Path tracen op de GPU: betere lichteffecten voor computerspellen?

Frank Erens

Academiejaar: 2013-2014

Computertechnologie staat niet stil. Vanaf de allereerste computer die een volledige kamer in beslag nam tot de smartphone in je zak: kleiner, sneller en goedkoper lijkt het motto. We zien dit duidelijk als we de spelconsole uit de jaren '80 – die nauwelijks 16 kleuren tegelijk konden weergeven – vergelijken met de hedendaagse computer, die miljoenen berekeningen tegelijk aan kan.

Processoren in computers worden steeds sterker. Maar is dit de enige manier om dingen te versnellen? Zouden we niet iets kunnen doen met die zwakkere processoren – die nu spotgoedkoop zijn?

De komst van de GPU

We zouden een stel kleine, trage processoren kunnen nemen en die allemaal samenvoegen tot één geheel. Deze processoren kunnen dan allemaal werken aan dezelfde taak. Jammer genoeg gaat dat enkel voor taken die makkelijk op te splitsen zijn over de processoren. Wat zijn dan zulke taken?

Het blijkt nu dat grafische taken – zoals het tekenen van 3D-beelden in computer games – enorm goed geschikt zijn voor dit. Moderne grafische processoren (ook wel *GPUs* genoemd, van het Engelse *Graphical Processing Unit*) zijn dan ook op deze manier opgebouwd: een verzameling van honderden, al niet duizenden, kleine goedkope processoortjes die allemaal in parallel draaien.

Het interessante hieraan is dat moderne GPUs *programmeerbaar* zijn: ze zijn niet enkel beperkt tot de taak van het tekenen van 3D beelden, we kunnen zelf nieuwe dingen bedenken die dan worden uitgevoerd door alle processoren. GPUs zijn zo niet langer gelimiteerd tot één vaste taak, maar kunnen gelijk wat doen.

Licht in de duisternis

Hedendaagse computerspellen maken gebruik van krachtige GPUs om indrukwekkende scenes te tekenen. Maar nog niet alles ziet er perfect uit. Schaduwen en reflecties zien er vaak lelijk uit, met harde randen of rare vormen. Dikwijls

zijn ze er zelfs gewoon niet, of enkel voor stilstaande figuren, zoals gebouwen in de achtergrond. Transparante voorwerpen zoals glazen zijn er gewoon niet of zien er nep uit. Kan het beter?

In de echte wereld zien we dingen doordat ze belicht worden. Geen licht – geen beeld. Lichtstralen komen uit een lamp, worden weerkaatst tussen allerlei oppervlakten, en komen dan uiteindelijk in je oog aan. We kunnen een programma schrijven dat het pad volgt die de lichtstralen door de 3D wereld nemen. Voor iedere straal wordt de richting en de kleur bijgehouden, en uiteindelijk wordt zo het beeld op je scherm gevormd. Dit is precies hetzelfde als wat er gebeurt in de echte wereld. Dit betekent dat we elk lichteffect waarheidsgetrouw kunnen nabootsten!

Maar niet te vroeg gejuicht. Voor al die lichtstralen te simuleren is er enorm veel rekenwerk nodig – zoveel dat de meeste games tot nu toe het niet hebben geprobeerd en op simpelere, maar visueel slechtere methoden zijn teruggevallen.

Iedere lichtstraal is onafhankelijk van iedere andere lichtstraal. Dit is prima geschikt voor een GPU, met al zijn individuele processoren. We kunnen dus de GPU nu herprogrammeren, zodat in plaats van zijn gewoonlijke methode van tekenen, deze nu de lichtmethode gaat gebruiken. Dat laat ons toe om snel en efficient beelden te krijgen die er veel mooier uitzien dan met de traditionele methodes.

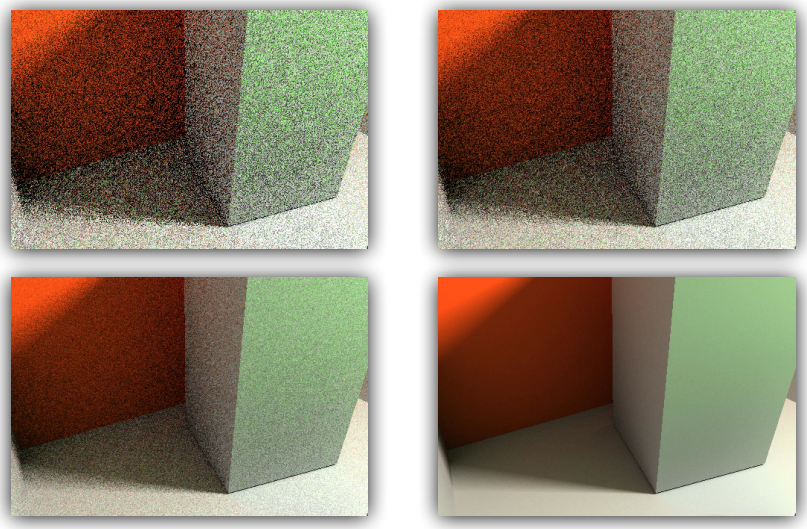
Storing in het signaal

Als het licht op een mat oppervlak komt, wordt het willekeurig rondgestrooid in alle richtingen. Je ziet dan het gemiddelde van al die stralen. In het echt is dat natuurlijk geen probleem: licht gaat zo snel dat het lijkt alsof je al het licht uit alle richtingen tegelijk binnenkrijgt. Een computer kan maar een beperkt aantal lichtstralen tegelijk simuleren.

Dit zorgt voor ruis. Hoe trager je computer, hoe langer het duurt voor dat de ruis wegtrekt. Iedere keer als je beweegt begint dit opnieuw. Afhankelijk van hoe complex de 3D-wereld is – met veel oppervlakken en veel lichtbronnen – kan dit van enkele seconden tot enkele minuten duren. Niet ideaal voor spellen.

We kunnen een paar trucjes toepassen om het proces te versnellen en zo de ruis ook te doen verminderen. Eén zo'n techniek is door de 3D-wereld op te delen in stukken. We delen de wereld in twee: als een straal maar door één helft gaat, kunnen we de andere helft buiten beschouwing laten. We hebben net het werk gehalveerd! Dit proces van opdelen in twee – iedere keer in kleinere stukjes – kunnen we blijven herhalen, tot op een bepaald moment de stukjes zo klein zijn dat verder opdelen de prestatie niet meer verbetert.

We kunnen het zelfs slimmer aanpakken: splits de wereld zo op dat er steeds zo veel mogelijk lege ruimte wordt afgezonderd. In de meeste 3D werelden is er veel lege ruimte: je kan immers niet door een blok vast beton gaan lopen! Een straal die door een lege ruimte gaat, raakt niets. We kunnen deze dan ook gewoon uit de berekeningen laten, wat weer een hoop werk voor die arme GPU scheelt.



Ruis is een ernstig probleem bij path tracen. Er bestaan oplossingen, maar nog geen die de ruis helemaal doet verdwijnen.

Helaas kunnen we op dit moment de ruis nog niet volledig wegkrijgen. In de toekomst – wanneer er vast en zeker snellere GPUs beschikbaar zijn – gaat het probleem van de ruis waarschijnlijk zichzelf verhelpen. Tot die tijd moeten we het dan nog maar doen met die lelijke schaduwen in onze games.