

Ray-Tracing: Het Ontwerpen van Realistische Lichtsimulaties

Een Profielwerkstuk

Door Sytze Reitsma

onder begeleiding van H. Mulder, MSc

Zaanlands Lyceum

Inhoudsopgave

Introductie: Wat is ray-tracing?

Doelstelling: Wat moet een ray-tracer kunnen?

Theorie: Welke wiskunde ligt achter de ray-tracer?

Toepassing: Hoe is de raytracer gemaakt?

Conclusie

Discussie

Introductie: Wat is ray-tracing?

Wat is ray-tracing? - Introductie

- Special FX
- 3D-animatie

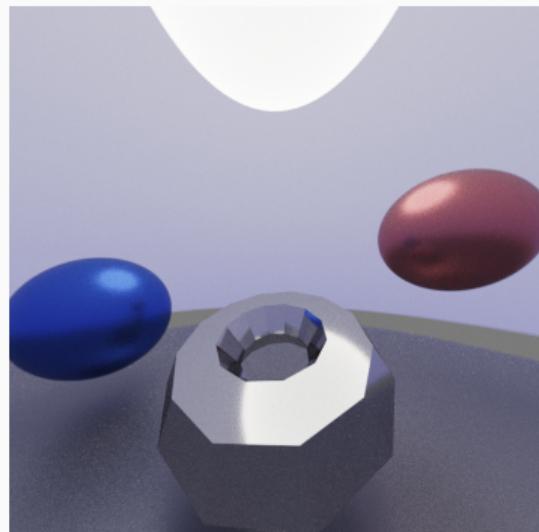
- Special FX
 - 3D-animatie



Wat is ray-tracing? - Invoer en Uitvoer

We willen een programma, dat:

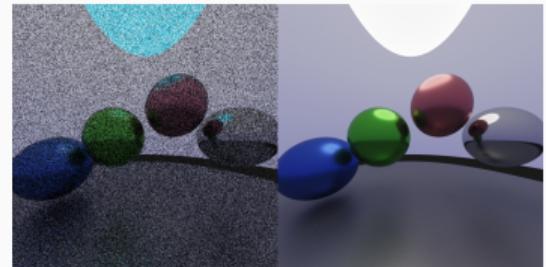
- Van ruimte met objecten naar een 2D-beeld gaat,
- vanuit materiaaleigenschappen, locaties en vormen
- We krijgen een foto (tabel met een kleur voor elke pixel, bijvoorbeeld 720x720)



Wat is ray-tracing? - Lichtsimulatie

Licht gedraagt zich als bewegend deeltje: een foton

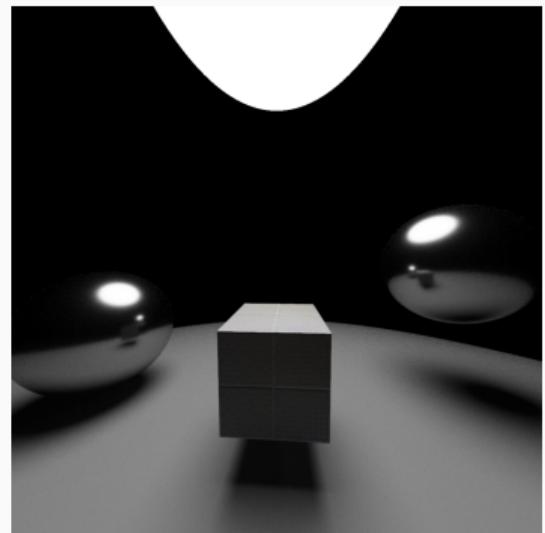
- Fotonen bewegen van een bron naar een kijker
- Fotonen bevatten een kleur
- Fotonen bewegen in een rechte lijn en kunnen reflecteren
- We moeten het pad van veel fotonen (=rays) traceren



*Lage vs. hoge hoeveelheid
getraceerde fotonen*

Wat is ray-tracing?

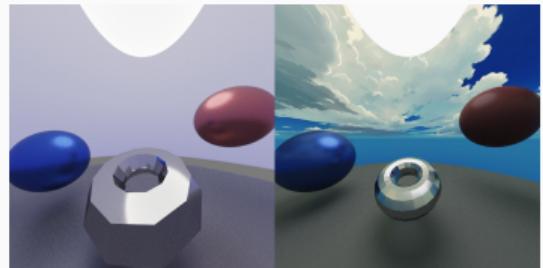
- Simulatie van lichtinval op een camera in een ruimte
- Fotonen van *camera* → *lichtbron*
- Invoer: Ruimtelijke informatie van objecten, en materiaaleigenschappen
- Uitvoer: 2D-'foto' van een virtuele ruimte



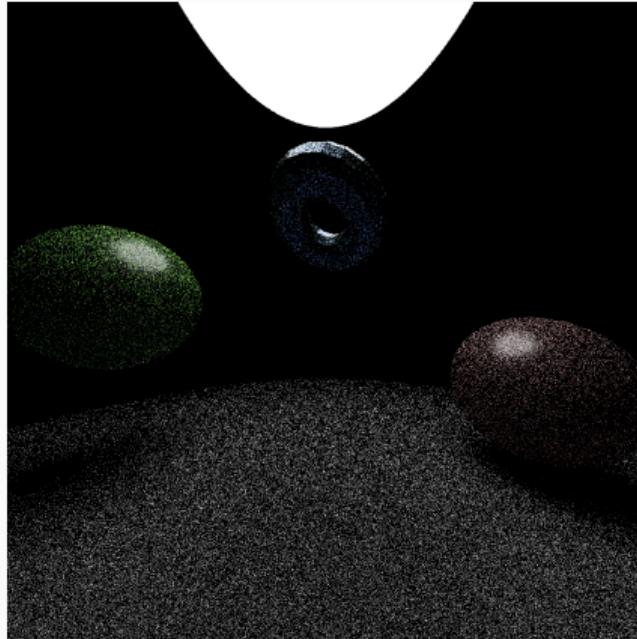
Doelstelling: Wat moet een ray-tracer kunnen?

Wat moet een ray-tracer kunnen?

1. Ondersteuning invoer ruimtelijke informatie
2. Ondersteuning lichtreflectie
3. Modulair en duidelijk te gebruiken
4. Optioneel: meer detail, realistischer licht, snelheidsoptimalisatie

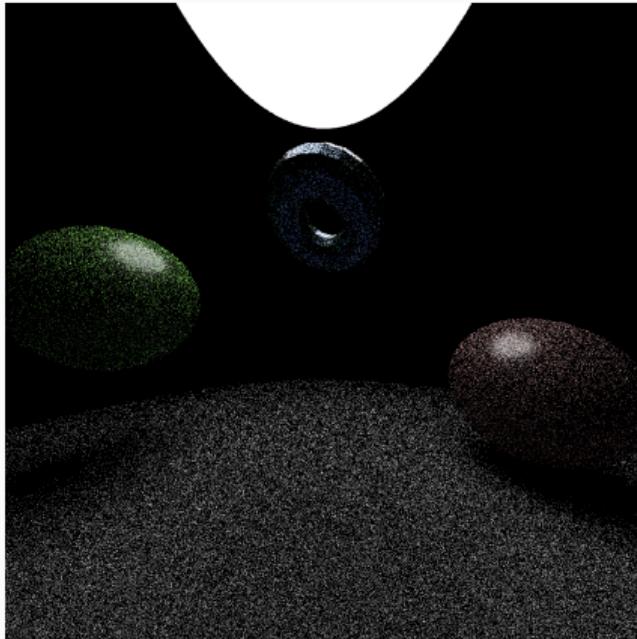


Sampling



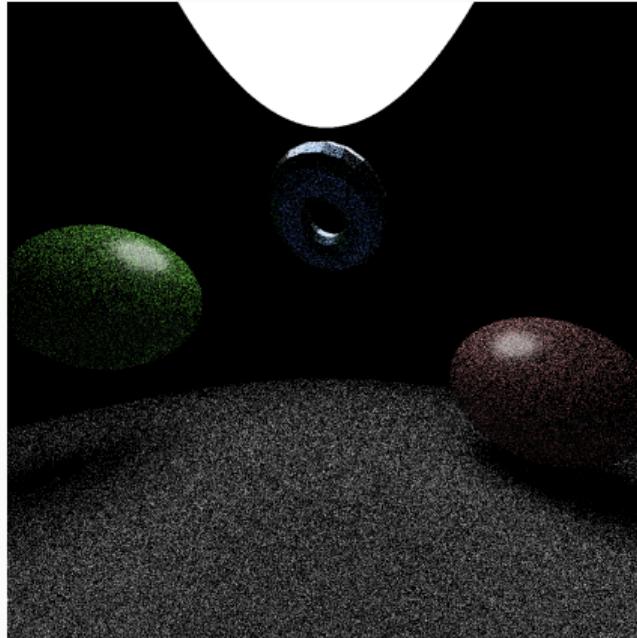
4 samples, $\sigma = 0.5\sigma_0$

Sampling



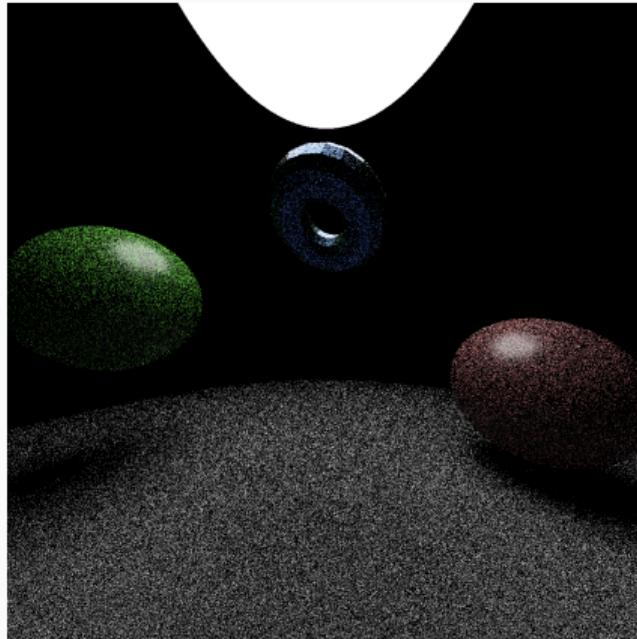
5 samples, $\sigma = 0.45\sigma_0$

Sampling



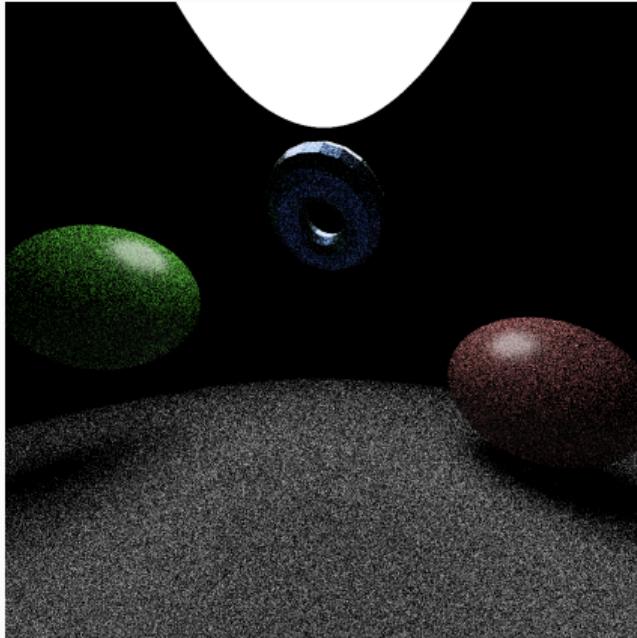
6 samples, $\sigma = 0.4\sigma_0$

Sampling



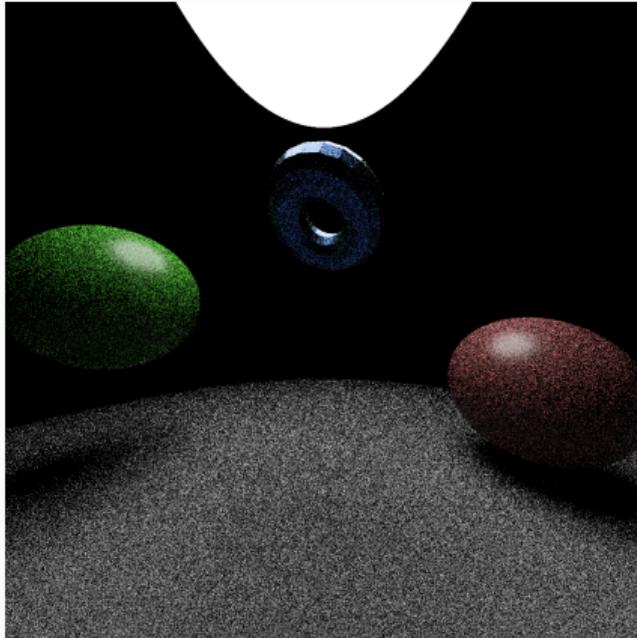
8 samples, $\sigma = 0.35\sigma_0$

Sampling



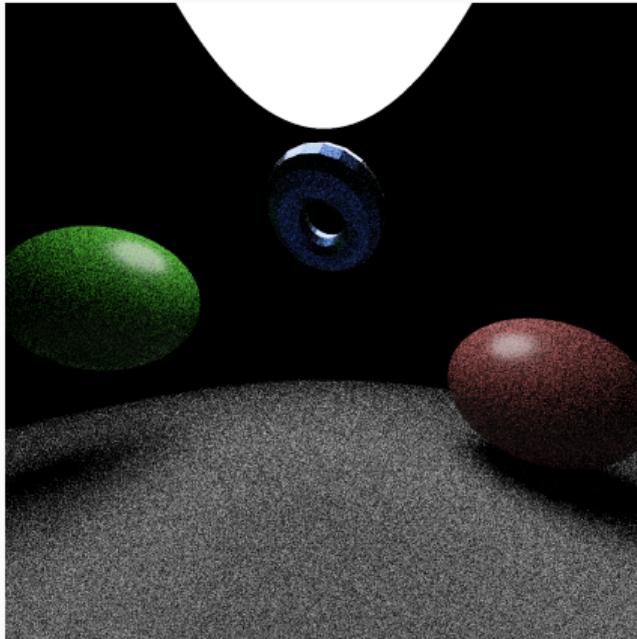
11 samples, $\sigma = 0.3\sigma_0$

Sampling



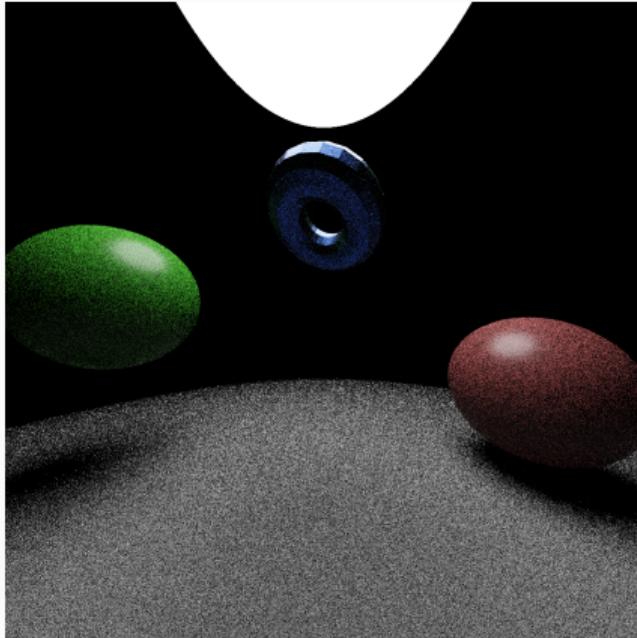
16 samples, $\sigma = 0.25\sigma_0$

Sampling



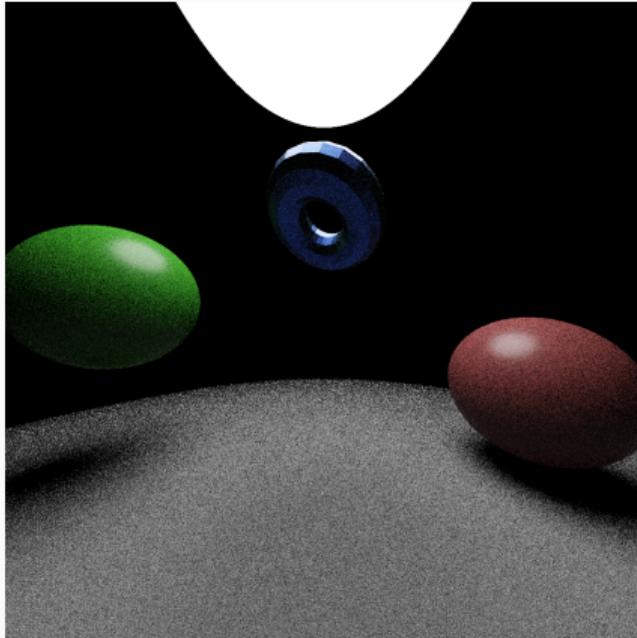
25 samples, $\sigma = 0.2\sigma_0$

Sampling



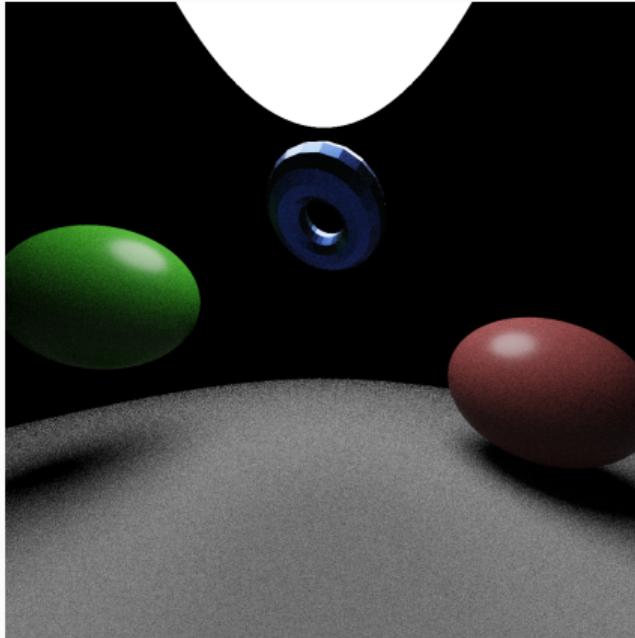
44 samples, $\sigma = 0.15\sigma_0$

Sampling



100 samples, $\sigma = 0.1\sigma_0$

Sampling



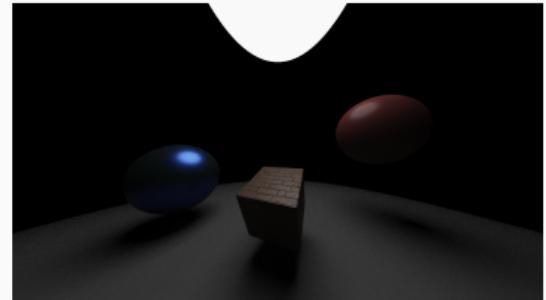
400 samples, $\sigma = 0.05\sigma_0$

Theorie: Welke wiskunde ligt achter de ray-tracer?

Welke wiskunde ligt achter de ray-tracer? - Overzicht

Een renderer moet een paar dingen kunnen:

1. Het snijpunt vinden tussen een lichtstraal en een object
2. Voorspellen wat de kans is dat een lichtstraal een bepaalde kant op stuiter, aan de hand van de materiaalsoort en de invalshoek
3. Weten hoe materiaalkleur kleur van een foton verandert



Welke wiskunde ligt achter de ray-tracer? - Snijding

*Snijdt een lichtstraal een driehoek? En zo ja,
waar?*

$$\mathbf{d}_1 = \mathbf{p}_2 - \mathbf{p}_0$$

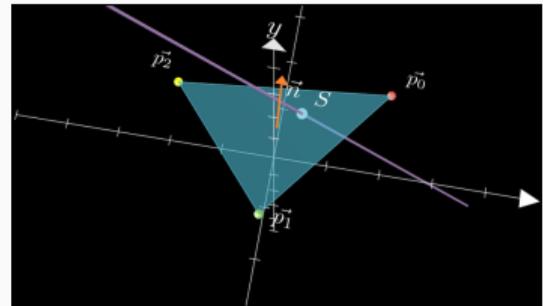
$$\mathbf{d}_2 = \mathbf{p}_1 - \mathbf{p}_0$$

$$\mathbf{n} = \mathbf{d}_1 \times \mathbf{d}_2$$

$$V : \langle \mathbf{n}, \mathbf{s} \rangle + d = 0$$

$$d = -\langle \mathbf{n}, \mathbf{p}_0 \rangle$$

$$V : \langle \mathbf{n}, \mathbf{s} \rangle - \langle \mathbf{n}, \mathbf{p}_0 \rangle = 0$$



Welke wiskunde ligt achter de ray-tracer? - Snijding

$$\mathbf{s} = \mathbf{o} + \mathbf{d}t$$

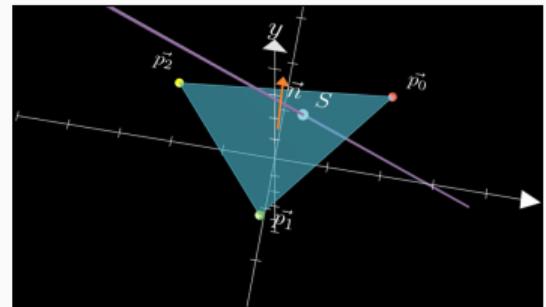
$$\langle \mathbf{n}, (\mathbf{o} + \mathbf{d}t) \rangle - \langle \mathbf{n}, \mathbf{p}_0 \rangle = 0$$

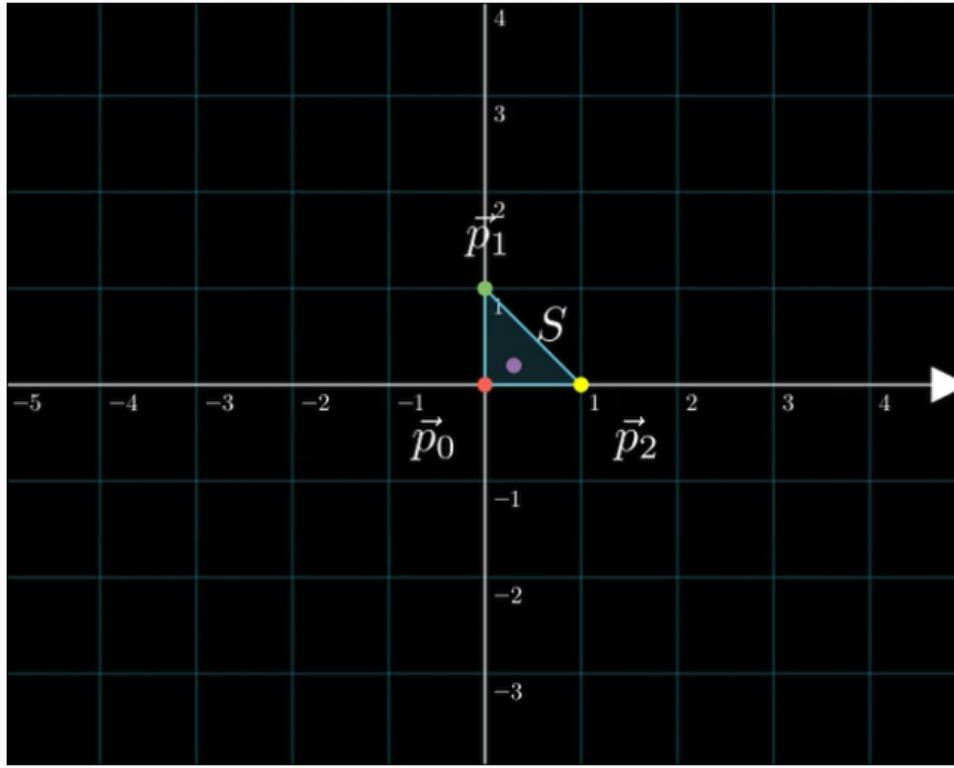
$$t_h = -\frac{\langle \mathbf{n}, \mathbf{o} \rangle - \langle \mathbf{n}, \mathbf{p}_0 \rangle}{\langle \mathbf{n}, \mathbf{d} \rangle}$$

$$\mathbf{S} = \mathbf{o} + \mathbf{d}t_s$$

$$M = \begin{pmatrix} \mathbf{d}_1 & \mathbf{d}_2 & \mathbf{n} \end{pmatrix}$$

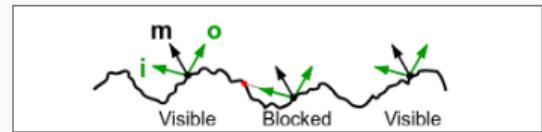
$$\mathbf{S}' = (\mathbf{S} - \mathbf{p}_0) \cdot M^{-1}$$





Welke wiskunde ligt achter de ray-tracer? - BRDF

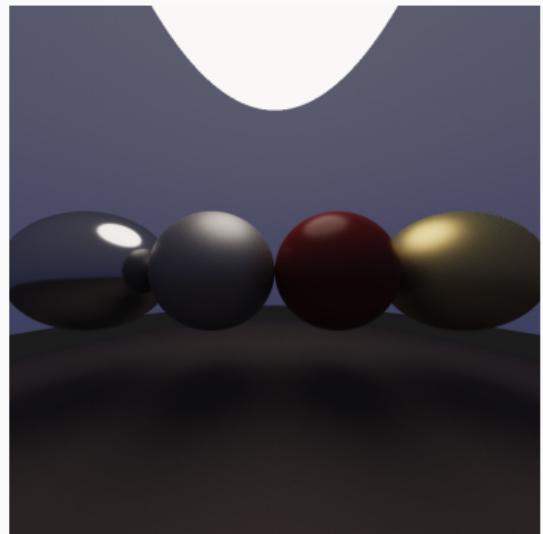
1. Omzetten attributen
materiaal → kansverdeling
2. Kiezen willekeurige richting aan de hand
hiervan



Welke wiskunde ligt achter de ray-tracer? - Lichtaccumulatie

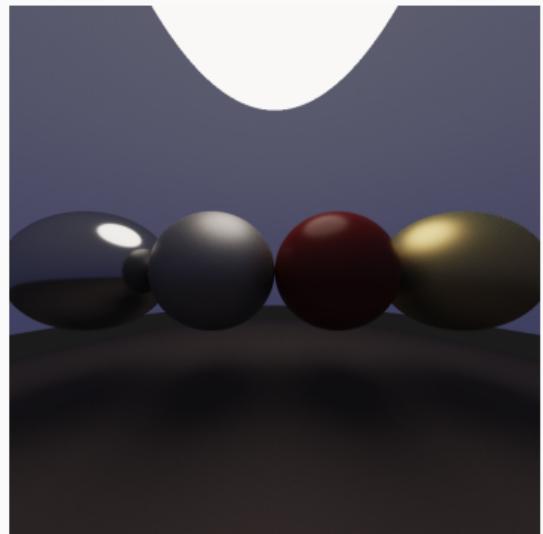
$$\mathbf{f} = \begin{pmatrix} \text{roodfactor} \\ \text{groenfactor} \\ \text{blauwfactor} \end{pmatrix}$$

$$\mathbf{L}_{\text{camera}} = \mathbf{L}_{\text{bron}} \circ \mathbf{f}_1 \circ \mathbf{f}_2 \cdot \dots$$



Welke wiskunde ligt achter de ray-tracer? - Lichtaccumulatie

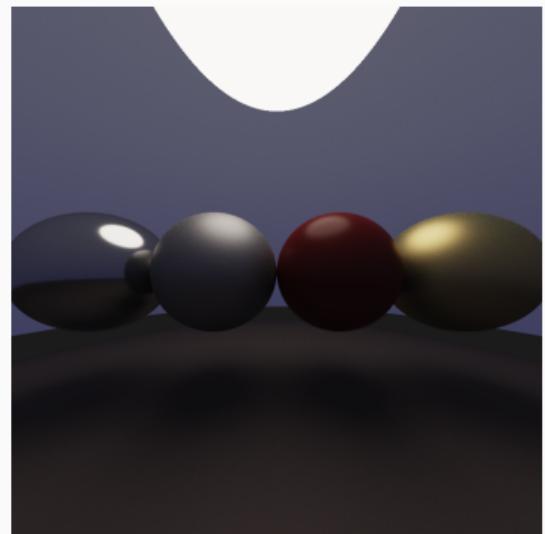
Voorbeeld: Een witte lamp schijnt via een rode bol en een grijze tafel op een camera.



Welke wiskunde ligt achter de ray-tracer? - Lichtaccumulatie

Voorbeeld: Een witte lamp schijnt via een rode bol en een grijze tafel op een camera.

$$\mathbf{L}_{\text{bron}} = \begin{pmatrix} 1.0 \\ 1.0 \\ 1.0 \end{pmatrix}, \mathbf{f}_{\text{bol}} = \begin{pmatrix} 1.0 \\ 0.0 \\ 0.0 \end{pmatrix}, \mathbf{f}_{\text{tafel}} = \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix}$$



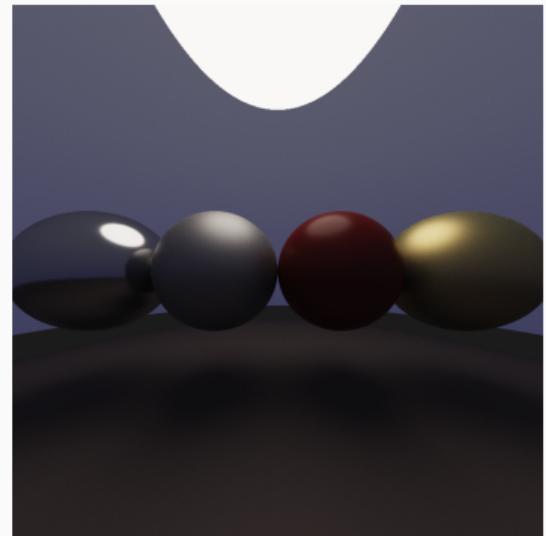
Welke wiskunde ligt achter de ray-tracer? - Lichtaccumulatie

$$\mathbf{L}_{\text{camera}} = \mathbf{L}_{\text{bron}} \circ \mathbf{f}_{\text{bol}} \circ \mathbf{f}_{\text{tafel}}$$

$$= \begin{pmatrix} 1.0 \\ 1.0 \\ 1.0 \end{pmatrix} \circ \begin{pmatrix} 1.0 \\ 0.0 \\ 0.0 \end{pmatrix} \circ \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix}$$

$$= \begin{pmatrix} 1.0 \\ 0.0 \\ 0.0 \end{pmatrix} \circ \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix}$$

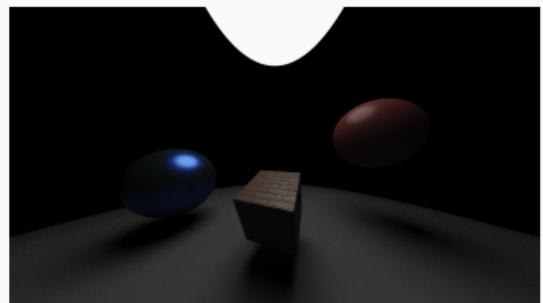
$$= \begin{pmatrix} 0.5 \\ 0.0 \\ 0.0 \end{pmatrix}$$



Toepassing: Hoe is de raytracer gemaakt?

Hoe is de raytracer gemaakt?

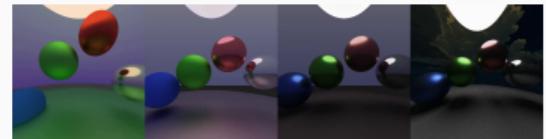
- Python op de CPU
- GLSL op de GPU
- main.py, shader.frag, conf_*.py



Conclusie

Conclusie

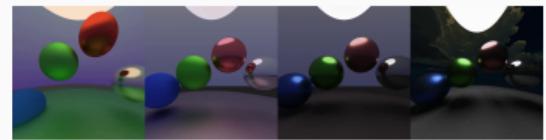
- Een raytracer geschreven in GLSL en Python
- Samenstelling van veel ingewikkelde algoritmen
- De doelen zijn bereikt, en verbeteringen er voorbij zijn toegevoegd



Discussie

Discussie

- Eigen interesse: goed, maar met lichte nadelen
- Onderzoeks methode had formeler kunnen zijn



Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

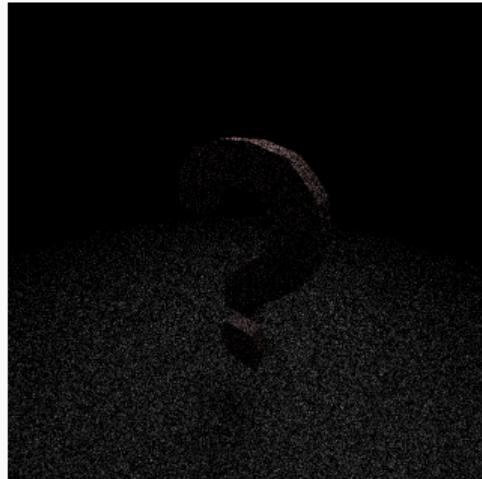
Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

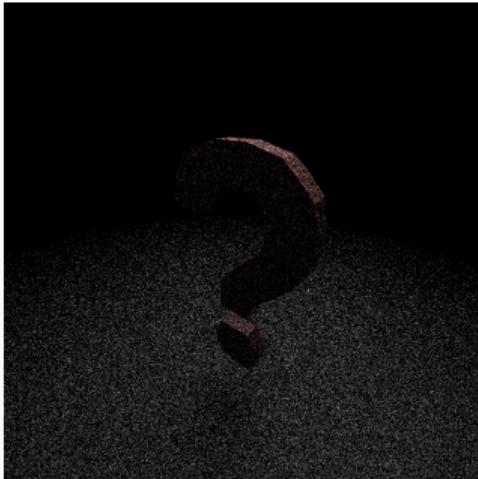
Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

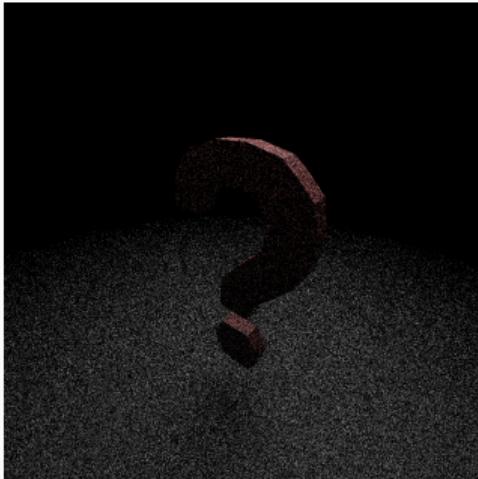
Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

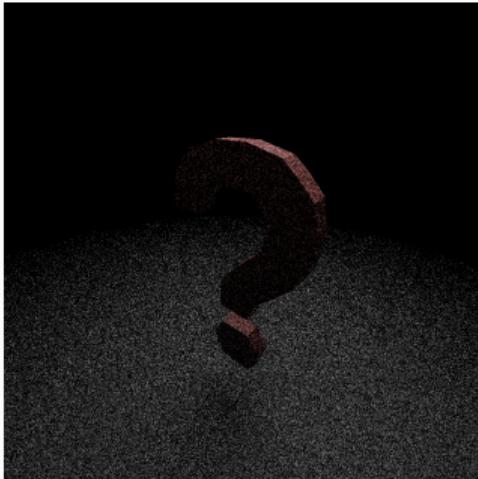
Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

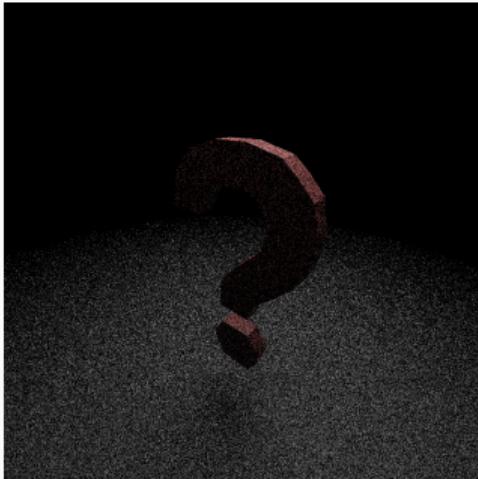
Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

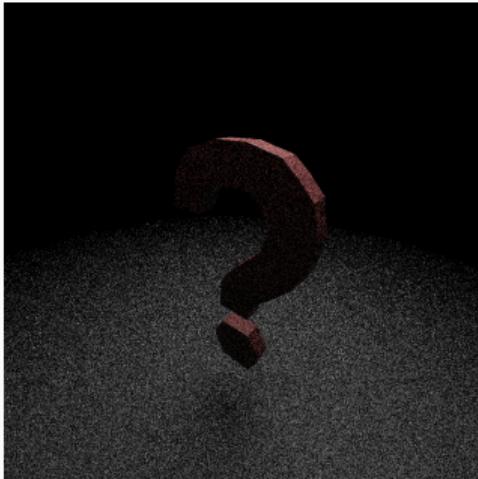
Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

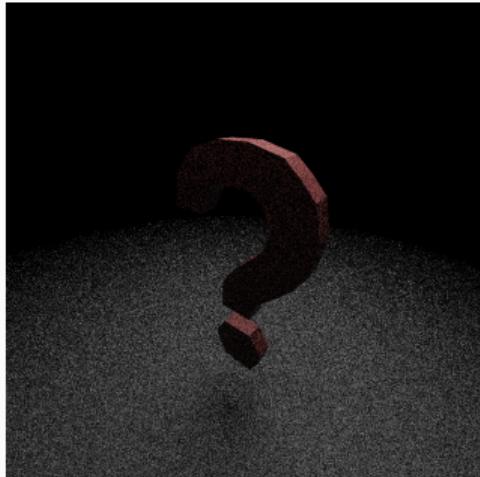
Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

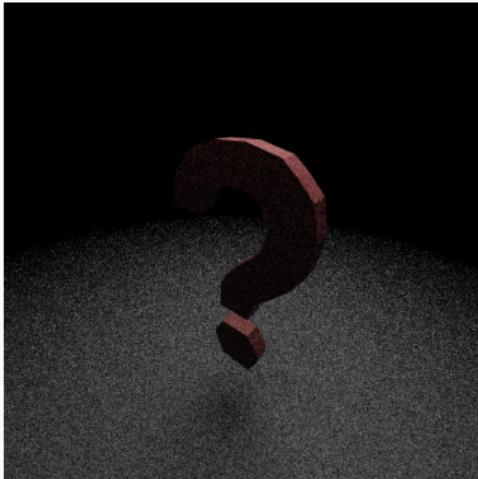
Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

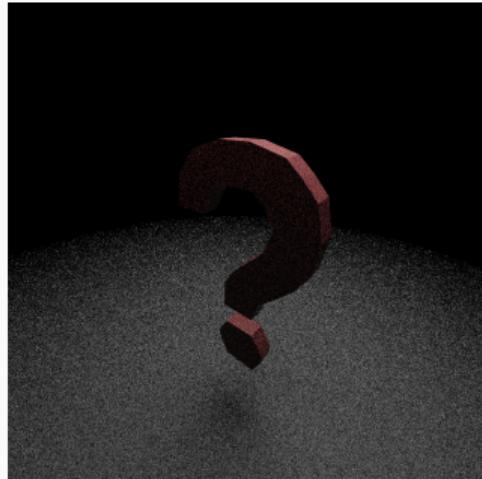
Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>

Vragen?



Vragen?

<https://github.com/BBernYY/PWS>