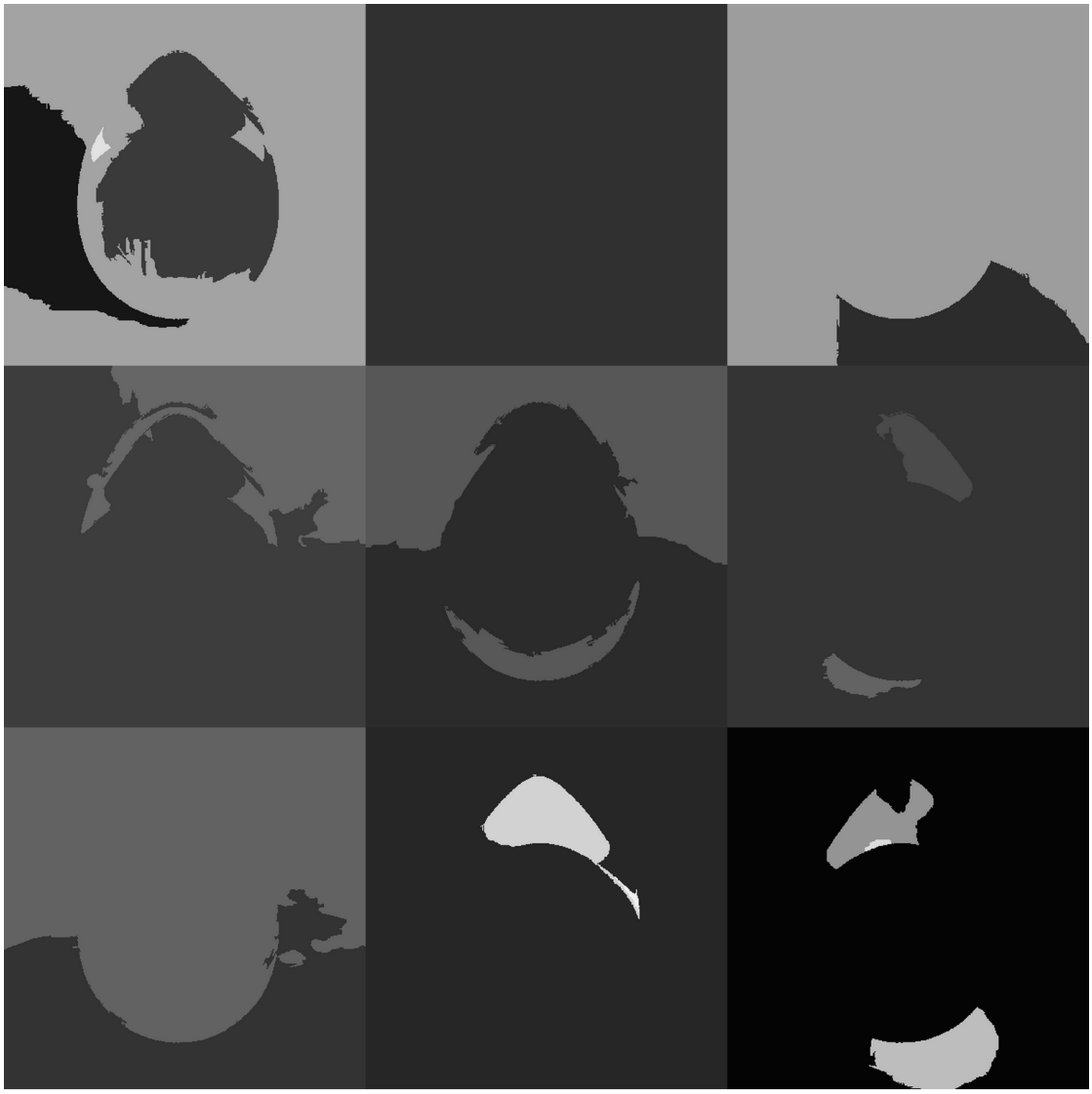
1. Er wordt een *gebied* beschouwd waar de rand aan grenst.
2. Het gebied wordt *uitgebreid* met een straal van n pixels.
3. Het gebied wordt *ingekrompen* met een straal van n pixels.

De waarde van n kan een natuurlijk getal zijn, maar dat hoeft niet.

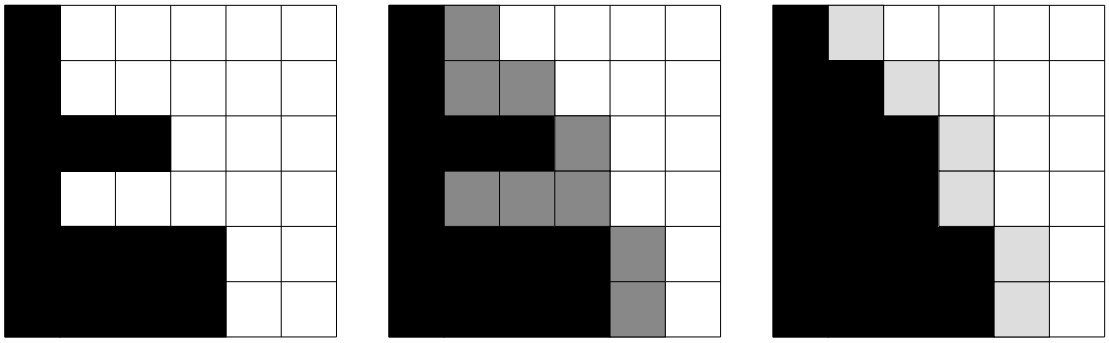
In figuur 5.22 is een voorbeeld uitgewerkt met een straal van één pixel. In 5.22(a) beschouwt men het zwarte gebied. Vervolgens breidt men in 5.22(b) het gebied uit met één pixel. De nieuwe pixels zijn aangeduid in het donkergrijs. Tenslotte wordt in 5.22(c) het gebied weer ingekrompen met één pixel. Het resultaat is een groter gebied met een meer afgeronde rand.

We merken op dat deze werkwijze om een rand af te ronden afhangt van welk gebied men beschouwt. Dat gebied kan namelijk enkel groter worden. In figuur 5.22 zijn er witte pixels zwart geworden. Hadden we het witte gebied beschouwd in plaats van het zwarte, zouden zwarte pixels wit geworden zijn. In onze implementatie hebben we dit probleem genegeerd en in niet nader bepaalde volgorde gebied per gebied afgerond. Andere resultaten kunnen dus bekomen worden door een andere volgorde te kiezen. Idealiter zouden we een techniek moeten gebruiken die deze asymmetrie niet vertoont.

In figuur 5.7 ziet men het effect van het afronden op een segmentatie. Er moet een correcte straal gekozen worden. Een te kleine straal wijzigt te weinig, een te grote straal

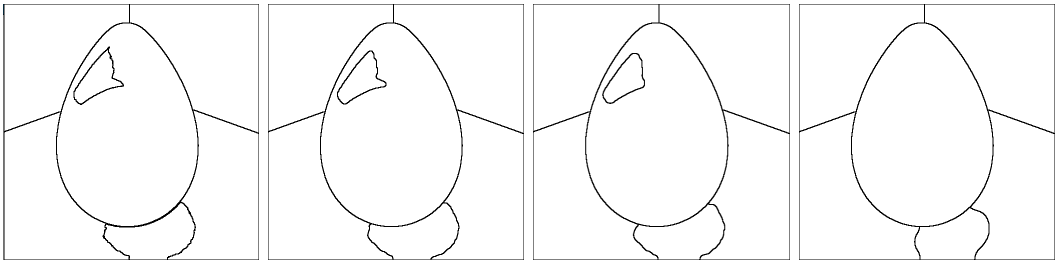


Figuur 5.21: Adaptieve minimale regiogrootte.



(a) Uitgangstoestand (b) Groeifase (c) Krimpfase

Figuur 5.22: Illustratie van het afronden van randen



(a) Geen afronding (b) Afronden met straal 4 (c) Afronden met straal 10 (d) Afronden met straal 30

vervormt scherpe randen.

We hebben er ook voor gezorgd dat geometrische grenzen niet gewijzigd worden.

5.8 Evaluatie

We evalueren nu de door ons ontwikkelde fotonsegmentatie. We hebben een segmentatie ontwikkeld die gebieden van uniforme lichtinformatie beter kan benaderen.

De meest rekenintensieve stap in het algoritme is het zoeken naar de schattingen. Als oplossing stellen we een adaptief zoeken naar schattingen voor. Hierdoor zal rekentijd in de preprocessing stap sterk verminderen. De rechthoekige interpolatie zal bij een te grote versnelling voor ongewenste artefacten zorgen. Hierdoor worden scherpe randen bekomen waarvoor de meest segmentatie algortimes heel gevoelig voor zijn.

Segmentatie via randdetectie was het eerste door ons geïmplementeerde segmentatie-algoritme. Door onoverkomelijke problemen hebben we deze aanpak verlaten en hebben een tweede segmentatie algoritme geïmplementeerd. Het mean shift algoritme is een robuust en doeltreffend algoritme. Het biedt een stevige basis voor de fotonsegmentatie.

Het rechtstreeks toepassen van mean shift is niet voldoende om een robuuste fotonsegmentatie te vinden. Enkele verfijningen zijn daarom voorgesteld.

De beste resultaten hebben we gekregen met een vaste bandbreedte. Dit is in overeenstemming met de bevindingen in [CM02]. Verdere verbeteringen leiden tot het verminderen van vlekken en het verbeteren van detectie van wazige overgangen. Een afrondinging van de segmentatie zorgt voor een verdere verbetering.

De vooropgestelde fotonsegmentatie vindt niet altijd goede resultaten. Dit mag ook niet verwacht worden van de segmentatie, hiervoor is de benaderende schatting (soms) te slecht. De ervaring leert echter dat in de meeste gevallen een bevredigende segmentatie bekomen wordt.

Hoofdstuk 6

Toepassing van de fotonsegmentatie

De bekomen segmentatie kunnen we nu toepassen in [DD05]. Daar werd er op een aantal niveaus een segmentatie gebruikt die nu kan vervangen worden.

Een eerste vervangbare segmentatie is de rechthoekige segmentatie. Deze werd ingevoerd om een betrouwbare schatting te bekomen van de lichtbijdragen over het hele beeld. In elke iteratiestap komt men dan de beste[1] schatting van het beeld door de gerenderde componenten op te tellen bij de benadering van de overige componenten op basis van deze rechthoekige segmentatie. Hiermee kan de thresholdmap opgesteld worden.

Daarnaast wordt dezelfde segmentatie toegepast om de volgorde van de componenten te bepalen. Het verschil in de luminantie tussen de geometrische en rechthoekige segmentatie is een maat voor prioriteit. Deze rechthoekige segmentatie vervangen we voor de twee gevallen in deel 6.2.

Vervolgens wordt er voor de gevisualiseerde benadering gebruik gemaakt van de geometrische segmentatie. Deze vervangen we op zijn beurt in deel 6.3.

Tenslotte bepaalt eveneens de geometrische segmentatie hoe de componenten worden opgedeeld in vlak van het beeld. De verfijning naar fotonsegmentatie behandelen we in deel 6.4.

Tabel 6.1 toont de opeenvolgende verfijningen.

[1] Beste in de zin dat de radiantie goed geschat wordt. Het bevat echter teveel artefacten om voor directe visualisatie gebruikt te worden.

Sectie	thresholdmap	prioriteiten	benadering	componenten
3	rechthoekig	rechthoekig	geometrisch	geometrisch
6.2	foton	foton	geometrisch	geometrisch
6.3	foton	foton	foton	geometrisch
6.4	foton	foton	foton	foton

Tabel 6.1: Opeenvolgende verfijningen: gebruikte segmentatie in functie van het doel

6.1 Segmentatie per lichtpadtype

De fotonsegmentatie kan op twee manieren gebruikt worden:

- één algemene segmentatie, of
- een segmentatie per lichtpadtype.

In [DD05] gebruikte men enkel rechthoekige en geometrische segmentatie. Deze is uiteraard gemeenschappelijk voor alle lichtpadtypes. De voor de hand liggend keuze is dan te segmenteren op het totaalbeeld van alle lichtpadtypes bekomen uit de photon mapping.

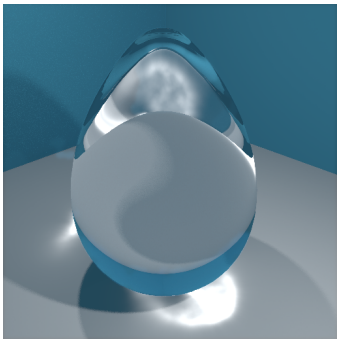
In het ideale geval vindt die segmentatie alle effecten in het beeld: schaduwen, caustics, ... We kunnen bijvoorbeeld voor lichtpadtypes opsplitsen volgens de schaduwen. Voor het lichtpadtype "direct licht" is deze opdeling zeer zinvol. De detectie van de schaduw is in het "caustic" lichtpadtype echter irrelevant. Dit wordt geïllustreerd in figuur 6.1 waar de segmenten uniform ingekleurd worden met hun gemiddelde radiantie. Voor de component (ondervlak, direct licht) is de segmentatie quasi perfect. Er valt amper verschil te merken tussen figuur 6.1(g) en figuur 6.1(h), terwijl het verband tussen figuur 6.1(i) en figuur 6.1(j) minder duidelijk is. Indien er een correlatie is tussen het causticlicht binnen en buiten het schaduwgebied is, zal dat eerder toevallig zijn.

We kunnen nagaan hoe zinvol de segmentatie van het volledige beeld is. Stel:

- er zijn per lichtpadtype twee gebieden te onderscheiden: met en zonder licht
- er zijn l lichtpadtypes
- de segmentatie wordt niet verfijnd met geometrische segmentatie.

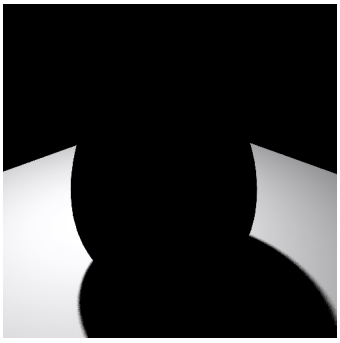
Dan worden er in ideale segmentatie 2^l gebieden onderscheiden in het totaalbeeld. Componenten worden ook nog eens verdeeld per lichtpadtype, wat resulteert in $l \cdot 2^l$ componenten.

We kunnen ook nagaan hoeveel nuttige componenten zijn. Per lichtpadtype zijn er twee gebieden. Voor alle lichtpadtypes samen is dat $2l$ componenten. Het weze duidelijk dat

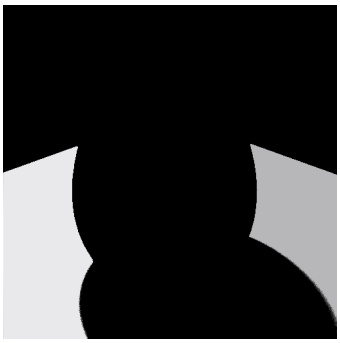


(e) De scène

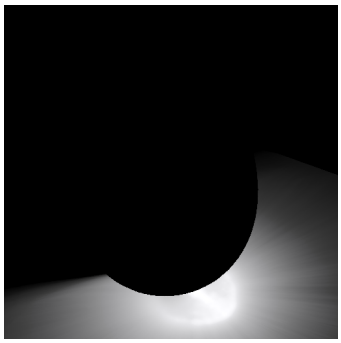
(f) Segmentatie van het ondervlak volgens de schaduw



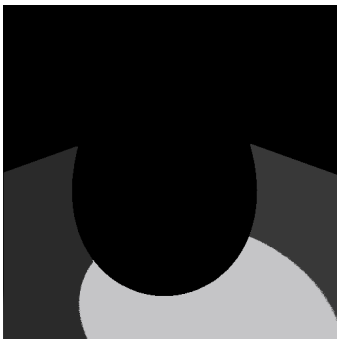
(g) De component (ondervlak, direct licht)



(h) Segmentatie van (ondervlak, direct licht)



(i) De component (ondervlak, caustic)



(j) Segmentatie van (ondervlak, caustic)

Figuur 6.1: De effecten van een gemeenschappelijke segmentatie

een ideale globale fotonsegmentatie van het beeld veel overtollige componenten zal produceren. De complexiteit is onaanvaardbaar indien men wil experimenteren met een groter aantal lichtpadtypes. Bovendien leert de ervaring ons dat een overtollige segmentatie eerder een storend effect zal opleveren in de tussenresultaten.

Nu werd er van de ideale situatie uitgegaan. We vertrekken echter van een beeld met fotonruis zoals in figuur 5.1. Er zullen vaak gebieden gevonden worden die toevallig wegens ruis wat helderder zijn en anderzijds gebieden niet gevonden worden omdat ze te subtiel zijn om te detecteren, of omdat sommige effecten elkaar overlappen. Door elk lichtpadtype apart te segmenteren krijgen we een betrouwbaardere segmentatie waarbij belangrijke effecten zeker gevonden worden.

Omwille van deze twee argumenten (overtollige componenten, mindere segmentatie) hebben we geopteerd om per lichtpadtype te segmenteren. Dit vereiste een grondige aanpassing in raamwerk BLEED ontwikkeld in [DD05].

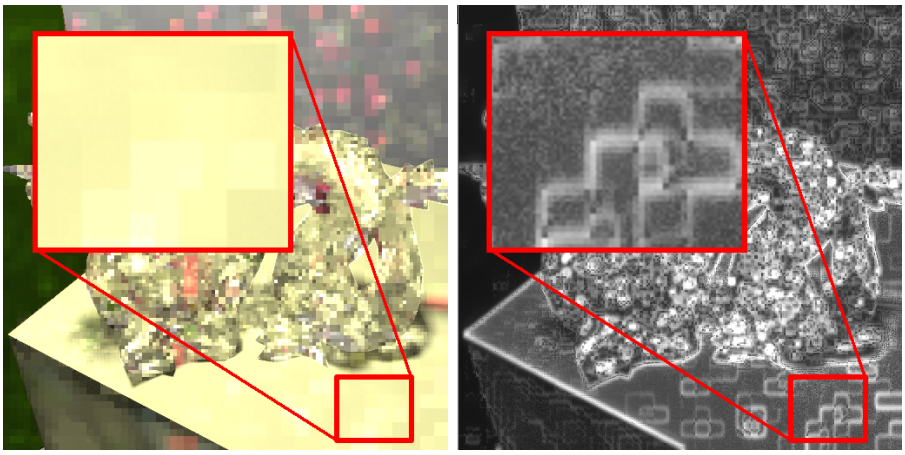
6.2 Berekening van thresholdmap en prioriteiten met fotonsegmentatie

Deze twee verfijningen worden tezamen behandeld omdat ze logisch uit elkaar volgen. De thresholdmap is de kaart van de drempelwaardes waarvoor het menselijk oog gevoelig is. Om dit te berekenen is een zo volledig mogelijk beeld van het resultaat nodig. Om de prioriteiten van de componenten te berekenen tracht men zo nauwkeurig mogelijk de variantie binnen een component te meten. Hier is dus eveneens een zo volledig mogelijk beeld mogelijk. Het spreekt voor zich dat de implementatie in BLEED hiervoor hetzelfde gebruikt. Een ontkoppeling van de twee berekeningen zou resulteren in een uitbreiding van het raamwerk en eveneens resulteren in overhead; er zouden dan twee benaderingen van het beeld berekend worden in plaats van één.

6.2.1 Thresholdmap

De threshholdmap is de gegevensstructuur die per pixel bijhoudt welke variantie toegelaten is opdat het menselijk oog geen ruis kan waarnemen, zoals besproken in deel 3.7.1. Deze methode heeft in theorie als invoer het volledig geconvergeerd beeld. Deze is uiteraard niet op voorhand bekend zodat men moet voortgaan op wat men al heeft. In [DD05] is dat al redelijk wat:

1. De reeds gerenderde componenten.
2. De benadering van de radiantie per voorwerp (geometrie).
3. De benadering van de radiantie per rechthoek.



(a) Schatting radiantie bij rechthoekige segmentatie

(b) Thresholdmap bij rechthoekige segmentatie

Figuur 6.2: Het effect van rechthoekige segmentatie op de thresholdmap

Aangezien een zo volledig mogelijk beeld gewenst is, is er in [DD05] geopteerd voor 1+3. De rechthoekige opdeling levert echter artefacten op. Deze zijn niet zo schadelijk aangezien ze niet dienen om te visualiseren. Wel zorgt het ervoor dat de drempelwaarde in de grensgebieden hoger is en binnen de gebieden lager is dan normaal, zie figuur 6.2. De menselijke perceptie voor fouten daalt namelijk met de frequentie in het beeld.

Door deze rechthoekige segmentatie te vervangen door fotonsegmentatie wordt dit probleem ietwat verholpen. De bekomen gebieden zijn over het algemeen groter en volgen beter de visuele effecten. De vergelijking wordt in figuur 6.3 getoond.

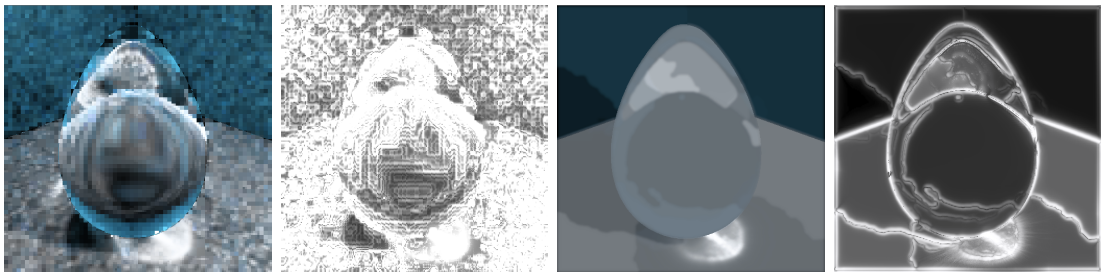
6.2.2 Prioriteiten

Bij het bepalen van de prioriteiten van de componenten wordt er vooral nagegaan of de benadering over geometrie wel klopt. In deel 3.6.2 werd hiervoor het verschil tussen benadering per voorwerp en benadering per rechthoek berekend.

Deze rechthoekige segmentatie vervangen door fotonsegmentatie heeft weinig invloed zolang de segmentatie correct gebeurd is.

6.2.3 Evaluatie

De twee behandelde verfijningen voeren een conceptuele verbetering in. De verwachte verbetering is echter minimaal en bijna onmogelijk na te gaan. De berekende drempel-



(a) Schatting radiantie bij rechthoekige segmentatie

(b) Thresholdmap bij rechthoekige segmentatie

(c) Schatting radiantie bij fotonsegmentatie

(d) Thresholdmap bij fotonsegmentatie

Figuur 6.3: Vergelijking thresholdmaps. Het betreft telkens de toestand na de eerste iteratie waar reeds één caustic is gerenderd. We merken op dat bij rechthoekige segmentatie de drempelwaarden vaak overschat worden.

waardes zullen niet spectaculair verbeteren en de prioriteiten zullen grotendeels bewaard worden.

Het concrete nut is het vermijden van de rechthoekige segmentatie. Aangezien we van plan zijn de fotonsegmentatie verder door te voeren in het raamwerk, is deze zeker nodig. Nu blijkt deze een goede vervanger te zijn voor de rechthoekige segmentatie. Door deze laatste niet te gebruiken, kan men besparen op tijd en geheugen voor deze structuur. Concrete cijfers hieromtrent zijn te vinden in hoofdstuk 7. Een voorbeeld kan men vinden in tabel 7.1.

6.3 Verfijning van de gevisualiseerde benadering met fotonsegmentatie

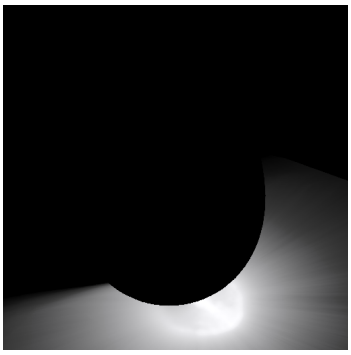
Een interessantere toepassing van de fotonsegmentatie is het berekenen van de benadering van niet-berekende componenten. Op figuren 6.4(d)–(f) ziet men dat een eenvoudig opsplitsing een drastische verbetering van de benadering kan opleveren.

Er moet echter opgepast worden met de extra segmentatie. Vroeger kon er aangenomen worden dat grenzen tussen gebieden een scherpe rand vertoonden aangezien het ook overgangen tussen verschillende voorwerpen waren. Met fotonsegmentatie zijn de gebieden nu meestal kleiner en vertoont de segmentatie ook grenzen binnen in de voorwerpen.

(a) Gerenderde component

(b) Benadering per voorwerp

(c) Benadering per fotonsegment



(d) Gerenderde component

(e) Benadering per voorwerp

(f) Benadering per fotonsegment

Figuur 6.4: Verbetering van de benadering door fotonsegmentatie: Bovenaan was de segmentatie reeds in orde en wordt het ook niet verfijnd. Onderaan negeerde de geometrische benadering een belangrijke caustic die met fotonsegmentatie wel gevonden wordt.

(a) Oorspronkelijke benadering

(b) Na het toepassen van een vervagingsfilter

(c) Na het toepassen van een gepartitioneerd vervagingsfilter

Figuur 6.5: Behandeling van kartelingen in de benadering: (a) toont de benadering van een caustic component: het wit is de kern van de caustic, het grijs is de staart en het zwart maakt geen deel uit van de component, (b) heeft de hele figuur vervaagd met als neveneffect dat er kleur is uitgewisseld binnen en buiten de component, (c) past gepartitioneerde vervaging toe zodat er enkel binnen de component vervaagd wordt.

6.3.1 Scherpe randen

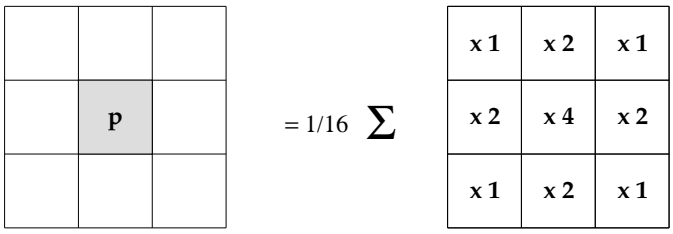
Een eerste probleem met de nieuwe grenzen is dat pixels aan de ene kant van de rand een bepaalde waarde hebben en de andere kant een andere waarde, mogelijk sterk verschillend. Door de wijze waarop gesegmenteerd wordt (per pixel, een pixel kan niet slechts voor de helft behoren tot een gebied) verkrijgt men kartelingen[2], zie figuur 6.5(a).

Naast de kartelingen is het ook zo dat de overgang ook gewoon abrupt is. Het zou best wel kunnen dat het eigenlijk een gradiënt moet zijn, die door de segmentatie opgedeeld in een paar discrete uniforme delen.

Men bekomt dus misschien wel een juister beeld (meer gedetailleerde benadering), maar het is visueel minder aantrekkelijk. Een eenvoudige oplossing hiervoor is het vervagen van het beeld.

Bij de geometrisch segmentatie waren de randen geen probleem. Men kan namelijk verwachten dat overgangen tussen verschillende voorwerpen discontinu zullen zijn. Er kan zelfs sterker gesteld worden: deze grenzen mogen niet vervaagd worden.

[2]Doorgaans wordt de term “jaggies” gebruikt voor kartelingen



Figuur 6.6: Toepassing van een gaussiaanse vervagingsfilter: de nieuwe radiantiewaarde voor pixel p is het gewogen gemiddelde van zijn buren.

Vervaging

Bij het vervagen heeft men ruwweg twee parameters: methode en straal. Qua methode hebben we gekozen voor een gaussiaanse convolutie matrix van dimensies 3×3

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

Door een beeld met deze matrix te convolueren bekomt men vervaging met straal 1. Een gaussiaanse vervaging van straal r kan men eenvoudig bekomen door r maal met deze matrix te convolueren.

In figuur 6.6 wordt deze matrix eenmaal toegepast op pixel p.

Op het eerste zicht zouden we de straal van de vervaging proportioneel met de afmeting nemen, maar dat heeft zijn nadelen. Bij grote beelden zou men dan de benadering over grote straal vervagen, wat in schril contrast zou staan tot de scherpe gerenderde componenten. In dit opzicht is het beter een vaste straal te nemen. Men kan argumenteren dat een vaste straal een klein beeld vernielt, maar kleine beelden zijn echter nooit een gewenst resultaat.

Bij deze vervaging is het het doel om de grenzen van twee gebieden binnen één voorwerp minder opvallend te maken. Zoals eerder vermeld, worden grenzen tussen verschillende voorwerpen echter wel verwacht scherp te zijn. Deze mogen hoe dan ook niet vervaagd worden, want dat zou betekenen dat licht van één voorwerp op een ander kan springen. Felle voorwerpen in in een donkere omgeving zouden dan een gloed geven aan de omringende voorwerpen, wat onaanvaardbaar is. Het verschil tussen het bekomen beeld en het gewenste beeld is te zien op figuren 6.5(b) en 6.5(c).

Gepartitioneerde vervaging

Om dit probleem op te lossen is de notie van gepartitioneerde vervaging ingevoerd. Aan het vervagingsalgoritme wordt een partitie meegegeven. Het algoritme wordt zo aange-

$$\begin{bmatrix} 1 & 2 & 2 \\ 1 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

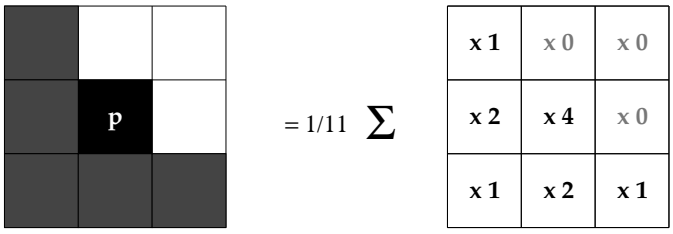
(a) Partitie

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

(b) Algemene convolutiematrix

$$\frac{1}{11}\begin{bmatrix} 1 & 0 & 0 \\ 2 & 4 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

(c) Lokale convolutiematrix



(d) Toepassing op pixel p

Figuur 6.7: Lokale invloed van een partitie op de convolutiematrix bij gepartitioneerde vervaging

past dat pixels enkel bijdragen kunnen krijgen die afkomstig zijn van dezelfde deelverzameling in die partitie. In figuur 6.7 wordt dit geïllustreerd voor de middelste pixel behorend bij verzameling 1 van de meegeleverde partitie. De partitie is in figuur 6.7(a) voorgesteld als een matrix. Elk deelverzameling van de partitie is de verzameling van elementen dat een gelijke waarde heeft. In figuur 6.7(d) wordt dit gevisualiseerd met kleuren: wit en donkergrijs/zwart.

6.3.2 Vorm van de randen

Een tweede punt waar aandacht aan moet besteed worden bij de nieuwe randen is de vorm ervan. Bij de bespreking van de vervaging wilden we kartelingen op pixel niveau vermijden. Er kunnen echter ook andere kartelingen ontstaan door een segmentatie die moeite heeft een goede opsplitsing te maken. Het afronden van randen is reeds besproken in deel 5.7.

6.3.3 Fotonbenadering

Fotonbenadering biedt een alternatief voor de uniforme benadering. Er wordt nagegaan hoe op een efficiënte manier een betere benadering kan verkregen worden uit de fotonsegmentatie. Op sommige plaatsen is deze benadering reeds zeer goed. Caustics kunnen efficiënt gerenderd worden door gebruik te maken van photon mapping wat de goede benadering in de fotonsegmentatie verklaart. Een hergebruik hiervan kan potentieel betere tussenresultaten geven dan een uniforme benadering.

Er wordt telkens gewerkt per lichtpadtype. De gevonden benadering is echter niet onmiddelijk bruikbaar. Deze bevat immers te veel vlekken. Een vervagingsfilter kan deze vlekken doen verdwijnen. Indien deze uniform wordt toegepast op de benadering van het lichtpadtype zal er echter geen goede benadering worden bekomen. Er zal een grote vervagingsfilter gekozen moet worden om alle vlekken te verwijderen. Hierdoor wordt de benadering zeer wazig. Dit wazige effect wordt door de menselijke waarnemer niet herkend als realistisch. Het voordeel van een fotonbenadering verdwijnt hierdoor.

In elke pixel heeft de schatting een andere *nauwkeurigheid*. Indien deze laag is, is het wenselijk dat de vervagingsstraal groot is. Hoge nauwkeurigheiden leiden tot zeer kleine vervagingsstralen. Een adaptief vervagingsalgoritme zorgt ervoor dat pixel vervaagd wordt met een andere vervagingsstraal. Hierdoor wordt er wel een potentieel betere benadering bekomen.

Het fotonbenaderingsalgortime ziet er als volgt uit:

1. Bepaal voor elke pixel wat de optimale vervagingsstraal is.
2. Pixels waarvan de benadering zeer slecht is, worden vervangen door de gemiddelde waarde van de radiantie in het gebied.
3. Voer het adaptief vervagingsalgortime uit.
4. Herschaal de benadering zodat de gemiddelde radiantie gelijk is aan de gemiddelde uniforme radiantie.

Dit algoritme wordt per lichtpadtype uitgevoerd.

Zoeken naar vervagingsstralen

De bedoeling is voor elke pixel de optimale vervagingsstraal te zoeken. De bijdrage van één foton kan invloed hebben op meerdere pixels. Indien de foton invloed heeft op veel pixels, is de benadering in die pixels minder goed dan indien de foton maar één pixel beïnvloedt.

We zoeken nu per pixel x de verste pixel y (in *Manhattan*-afstand) waarvoor geldt dat in y een foton zit dat ook in x zit. In elke pixel worden verwijzingen naar fotonen die bijdragen tot deze pixel bijgehouden. De fotonen worden best per lichtpadtype opgeslagen zodat er sneller gezocht kan worden. Dit zoeken is de traagste stap van de fotonbenadering. Om het beeld (256×256 pixels) met het glazen ei te renderen duurt deze stap op een Pentium 4 2,0 GHz gemiddeld 400 seconden. De totale preprocessingtijd is gelijk aan ongeveer 800 seconden.

Vervanging slechte benaderingen

Pixels waarvan de benadering zeer slecht is, hebben een zeer grote vervagingsstraal nodig. Voor goede resultaten moet deze straal groter gekozen worden als de gevonden vervagingsstraal. Hierdoor duurt het adaptief vervagingsalgoritme langer. Deze benadering zal voor grote stralen steeds meer in de buurt van de uniforme benadering komen.

Pixels waarvan de benadering zeer slecht is, worden daarom vervangen door de uniforme benadering in het gebied. Dit zijn pixels waarvoor de vervagingsstraal $r(i)$ groter is dan een waarde r_{max}. We hebben r_{max} gekozen als een kwart van de maximale gevonden vervagingsstraal. Het aantal nodige vervagingen daalt hierdoor sterk.

Om zichtbare artefacten tegen te gaan, worden deze benaderingen ook nog vervaagd met straal r_{max}.

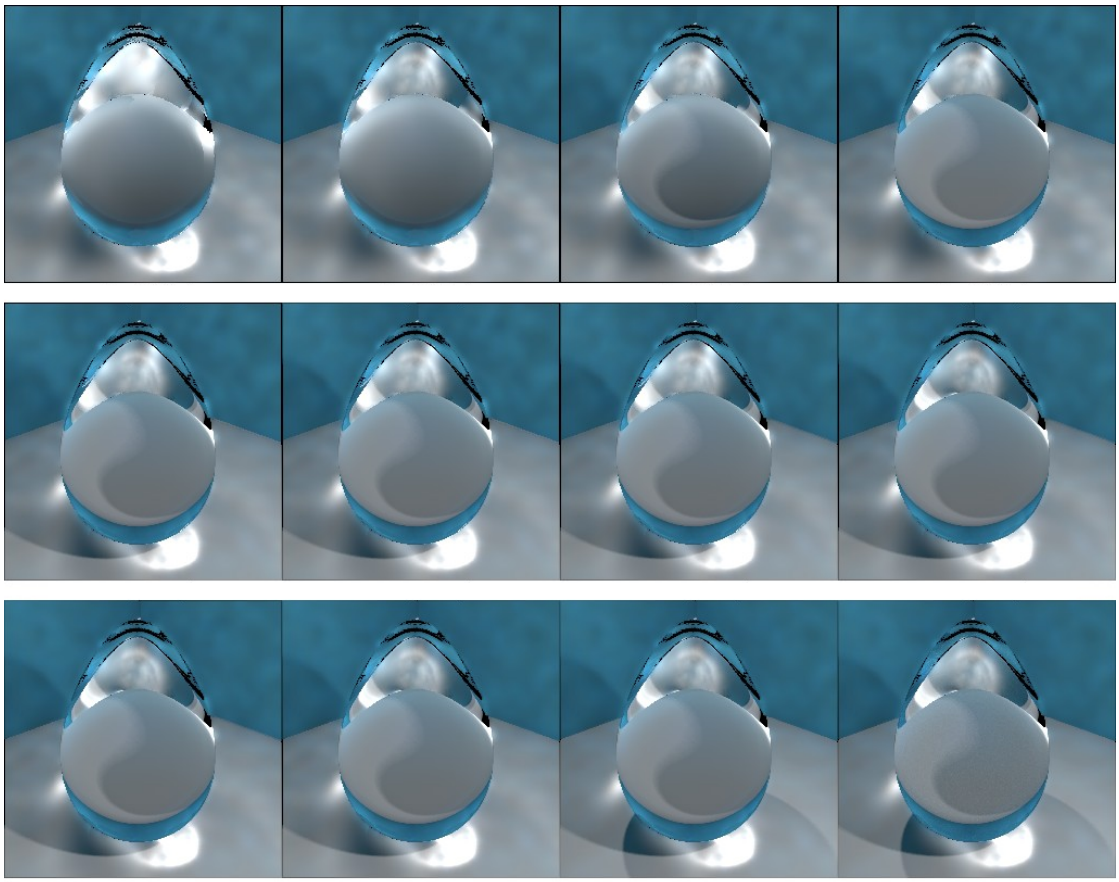
Adaptief vervagingsalgoritme

Er wordt r_{max} keer een vervagingsfilter uitgevoerd op de benaderde schatting. Per vervaging worden de pixels waarvan de vervagingsstraal gelijk is aan de huidige vervagingstraal, gekopieerd naar het eindresultaat. Dit vervagen gebeurt met behulp van de Gaussiaanse filter die reeds eerder werd besproken. Uiteindelijk zijn alle pixels adaptief vervaagd.

Toepassing

In figuur 6.3.3 hebben we de fotonbenadering toegepast op een beeld (256×256). De caustics en de speculaire reflecties van de caustics in het ei zijn reeds zeer goed benaderd. Dit wordt aangetoond door de opeenvolgende gerenderde componenten: deze componenten zullen weinig verbeterd worden. Bij de berekening van prioriteiten wordt hier geen rekening mee gehouden. Voor een optimaal gebruik van de fotonbenadering is dit echter wel noodzakelijk.

De kwaliteit van de schatting is sterk verbeterd, maar is nog niet perfect. Op de muren en op de grond zijn lichte vlekken te zien. Dit is het gevolg van de indirecte diffuse



Figuur 6.8: Toepassing van de fotonbenadering. De combinatie van gerendere en benaderde componenten is te zien in opeenvolgende stappen. De grote caustic op het grondvlak is reeds gerenderd in de eerste stap.

belichting die moeilijk te benaderen is. Hier geeft een uniforme benadering veel betere resultaten.

De extra overhead door het gebruik van de fotonbenadering in de renderingstap is te verwaarlozen.

Een nadeel van deze aanpak is dat er zeer veel geheugen nodig is. Op een 32-bit computer heeft een verwijzing of *pointer* 4 bytes in het geheugen nodig. Indien er gezocht wordt naar de 160 dichtste fotonen zal er voor een beeld bestaande uit 512×512 pixels minstens 160.000 bytes extra in het geheugen moeten worden gealloceerd. Om een prentje met 10 verschillende lichtpadtypes te renderen vraagt de fotonbenadering dus een kleine twee gigabyte geheugen. Hoewel geheugen de laatste jaren steeds goedkoper is geworden, is dit onaanvaardbaar.

6.3.4 Evaluatie

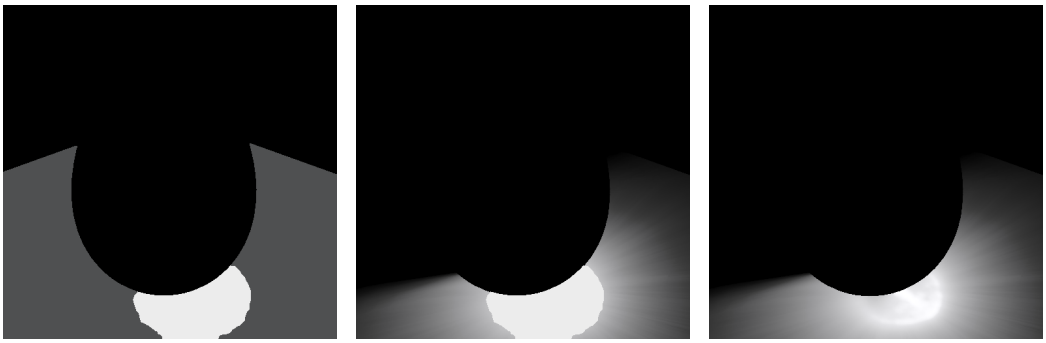
Het voordeel van het verfijnen van de benaderingen is dat de benaderingen ook belangrijke effecten uit de scène kunnen tonen, in de mate dat deze correct gevonden werden door de segmentatie.

Wanneer we deze verfijning implementeren met uniforme benaderingen, verkrijgen we in het begin betere beelden aangezien de benadering beter is. Het systeem zit echter zo in elkaar dat benaderingen die vroeger slecht waren eerst werden vervangen. Na enkele stappen is het verschil dus verdwenen. Uiteindelijk zal het beeld identiek zijn: enkel de voorlopige benadering is anders. Om voordeel te halen uit deze methode zal de preprocessing (densiteitsschatting, fotonsegmentatie) snel genoeg moeten zijn.

Fotonbenadering is een alternatief voor de uniforme benadering. Toepassing hiervan levert voor bepaalde lichtpadtypes een zeer sterke verbetering. Het grote geheugengebruik en de mindere resultaten voor indirecte lichtpadtypes maken deze methode echter minder aantrekkelijk.

6.4 Verfijning van de rendercomponenten met fotonsegmentatie

Een laatste punt waar de geometrische segmentatie gebruikt wordt, is bij het opdelen in componenten. Aangezien de fotonsegmentatie een verdere opdeling is van de geometrische segmentatie, betekent dat componenten verder opgedeeld worden. Vergeleken met de vorige methode is het enige verschil dus dat het renderwerk fijner is verdeeld over componenten, zoals in figuur 6.9. De nieuwe componenten hoeven echter niet in die volgorde of na elkaar gerenderd worden. De berekeningen kunnen verweven worden met andere componenten.



(a) Stap 1 (b) Stap 2 (c) Stap 3

Figuur 6.9: Fijnere verdeling van de componenten: Bij componenten per voorwerp gaat men rechstreeks van (a) naar (c). De fotonsegmentatie vindt de kern van de caustic zodat deze component in twee wordt gesplitst. Resultaat is de extra tussenstap (b).

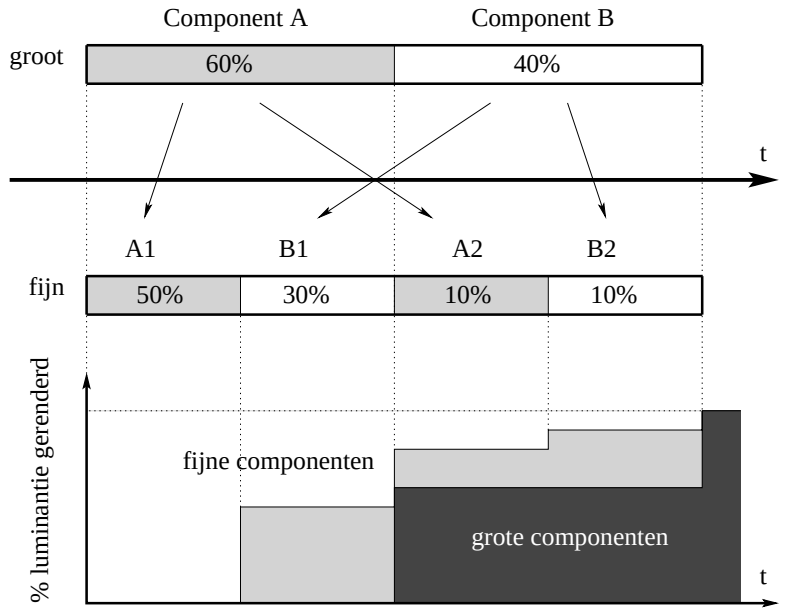
Door de componenten fijner op te splitsen, wordt het renderwerk ook verder verfijnd. Een voorbeeld van het nut hiervan is bij een voorwerp waar enkel een klein deel van het zichtbare oppervlak een bepaald lichteffect (bijvoorbeeld een caustic) bevat en de rest quasi onbeschenen is. De prioriteit van het onbeschenen deel zal zeer laag zijn en dus pas op het einde gerenderd worden. Een voorbeeld wordt geïllustreerd in figuur 6.10. Er zijn twee componenten: A en B. Beide bestaan ze uit een belangrijke en minder belangrijke subcomponent: resp. A1, B1 en A2 en B2. Door de verfijning wordt de volgorde van het renderen veranderd, zodat de belangrijkste delen (A1 en B2) eerst worden gerenderd (in plaats van eerst A1 en A2 samen als A). Er wordt verondersteld dat elke component evenveel tijd nodig heeft om te renderen.

6.4.1 Prioriteit

Deze aanpak geeft problemen bij de berekening van de prioriteit van de componenten. Voorheen werd namelijk het verschil tegenover de fijnere onderverdeling (zij het de rechthoekige of fotonsegmentatie) genomen als maat voor prioriteit. Nu is er echter nog maar één niveau: de fotonsegmentatie. Er moet een alternatieve heuristiek gevonden worden om de nieuwe componenten te sorteren.

Totale luminantie

Een eerste mogelijkheid voor prioriteitsheuristiek is de totale luminatie van een component. Logisch gezien zijn zo'n componenten het meest zichtbaar, en daar kan dus de grootste fout in de benadering in bestaan. Deze fouten zullen ook het meest opvallen.



Figuur 6.10: De versnelling door het gebruik van fijnkorrelige componenten

Er wordt dus niet meer expliciet rekening gehouden met de mogelijke variantie binnen een gebied. Dit is ook redelijk logisch omdat de segmentatie ervoor moet zorgen dat die variantie binnen een gebied laag gehouden wordt.

Gemiddelde luminantie

Een probleem met de totale luminantie is dat kleine gebieden die minder totale luminantie hebben pas laatst zullen behandeld worden. Met een eenvoudige redenering kan aangetoond worden dat er best gemiddelde luminantie gebruikt wordt.

Stel component $\mathbf{C}$ met oppervlakte $\mathbf{G}$ en luminantie $\mathbf{L}$. Deze wordt opgesplitst in twee even grote delen met evenveel licht:

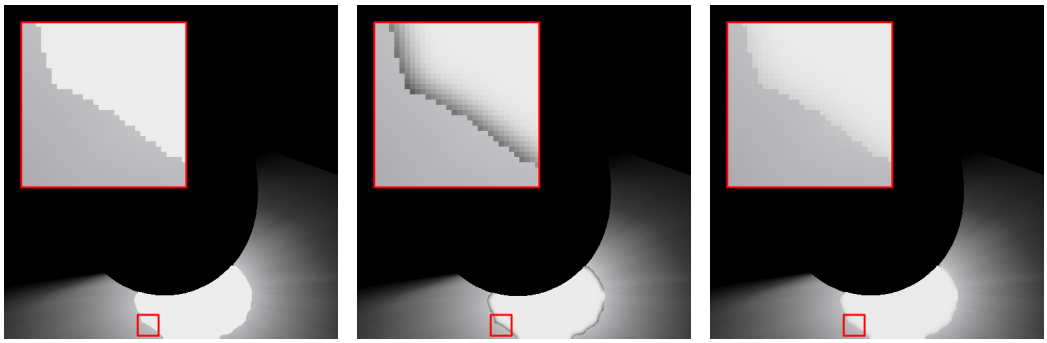
$$\mathbf{C_A}(\mathbf{G_A}, \mathbf{L_A}) \quad \text{en} \quad \mathbf{C_B}(\mathbf{G_B}, \mathbf{L_B})$$

waarbij

$$\mathbf{G_A} = \mathbf{G_B} = \mathbf{G}/2 \quad \text{en} \quad \mathbf{L_A} = \mathbf{L_B} = \mathbf{L}/2$$

Indien de twee nieuwe componenten na elkaar worden gerenderd, is dat net hetzelfde als het renderen van de oorspronkelijke component. Aangezien de nieuwe componenten nagenoeg identiek zijn moet de prioriteit gelijk zijn aan de oorspronkelijke:

$$\mathbf{P} = \mathbf{P_A} = \mathbf{P_B}$$



(a) Onbewerkt, met scherpe rand

(b) Gepartitioneerde vervaging

(c) Gemaskeerde gepartioneerde vervaging

Figuur 6.11: Behandeling van randen tussen gerenderde en benaderde componenten

Ons startpunt was dat de prioriteit afhankelijk is van de luminantie:

$$\frac{\mathbf{P_i}}{\mathbf{P_j}} = \frac{f(\mathbf{L_i})}{f(\mathbf{L_j})} \quad \Rightarrow \frac{\mathbf{P_A}}{\mathbf{P}} = \frac{1}{2} \quad \text{voor} \quad f : x \mapsto K \cdot x$$

Indien het verband lineair is, dan zal de prioriteit inderdaad dalen bij componenten met een kleiner gebied. Een manier om dit op te lossen is de oppervlakte erbij te betrekken om de halvering te neutraliseren:

$$\frac{\mathbf{P_i}}{\mathbf{P_j}} = \frac{f(\mathbf{L_i}/\mathbf{G_i})}{f(\mathbf{L_j}/\mathbf{G_j})} \quad \Rightarrow \frac{\mathbf{P_A}}{\mathbf{P}} = 1 \quad \text{voor} \quad f : x \mapsto K \cdot x$$

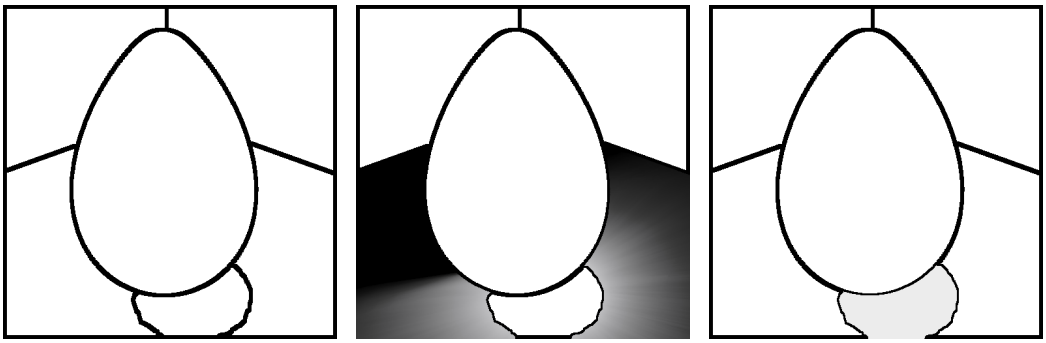
De prioriteit moet dus een monotoon stijgende functie zijn van luminantie op het oppervlak van het gebied.

6.4.2 Nieuwe randen

Net zoals bij de vorige verfijning, ontstaan hier ook nieuwe randen die apart moeten behandeld worden. Het is nu mogelijk dat een geometrisch gebied zowel gerenderde als benaderde componenten bevat, zoals in figuur 6.9(b) of 6.11(a). De randen tussen deze twee types gebieden zullen blijken een bijzondere behandeling nodig te hebben. We analyseren wat de problemen en oplossingen zijn.

Niets veranderen

Met de vorige vervagingsmethode wordt er enkel vervaagd binnen een voorwerp. Het kan nu echter zijn dat een gebied binnen dat voorwerp volledig gerenderd is en een ander



(a) De segmentatie (b) Een gerenderde component (c) Een benaderde component

Figuur 6.12: Segmentatie van het lichtpadtype caustics

gebied nog benaderd wordt. In het benaderde beeld ontbreekt nu informatie waar er gerenderd is geweest. Het resultaat van het vervagen is dat er een zwarte rand ontstaat rond die grens, zoals in figuur 6.11(b).

Fotonsegmentatie als partitie voor de gepartitioneerde vervaging

Dit breidt de partitionering uit naar elke rand van de fotonsegmentatie. De zwarte rand is nu verholpen, maar de vervaging wordt nu enkel nog toegepast in een gebied met uniforme kleuren, wat dus weer 6.11(a) oplevert. Met andere woorden: het is een nul-operatie.

Gemaskeerde gepartitioneerde vervaging

Gerenderde deelgebieden binnen in voorwerp zijn nieuw en leveren problemen op. Tot nog toe werd de vervaging uitgevoerd op de benadering alleen. Om die gerenderde gebieden in rekening te kunnen brengen is het dus nodig om de vervaging in een later stadium toe te passen; in aanwezigheid van de gerenderde gebieden.

We beschouwen eerst het probleem binnen één lichtpadtype. Hierbij zijn de gebieden van componenten niet-overlappend, afgezien van de pixel overlap bij de geometrische opdeling. Figuur 6.12(a) illustreert de opdeling in componenten voor de caustics van de twee lichtbron.

Er zijn twee soorten gebieden;

1. gebieden die gerenderd zijn (figuur 6.12(b))

2. gebieden die deels of volledig gerenderd zijn (figuur 6.12(c))

We kunnen de volgende soorten randen onderscheiden;

1. randen van voorwerpen (afkomstig van de geometrische segmentatie),
2. randen binnen voorwerpen
   (a) tussen benaderd en benaderd
   (b) tussen benaderd en gerenderd
   (c) tussen gerenderd en gerenderd

Randen van type 1 en type 2a worden correct behandeld door de gepartitioneerde vervaging. Nieuw zijn type 2b en 2c.

Tussen twee gerenderde gebieden (2c) is geen vervaging nodig. Het bevindt zich namelijk reeds in de eindtoestand. Speciale aandacht is nodig tussen een benaderd en een gerenderd gebied (2b). Het gerenderd gebied mag niet vervaagd worden omdat het een flou beeld zou opleveren. (Bij een benadering is dit geen probleem omdat het al een uniforme kleur heeft.) Deze zichtbare rand is echter niet gewenst. Het probleem hebben we kunnen halveren door de vervaging te maskeren zodat deze enkel de pixels van de benadering mag herberekenen. Een resultaat is te zien op figuur 6.11(c).

Men ziet dat de rand niet helemaal is verdwenen. Stel dat het gerenderd gebied een uniforme kleur k_r heeft en de benadering een uniforme kleur k_u. Wegens symmetrie in gaussiaanse vervaging zullen pixels op rand aan de kant van de benadering kleur $\frac{k_r+k_u}{2}$ hebben. Pixels op rand aan de kant van het gerenderd gebied hebben nog steeds kleur k_r. We verkrijgen dus een disconuïteit van $|\frac{k_r-k_u}{2}|$, wat twee keer zo klein is dan de initiële discontinuïteit in de rand. De situatie is verbeterd, maar niet helemaal opgelost.

Het aangepast algoritme betekent concreet voor de convolutiematrix:

benaderde pixel Behoud de matrix van de vorige methode

- Start met een gaussische 3×3 vervagingsmatrix.
- Snijd deze matrix volgens de partitie berekend uit de geometrisch segmentatie.

gerenderde pixel Gebruik de triviale convolutiematrix

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Deze methode is nu uitgewerkt per lichtpadtype. Deze aanpak vereist echter ook dat het hele renderproces de lichtpadtypes apart behandelt en blijft behandelen. Zo moet bijvoorbeeld de thresholdmap dan ook per lichtpadtype berekend worden. Dit alles brengt een hogere overhead en compliceert de implementatie in BLEED enorm. Daarom is er geopteerd om het nieuw vervagingsalgoritme aan te passen zodat het kan werken op het beeld met alle bijdrages van de lichtpadtypes samen.

Wanneer het samengesteld beeld wordt beschouwd, kunnen we een paar soort pixels onderscheiden:

1. Pixels waar alle componenten nog benaderd worden.
2. Pixels waar een paar componenten reeds gerenderd zijn.
3. Pixels waar alle componenten reeds gerenderd zijn.

We beslissen de twee laatste soorten gelijk te behandelen. De eerstgerenderde componenten zijn namelijk deze met de grootste intensiteit. We kunnen dus verwachten dat na de de eerstgerenderde component de benadering van de overige componenten redelijk zacht zal verlopen. Door de mogelijke overblijvende variaties te verwaarlozen, beschouwen we die pixel als een pixel waar alle componenten reeds gerenderd zijn.

Het samenvoegen van de lichtpadtypes had als gevolg dat er een nieuwe type pixel (2) ontstond. Deze hebben we nu geklasseerd bij een ander type, zodat het gemaskeerd gepartitioneerd vervagen weer kan toegepast worden.

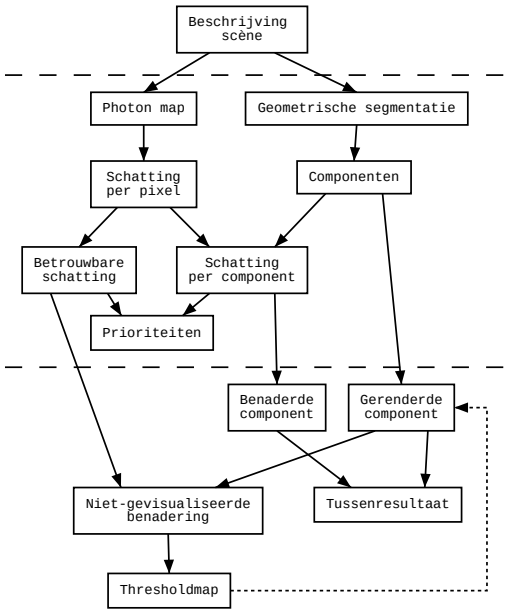
6.5 Vergelijking werking methodes

Om samen te vatten gaan we na hoe de werking van de oorspronkelijke methode [DD05] is gewijzigd. We vergelijken de informatiestromen in de algoritmes in figuur 6.13. De schema's worden telkens verdeeld in drie delen: invoer (beschrijving van de scène), voorbereidende fase of preprocessing en ten laatste het iteratieve gedeelte.

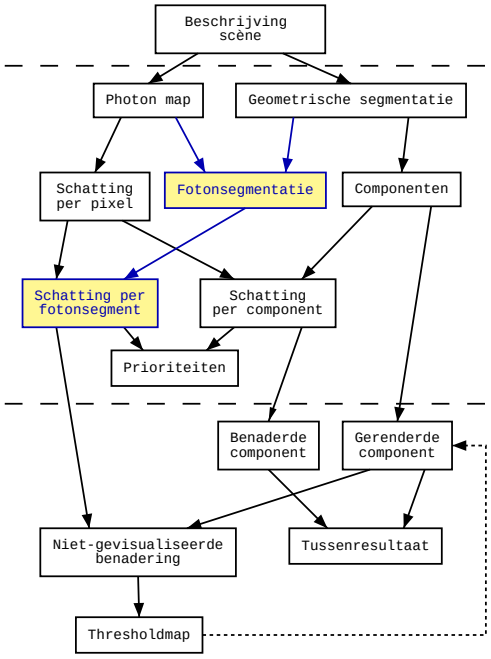
6.5.1 Oorspronkelijke methode

In figuur 6.13(a) zien we de informatiestromen in de oorspronkelijke methode. We overlopen het kort.

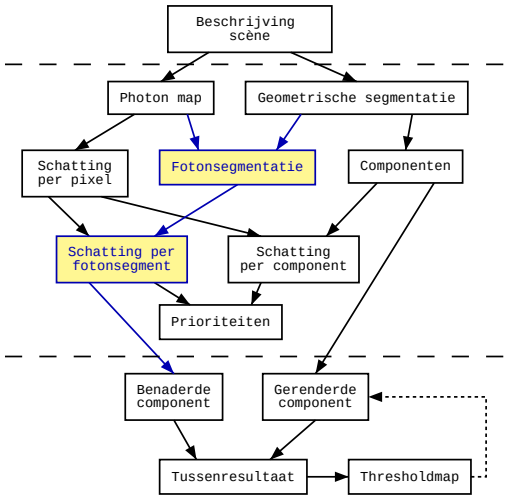
Als invoer hebben we de beschrijving of modellering van de scène. Hiermee wordt een photon map opgesteld waaruit per pixel van het beeld de lichtpadbijdragen worden geschat. Anderzijds gebruiken we die beschrijving ook om het beeld op te splitsen volgens de voorwerpen in de scène die dan mee de componenten bepalen.



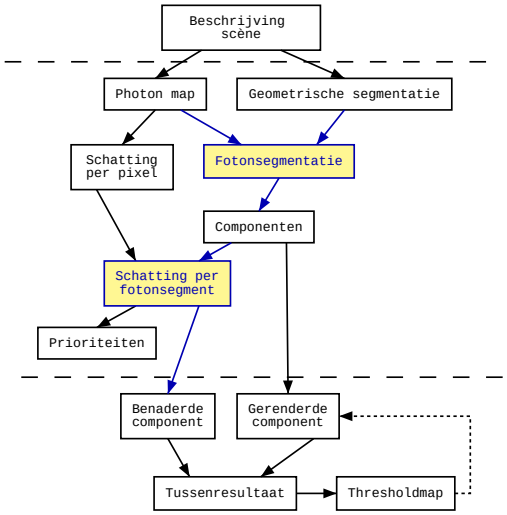
(a) Oorspronkelijke methode



(b) Aangepaste thresholdmap en prioriteiten



(c) Verfijnde benadering



(d) Verfijnde componenten

Figuur 6.13: Informatiestroom bij de verschillende methodes

Er worden twee benaderingen geconstrueerd. De ene volgens de rechthoekige segmentatie om een betrouwbare benadering te bekomen. De andere volgens de componenten om benaderingen te bekomen die gevisualiseerd kunnen worden. Uit het verschil van deze twee stellen we de prioriteiten op om te componenten te renderen.

Door de betrouwbare benadering en de reeds gerenderde componenten te combineren, bekomt men een zo volledig mogelijk beeld waaruit de thresholdmap kan berekend worden. Deze bepaalt hoe lang de behandelde component moet berekend worden. Een combinatie van de benadering per component en reeds gerenderde componenten levert een tussenresultaat op. Deze tussenresultaten worden geïllustreerd in bijlage B.

6.5.2 Aangepaste thresholdmap en prioriteiten

We overlopen de verschillen in figuur 6.13(b). Ten eerste is er een fotonsegmentatie ingevoerd, berekend uit de photon map. Deze wordt bijgestuurd door de geometrische segmentatie zodat belangrijke randen zeker gevonden worden.

Vervolgens wordt de betrouwbare benadering met rechthoekige segmentatie vervangen door een schatting per fotonsegment.

6.5.3 Verfijnde benadering

Bij deze verfijning worden in figuur 6.13(c) de gevisualiseerde benaderingen berekend uit de fotonsegmentatie in plaats van in de geometrisch segmentatie. Bijgevolg zijn de "niet-gevisualiseerd benadering" en de tussenresultaten gelijk. De thresholdmap kan dus berekend worden uit het tussenresultaat.

6.5.4 Verfijnde componenten

In figuur 6.13(d) wordt voor de componenten ook gebruik gemaakt van de fotonsegmentatie. Bijgevolg is er nog slechts één schatting, namelijk per fotonsegment. Zoals eerder aangehaald kunnen we hierdoor voor die prioriteiten niet meer vergelijken tussen twee verschillende schattingen.

Hoofdstuk 7

Resultaten

We hebben een bestaande methode om incrementeel en tijdsgebonden te renderen [DD05] overgenomen en aangepast in hoofdstuk 6 om het proces bij te sturen aan de hand van meer zinvolle concepten, met name fotonsegmentatie, besproken in hoofdstuk 5.

Ten eerste vergelijken we in deel 7.1 hoe de verfijnde methode zich gedraagt tegenover path tracing met photon mapping. We baseren ons op de vergelijking die uitgevoerd is in de oorspronkelijke paper.

Verder gaan we in deel 7.2 en volgende na of de verfijningen effectief de methode verbeteren. Door de verlengde preprocessing is er namelijk meer tijd nodig om de (tussen)resultaten te bekomen.

7.1 Vergelijking met path tracing

In dit deel gaan we na hoe de methode uit [DD05] en onze methodes zich gedragen tegenover path tracing met photon mapping.

Terwijl onze aangepaste methode incrementeel werkt, is dat niet het geval voor path tracing. Men kan path tracing echter op eenvoudige manier wijzigen zodat men wel tussenresultaten bekomt. We voeren als het ware opnieuw componenten in. Er zijn twee triviale opsplitsingen:

per gebied Deze opsplitsing is reeds besproken in deel 3.3. Men rendert pixel per pixel. De tussenresultaten bevatten dus maar een deel van de pixels van het eindresultaat.

per bemonstering Path tracing zal elke pixel meerdere malen bemonsteren, ten einde de ruis wegens Monte Carlo (deel A) te reduceren. Voor n bemonsteringen per pixel, bekomt men hiermee n tussenresultaten. Tussenresultaat i heeft in elke

pixel i bemonsteringen. Zo kunnen figuren 2.5(a), 2.5(b) en 2.5(c) op pagina 16 beschouwd worden als het respectievelijk vierde, zestiende en honderdachtentwintigste tussenresultaat bij path tracing.

De keuze is hier gemaakt om *per bemonstering* op te splitsen. Deze werkwijze biedt namelijk in de tussentijdse beelden een volledig, zij het ruizig, beeld. De opsplitsing in gebieden geeft daarentegen een onvolledig beeld. Er is namelijk geen informatie aanwezig om niet-gerenderde componenten te kunnen vervangen door hun benadering, in tegenstelling tot de aanpak aangebracht in hoofdstuk 3.

De vergelijking tussen path tracing en rendercomponenten werd reeds uitgevoerd in [DD05]. Aangezien het basisprincipe grotendeels hetzelfde is gebleven bij onze verfijningen, kunnen we deze vergelijking overnemen. In de volgende delen zal de paper met onze methodes vergeleken worden.

Initieel heeft path tracing een duidelijk voordeel. Het algoritme mag na het opstellen van de photon map meteen beginnen renderen terwijl onze methode eerst een visualisatie moet maken van de fotonen. Dit is nodig teneinde de segmentatie en daaropvolgende analyse te kunnen uitvoeren. Dit voordeel moet echter gerelativeerd worden. De eerste beelden van path tracing bevatten veel ruis zodanig dat ze niet bruikbaar zijn.

Reeds vanaf het eerste geproduceerde beeld door de incrementele methode is er duidelijk voordeel tegenover path tracing zichtbaar. Hoewel er dan ook al minder ruis zichtbaar is bij path tracing, toont de incrementele methode al een ruisloos beeld waar elk voorwerp reeds een schatting van de radiantie draagt.

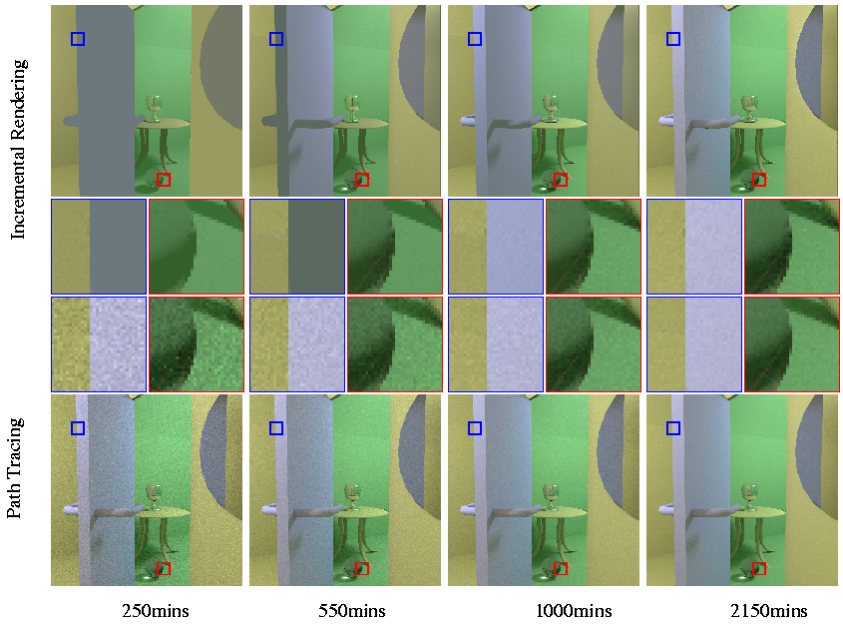
Figuren 7.1 en 7.2 passen deze vergelijking toe voor de basismethode [DD05]. Voor de verdere vergelijking kunnen we figuur 7.2 beschouwen. Bovenaan wordt er incrementeel gesynthetiseerd, terwijl onderaan path tracing wordt toegepast. Na 19 minuten geeft path tracing een volledig maar ruizig beeld. De andere methode heeft enkel nog maar twee caustic-componenten kunnen visualiseren. In de ingezoomde beelden in het midden van de figuur kan men zien hoe bij incrementeel renderen er telkens een schaduw bijkomt, terwijl bij path tracing de ruis vermindert. Figuur 7.1 doet net hetzelfde voor een andere scène waar men tot dezelfde bevindingen kan komen.

7.2 Vergelijking van de verschillende methodes

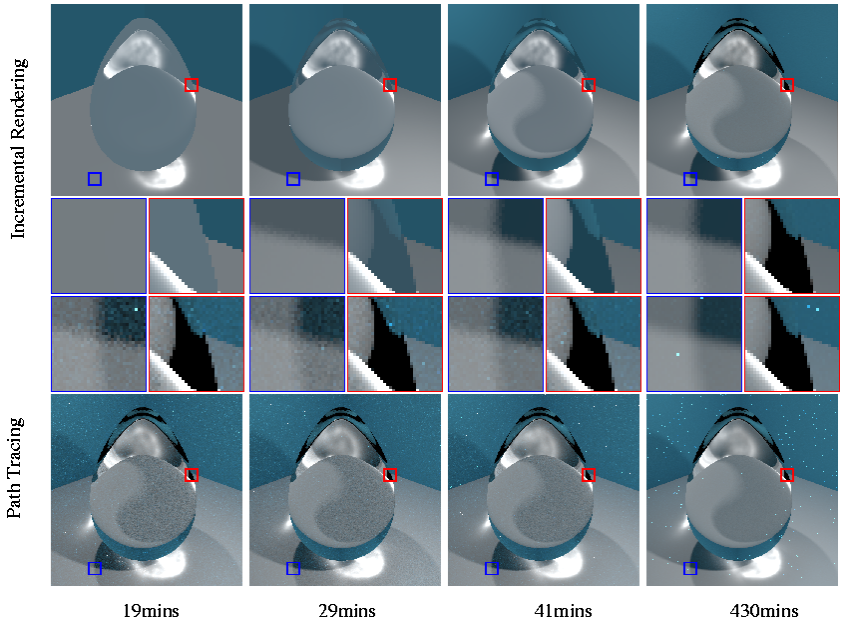
De resultaten van de voorgestelde verfijningen in hoofdstuk 6 kunnen sterk afhangen van de scène die gesynthetiseerd wordt. Daarom wordt er geen algemene vergelijking gegeven, maar wordt scène per scène besproken wat de impact van de methode is.

De oorspronkelijke methode duiden we aan als OM.

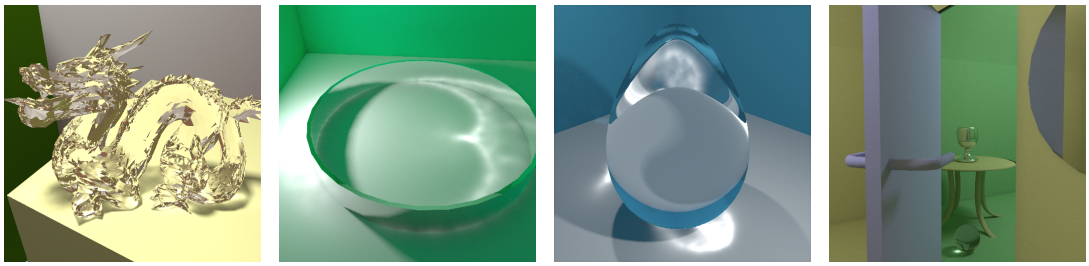
In hoofdstuk 6 hebben we driemaal de fotonsegmentatie ingevoerd.



Figuur 7.1: Vergelijking tussen incrementeel renderen en een normaal path tracing algoritme. Uit [DD05].



Figuur 7.2: Vergelijking tussen incrementeel renderen en een normaal path tracing algoritme. Uit [DD05].



(a) Glazen draak (b) Metalen ring (c) Glazen ei (d) Kamer

Figuur 7.3: Testscènes

- In deel 6.2 hebben we de rechthoekige segmentatie vervangen door fotonsegmentatie. Dit beïnvloedt de berekening van thresholdmap en prioriteiten niet fundamenteel. Resultaten hieromtrent hebben dus weinig zin.
- In deel 6.3 is de benadering verfijnd met behulp van de fotonsegmentatie. Dit is de eerste methode die we zullen nagaan. Afgekort: VB.
- Tenslotte hebben we in deel 6.4 ook de componenten verfijnd. Ook de resultaten van deze methode zullen we bestuderen. Afgekort: VC.

Verder zullen we nog speciaal aandacht besteden aan de adaptieve bemonstering van de photon map om de fotonsegmentatie te versnellen. Methodes die deze versnelling gebruiken zullen we aanduiden met +AB.

Eerst nemen we een scène met een glazen draak (figuur 7.3(a)). Een groot glazen object zorgt voor een grote, dure[1] geometrische component. Gebruik van fotonsegmentatie voor de verfijning van de componenten moet hier dus voordeel uit halen aangezien het rekenwerk in meer gelijke delen verdeeld wordt.

De volgende scène toont een metalen ring (figuur 7.3(b)). Door het invallend licht ontstaan lang uitgerekte reflectieve caustics. Deze effecten zijn de belangrijkste elementen uit de scène en hebben dus het meeste aandacht nodig. Met verfijnde componenten moet het mogelijk zijn deze als prioriteit te stellen.

Vervolgens beschouwen we een scène met een glazen ei en twee lichtbronnen (figuur 7.3(c)). De scène wordt getypeerd door de aanwezigheid van duidelijke effecten. Zo zijn er twee grote schaduwen en is er een afgelijnde brekingscaustic. De verfijnde benadering moet hier voordeel uit kunnen halen door reeds in het begin deze effecten te tonen.

Tenslotte tonen we een zicht door een kamerdeur (figuur 7.3(d)). Geometrische segmentatie van het beeld zal de deur als een geheel beschouwen. Hier kan men verwachten dat

[1] Duur in termen van processortijd

de verfijnde benadering de deur reeds in het begin een herkenbare vorm zal geven.

7.3 Scène met glazen draak

Eerst geven we de bekomen beelden zelf in deel 7.3.1. Vervolgens worden de methodes en parameters toegelicht in deel 7.3.2. Daarna worden zowel de preprocessing stap (deel 7.3.3) als het renderen besproken (deel 7.3.4).

7.3.1 Resultaten

Voor de volledige resultaten van de besproken incrementele methodes refereren we naar bijlage B. In figuren B.1 tot B.5 (pagina 122 tot 126) vindt men de tussenresultaten voor een scène met een glazen draak bekomen met verschillende methodes.

7.3.2 Parameters

De figuren zijn bekomen met de onderstaande parameters. Parameters die niet veranderen worden niet telkens herhaald.

1. Figuur B.1 gebruikt de methode uit [DD05].
   - Thresholdmap: benadering van het beeld aan de hand van rechthoekige segmentatie
   - Prioriteit: verschil tussen geometrische en rechthoekige segmentatie
   - Benaderingen: geometrisch segmentatie
   - Componenten: geometrisch segmentatie
   - Resolutie: 512×512
   - Strategie: incrementeel
2. Figuur B.2 verfijnt de benadering.
   - Thresholdmap: fotonsegmentatie
   - Prioriteit: verschil tussen fotonsegmentatie en geometrische segmentatie
   - Benaderingen: fotonsegmentatie, vervaging met straal 20 pixels
   - Componenten: geometrische segmentatie
   - Fotonsegmentatie:
     - afronden met straal 10 pixels
3. Figuur B.3 versnelt de radiantieschatting.

methode	rechthoekige segm. (s)	radiantieschatting (s)	mean shift (s)	totaal
OM	45	180	0	280
VB	0	960	90	1140
VB+AB	0	660	90	830
VC+AB	0	660	90	830

Tabel 7.1: Preprocessing tijden voor de verschillende methodes. Enkel de meest relevante en verschillende onderdelen van de preprocessing worden weergegeven.

- Fotonsegmentatie:
  - afronden met straal 10 pixels
  - adaptieve bemonstering voor de radiantieschatting met 75% overlapping van de fotonbollen

4. Figuur B.4 en B.5 verfijnen de componenten.
   - Thresholdmap: fotonsegmentatie
   - Prioriteit: gemiddelde luminantie
   - Benaderingen: fotonsegmentatie, vervaging met straal 20 pixels
   - Componenten: fotonsegmentatie

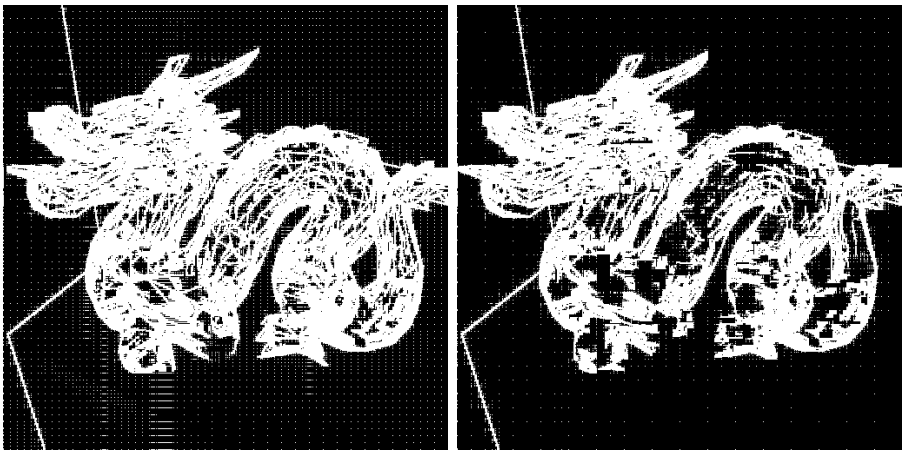
7.3.3 Preprocessing

Oorspronkelijke methode

Figuur B.1 gebruikt de rechthoekige segmentatie voor zowel de thresholdmap als de prioriteiten. De radiantieschatting om deze segmentatie uit te voeren duurde 180 seconden. De daaropvolgende berekening van de rechthoekige segmentatie zelf duurde 45 seconden. Wanneer men het opbouwen van de photon map (dat gemeenschappelijk is voor alle methodes) erbij rekent, komt men tot een preprocessing van 280 seconden.

Verfijnde benadering

Voor figuur B.2 werd de rechthoekige segmentatie verwijderd en vervangen door de fotonsegmentatie. Dat levert dus ten eerste een winst op van 45 seconden. De fotonsegmentatie vereist echter een meer uitgebreide radiantieschatting, zodat de benodigde tijd hiervoor verhoogt van 180 seconden naar 960 seconden. In 960 seconden bekomen we voor elke pixel een schatting van de lichtpadbijdragen. De segmentatie zelf duurt 90 seconden. De totale tijd stijgt dus tot 1140 seconden.



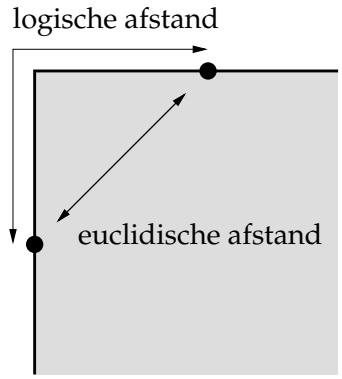
(a) Overlap van 85% (b) Overlap van 60%

Figuur 7.4: De bemonsteringen voor de versnelde radiantieschatting voor de draak-scène

Adaptieve bemonstering

Voor de volgende figuur B.3 werd getracht deze sterk gestegen preprocessingtijd te verlagen. We hebben gezien dat een adaptieve bemonstering van de photon map een significante snelheidswinst kan opleveren. Figuur 7.4 toont de bemonsteringen voor een overlappingsgraad van 85% en 60% (zie ook deel 5.2). Het spelen met deze parameter heeft vooral invloed op de wanden. De draak reflecteert en refracteert namelijk de afgeschoten stralen in die mate dat de afstand van de uiteindelijk intersecties voor de draak-pixels zeer sterk uit elkaar liggen. Met 75% overlapping van de bollen hebben we de radiantie schatting van 960 seconden naar 660 seconden kunnen reduceren. Dit levert een totale preprocessing van 830 seconden op, oftewel algemeen 32% snelheidswinst in de preprocessing. Het is echter nog steeds 170% trager dan de preprocessortijd voor de methode [DD05]. Dit is redelijk normaal aangezien er een betrouwbaardere radiantieschatting nodig is om de fotonsegmentatie stabiel te doen functioneren.

Het toepassen van deze versnelling komt hier echter niet zonder afstraffing. De lichtpadbijdragen in het beeld worden lokaal lineair benaderd over een rechthoek. Indien het overlappingspercentage laag wordt gekozen, worden deze rechthoeken te groot. Hierdoor verdwijnen sommige eigenschappen van het beeld, zoals de ribbe van de kubus waarop de draak steunt, zie figuur 7.6. De reden dat deze ribbe verslechtert is de heuristiek van de adaptieve bemonstering. Deze gebruikt de euclidische afstand (d) tussen de intersectiepunten. De "logische" of intuïtieve afstand is de afstand over het oppervlak, zoals in figuur 7.5. We voelen aan dat deze afstand zou moeten gebruikt worden en niet de euclidische afstand. Voor een hoek van 90° kan de euclidische afstand tot een fractie $\frac{\sqrt{2}}{2}$ van de logische afstand gereduceerd zijn. Daarom zal bij zulk een hoek de schatting zo'n



Figuur 7.5: Logische en euclidische afstand bij versnelde schatting

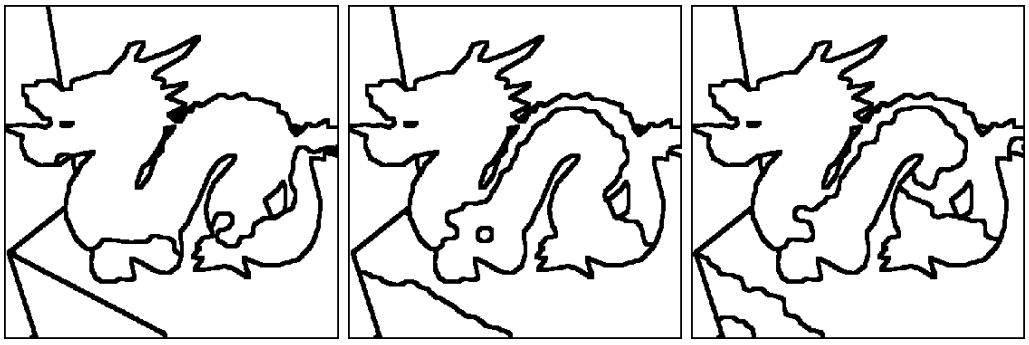
(a) Volledige bemonstering (b) 85% overlap (c) 60% overlap

Figuur 7.6: Invloed van de bemonsteringsgraad op de initiële schatting van lichtinformatie voor fotonsegmentatie

30% $\left(\text{ofwel } 1 - \frac{\sqrt{2}}{2}\right)$ verslechteren.

De resulterende segmentatie vindt men terug in figuur 7.7. Een dalende overlappingsgraad blijft de draak goed schatten. De ribbe van de kubus wordt echter hoekig.

De vraag kan gesteld worden waarom zo'n duidelijke grens in een voorwerp niet gevonden kan worden door geometrische segmentatie. De kubus is gemodelleerd als één voorwerp en wordt dus ook als dusdanig herkend. We hadden kunnen overwegen om discontinuïteiten te zoeken in voorwerpen om bijkomende betrouwbare segmentatie te bekomen, maar het probleem is niet zo eenvoudig. Door discontinuïteiten te herkennen als randen in de segmentatie zou de draak opgedeeld worden in vele kleine componenten, wat excessief is. Om goede resultaten te bekomen gaat men best de scène gericht modelleren. In dit geval betekent dit dat de wanden van de kubus best als aparte voorwerpen worden ingegeven.



(a) Volledige bemonstering (b) 85% overlap (c) 60% overlap

Figuur 7.7: Invloed van de bemonsteringsgraad op de segmentatie

Verfijnde componenten

Figuren B.4 en B.5 werden bekomen door eveneens de componenten op te splitsen. De preprocessing blijft even lang duren aangezien evenveel informatie nodig is.

De samenvattende vergelijking van de preprocessingtijden vindt men in tabel 7.1

7.3.4 Rendering

Oorspronkelijke methode

Vervolgens kunnen we de timing van de visualisaties vergelijken. Bij figuur B.1 worden eerst een paar onbelangrijke componenten berekend, zoals men kan zien op figuur B.1(b). In stap 3 wordt het directe licht op het blok berekend. Aangezien de scène een oppervlaktelichtbron bevat, ontstaan er zachte schaduwen[2]. Deze veroorzaken ruis zodat de benodigde rendertijd hiervoor redelijk groot is.

Deze rekentijd is echter klein ten opzichte van de tijd nodig voor stap 6 en vooral stap 7. In deze stappen worden de belangrijkste[3] componenten berekend, namelijk het direct licht gezien door de draak. Gezien de complexiteit van de interacties van de lichtbijdragen met het reflecterend en brekend glas, zijn dit de duurste componenten. Merk op dat bij de berekening van de prioriteit van de componenten geen rekening werd gehouden met

[2]Zachte schaduwen gaan gradueel over van licht naar donker. Puntlichtbronnen veroorzaken harde schaduwen waarbij de overgang tussen licht en donker zeer abrupt is.

[3]Belangrijkst voor de menselijke toeschouwer. Het voorwerp staat centraal en trekt de aandacht. Om zo'n klasse van prioriteiten te berekenen zou totaal verschillende heuristieken moeten ontwikkeld worden.

de kost van de componenten. Mocht de rekentijd voorspeld kunnen worden, zou men daarmee de heuristiek van de prioriteiten kunnen verfijnen.

Verfijnde benadering

Het eerste wat opvalt in figuren B.2 en B.3 is dat de initïele schatting van de draak beter is. Het resultaat bekomen na 830 seconden (figuur B.2(a) stap 0) is beter dan het resultaat dat men bekwam bij de vorige methode. Na stap 4 wordt de zware component aangevat waardoor er een hele tijd geen nieuw tussenresultaat zal bekomen worden. Dit wekt de indruk dat het algoritme minder goed presteert. Later blijkt echter dat dit niet het geval is.

De conclusie van deze eerste resultaten is dat dure componenten een grote invloed hebben op de tussenresultaten aangezien er een tijdlang niets meer wordt geproduceerd. Men kan dus verwachten dat het verfijnen van de componenten de stabiliteit van het proces zal verbeteren.

Verfijnde componenten

Wanneer men figuren B.4 en B.5 beschouwt, valt het meteen op dat er veel meer componenten zijn. Segmentatie van componenten per geometrie levert 24 componenten (waarvan 20 getoond) op, terwijl segmentatie volgens de fotonsegmentatie 57 componenten (waarvan 40 getoond) oplevert. We zien inderdaad dat er per stap een kleiner deel van het beeld berekend wordt. Meer tussenresultaten betekent minder periodes waarin men lang moet wachten op beeld. Zo kan het renderproces meer onderbroken worden. Vergelijk met path tracing waar er in principe moet gewacht worden tot volledige convergentie eer een bruikbaar beeld bekomen wordt.

Terwijl de oorspronkelijke methode in B.1(a) tussen stap 6 en 7 lang moet wachten, kan de methode met verfijnde componenten in B.5(a) de beelden 24 en 25 genereren, dewelke verder gesynthetiseerd zijn dan beeld 6 van B.1(a).

7.4 Scène met metalen ring

In tegenstelling tot de scène met de glazen draak, zijn voorwerpen hier wel opgesplitst in al hun vlakken. Zo bestaat de ring uit drie geometrieën: buitenkant, binnenkant en bovenkant. Dit zal de geometrische segmentatie en bijgevolg ook de fotonsegmentatie een sterke duw in de rug geven om tot een zinvolle opsplitsing te komen.

methode	rechthoekige segm. (s)	radiantieschatting (s)	mean shift (s)	totaal
OM	70	130	0	240
VB	0	1240	90	1370
VB+AB	0	250	90	370
VC+AB	0	250	90	370

Tabel 7.2: Preprocessing tijden voor de verschillende methodes. Enkel de meest relevante en verschillende onderdelen van de preprocessing worden weergegeven.

7.4.1 Resultaten

De volledige resultaten zijn terug te vinden in bijlage B: figuur B.6 tot B.8 (bladzijde 128 tot 130).

De gebruikte methodes zijn analoog aan het vorig voorbeeld, maar minder uitgebreid. Dezelfde parameters werden gebruikt.

Figuur B.6 Methode uit [DD05].

Figuur B.7 Verfijnde benadering (met adaptieve bemonstering).

Figuur B.8 Verfijnde componenten (met adaptieve bemonstering).

7.4.2 Preprocessing

Tabel 7.2 toont de preprocessingtijden voor de verschillende gevallen. Er is ook nagegaan wat de timing zou zijn moest er geen adaptieve bemonstering gebruikt worden ter versnelling van de radiantieschatting. Hierbij merkt men een enorme impact op de tijden. De tijd nodig voor de radiantieschatting daalt van 1240 seconden tot 250 seconden, een daling van 79%. Op de totale preprocessing zorgt het voor een daling van 72%. In tegenstelling tot de scène met de draak, verslechtert de verminderde bemonstering van de photon map de fotonsegmentatie niet omdat de te vinden grenzen in deze scène niet duidelijk afgelijnd zijn. De caustics vertonen een gradueel verloop.

7.4.3 Rendering

Oorspronkelijke methode

In figuur B.6 valt het op dat er twee zeer belangrijke componenten zijn: ten eerste de directe belichting op het ondervlak, berekend in stap 9, en ten tweede de caustics eveneens op het ondervlak, berekend in stap 10. Deze tussenresultaten werden berekend na 1806 respectievelijk 2043 seconden.

Verfijnde benadering

Figuur B.7 toont de verfijnde benadering. Dankzij de adaptieve bemonstering is tegenover de oorspronkelijke methode een achterstand van slechts een 140-tal seconden.

Het voordeel van de verfijnde benadering is dat lichteffecten binnen een voorwerp ook reeds in het begin kunnen gevisualiseerd worden. Dit voordeel is in dit voorbeeld twijfelachtig, aangezien caustics moeilijk kunnen voorgesteld worden door uniforme gebieden. Dit valt te zien op figuren B.7(a) 0 tot B.7(a) 4.

Het verschil in volgorde van componenten met de oorspronkelijke methode (zie figuur B.6(b) en B.7(b)) is te verklaren door de vervanging van de rechthoekige segmentatie door fotonsegmentatie voor de berekening van de prioriteiten van de componenten. In figuur B.7(a) 0 ziet men dat het verschil in radiantie tussen caustic en niet-caustic gebied veel groter is dan het verschil tussen schaduw en directe belichting. De radiantie in het schaduwgebied lijkt overschat. Dit zorgt voor een groter verschil tegenover de geschatte radiantie per voorwerp, zodat de caustic-component voorrang (5) krijgt op de directe-belichtingscomponent (13).

Zo krijgt men een beeld met caustics reeds na 1046 seconden berekend, terwijl een beeld met ook directe belichting op het ondervlak wordt uitgesteld tot 3767 seconden.

Verfijnde componenten

We kunnen nagaan hoe de belangrijkste componenten van hierboven worden opgesplitst met deze methode. De directe belichting op het ondervlak is niet gesegmenteerd, terwijl de caustics op de tafel wel werden afgesplitst, zoals men kan concluderen uit de vergelijking van figuren B.6(b) 10 en B.8(b) 5. De nieuwe component die de caustics bevat wordt reeds in stap 5 (figuur B.8(a) 5), terwijl het complementaire deel in de eerste 20 stappen niet aan de beurt komt. Dit illustreert het mechanisme besproken in figuur 6.10 over het versnellen dankzij verfijnde componenten. Na 1069 seconden (stap 6) zijn zowel de caustics als de schaduwen gevisualiseerd.

Men merkt echter ook artefacten op het beeld. Dit fenomeen werd reeds besproken in deel 6.4.2. Het is in principe te verhelpen zonder beduidend meer rekentijd door de gemaskeerde gepartitioneerde vervaging verder te verfijnen.

Ter volledigheid kan men een anomalie merken in stap 10 en stap 14. Deze vergen abnormaal veel rekentijd in vergelijking met de oorspronkelijke methode. Stap 10 vergt hier 6314 seconden, terwijl dat dit in figuur B.6 werd berekend in 92 seconden (stap 16). Dit is waarschijnlijk te wijten aan een fout in de berekening van de toegelaten fout volgens de perceptuele metriek.

Tabel 7.3 vat de tijden samen die nodig waren om de schaduw en de caustic in het beeld

methode	schaduw (s)	caustic (s)	beide (s)
OM	1606	2043	2043
VB+AB	3767	1046	3767
VC+AB	1069	722	1069

Tabel 7.3: Benodigde tijd om een bepaald effect te bekomen in de scène met metalen ring.

methode	rechthoekige segm. (s)	radiantieschatting (s)	mean shift (s)	totaal
OM	115	155	0	305
VB	0	1240	110	1415
VB+AB	0	380	110	560
VC+AM	0	380	110	560

Tabel 7.4: Preprocessing tijden voor de verschillende methodes. Enkel de meest relevante en verschillende onderdelen van de preprocessing worden weergegeven.

te bekomen. We zien dat de methode met verfijnde componenten het best scoort.

7.5 Scène met glazen ei

7.5.1 Resultaten

De volledige resultaten zijn terug te vinden in bijlage B: figuur B.9 tot B.11 (bladzijde 132 tot 134).

7.5.2 Preprocessing

Deze scène is veel regelmatiger dan de scène van de glazen draak. Stralen geschoten door het ei worden grotendeels gegroepeerd, als ware het een bolle lens. Bij de draak werden de stralen veel meer verspreid zodat de adaptieve bemonstering voor de lichtinformatieschatting weinig invloed had. Hier kan men verwachten dat deze versnellingsmethode beter zal werken.

Tabel 7.4 vat de preprocessing timing samen. Met behulp van adaptieve bemonstering halveert de preprocessing tijd. Uiteindelijk is de benodigde tijd toch bijna verdubbeld tegenover de oorspronkelijke methode.

Het is interessant om op te merken dat de segmentatie op zich sneller is via mean shift (gebruik maken van lichtinformatie) dan de rechthoekige adaptieve opdeling van het beeld, terwijl de berekeningen voor het laatste meer voor de hand liggen.

7.5.3 Rendering

Oorspronkelijke methode

In figuur B.9 kan men wederom de methode [DD05] zien. Men merkt opnieuw een anomalie op in stap 14 waar er een enorme rekentijd is. Het probleem is gerelateerd met de wijze van berekening van het aantal pixels dat nog moet convergeren. De gebruikte methode blijkt in sommige gevallen instabiel.

Verfijnde benadering

In deze scène zijn duidelijke lichteffecten aanwezig. Zo zijn er twee duidelijk afgelijnde schaduwen en paar caustics. Na 560 seconden (figuur B.10(a) 0) bekomt men reeds een schatting die rekening houdt met de twee schaduwen en de belangrijkste caustics. Op dat moment heeft men in de oorspronkelijke methode nog altijd het eerste beeld (dat reeds beschikbaar was na 305 seconden). Daarop zijn deze effecten niet zichtbaar. Na verloop van tijd wordt de methode wel ingehaald aangezien de benaderingen in belang dalen en er tenslotte een langere preprocessing was bij de verfijnde benadering.

Verfijnde componenten

Het voordeel van fijnkorrelige componenten kan men zien aan de hand van de eerste stap van deze en de vorige methode. Zowel in figuur B.10(a) 1 als in B.11(a) 1 wordt de grote caustic op het ondervlak gerenderd. Een vergelijking tussen B.10(b) 1 en B.11(b) 1 toont dat bij de verfijnde benadering enkel de kern van de caustic wordt gerenderd. Dat is een goede zaak aangezien dit deel het belangrijkst is. We bekomen een gelijkaardig beeld met beide methodes in deze stap, terwijl de rekentijd nu met 241 seconden verminderd is.

Opnieuw ziet men ook dat er meer tussenresultaten bekomen worden. In 2500 seconden zijn er reeds 20 beelden in plaats van 12.

7.6 Scène met kamers

De volledige resultaten zijn terug te vinden in bijlage B: figuur B.12 tot B.15 (bladzijde 136 tot 139).

In dit laatste voorbeeld zijn slechts twee methodes toegepast: [DD05] en de verfijnde benaderingen.

methode	rechthoekige segm. (s)	radiantieschatting (s)	mean shift (s)	totaal
OM	200	180	0	520
VB	0	1000	150	1310
VB+AB	0	130	150	440

Tabel 7.5: Preprocessing tijden voor de verschillende methodes. Enkel de meest relevante en verschillende onderdelen van de preprocessing worden weergegeven.

(a) Volledige bemonstering (b) Adaptieve bemonstering (c) Bemonsteringen

Figuur 7.8: Invloed van adaptieve bemonstering bij de scène met kamers

7.6.1 Preprocessing

Deze scène blijkt zeer geschikt voor de adaptieve bemonstering van lichtpadbijdrages. Men verkrijgt zelfs een snellere schattingsfase dan in de oospronkelijke paper [DD05].

Tabel 7.5 vat de preprocessing timing samen.

In figuur 7.8 ziet men dat het effect van adaptieve bemonstering beperkt blijft tot een vervaging van een ribbe van deur. Uiteindelijk bekomt men dus een gelijkaardig beeld in een zevende van de tijd omdat er veel minder bemonsterd wordt.

7.6.2 Rendering

Een belangrijk deel van de scène is de deur. Deze is gemodelleerd als zijnde één voorwerp. Logisch gezien is deze in het eerste beeld van figuur B.12 onherkenbaar. In figuur B.14 valt de deur reeds te raden. Hetzelfde geldt voor de reflectie van de deur in de spiegel rechtsboven.

Verder ziet men een verbetering in de volgorde van de componenten betreffende de

deur. De oorspronkelijke methode plaats ze in stap 24 en 28, terwijl met de verfijnde benadering dat reeds in stap 10 en 13 gebeurt. Dit verschil valt te verklaren uit het feit dat de berekening van de prioriteit nu met fotonsegmentatie gebeurde in plaats van rechthoekige segmentatie.

Hoofdstuk 8

Besluit

8.1 Overzicht eigen werk en bekomen resultaten

Er werd vertrokken van een incrementele methode [DD05] om een beeld adaptief te berekenen. Deze methode splitst eerst het rekenwerk op in componenten. Initieel wordt voor elke component een benadering gemaakt. Telkens een component gerenderd is, vervangt men de benadering door het gerenderd beeld. Zo verkrijgt men stelselmatig een beter beeld. De tussenresultaten zijn bruikbaar aangezien ze telkens volledig zijn (dankzij de benaderingen) en bovendien geen ruis bevatten.

Het verdelen in componenten is in [DD05] zeer ruw. Elk gemodelleerd voorwerp vormt een aparte component. Voorwerpen kunnen meerdere lichteffecten zoals schaduw en caustics bevatten. Door de benadering per voorwerp op te stellen, worden deze genegeerd.

Er wordt ook een tweede segmentatiemethode aangeboden. Het beeld wordt adaptief rechthoekig opgedeeld op basis van een intiële radiantieschatting, zodat een betrouwbare schatting wordt bekomen. Deze segmentatie levert vele kleine rechthoeken op. Dit kan niet gebruikt worden voor directe visualisatie (voor de benadering) aangezien dit veel visuele artefacten zou opleveren.

Om het beeld beter te kunnen opdelen, werd een nieuwe segmentatie, de fotonsegmentatie, ontwikkeld in hoofdstuk 5. Deze voert net zoals de vorige methode een segmentatie uit aan de hand van een initiële radiantieschatting, maar de vorm van de gebieden is hier volkomen vrij. Een betere segmentatie wordt hierdoor bekomen.

De basis van de fotonsegmentatie is het mean shift algoritme, een niet-parametrische clusteringstechniek die gebruikt wordt om beelden te segmenteren. De beelden worden aan de hand van een initiële radiantieschatting bekomen. We baseerden ons op een bestaande mean shift implementatie die we aangepast om te kunnen segmenteren volgens lichtinformatie. Een aantal verfijningen waren echter noodzakelijk om een robuuster al-

goritme te verkrijgen. Weinig, grote gebieden zijn te verkiezen boven vele kleine gebieden.

We hebben een aantal methodes voorgesteld om de bandbreedteparameters te schatten zodat er per scéne steeds een goede bandbreedte genomen wordt. We hebben ervoor gezorgt dat de segmentatie van grote, donkere gebieden minder moeizaam verloopt. Om vlekken en minder belangrijke gebieden te verwijderen werd een adaptieve minimale regiogrootte ingevoerd. Kleine gebieden met een laag relatief belang worden samengevoegd met grote gebieden. Een verdere verfijning wordt verkregen door scherpe randen van gebieden af te ronden.

Om de fotonsegmentatie te kunnen uitvoeren is een betere initiële schatting van de radiantie in het beeld nodig. Hierdoor wordt de voorbereidende stap in de voorgestelde incrementele methode veel trager. We hebben een adaptieve bemonstering ontwikkeld die we gebruiken om de schatting te versnellen. In sommige scènes verliep de voorbereidende fase hierdoor zelfs sneller dan in de oorspronkelijke methode.

Deze segmentatie hebben we in [DD05] toegepast op een paar verschillende niveaus. Ten eerste ter verfijning van de benadering. De benadering werd opgesteld per nieuw gebied. Aangezien de fotonsegmentatie visuele effecten zoals schaduwen en caustics kan detecteren en afzonderen, hebben we de eerste tussenresultaten van de methode kunnen verbeteren, zoals we kunnen zien in de vergelijking tussen de eerste beelden van figuren B.9(a) en B.10(a). Als alternatief voor de uniforme benadering hebben we een fotonbenadering voorgesteld. Een verbetering is vooral merkbaar voor lichtpadtypes waarvoor de initiële radiantieschatting al redelijk goed is (zie figuur 6.3.3. Helaas is deze aanpak niet bruikbaar door het hoog geheugengebruik.

Ten tweede werd de fotonsegmentatie gebruikt om de componenten te verfijnen. Sommige delen van voorwerpen kunnen namelijk belangrijker zijn dan andere. Kleinere componenten betekent ook dat we minder lang moeten wachten op tussenresultaten. Zo heeft men in de oorspronkelijke methode meer dan 2000 seconden nodig om een redelijk beeld te bekomen van een scène met metalen ring (figuur B.6(a) 10). Door componenten te kunnen definiëren over belangrijke gebieden krijgen we sneller een aanvaardbaar[1] beeld, namelijk 1050 seconden (figuur B.8(a) 6).

8.2 Verfijningen en uitbreidingen

8.2.1 Verbetering van gemaskeerde gepartioneerde vervaging

In deel 6.4.2 hebben we het vervagingsalgoritme verfijnd zodat we het konden toepassen in ons raamwerk. We hebben echter ook opgemerkt dat de vervaging de randen tussen benaderde gebieden en berekende gebieden enkel halveert, niet verwijdert. Dit levert

[1] We merken op dat er wel artefacten aanwezig zijn in het beeld. Deze kunnen in principe zonder beduidend extra rekenwerk verbeterd worden om die zichtbare randen weg te werken.

artefacten op, zoals in figuur B.8(a) 6.

We kunnen twee denkpistes aangeven. Ofwel wijzigen we de benadering zodat deze volledig naar het gerenderd deel convergeert, ofwel wijzigen we het gerenderd deel zodat deze ook evolueert naar de gemiddelde radiantie op de rand.

8.2.2 Meer randen afleiden uit de geometrische informatie

Sommige voorwerpen vertonen scherpe randen. Het zou mogelijk moeten zijn om deze randen te detecteren om zo de fotonsegmentatie bij te sturen. Er moet wel opgepast worden met voorwerpen die uit vele kleine facetten bestaan. Deze mogen niet aanleiding geven tot een explosie van het aantal gebieden.

Een alternatief is vereisen dat bij de modellering van de scène manueel gekozen wordt welke voorwerpen beduidende randen hebben.

8.2.3 Voorspellen van de rendertijd

In figuur 6.10 hebben we grafisch getoond hoe kleinere componenten tijdswinst kunnen opleveren. We gingen daar echter van uit dat elke component evenveel tijd nodig heeft. Een component met veel bijdrage moet dan hoe dan ook voor een component met minder bijdrage berekend worden. Dit hebben we dan bijgestuurd door ook rekening te houden met het oppervlak van het gebied van de component, zodat intensiteit de doorslaggevende factor werd.

In feite gebruikten we deze oppervlakte als schatting van de kost van het berekenen. De bedoeling is in beperkte tijd zo veel mogelijk bijdrage tot het beeld te berekenen. Met andere woorden worden de componenten best geordend volgens

$$\text{prioriteit} = \frac{\text{bijdrage}}{\text{rekentijd}}$$

De bijdrage hebben we met de photon map kunnen schatten. Om de rekentijd te schatten moeten er nieuwe methodes ontwikkeld worden.

8.2.4 Verbeterde fotonbenadering

De fotonbenadering in zijn huidige vorm is niet bruikbaar. Indien het geheugengebruik teruggeschroeft kan worden, kan er een alternatief gevonden worden voor de uniforme benadering per gebied.

Een mogelijk betere oplossing wordt bekomen door gebruik te maken van informatie uit de adaptieve bemonstering ter versnelling van de radiantieschattingstap in de fotonsegmentatie. Deze berekent de benaderingen op een adaptieve manier en interpoleert tussen

de berekende waarden. Uitgaande van informatie over de interpolatie, kan een schatting bekomen worden van de vervagingsstraal. Een groot voordeel van deze methode is dat er zeer weinig extra geheugen nodig is.

8.2.5 Fout in perceptuele metriek

Af en toe heeft een component abnormaal veel tijd nodig om gerenderd te worden. Dit wordt waarschijnlijk veroorzaakt door verkeerd om te gaan met de accumulerende fout in de perceptuele metriek. We zouden kunnen nagaan wat er misloopt om dit probleem te verhelpen.

8.2.6 Compensatie hoeken bij adaptieve bemonstering

In deel 7.3.3 hebben we ondervonden dat de adaptieve bemonstering van de photon map moeilijkheden heeft met hoeken. Voor een bepaalde hoek hebben we berekend hoeveel percent de overlapping werd overschat. Dit moet eenvoudig op te compenseren zijn, aangezien de intersecties gekend zijn. Intersecties hebben normalen, waarmee de hoek tussen de twee vlakken kunnen berekend worden.

Bijlage A

Monte Carlo integratie

Een veel gebruikte numerieke integratietechniek is Monte Carlo integratie. Deze methode is gebaseerd op kansrekenen. In deze bijlage bespreken we de belangrijkste concepten van deze methode.

De basisbegrippen uit de kansrekening waarop Monte Carlo integratie steunt, worden in deel A.1 besproken. In deel A.2 wordt het principe van Monte Carlo integratie uitgelegd.

Verdere informatie over de techniek is te vinden in [KW86].

A.1 Kansrekenen

Er wordt een overzicht gegeven van de kanstheorie die gebruikt wordt bij Monte Carlo integratie.

Een toevalsvariabele x kan een aantal waarden aannemen. Indien dit aantal eindig is, wordt x een discrete toevalsvariabele genoemd. Anders is x een continue toevalsvariabele.

A.1.1 Discrete variabelen

De gebeurtenis dat x een bepaalde waarde x_i aanneemt, heeft een bepaalde kans p_i. Deze kans ligt tussen nul en é'en. Indien p_i gelijk is aan nul, treedt de gebeurtenis nooit op. De gebeurtenissen die horen bij kansen die gelijk zijn aan é'en, treden altijd op.

De kans dat juist één van twee gebeurtenissen optreedt is steeds kleiner of gelijk aan de som van afzonderlijke kansen. Indien de éne gebeurtenis niet kan optreden wanneer de andere is opgetreden en omgekeerd, dan zijn de twee gebeurtenissen mutueel exclusief.

Er geldt dan dat

$$P(\text{gebeurtenis}_1 \text{ of gebeurtenis}_2) = P(\text{gebeurtenis}_1) + P(\text{gebeurtenis}_2)$$

De som van alle mutueel exclusieve gebeurtenissen is steeds gelijk aan één.

De verwachte of gemiddelde waarde is gelijk aan

$$E[x] = \sum_{i}^{n} x_i p_i$$

p_i is de kans dat de gebeurtenis $x = x_i$ optreedt. Variantie (σ^2) is een maat om de afwijking van de uitkomst is ten opzichte van het gemiddelde te meten:

$$\sigma^2[x] = E[(x - E(x))^2]$$

Een andere maat om afwijkingen ten opzichte van het gemiddelde te meten, is de standaard deviatie σ. Deze is gedefinieerd als de vierkantswortel van de variantie.

Continue variabelen

De kansdichtheidsfunctie $p(x)$ duidt aan wat de kans is dat de waarde van een toevalsvariabele ligt tussen $[x, x + dx]$. Deze heeft als eigenschappen:

$$p(x) \geq 0$$

$$\int_{-\infty}^{+\infty} p(x)dx = 1$$

De verwachte waarde en de variantie zijn gelijk aan

$$E[x] = \int xp(x)dx$$

$$\sigma^2 = \int (x - E[x])^2 \, p(x)dx$$

De cumulatieve kansverdelingsfunctie $P(y)$ duidt aan wat de kans dat een toevalsvariabele een waarde heeft die kleiner of gelijk is dan y:

$$P(y) = \int_{-\infty}^{y} p(x)dx$$

Een voorbeeld van een kansverdelingsfunctie is de uniforme verdeling. De kansverdelingsfunctie is hier gelijk aan een constante.

A.2 Numerieke integratie

Stel dat de volgende intergraal moet worden uitgerekend:

$$I = \int_a^b f(x)dx$$

Er worden volgens een bekende kansverdeling N punten genomen van de te integreren functie. Uit deze punten kan een schatter van de integraal worden berekend. Deze schatter wordt voorgesteld als $\langle I \rangle$.

$$\langle I \rangle = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)}$$

De verwachte waarde van de schatter is gelijk aan

$$E[\langle I \rangle] = E\left[\frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)}\right]$$

De verwachte waarde van een som van toevalsvariabelen is gelijk aan de som van de verwachte waarden. Er wordt dus verkregen:

$$E[\langle I \rangle] = \frac{1}{N} \sum_{i=1}^{N} E\left[\frac{f(x_i)}{p(x_i)}\right] = \frac{1}{N} N \int \frac{f(x_i)}{p(x_i)} p(x_i)dx = I$$

Een schatter wordt onvervalst[1] genoemd indien de verwachtte waarde van de schatter gelijk is aan de gezochte waarde. Monte Carlo integratie levert dus een onvervalste schatter van de integraal. Schatters die niet onvervalst zijn, maar waar de vervalsing [2] verdwijnt voor een oneindig aantal bemonsteringen wordt een consistent schatter genoemd.

De variantie van deze schatter is gelijk aan

$$\sigma^2 = \frac{1}{N} \int \left(\frac{f(x)}{p(x) - I}\right)^2 p(x)dx$$

De variantie zal dalen met het aantal bemonsteringen dat genomen wordt. De standaarddeviatie daalt met $\sqrt{N}$. De variantie kan geschat worden aan de hand van

$$\sigma^2_{\text{schatting}} = \frac{\frac{1}{N} \sum_{i=1}^{N} \left(\frac{f(x_i)}{p(x_i)}\right)^2 - \left(\frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)}\right)^2}{N - 1}$$

De centrale limietstelling stelt dat de waarden van de schatter normaal verdeeld zijn voor $N \to \infty$. De kans dat de geschatte waarde binnen drie keer de standaard deviatie

[1] De meer gebruikte Engelse term hiervoor is *unbiased*.
[2] De meer gebruikte Engelse term hiervoor is *bias*.

ligt, is gelijk aan 99.7 % De fout daalt met $1/\sqrt{N}$. Voor grotere waarde van N wordt er dus steeds een grotere nauwkeurigheid verkregen. De centrale limietstelling is wel enkel geldig voor grote N. Het is niet echter niet duidelijk hoe groot N moet zijn. De meeste Monte Carlo technieken veronderstellen daarom dat N groot genoeg is. Voor lage waarden van N is er echter speciale aandacht nodig.

Monte Carlo integratie bestaat uit 3 stappen

1. Bemonster een punt aan de hand van een kansdichtheidsfunctie.
2. Bereken de waarde van de functie in dat punt.
3. Bereken het gemiddelde van de gewogen waarden.

Voor technieken om de fout op de schatting te verminderen wordt er verwezen naar [KW86].

Bijlage B

Illustraties bij het hoofdstuk resultaten

Deze bijlage bevat de illustratie van de methodes toegepast op de testscènes. Bij elke methode wordt er telkens slechts de eerste 20 of 40 resultaten getoond aangezien het beeld na zoveel stappen meestal reeds geconvergeerd is.

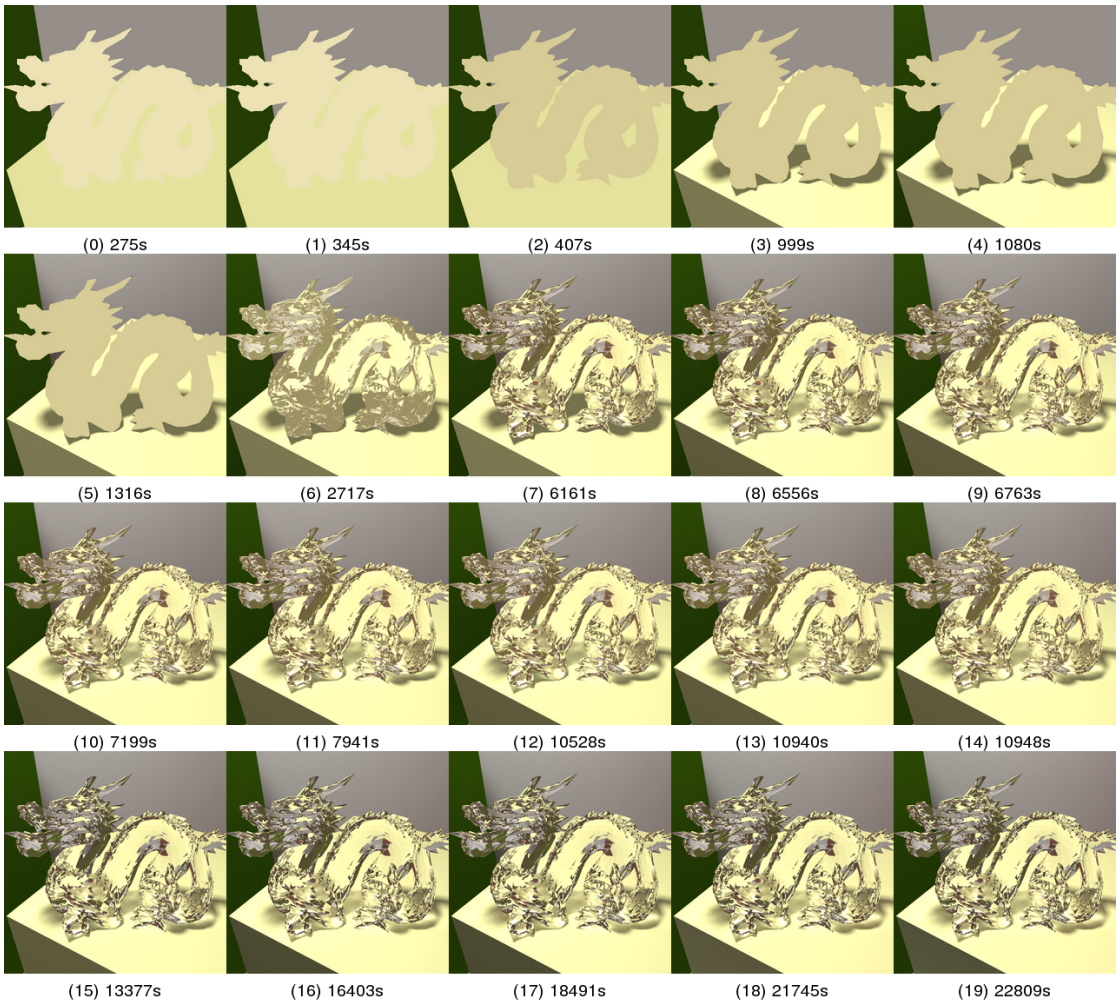
Elk tussenresultaat gaat gepaard met het nummer van de iteratie en de geaccumuleerde tijd (in seconden) die nodig was om het beeld te berekenen. Onder de tussenresultaten vindt men de componenten die in die bepaalde fase van het renderproces werden berekend.

B.1 Glazen draak

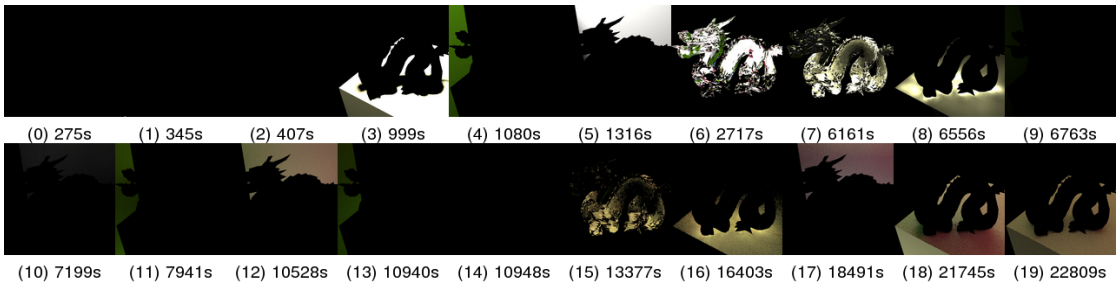
De scène bevindt zich in een cornell cube[1]. Op een van de kubussen is een glazen draak geplaatst. De scène wordt belicht door een oppervlaktelichtbron.

Figuur B.1 is bekomen met de oorspronkelijke methode [DD05], figuur B.2 met verfijnde benadering en figuur B.3 met verfijnde benadering en adaptieve bemonstering. Figuur B.4 en B.5 tonen telkens 20 van de eerste 40 stappen van de toepassing van de methode met verfijnde componenten.

[1] Een cornell cube is een typische computer graphics scène. Een kubus open aan een kant bevat twee kleinere kubussen

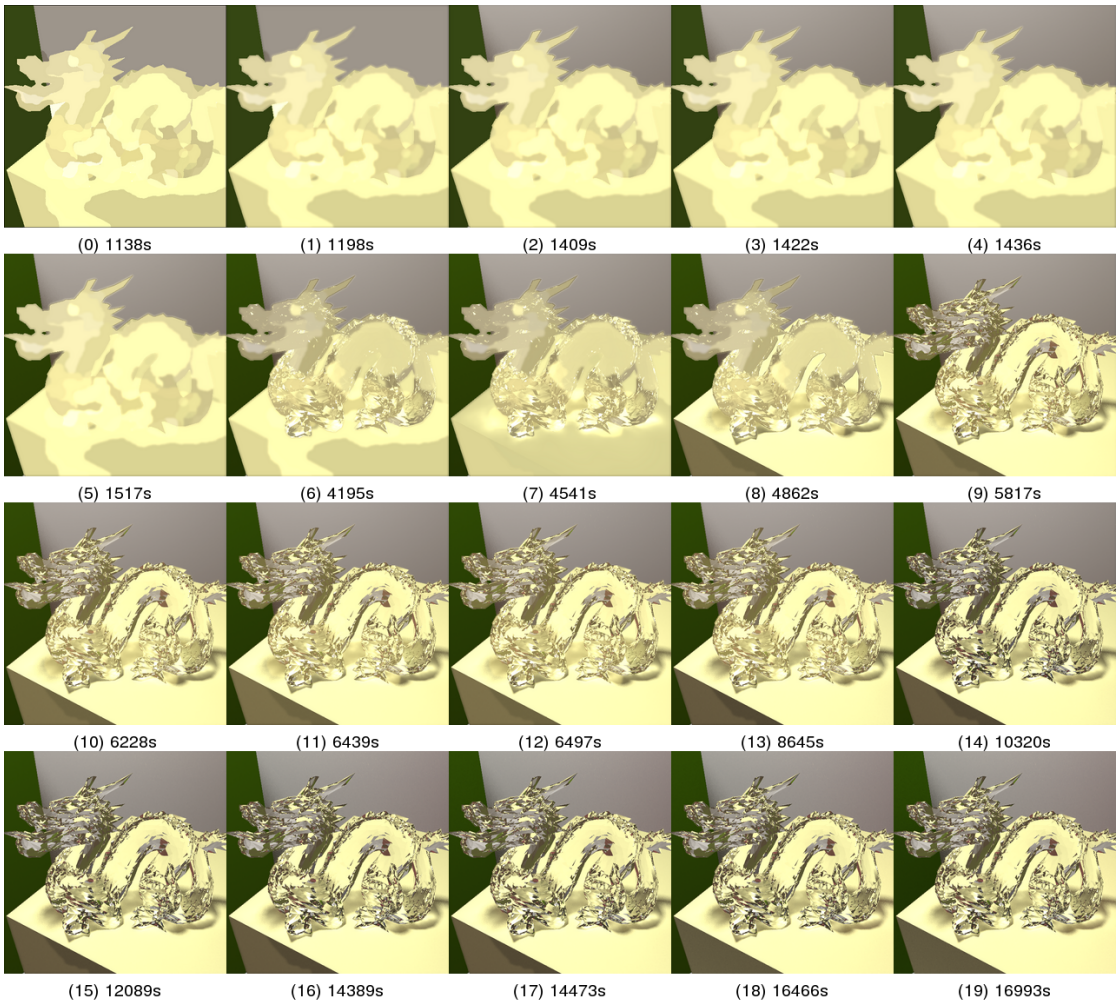


(a) Opeenvolgende tussenresultaten

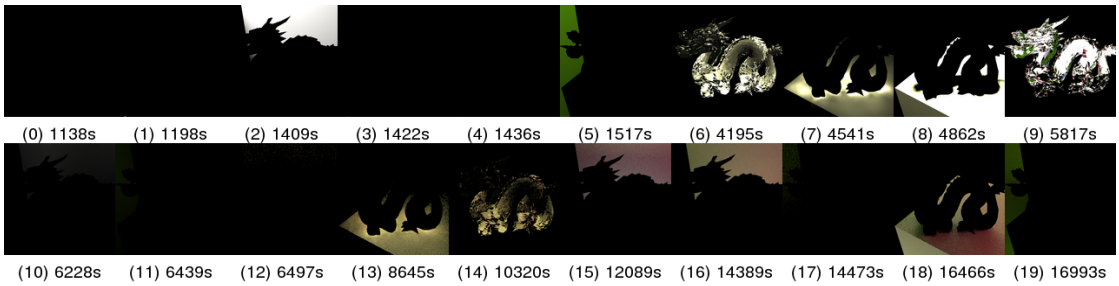


(b) De berekende componenten voor elke tussenstap

Figuur B.1: Scène met glazen draak berekend met methode beschreven in [DD05].

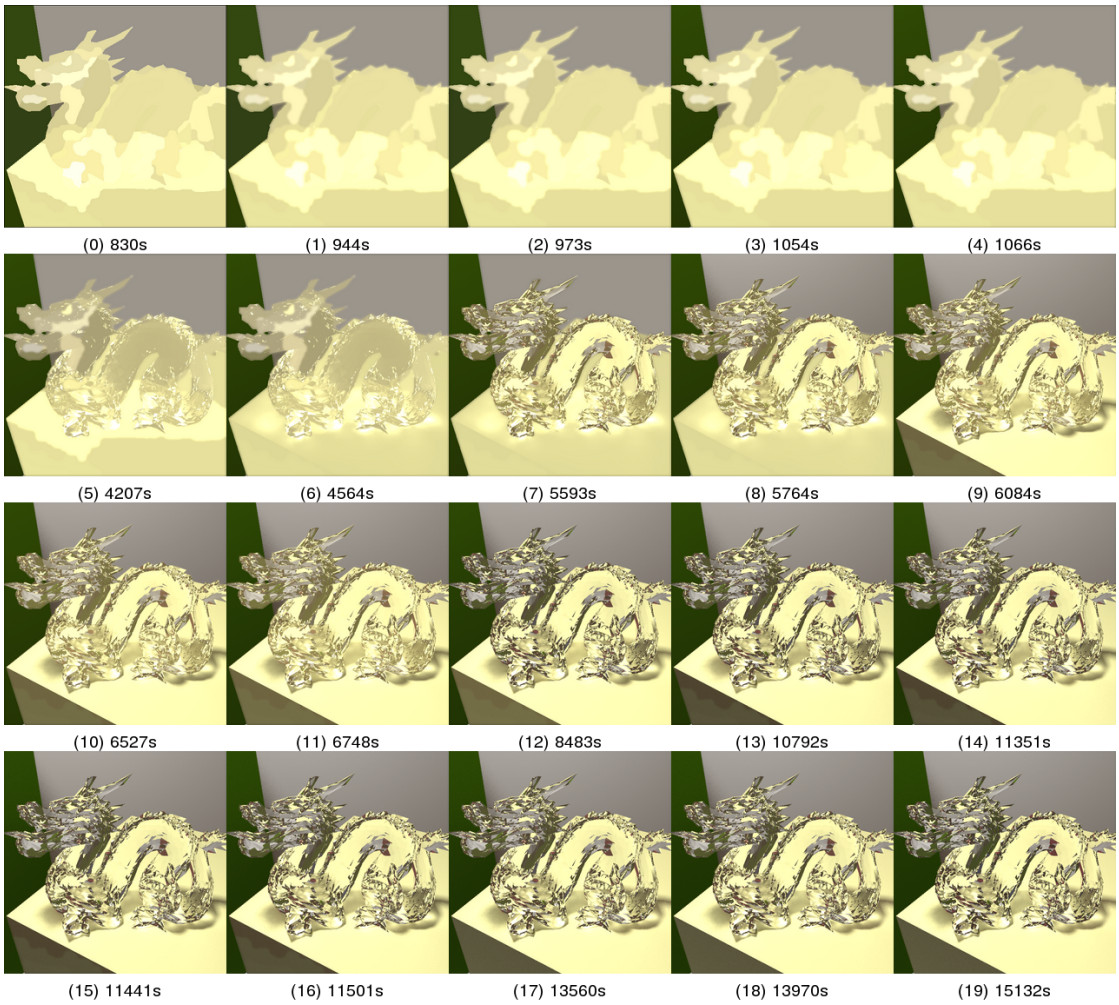


(a) Opeenvolgende tussenresultaten

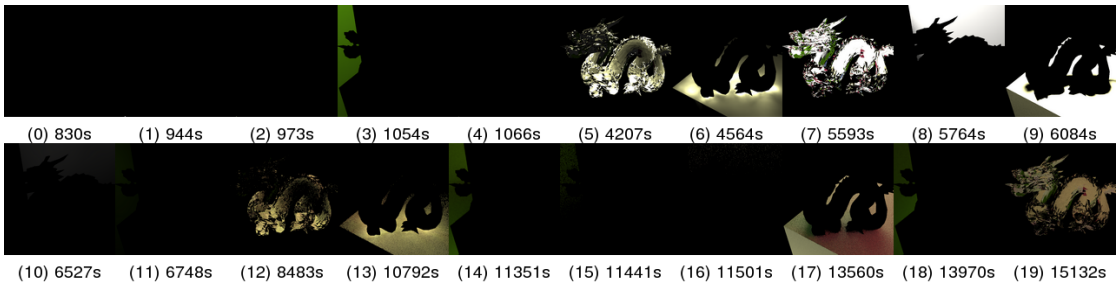


(b) De berekende componenten voor elke tussenstap

Figuur B.2: Scène met glazen draak berekend met verfijnde benadering.

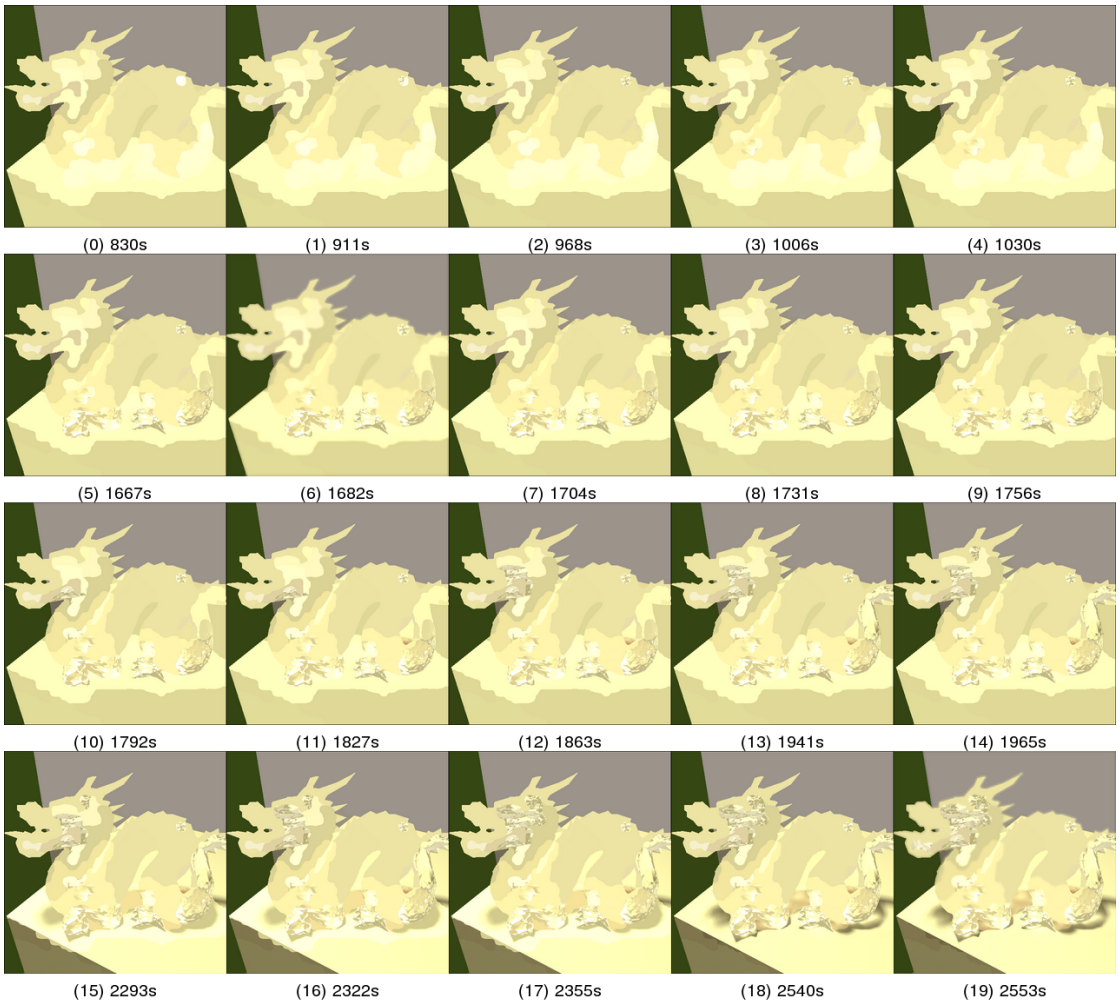


(a) Opeenvolgende tussenresultaten

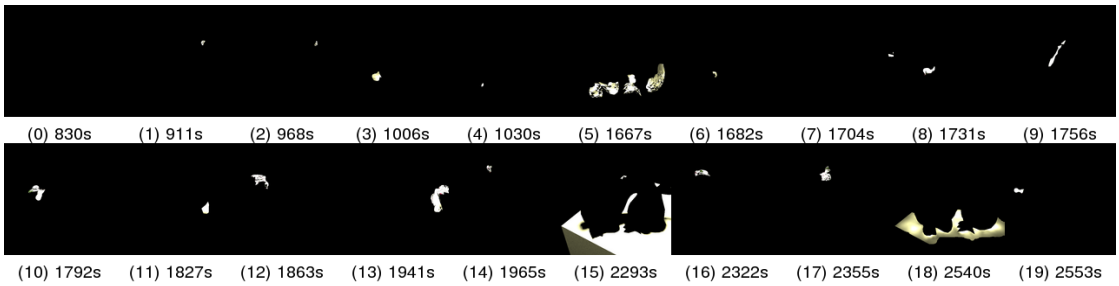


(b) De berekende componenten voor elke tussenstap

Figuur B.3: Scène met glazen draak berekend met verfijnde benadering en versnelde radiantieschatting.

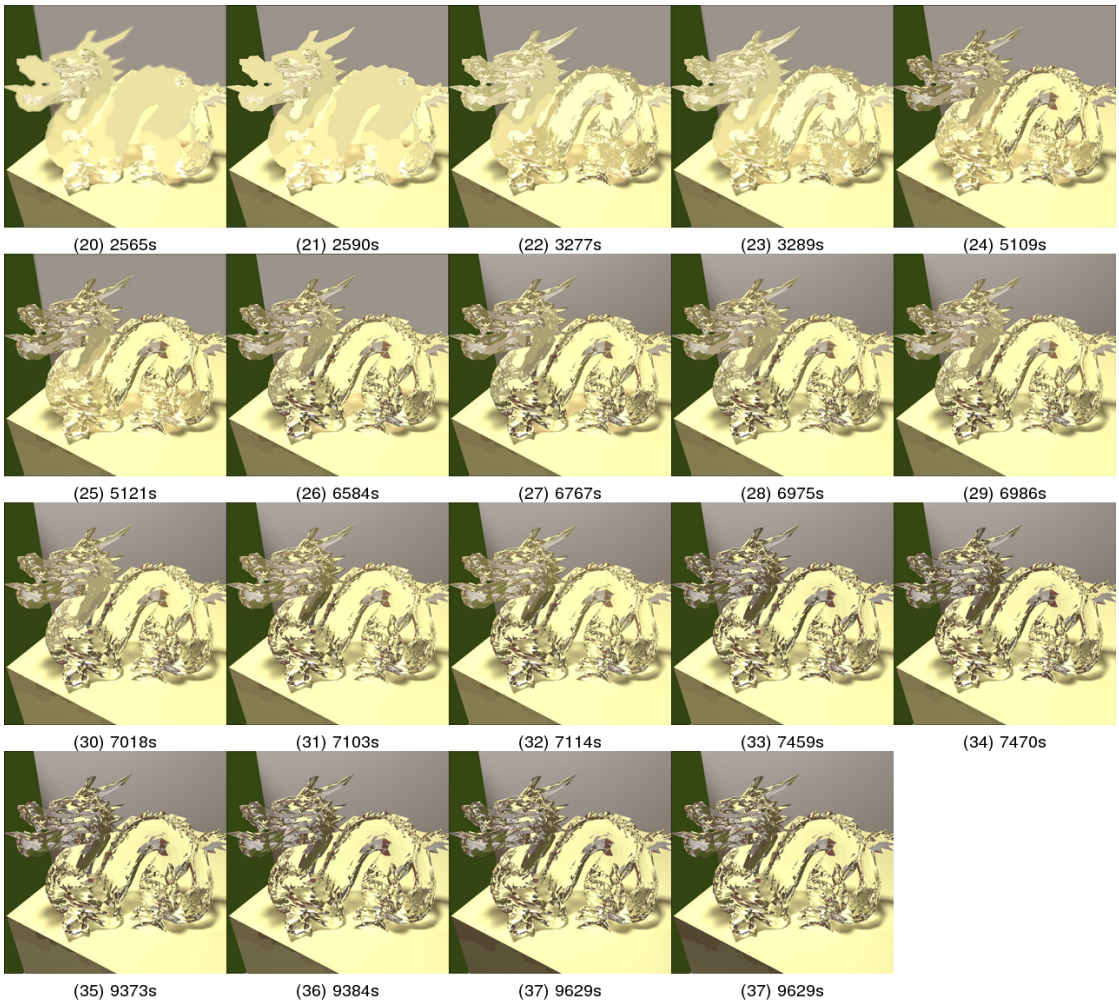


(a) Opeenvolgende tussenresultaten

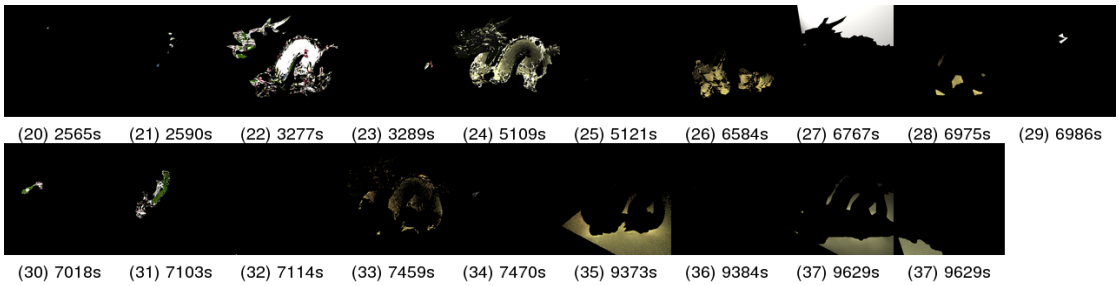


(b) De berekende componenten voor elke tussenstap

Figuur B.4: Scène met glazen draak berekend met verfijnde benadering, versnelde radiantieschatting en verfijnde componenten. Stap 1 tot 20.



(a) Opeenvolgende tussenresultaten



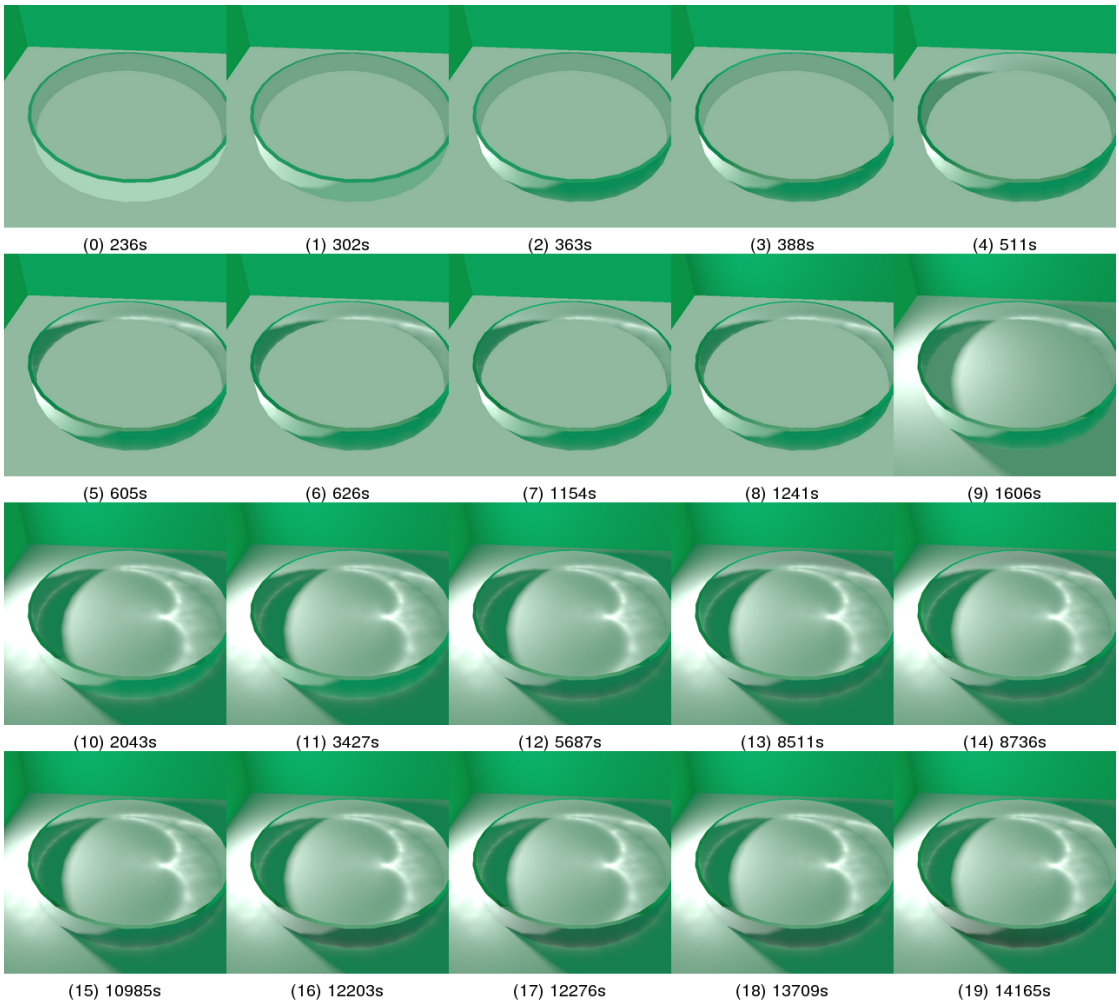
(b) De berekende componenten voor elke tussenstap

Figuur B.5: Scène met glazen draak berekend met verfijnde benadering, versnelde radiantieschatting en verfijnde componenten. Stap 21 tot 40.

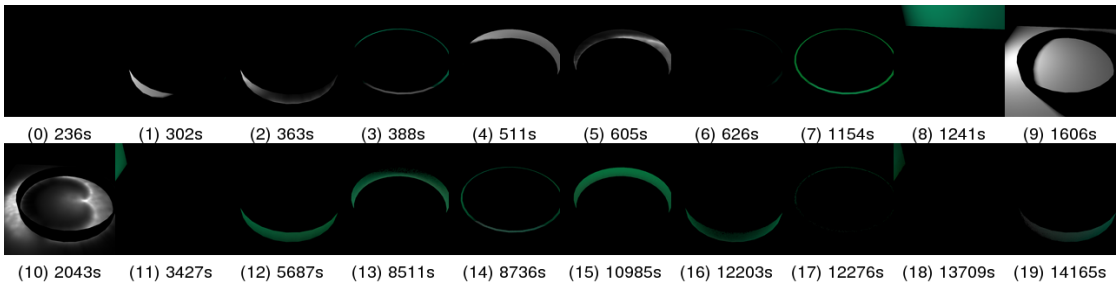
B.2 Metalen ring

In een hoek gevormd door drie diffuse vlakken wordt een metalen ring geplaatst. Door de inval van het licht op deze ring onstaat op het ondervalk reflectieve caustics.

Figuur B.6 illustreert de oorspronkelijke methode [DD05], figuur B.7 de verfijnde benadering en figuur B.8 de methode met verfijnde componenten.

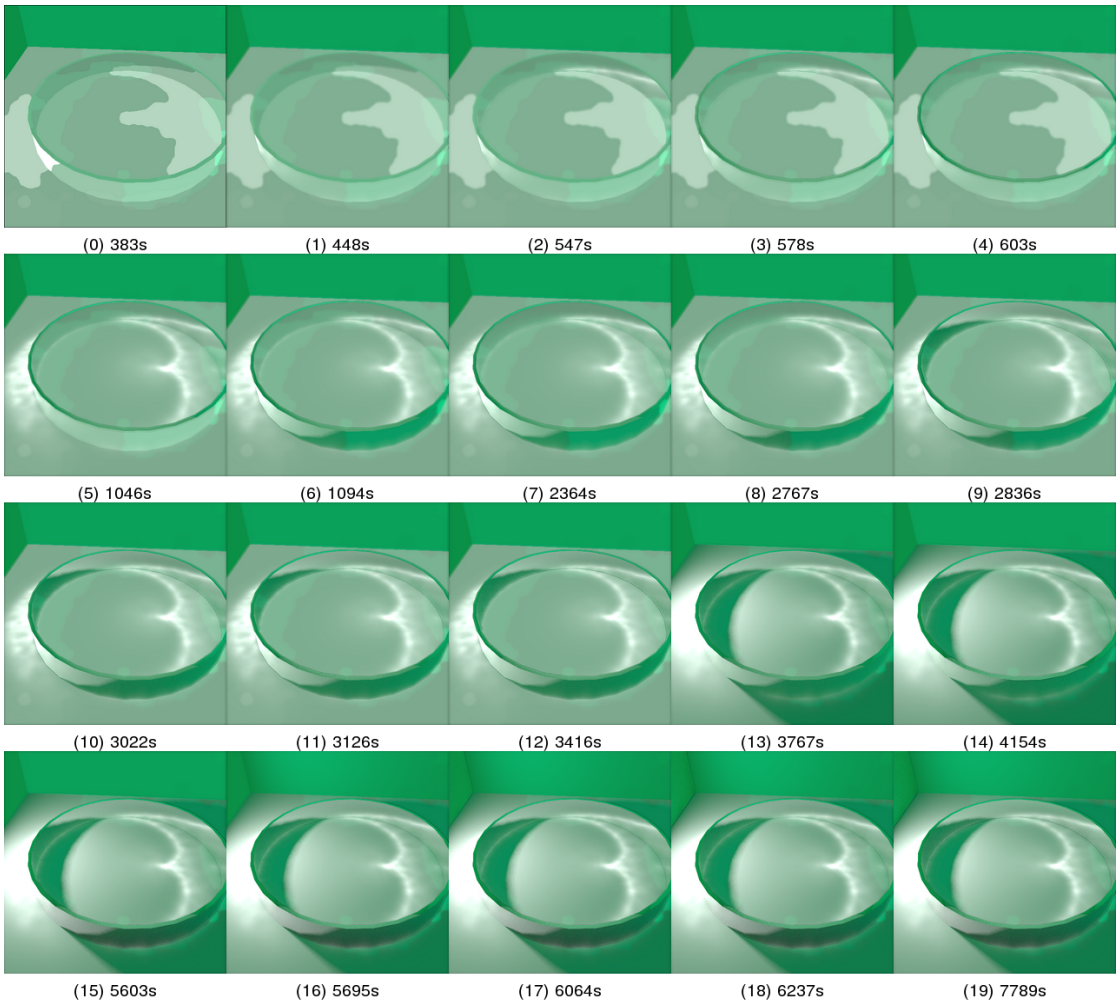


(a) Opeenvolgende tussenresultaten

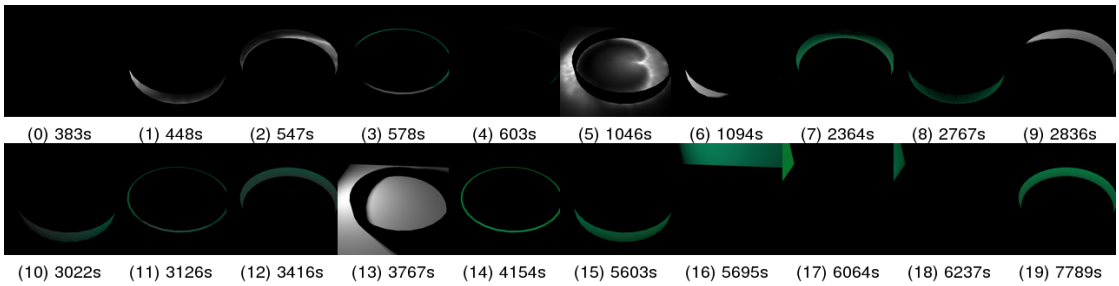


(b) De berekende componenten voor elke tussenstap

Figuur B.6: Scène met metalen ring berekend met methode beschreven in [DD05].

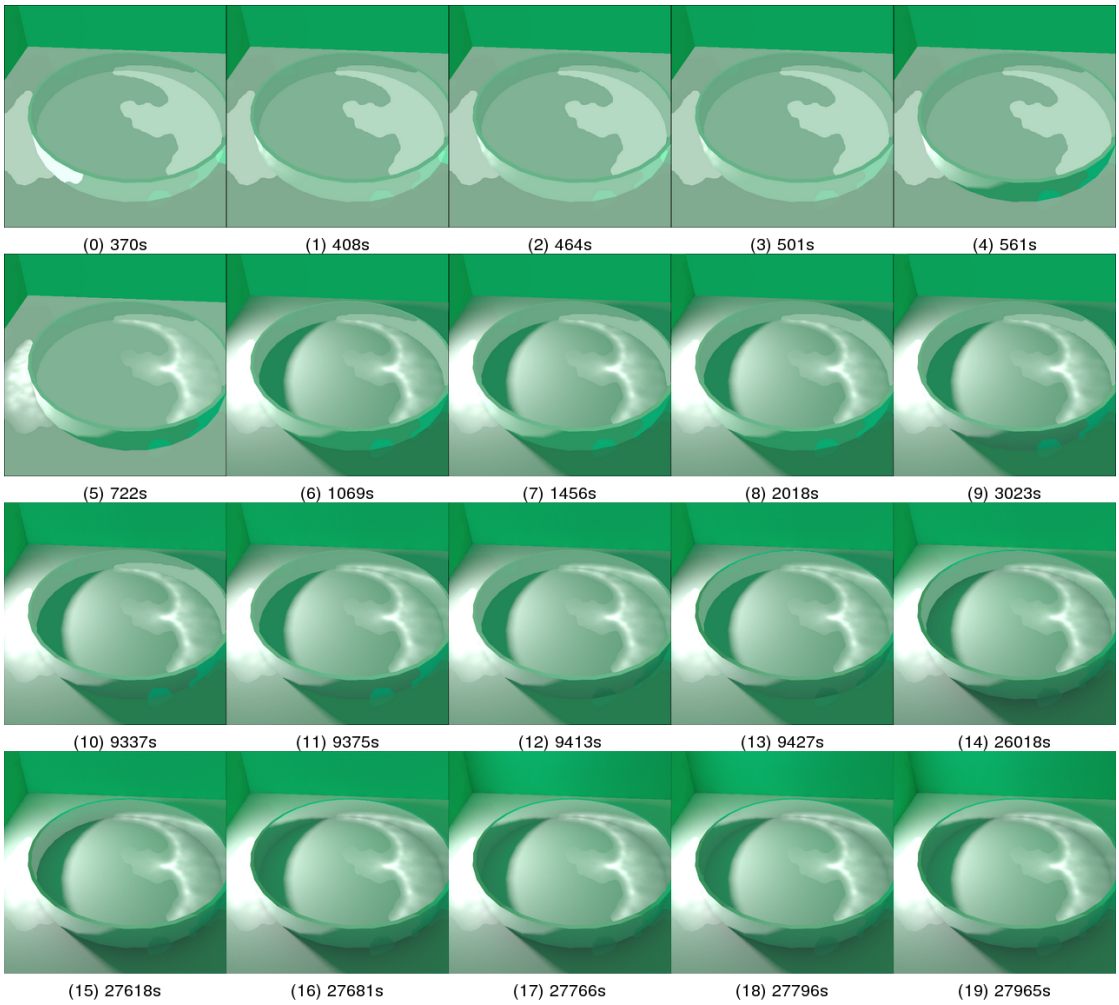


(a) Opeenvolgende tussenresultaten

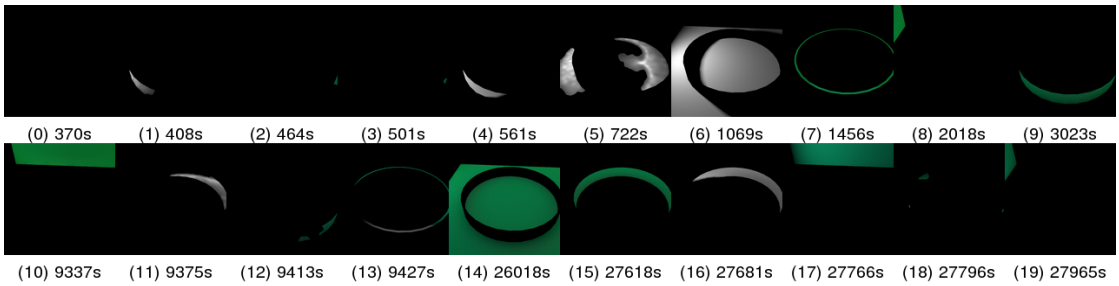


(b) De berekende componenten voor elke tussenstap

Figuur B.7: Scène met metalen ring berekend met verfijnde benadering.



(a) Opeenvolgende tussenresultaten



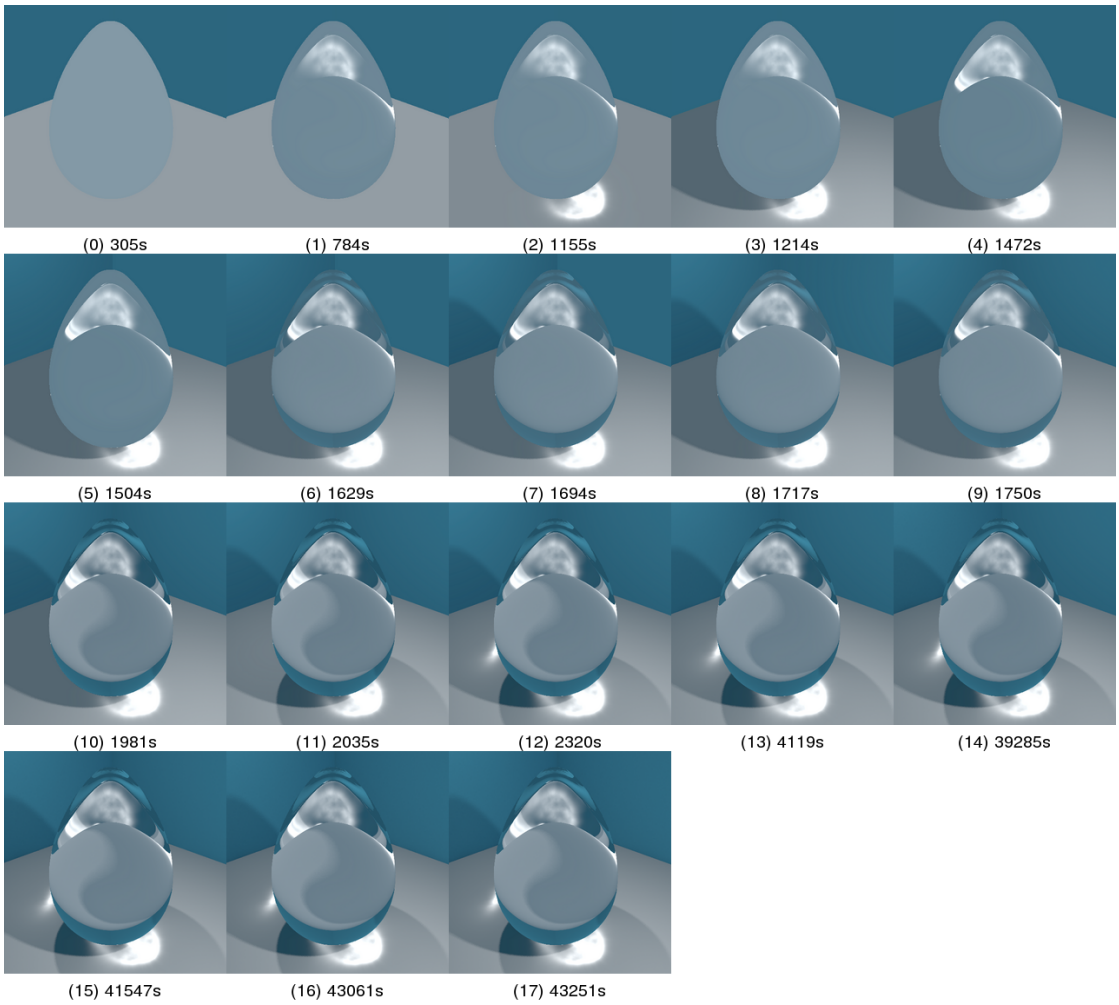
(b) De berekende componenten voor elke tussenstap

Figuur B.8: Scène met metalen ring berekend met verfijnde benadering en verfijnde componenten. Stap 1 tot 20.

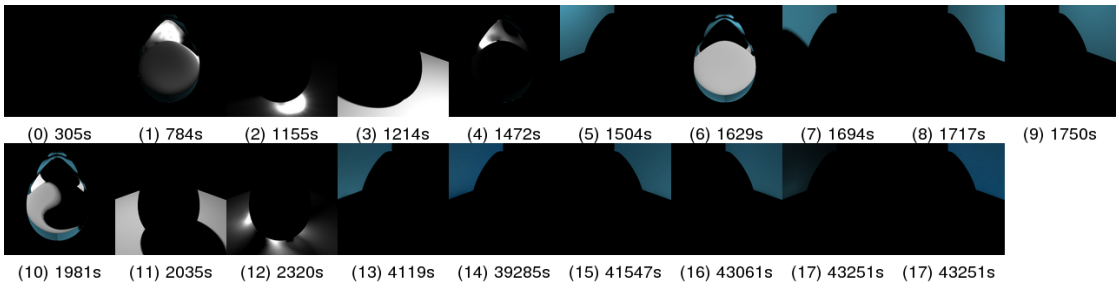
B.3 Glazen ei

De scène is een glazen ei in een cornell cube. Er zijn twee puntlichtbronnen met elk hun eigen schaduw en refractieve caustic op het ondervlak.

Figuur B.9 illustreert de oorspronkelijke methode [DD05], figuur B.10 de verfijnde benadering en figuur B.11 de methode met verfijnde componenten.

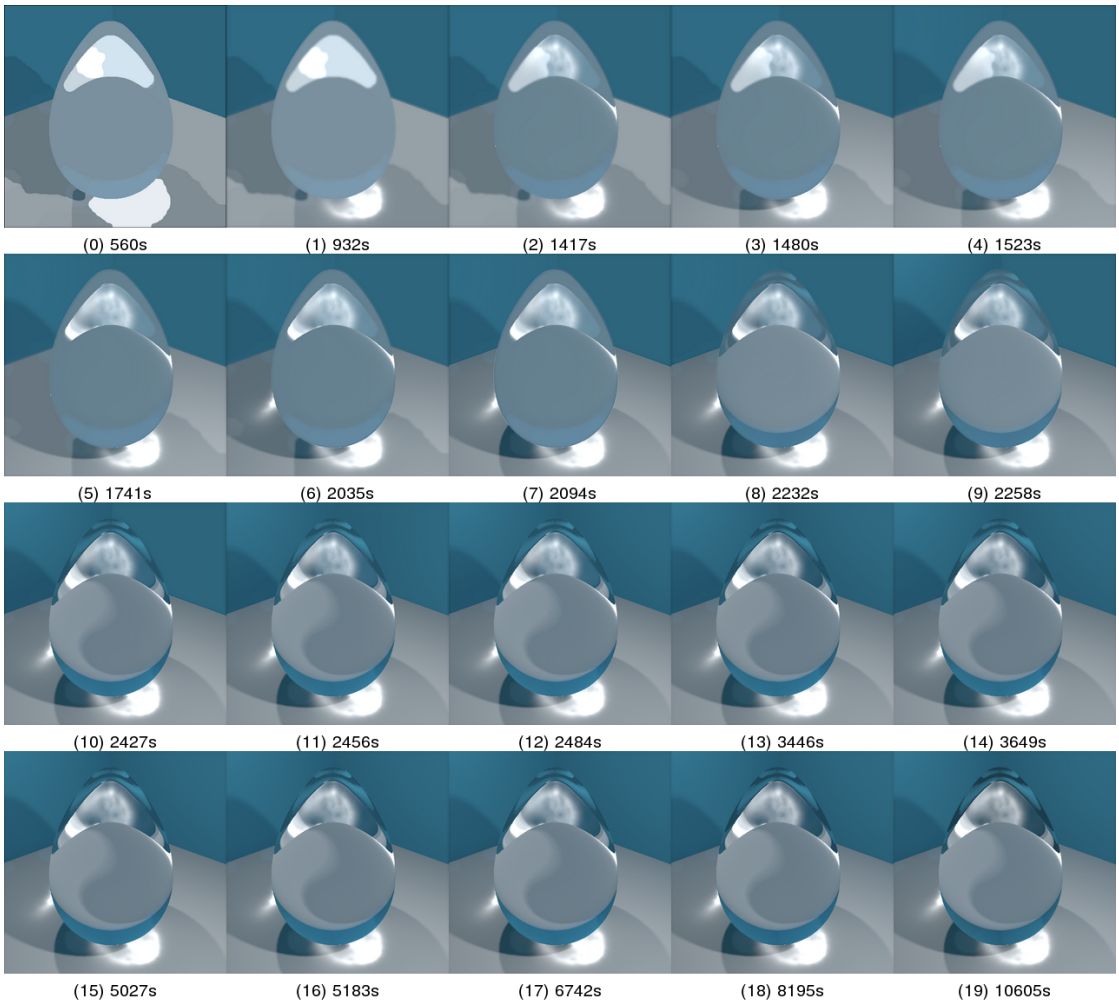


(a) Opeenvolgende tussenresultaten

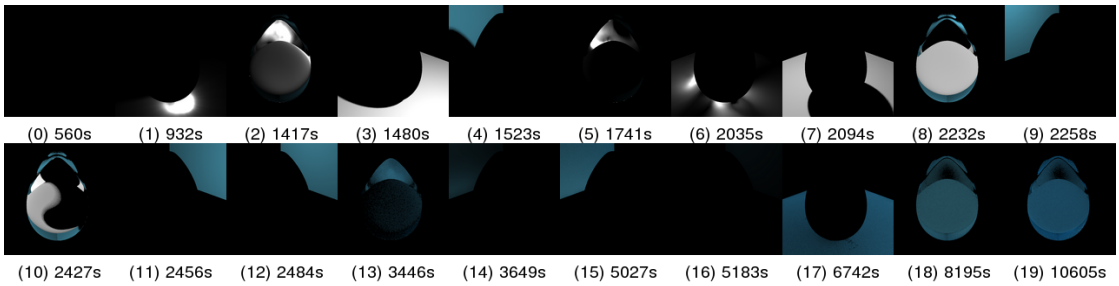


(b) De berekende componenten voor elke tussenstap

Figuur B.9: Scène met glazen ei berekend met methode beschreven in [DD05].

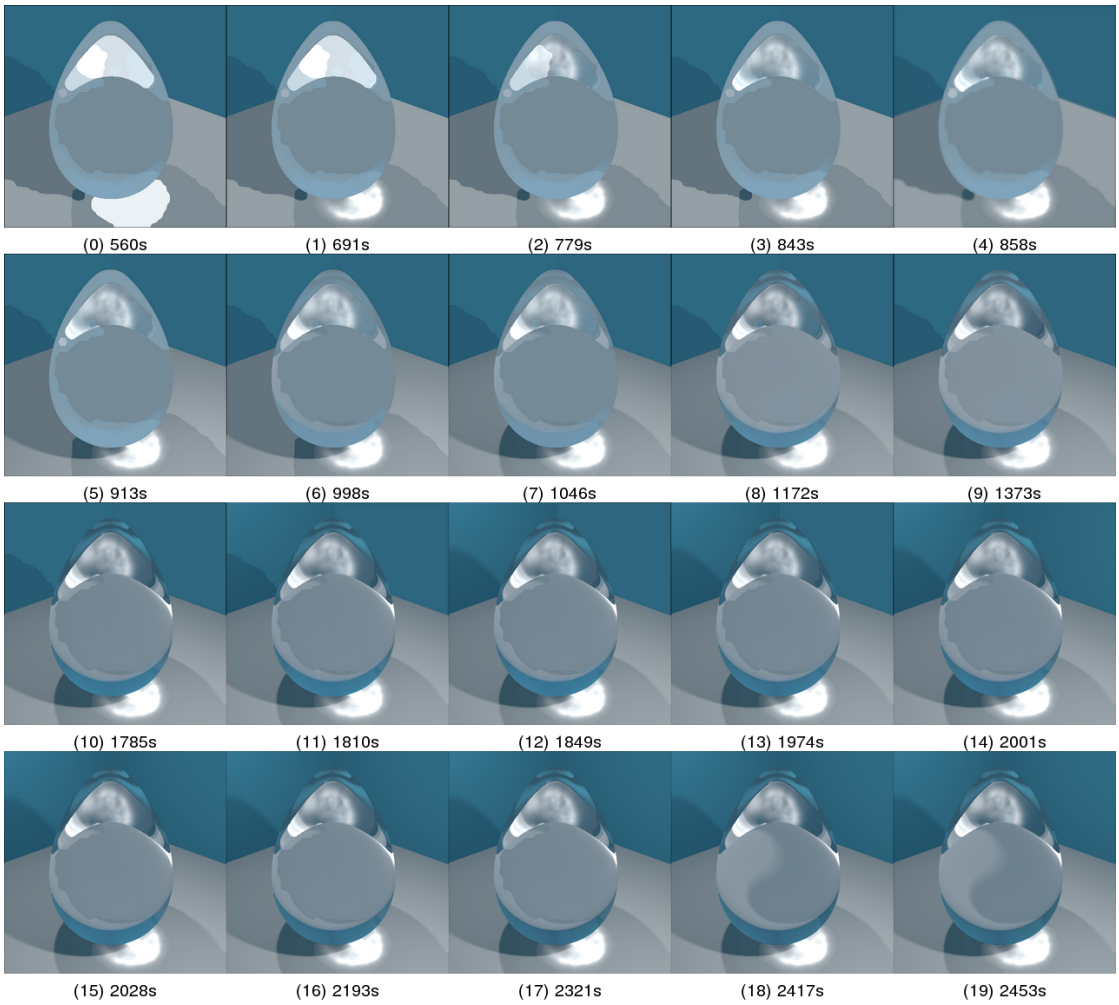


(a) Opeenvolgende tussenresultaten

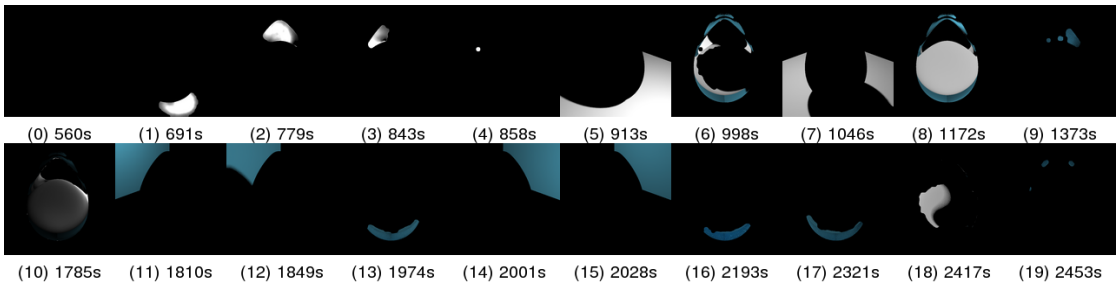


(b) De berekende componenten voor elke tussenstap

Figuur B.10: Scène met glazen ei berekend met verfijnde benadering.



(a) Opeenvolgende tussenresultaten



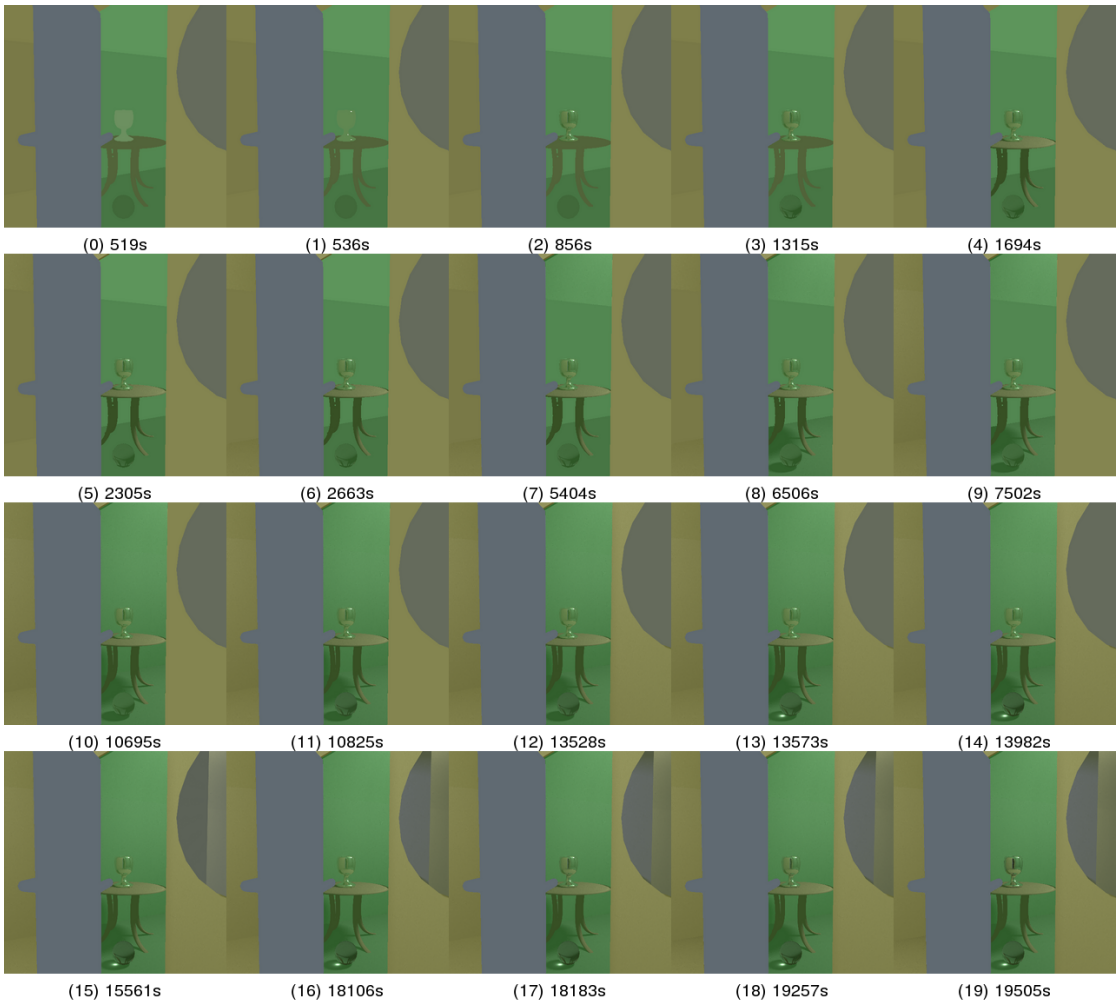
(b) De berekende componenten voor elke tussenstap

Figuur B.11: Scène met glazen ei berekend met verfijnde benadering en verfijnde componenten.

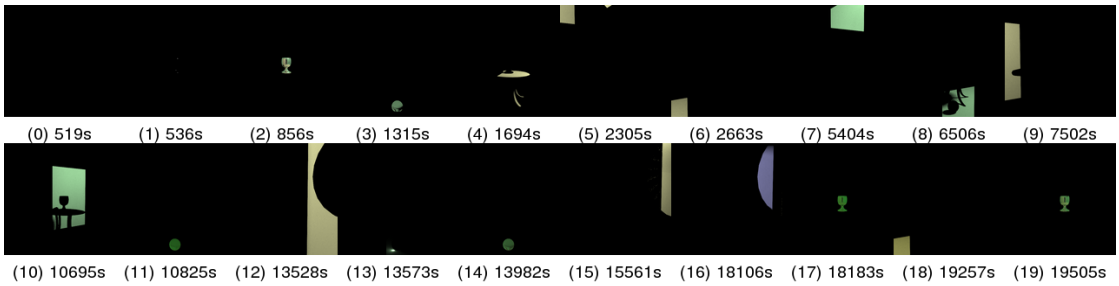
B.4 Kamers

Een zicht van een kamer door een kier van een deur. In de kamer waar de camera zich bevindt is er daardoor hoofdzakelijk indirecte belichting. Deze kamer bevat ook aan de rechterkant een spiegel die de direct beschenen kant van de deur weerspiegelt. In de kamer waar naarbinnen wordt gekeken onderscheiden we een tafel met daarop een spiegelende beker en op de grond een glazen bol die er een caustic op werpt.

Figuur B.12 en B.13 tonen elk 20 stappen van de oorspronkelijke methode [DD05]. Figuur B.14 en B.15 doen dit voor de methode met verfijnde benadering.

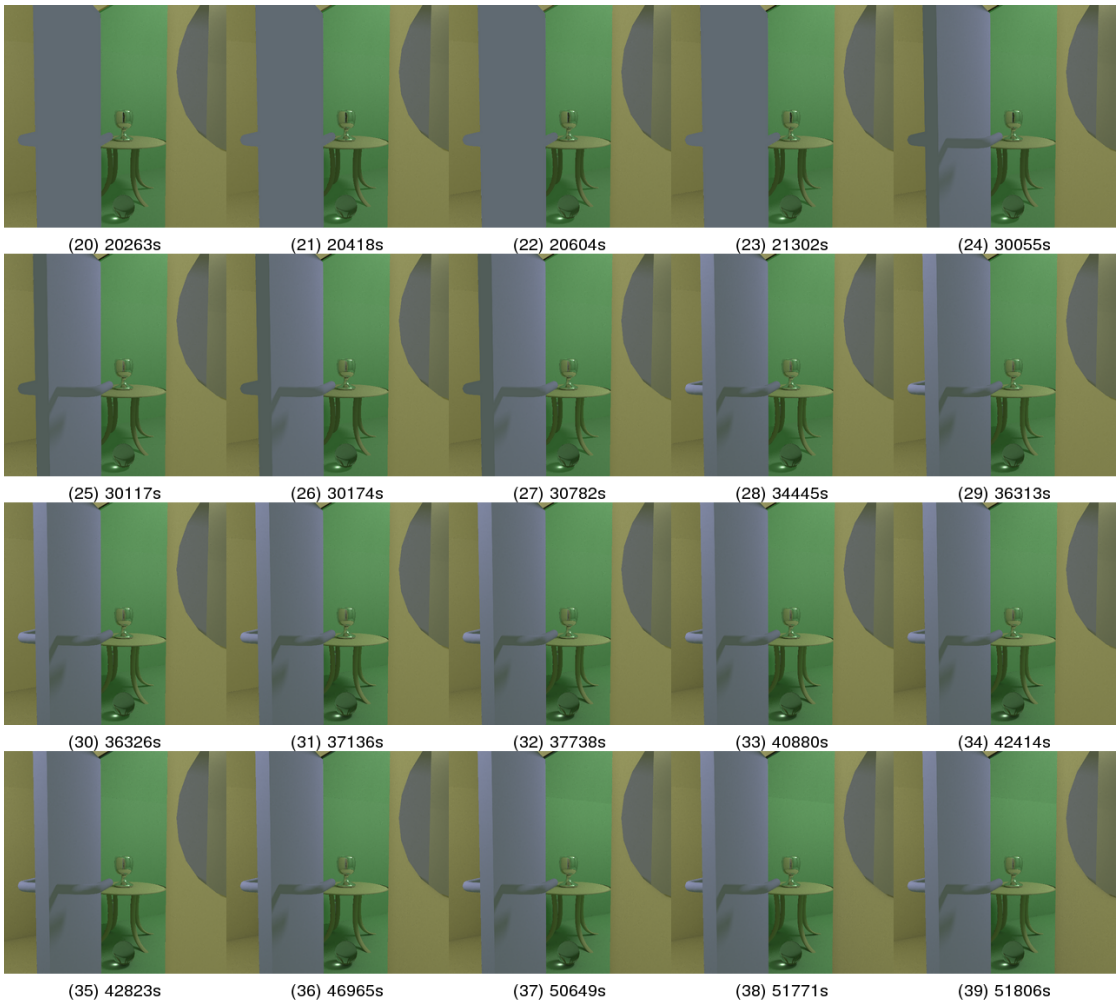


(a) Opeenvolgende tussenresultaten

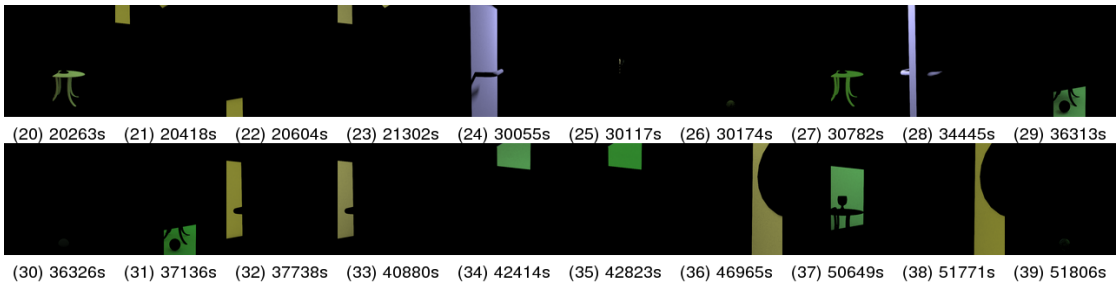


(b) De berekende componenten voor elke tussenstap

Figuur B.12: Scène met kamers berekend met methode beschreven in [DD05]. Stap 0 tot 19.

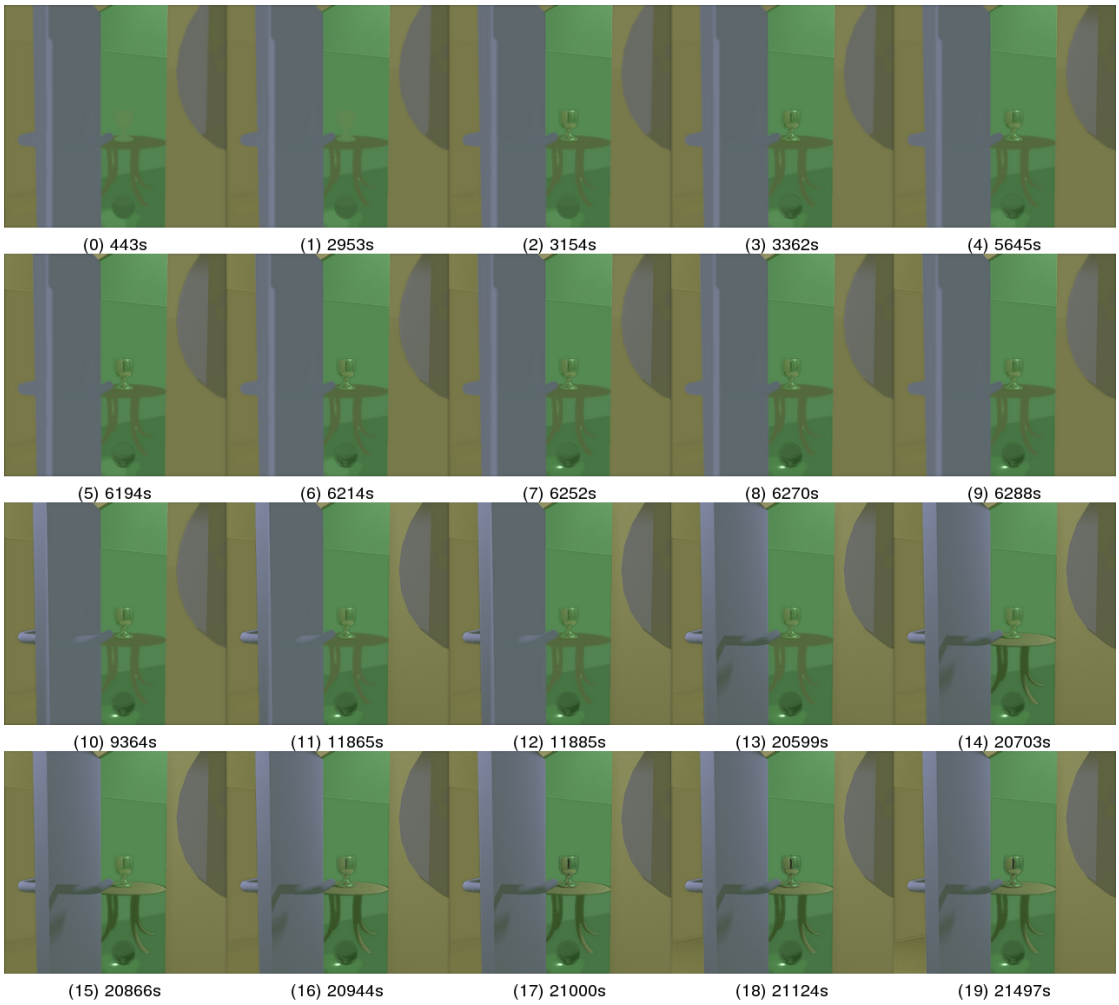


(a) Opeenvolgende tussenresultaten

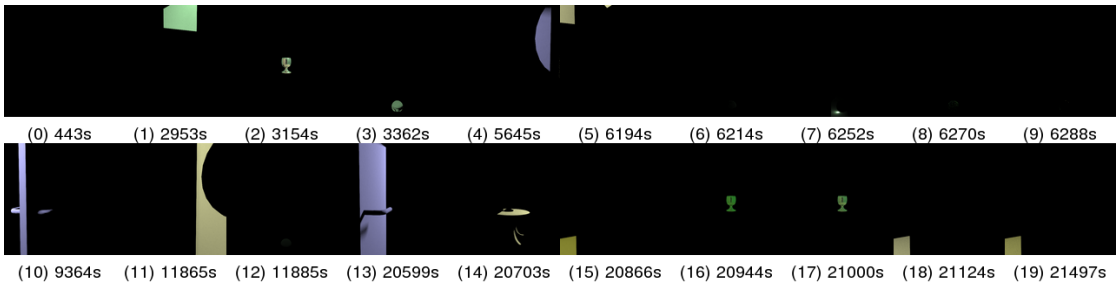


(b) De berekende componenten voor elke tussenstap

Figuur B.13: Scène met kamers berekend met methode beschreven in [DD05]. Stap 20 tot 39.

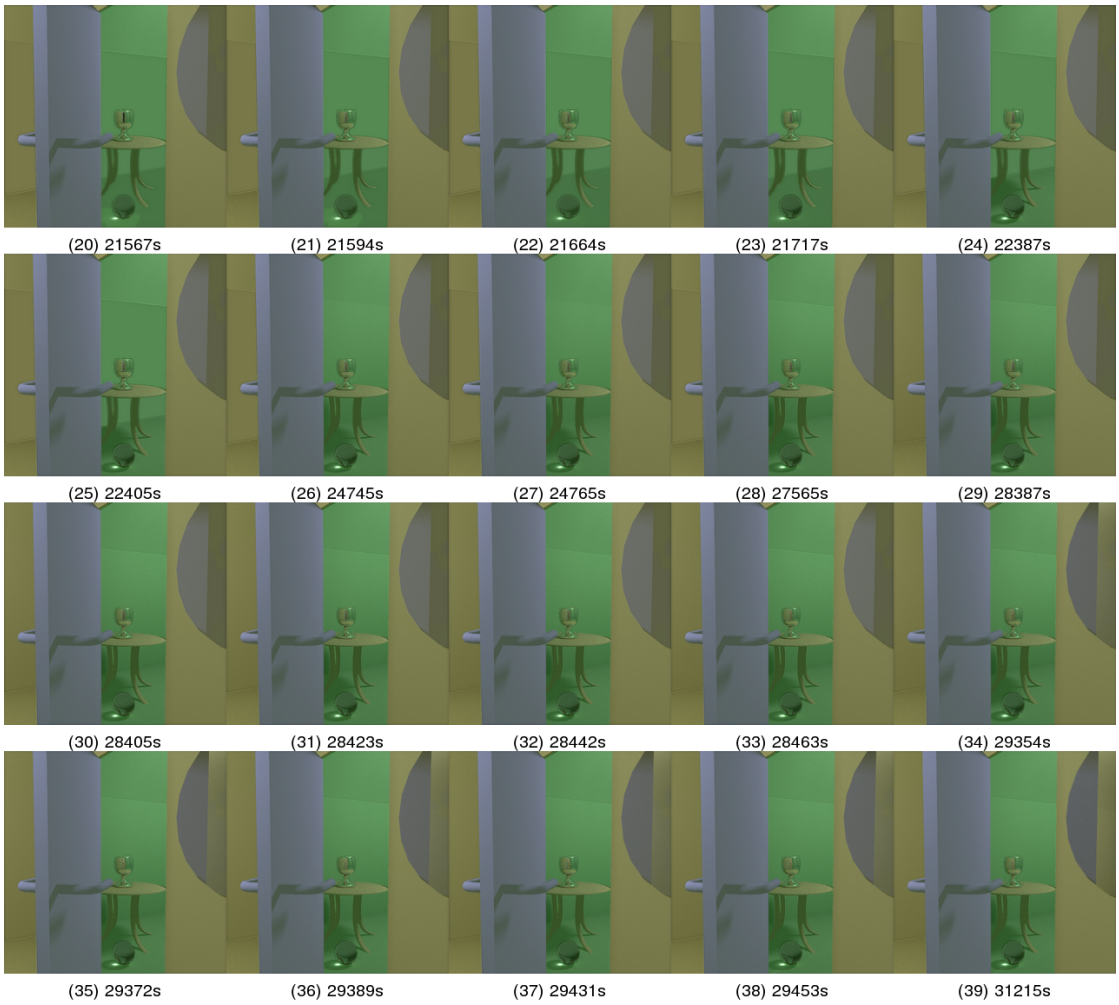


(a) Opeenvolgende tussenresultaten

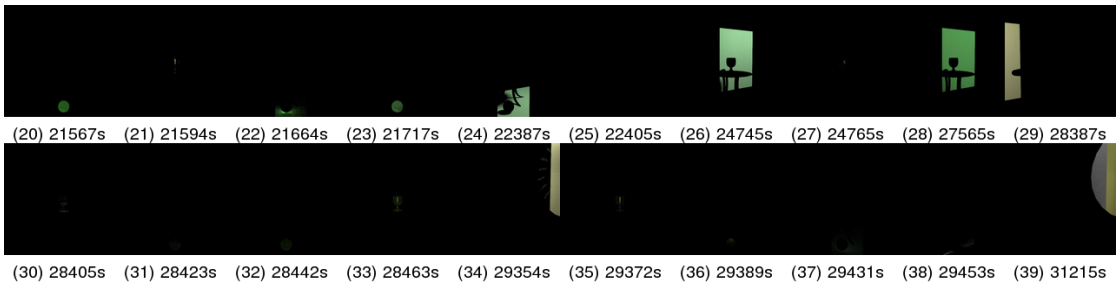


(b) De berekende componenten voor elke tussenstap

Figuur B.14: Scène met kamers berekend met verfijnde benadering. Stap 0 tot 19.



(a) Opeenvolgende tussenresultaten



(b) De berekende componenten voor elke tussenstap

Figuur B.15: Scène met kamers berekend met verfijnde benadering en verfijnde componenten. Stap 20 tot 39.

Lijst van tabellen

2.1 Belangrijkste radiometrische grootheden 8

3.1 Notaties voor interacties met oppervlakken in een lichtpad 30

3.2 Reguliere expressies 31

3.3 Lichtpadtypes als reguliere expressies 31

6.1 Opeenvolgende verfijningen: gebruikte segmentatie in functie van het doel 76

7.1 Preprocessing tijden voor de scène met de glazen draak. 102

7.2 Preprocessing tijden voor de scène met metalen ring 107

7.3 Benodigde tijd om een bepaald effect te bekomen in de scène met metalen ring. 109

7.4 Preprocessing tijden voor de scène met glazen ei. 109

7.5 Preprocessing tijden voor de scène met kamers. 111

Lijst van figuren

2.1 Voorbeelden van realistische computergegeneerde beelden 6
2.2 Voorstelling van de BRDF van isotrope materialen 11
2.3 Beeldsynthese schematisch voorgesteld 13
2.4 2D-doosfilter 13
2.5 Resultaten van het path tracing algoritme 16
2.6 Soorten photon maps 19
2.7 Fotonen in de photon map 21
2.8 Schatting van de radiantie aan de hand van de photon map 21
2.9 Rechstreekse visualisatie van de photon map 23
2.10 Photon mapping 24
2.11 Metropolis lichttransport algortime 25

3.1 Overzicht van de aanpak. (uit [DD05]) 28
3.2 Reguliere expressies voor lichtpadtypes 32
3.3 Voorbeeld van geometrische segmentatie 34
3.4 Deelbemonsterende piramide bij rechthoekige segmentatie 36
3.5 Rechthoekige segmentatie 37
3.6 Prioriteit van een component 37
3.7 Berekening van prioriteiten van componenten 38
3.8 Principe van het incrementeel renderen 39

3.9 Voorbeeld van een thresholdmap 40

3.10 Het principe van tijdsgebonden renderen 41

5.1 Visualisatie van schatting lichtpadtypes 46

5.2 Rechthoekige interpolatie 48

5.3 Intersectie van foton-bollen 49

5.4 Gemeenschappelijk volume van van bollen 50

5.5 Adaptieve onderverdeling voor snelle lichtschatting 51

5.6 Voorbeeld van versnelde lichtschatting 52

5.7 Relatieve en absolute bijdragen van twee lichtpadtypes 53

5.8 Enkele randdetectiefilters. 54

5.9 Randfilters uit figuur 5.8 toegepast op een beeld. De gefilterde beelden zijn geïnverteerd. 55

5.10 Randdetectie 56

5.11 Voorstelling van een beeld met één kleurendimensie door middel van een puntenwolk. 57

5.12 Enkele veelgebruikte kernelfuncties 58

5.13 Voorbeeld van een beeld gefilterd met het mean shift algoritme 61

5.14 Voorbeeld van een beeld gesegmenteerd met het mean shift algoritme . . . 61

5.15 Invloed van keuze bandbreedteparameters op resultaat mean shift algoritme 62

5.16 Schatting bijdrages verschillende lichtpaden 64

5.17 Schatting via adaptieve bandbreedte per pixel. 66

5.18 Behoud van minder belangrijke grote gebieden. 68

5.19 Behoud van minder belangrijke grote gebieden, *niet* toegepast. 69

5.20 Bepaling van adaptieve minimale regiogrootte 71

5.21 Adaptieve minimale regiogrootte. 72

5.22 Illustratie van het afronden van randen 73

6.1 De effecten van een gemeenschappelijke segmentatie 77

6.2 Het effect van rechthoekige segmentatie op de thresholdmap 79

6.3 Vergelijking thresholdmaps 80

6.4 Verbetering van de benadering door fotonsegmentatie 81

6.5 Behandeling van kartelingen in de benadering 82

6.6 Toepassing van een gaussiaanse vervagingsfilter 83

6.7 Lokale invloed van een partitie op de convolutiematrix bij gepartitioneerde vervaging 84

6.8 Toepassing van de fotonbenadering 87

6.9 Fijnere verdeling van de componenten 89

6.10 De versnelling door het gebruik van fijnkorrelige componenten 90

6.11 Behandeling van randen tussen gerenderde en benaderde componenten . . 91

6.12 Segmentatie van het lichtpadtype caustics 92

6.13 Informatiestroom bij de verschillende methodes 95

7.1 Vergelijking tussen incrementeel renderen en een normaal path tracing algoritme. 99

7.2 Vergelijking tussen incrementeel renderen en een normaal path tracing algoritme. 99

7.3 Testscènes 100

7.4 De bemonsteringen voor de versnelde radiantieschatting voor de draak-scène 103

7.5 Logische en euclidische afstand bij versnelde schatting 104

7.6 Invloed van de bemonsteringsgraad op de initiële schatting van lichtinformatie voor fotonsegmentatie 104

7.7 Invloed van de bemonsteringsgraad op de segmentatie 105

7.8 Invloed van adaptieve bemonstering bij de scène met kamers 111

B.1 Scène met glazen draak berekend met methode beschreven in [DD05]. . . 122

B.2 Scène met glazen draak berekend met verfijnde benadering. 123

B.3 Scène met glazen draak berekend met verfijnde benadering en versnelde radiantieschatting. 124

B.4 Scène met glazen draak berekend met verfijnde benadering, versnelde radiantieschatting en verfijnde componenten. Stap 1 tot 20. 125

B.5 Scène met glazen draak berekend met verfijnde benadering, versnelde radiantieschatting en verfijnde componenten. Stap 21 tot 40. 126

B.6 Scène met metalen ring berekend met methode beschreven in [DD05]. . . 128

B.7 Scène met metalen ring berekend met verfijnde benadering. 129

B.8 Scène met metalen ring berekend met verfijnde benadering en verfijnde componenten. Stap 1 tot 20. 130

B.9 Scène met glazen ei berekend met methode beschreven in [DD05]. 132

B.10 Scène met glazen ei berekend met verfijnde benadering. 133

B.11 Scène met glazen ei berekend met verfijnde benadering en verfijnde componenten. 134

B.12 Scène met kamers berekend met methode beschreven in [DD05]. Stap 0 tot 19. 136

B.13 Scène met kamers berekend met methode beschreven in [DD05]. Stap 20 tot 39. 137

B.14 Scène met kamers berekend met verfijnde benadering. Stap 0 tot 19. . . . 138

B.15 Scène met kamers berekend met verfijnde benadering en verfijnde componenten. Stap 20 tot 39. 139

Bibliografie

[BM98] Mark R. Bolin and Gary W. Meyer. A perceptually based adaptive sampling algorithm. In *Proceedings of ACM SIGGRAPH 1998*, pages 299–309. ACM Press, 1998.

[CGM02] Christopher M. Christoudias, Bogdan Georgescu, and Peter Meer. Synergism in low level vision. In *ICPR (4)*, pages 150–155, 2002.

[CM02] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(5):603–619, 2002.

[CRM01] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. The variable bandwidth mean shift and data-driven scale selection. In *ICCV*, pages 438–445, 2001.

[CT82] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. 1(1):7–24, January 1982.

[Dal93] Scott Daly. The visible differences predictor: an algorithm for the assessment of image fidelity. In *Digital images and human vision*, pages 179–206. MIT Press, 1993.

[DBB03] Philip Dutré, Kavita Bala, and Philippe Bekaert. *Advanced Global Illumination*. A. K. Peters, Ltd., 2003.

[DD05] Olivier Dumont and Philip Dutré. Light transport component analysis & rendering. Pending submission for SIGGRAPH 2005, 2005.

[FBG02] Sebastian Fernandez, Kativa Bala, and Donald P. Greenberg. Local illumination environments for direct lighting acceleration. In P. Debevec and S. Gibson, editors, *Rendering Techniques 2002 (Proceedings of the Thirteenth Eurographics Workshop on Rendering)*. Springer-Verlag, June 2002.

[Hec90] Paul S. Heckbert. Adaptive radiosity textures for bidirectional ray tracing. In *Computer Graphics (Proceedings on ACM SIGGRAPH 90)*, pages 145–154. ACM Press, 1990.

[Hil00] Francis J. Hill. *Computer Graphics Using OpenGL.* Prentice Hall PTR, Upper Saddle River, NJ, USA, 2000.

[HMSW04] W. Härdle, M. Müller, S. Sperlich, and A. Werwatz. *Nonparametric and Semiparametric Models.* Springer-Verlag, first edition, 2004.

[HTSG91] XiaoDong He, Kenneth E. Torrance, François Sillion, and Donald P. Greenberg. A comprehensive physical model for light reflection. *Computer Graphics*, 25(4), August 1991. Proceedings SIGGRAPH '91 in Las Vegas (USA).

[Jen96] Henrik Wann Jensen. Global Illumination Using Photon Maps. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, pages 21–30. Springer-Verlag/Wien, 1996.

[Jen01] Henrik Wann Jensen. *Realistic Image Synthesis Using Photon Mapping.* A. K. Peters, Natick, MA, 2001.

[Kaj86] James T. Kajiya. The rendering equation. In *Computer Graphics (Proceedings of ACM SIGGRAPH 86)*, pages 143–150. ACM Press, 1986.

[KMH95] Craig Kolb, Don Mitchell, and Pat Hanrahan. A realistic camera model for computer graphics. *Computer Graphics*, 29(Annual Conference Series):317–324, 1995.

[KW86] Malvin H. Kalos and Paula A. Whitlock. *Monte Carlo methods. Vol. 1: basics.* Wiley-Interscience, New York, NY, USA, 1986.

[LFTG97] Eric P. Lafortune, Sing-Choong Foo, Kenneth E. Torrance, and Donald P. Greenberg. Non-linear approximation of reflectance functions. In *SIGGRAPH*, pages 117–126, 1997.

[LW93] Eric P. Lafortune and Yves D. Willems. Bi-directional Path Tracing. In H. P. Santo, editor, *Proceedings of Third International Conference on Computational Graphics and Visualization Techniques (Compugraphics '93)*, pages 145–153, Alvor, Portugal, December 1993.

[ON95] M. Oren and S.K. Nayar. Generalization of the lambertian model and implications for machine vision. *IJCV*, 14(3):227–251, April 1995.

[Pra91] W. K. Pratt. *Digital Image Processing.* John Wiley & Sons, New York, second edition, 1991.

[SFWG04] William A. Stokes, James A. Ferwerda, Bruce Walter, and Donald P. Greenberg. Perceptual illumination components: a new approach to efficient, high quality global illumination rendering. *ACM Transactions on Graphics*, 23(3):742–749, 2004.

[Sip01] Michael Sipser. *Introduction to the Theory of Computation.* PWS Publishing Company, 20 Park Plaza, Boston, MA 02116, 2001.

[SW99] F. Suykens and Y. D. Willems. Weighted multipass methods for global illumination. In *Computer Graphics Forum (Proc. Eurographics '99)*, volume 18, pages C–209–C–220, September 1999.

[VG97] Eric Veach and Leonidas J. Guibas. Metropolis light transport. In *Proceedings of ACM SIGGRAPH 1997*, volume 31, pages 65–76, 1997.

[WH92] Greg Ward and Paul Heckbert. Irradiance gradients. In *Eurographics Rendering Workshop*, pages 85–98, May 1992.

[WTP01] Alexander Wilkie, Robert F. Tobler, and Werner Purgathofer. Combined rendering of polarization and fluorescence effects. Technical Report TR-186-2-01-11, Institute of Computer Graphics and Algorithms, Vienna University of Technology, apr 2001.