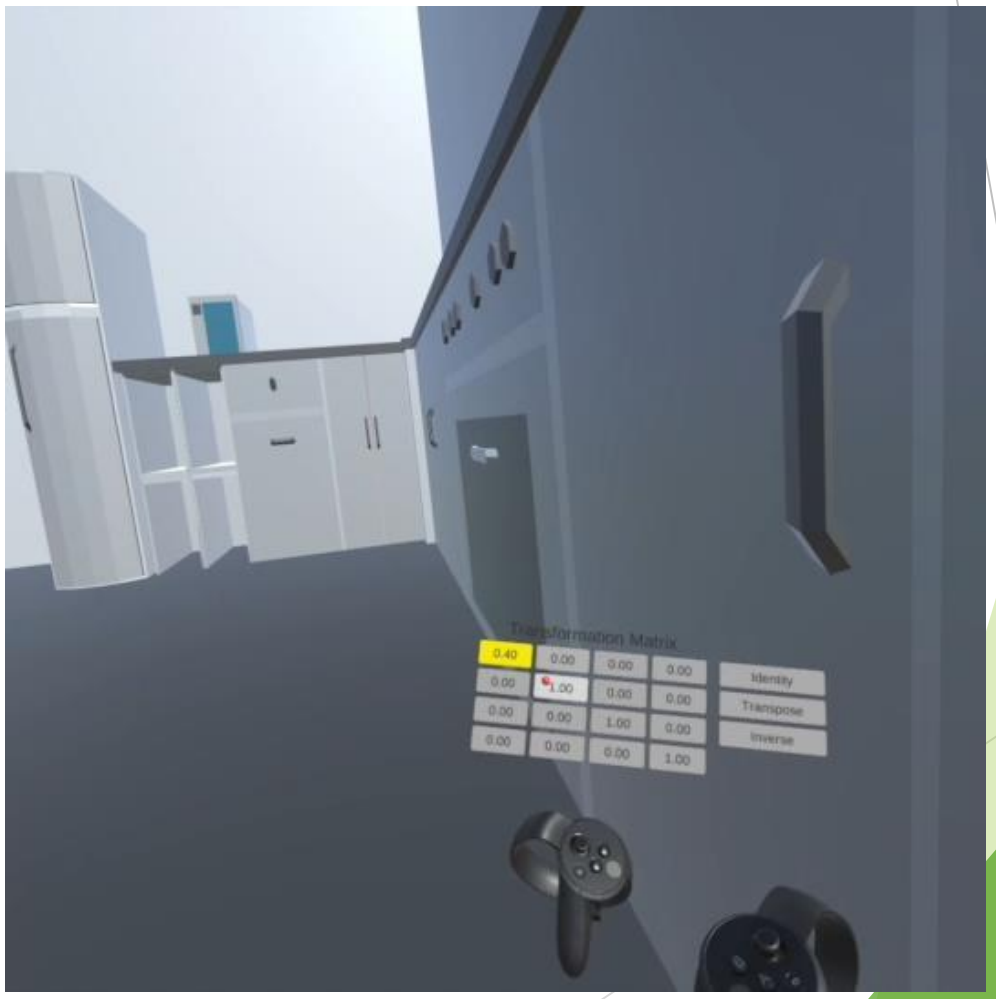
VR Experiment: grafische pijplijn in VR

- Testpersonen gezocht
- A.s. vrijdag 23 september in Visualisation Lab tussen 09:00 en 17:00
- 15 minuten max
- Schrijf in via dit formulier!



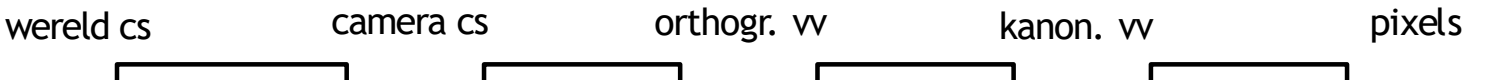
Graphics en Game Technologie

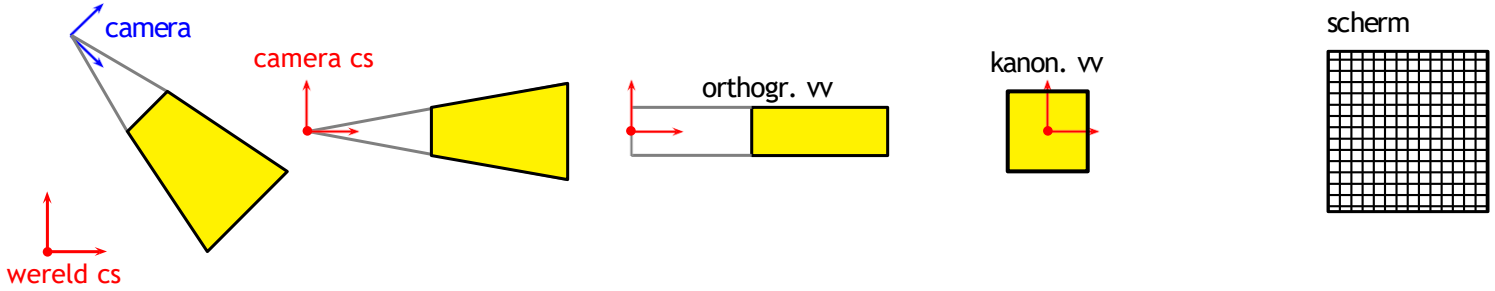
6. Belichting en raytracing

Robert Belleman
Computational Science Lab
Universiteit van Amsterdam
R.G.Belleman@uva.nl
(voeg a.u.b. “[GGT]” toe aan subject)

Pixar’s Luxo and Jr.

Grafische pijplijn





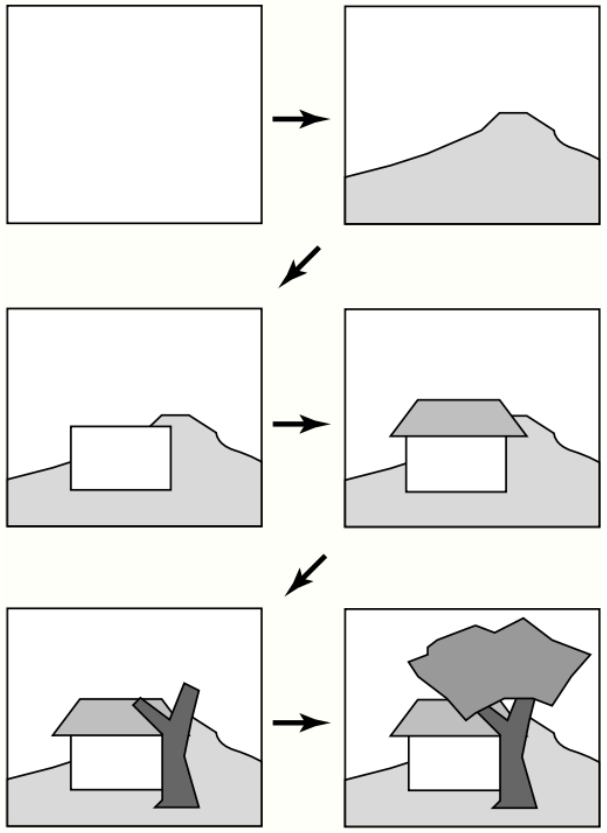
$$\begin{pmatrix} x_u & y_u & z_u & 0 \\ x_v & y_v & z_v & 0 \\ x_w & y_w & z_w & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -x_e \\ 0 & 1 & 0 & -y_e \\ 0 & 0 & 1 & -z_e \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & -fn \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{2}{\mathrm{r}-\mathrm{l}} & 0 & 0 & 0 \\ 0 & \frac{2}{\mathrm{t}-\mathrm{b}} & 0 & 0 \\ 0 & 0 & \frac{2}{\mathrm{n}-\mathrm{f}} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -\frac{l+r}{2} \\ 0 & 1 & 0 & -\frac{b+t}{2} \\ 0 & 0 & 1 & -\frac{n+f}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{n_x}{2} & 0 & 0 & \frac{n_x-1}{2} \\ 0 & \frac{n_y}{2} & 0 & \frac{n_y-1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Overzicht

- Occlusie: painter's algoritme en z-buffer
- Belichtingsmodellen
  - Diffuse reflectie en omgevingslicht
  - Flat shading en Gouraud shading
  - Spiegel reflectie
  - Phong shading
- Ray tracing en recursieve ray tracing

Occlusie

Painter's algoritme: van back-to-front



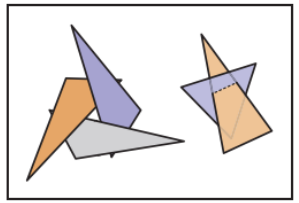


Figure 8.9. Two occlusion cycles, which cannot be drawn in back-to-front order.

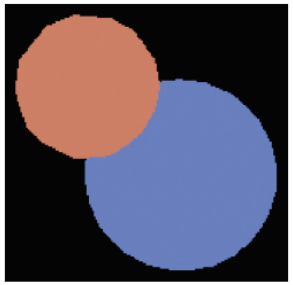
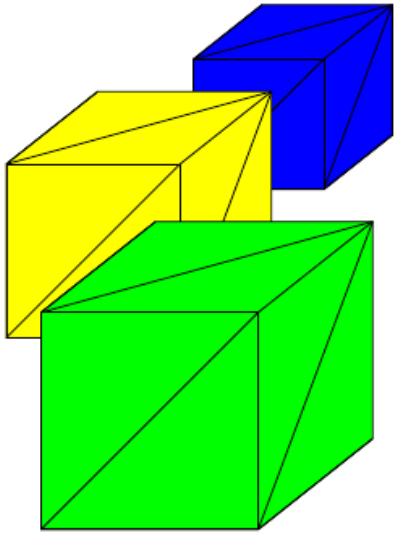


Figure 8.10. The result of drawing two spheres of identical size using the minimal pipeline. The sphere that appears smaller is farther away but is drawn last, so it incorrectly overwrites the nearer one.

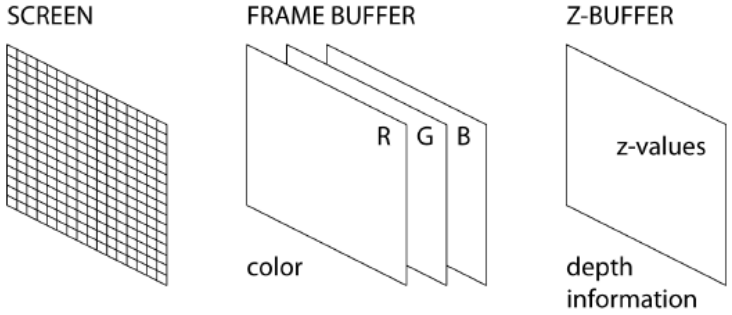
z-buffer of diepte-buffer

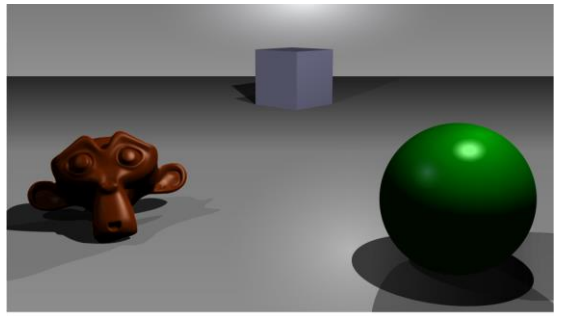
- z-buffer vaak gebruikt voor hidden surface removal
- Extra opslag van diepte informatie



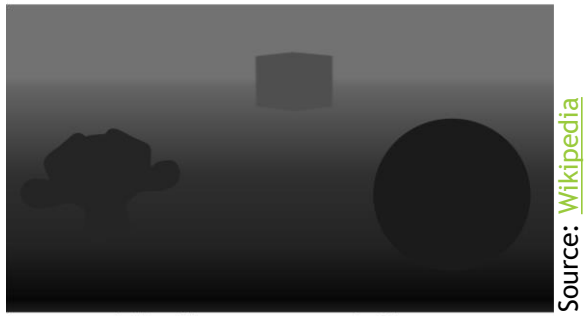
z-buffer algoritme: basis idee

Naast frame buffer (pixels met kleurinformatie) ook een z-buffer van dezelfde grootte, om diepte informatie bij te houden





A simple three-dimensional scene

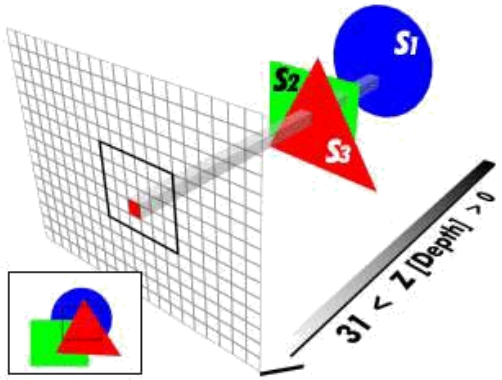


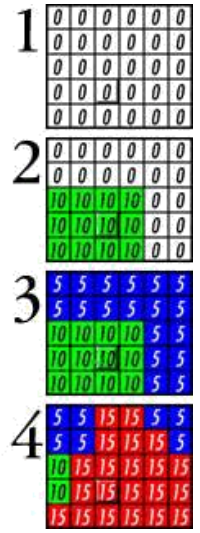
Z-buffer representation

Source: Wikipedia

z-buffer algoritme

- z-buffer algoritme:
  1. zet achtergrond kleur in rgb-buffer en verste z-waarde in z-buffer
  2. voor ieder polygoon: rasterizeer
     1. voor elk pixel:
        - als z-waarde dichter bij camera is dan opgeslagen z-waarde
          - zet nieuwe kleur in rgb-buffer en nieuwe z-waarde in z-buffer



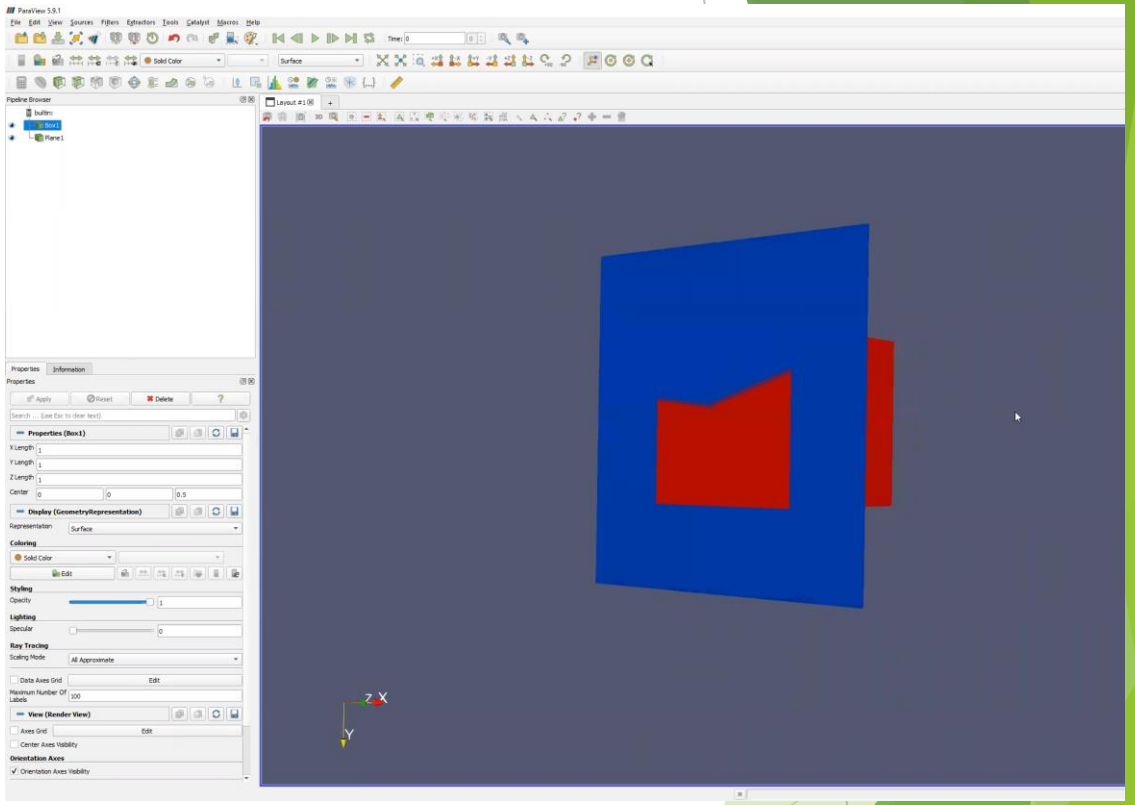


Source

- z-waarden voor hoekpunten berekend, voor overige pixels geïnterpoleerd

z-buffer: voor- en nadelen

1. Veel geheugen nodig om z-waarden te bewaren
2. Kan in hardware geïmplementeerd worden
3. Geen limiet aan aantal polygonen
4. Implementatie heel simpel
5. Kan geen transparante oppervlakken aan
6. Beperkte resolutie van de z-buffer kan leiden tot "z-fighting"



"Z-fighting": coplanar polygons that have near-similar or identical values in the z-buffer.

z-buffer versus ray tracing

z-buffer ("object-order" rendering)

```
for each object
  for each pixel inside objects projection
    if depth is closer than z-buffer value
      update color buffer with objects shading
      update z-buffer
```

ray tracing ("image-order" rendering)

```
for each pixel
  for each object
    if ray through pixel intersects object
    and is closer than saved depth
      compute shading
      update color
```

Belichtingsmodellen

Belichtingsmodellen

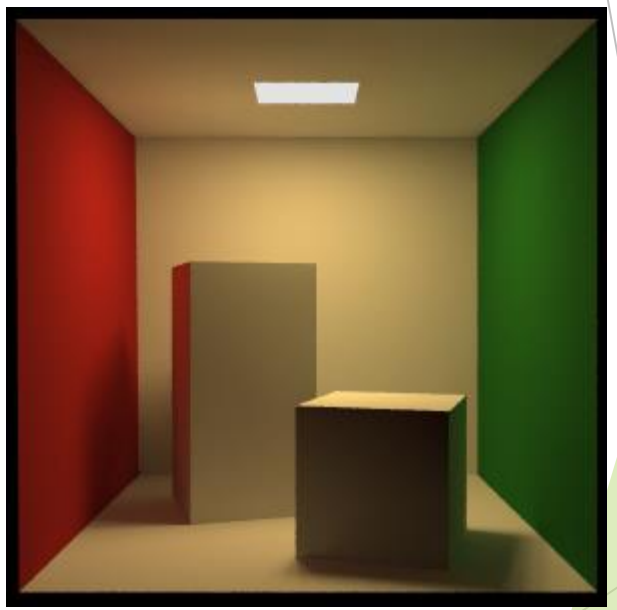
Belichtingsmodel berekent lichteffecten van oppervlak gebruikmakend van optische eigenschappen van oppervlak, zoals:

- transparantie
- kleurweerspiegelingscoëfficiënten
- textuur-parameters

en is afhankelijk van geometrie:

- richting van lichtbron
- richting van oog waarnemer
- oriëntatie (normaal) van oppervlak

Belichtingsmodel berekent hoeveel licht verspreid of weerspiegeld wordt en ons oog bereikt vanaf elk deel van een oppervlak.



The Cornell Box
Source

Lichtbronnen en oppervlak

Twee typen lichtbronnen:

- puntlichtbronnen
- omgevingslicht (ambient)

Licht schijnt op oppervlak en wordt

- deels geabsorbeerd (mat oppervlak)
- deels weerspiegeld (glimmend oppervlak)
- deels doorgelaten (transparant oppervlak)

Deel van het licht dat weerspiegeld wordt en het oog bereikt is wat we zien.

Weerspiegeld licht

Licht van lichtbron kan weerspiegeld worden:

- diffuse reflectie: uniform verspreid in alle richtingen
- spiegelreflectie: vooral in één richting verspreid

Meeste oppervlakken combinatie van 2 typen van reflecties: totale weerspiegelde licht is som van diffuse component en spiegelcomponent.

Voor elk punt op oppervlak berekenen we de grootte van elke component die ons oog bereikt.

1. Diffuse of Lambertian reflectie

- licht van puntlichtbron valt op deel oppervlak
- deel van licht wordt weerspiegeld in alle richtingen evenveel
- deel daarvan bereikt ons oog met kleur c

Mat oppervlak ziet er vanaf alle richtingen even helder uit, omdat licht in alle richtingen evenveel gereflecteerd wordt

1. Diffuse of Lambertian reflectie

Lambertian cosine law:

kleur c van oppervlak is evenredig met hoek tussen normaal $\boldsymbol{n}$ van het oppervlak en richting $\boldsymbol{l}$ van de lichtbron (directional light)

$$c \sim \cos\theta \text{ of } \quad c \sim \boldsymbol{n} \cdot \boldsymbol{l}$$

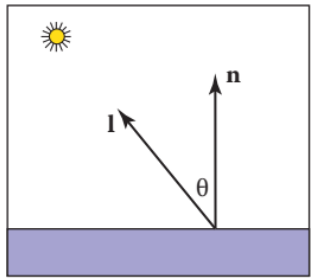


Figure 10.1. The geometry for Lambert's law. Both **n** and **l** are unit vectors.

1. Diffuse of Lambertian reflectie

Voor Lambertian oppervlak is hoeveelheid licht gezien door viewer:

- onafhankelijk van view
- evenredig met $\cos\theta$

Diffuse belichtingsvergelijking:

$$c = c_r c_l \boldsymbol{n} \cdot \boldsymbol{l} = c_r c_l \cos\theta$$

c is waargenomen kleur

c_r is diffuse-reflectie-coëfficiënt van oppervlak (tussen 0 en 1)

c_l is intensiteit van puntlichtbron (constant voor alle voorwerpen)

θ is hoek van inval (tussen 0° en 90°)

Kleur c tussen 0 en 1

$$c = c_r c_l \boldsymbol{n} \cdot \boldsymbol{l}$$

Wat als $\boldsymbol{n} \cdot \boldsymbol{l}$ negatief?

Als normaalvector in punt in andere richting, dan $\boldsymbol{n} \cdot \boldsymbol{l}$ negatief

In dit geval: $c = c_r c_l \max(0, \boldsymbol{n} \cdot \boldsymbol{l})$

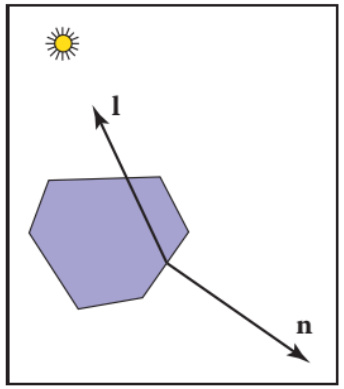


Figure 10.2. When a surface points away from the light, it should receive no light. This case can be verified by checking whether the dot product of **l** and **n** is negative.

2. Omgevingslicht of ambient light

Probleem: punt met normaal weg van lichtbron is zwart

- diffuse reflectie: zwarte schaduwen die onrealistisch en hard zijn

Oplossing: voeg constante kleur toe aan elk object

- schaduwen verzachten met omgevingslicht of ambient light
- achtergrondlichtbron die in alle richtingen uniform licht verspreidt

c_a is intensiteit achtergrondlichtbron (constant voor alle voorwerpen)

2. Omgevingslicht of ambient light

Ambiente component c_a toevoegen aan belichtingsvergelijking:

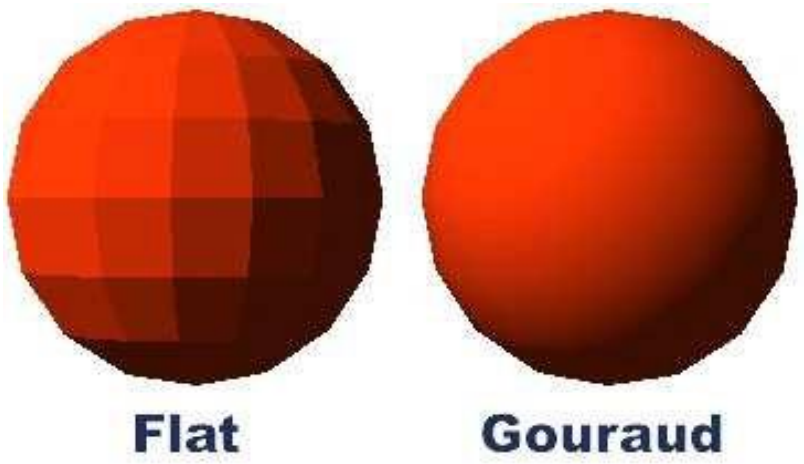
$$c = c_r(c_a + c_l \max(0, \boldsymbol{n} \cdot \boldsymbol{l}))$$

c de resulterende intensiteit ten gevolge van ambiente en diffuse component

c_a door experimenten bepaald, correspondeert niet met fysische eigenschap

c_r is diffuse-reflectie-coëfficiënt van oppervlak (tussen 0 en 1)

Flat shading en Gouraud shading



Flat shading

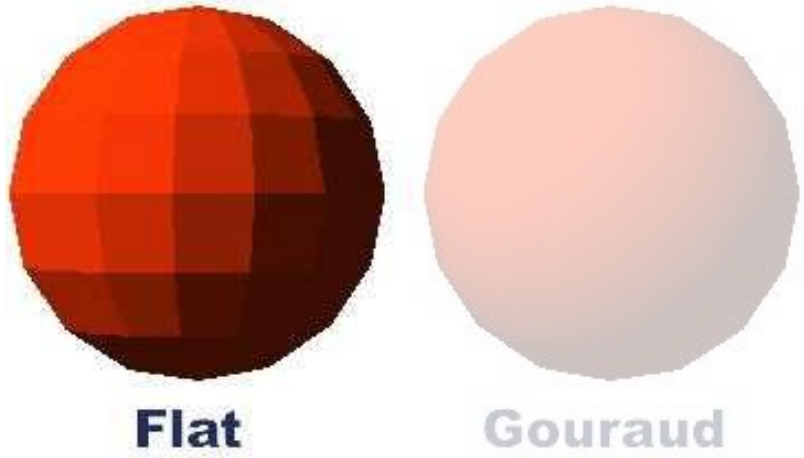
Berekenen belichting van polygoon-oppervlakken

Stap 1:

bepaal normaalvector van elk polygoon

Stap 2:

bereken kleur van hele polygoon met belichtingsmodel



Gouraud shading

Berekenen belichting van polygoon-oppervlakken

Stap 1:

bepaal normaalvector in elk hoekpunt polygoon

Stap 2:

bereken kleur van elk hoekpunt met behulp van gevonden hoekpuntnormaalvectoren met belichtingsmodel

Stap 3:

interpoleer hoekpuntkleuren over hele gebied polygoon

Normaalvectoren bij Gouraud shading

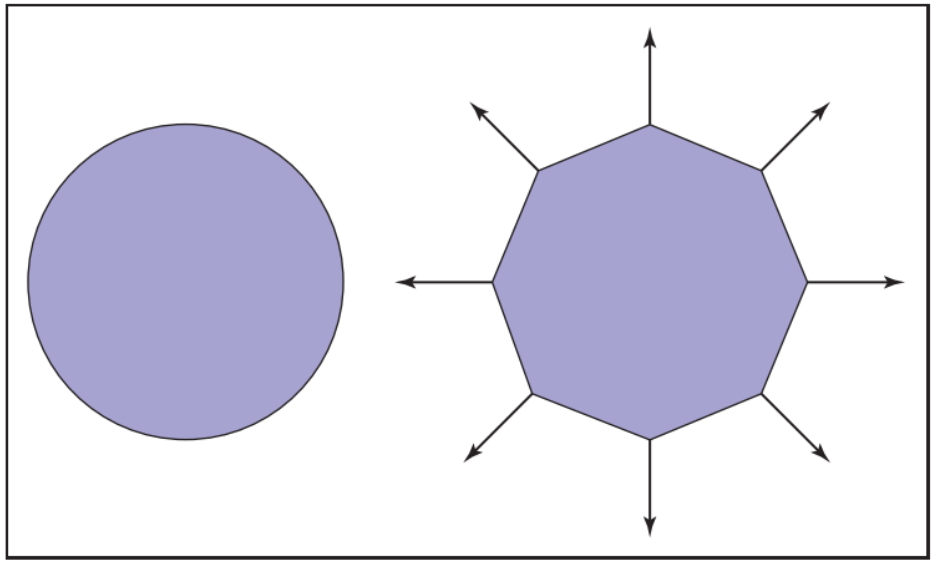


Figure 10.4. A circle (left) is approximated by an octagon (right). Vertex normals record the surface normal of the original curve.

3. Spiegelreflectie

Voorwerpen verspreiden licht niet uniform in alle richtingen daarom spiegelcomponent toevoegen

Spiegelreflecties veroorzaken glansplekken (glimmende voorwerpen)

Verlicht appel met helder wit licht:

- glansplek door spiegelreflectie
- rest door diffuse reflectie

Op glansplek lijkt appel niet rood, maar wit (Kleur invallende licht)

Model voor spiegelreflectie: Phong model

3. Spiegelreflectie: Phong model

- Glimmende voorwerpen weerspiegelen licht vooral in één richting $\boldsymbol{r}$
- Als viewer ($\boldsymbol{e}$) in de buurt van teruggekaatste richting (σ is klein), dan highlight te zien

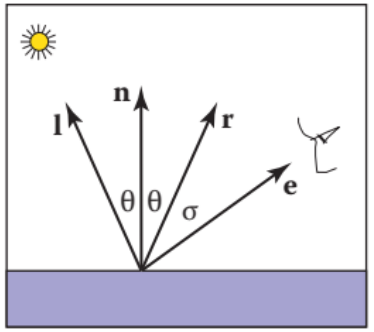


Figure 10.5. The geometry for the Phong illumination model. The eye should see a highlight if σ is small.

Phong model

Gegeven terugkaatsrichting $\boldsymbol{r}$, welke functie levert grote waarde bij $\boldsymbol{r}$ en kleine waarden verder daarvandaan?

$$c = c_l(\boldsymbol{e} \cdot \boldsymbol{r})$$

Twee problemen: inproduct kan negatief zijn en highlight te breed

Oplossing:

$$c = c_l \max(0, \boldsymbol{e} \cdot \boldsymbol{r})^p$$

met p Phong exponent

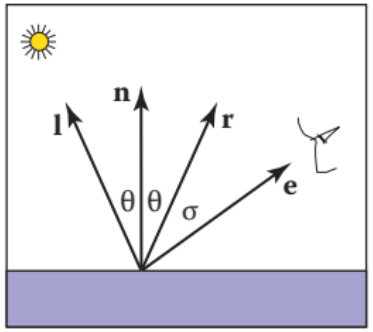


Figure 10.5. The geometry for the Phong illumination model. The eye should see a highlight if σ is small.

Effect Phong exponent

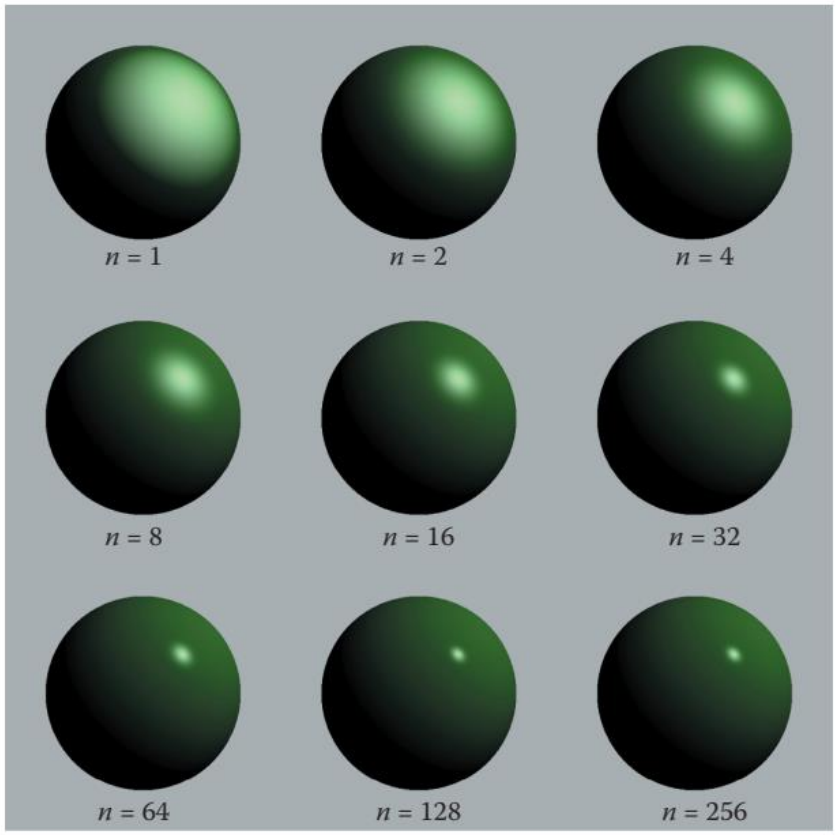


Figure 10.6. The effect of the Phong exponent on highlight characteristics. This uses Equation (10.5) for the highlight. There is also a diffuse component, giving the objects a shiny but non-metallic appearance. *Image courtesy Nate Robins.*

Phong belichtingsmodel

Kleur c van oppervlak berekend met:

$$c = c_r(c_a + c_l \max(0, \boldsymbol{n} \cdot \boldsymbol{l})) + c_l \max(0, \boldsymbol{e} \cdot \boldsymbol{r})^p$$

heet het Phong belichtingsmodel, omdat het de spiegelcomponent van Phong bevat

Berekenen van $\boldsymbol{r}$ en $\boldsymbol{e} \cdot \boldsymbol{r}$ rekenintensief, vervangen door $\boldsymbol{h} \cdot \boldsymbol{n}$ (altijd positief) halfway vector $\boldsymbol{h}$

$$\boldsymbol{h} = \frac{\boldsymbol{e} + \boldsymbol{l}}{|\boldsymbol{e} + \boldsymbol{l}|}$$

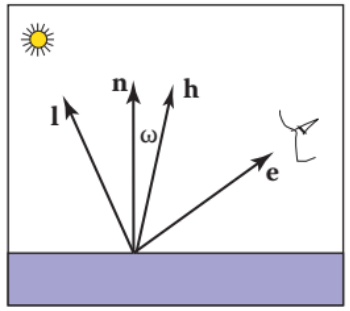


Figure 10.8. The unit vector **h** is halfway between **l** and **e**.

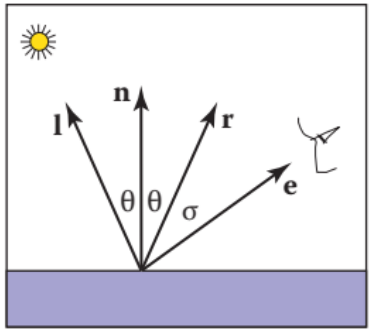


Figure 10.5. The geometry for the Phong illumination model. The eye should see a highlight if σ is small.

Halfway vector $\boldsymbol{h}$ tussen $\boldsymbol{l}$ en $\boldsymbol{e}$

Highlight als $\boldsymbol{h}$ in de buurt van $\boldsymbol{n}$, dus als $\cos\omega = \boldsymbol{h} \cdot \boldsymbol{n}$ dichtbij 1

Kleur c van oppervlak berekend met:

$$c = c_r(c_a + c_l \max(0, \boldsymbol{n} \cdot \boldsymbol{l})) + c_l \max(0, \boldsymbol{e} \cdot \boldsymbol{r})^p \Rightarrow$$

$$c = c_r(c_a + c_l \max(0, \boldsymbol{n} \cdot \boldsymbol{l})) + c_l(\boldsymbol{h} \cdot \boldsymbol{n})^p$$

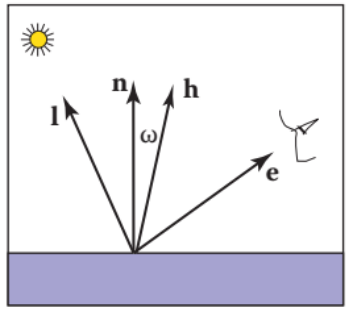
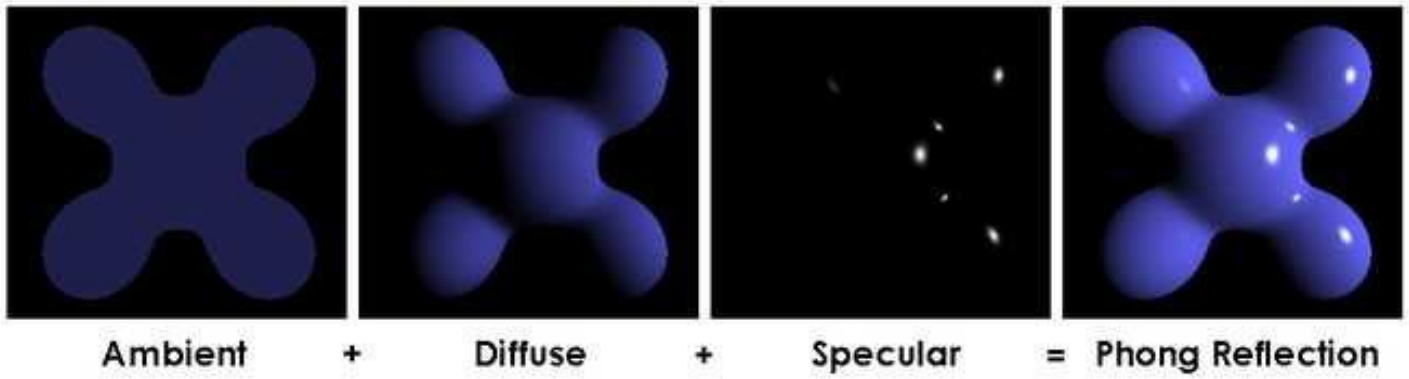


Figure 10.8. The unit vector **h** is halfway between **l** and **e**.

Phong belichtingsmodel

Phong belichtingsmodel is empirisch model voor locale belichting



Visual illustration of Phong equation: here light is white, ambient and diffuse colors are both blue, and specular color is white, reflecting almost all of the light hitting the surface, but only in very narrow highlights. The intensity of the diffuse component varies with the direction of the surface, and ambient component is uniform (independent of direction).
Source: Wikipedia

Phong shading (normaalvector $\boldsymbol{n}$ interpoleren)

Berekenen belichting van polygoon-oppervlakken

Stap 1:

bepaal normaalvector in elk hoekpunt polygoon

Stap 2:

interpoleer hoekpuntnormaalvectoren over hele gebied polygoon

(Voor driehoek: barycentrische coördinaten gebruiken!: $\boldsymbol{n} = \alpha \boldsymbol{n_0} + \beta \boldsymbol{n_1} + \gamma \boldsymbol{n_2}$)

Stap 3:

bereken kleur voor elk punt met behulp van geïnterpoleerde normaalvector met belichtingsmodel

Per-vertex (Gouraud) en per-fragment (Phong) shading

Figure 8.13. Two spheres drawn using per-vertex (Gouraud) shading. Because the triangles are large, interpolation artifacts are visible.

Per-fragment shading is sometimes called Phong shading, which is confusing because the same name is attached to the Phong illumination model.

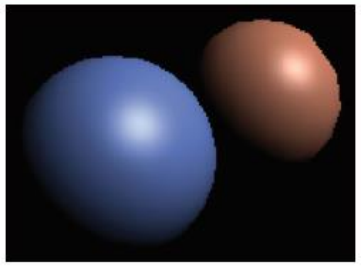


Figure 8.14. Two spheres drawn using per-fragment shading. Because the triangles are large, interpolation artifacts are visible.

FCG 8.2.4+5

Raytracing

Source

Ray tracing

- bereken straal van viewpunt door pixelcentrum
- bepaal snijpunt met eerste object geraakt door straal
- bereken kleur van pixel

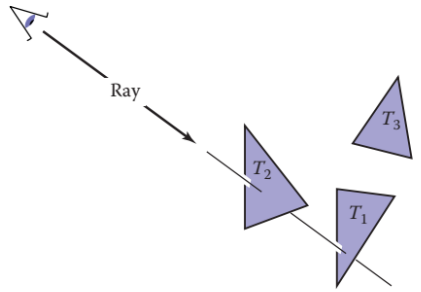
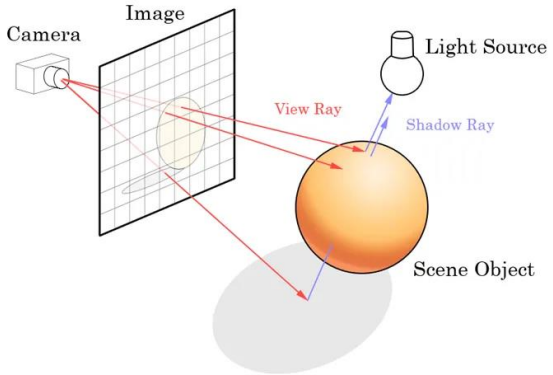


Figure 4.1. The ray is "traced" into the scene and the first object hit is the one seen through the pixel. In this case, the triangle T_2 is returned.



Source

Ray tracing

Ray tracing is methode voor produceren realistische beelden

Bepaalt zichtbare oppervlakken op pixel niveau

Voordeel ray tracing: recht toe recht aan om schaduwen en weerspiegelingen te berekenen

Ray tracing ook gebruikt voor visible surface detection en interactie met objecten

Wheely bin seems to float in the air
Source

Clearest ice you will ever see
Source

Sun directly overhead in Hawai
Source

Viewing volume voor ray tracing

- Rays door e snijden viewplane $w = n$

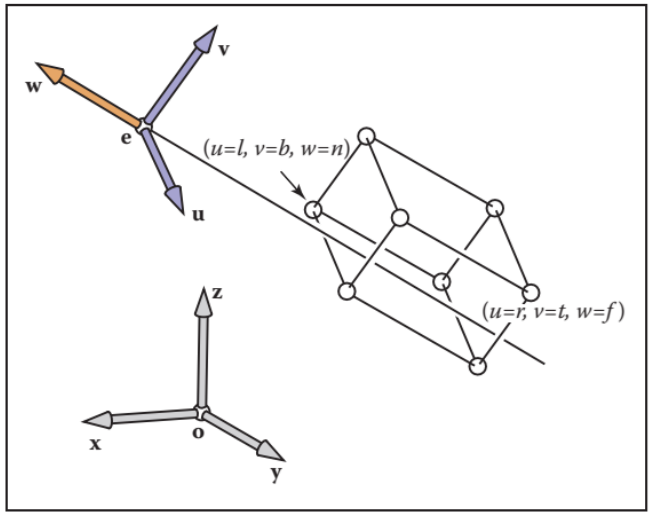


Figure 7.7. For arbitrary viewing, we need to change the points to be stored in the "appropriate" coordinate system. In this case it has origin **e** and offset coordinates in terms of **uvw**.

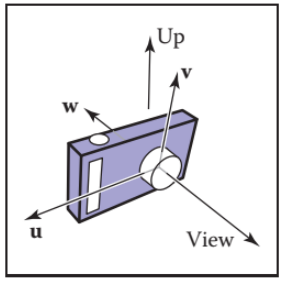


Figure 4.8. The vectors of the camera frame, together with the view direction and up direction. The **w** vector is opposite the view direction, and the **v** vector is coplanar with **w** and the up vector.

Rays door pixel centra

Snijpunt rays met viewplane $w = n$ corresponderen met pixels in schermcoördinaten

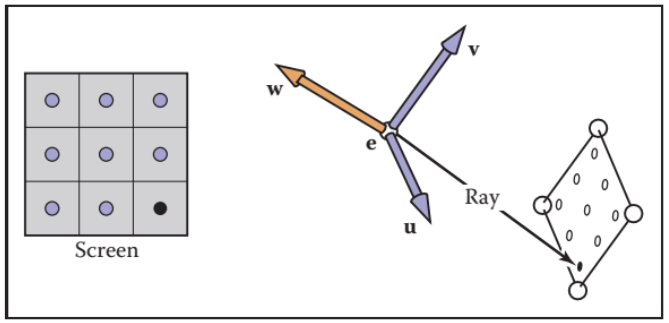
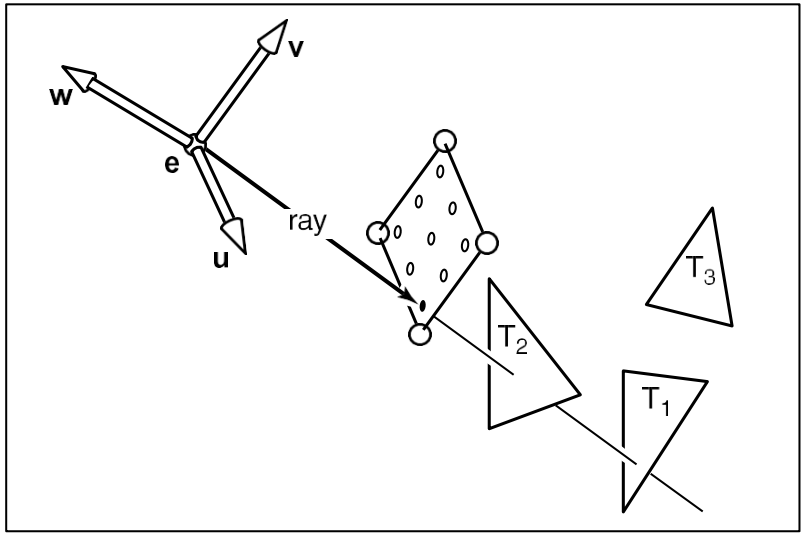


Figure 4.7. The sample points on the screen are mapped to a similar array on the 3D window. A viewing ray is sent to each of these locations.

Dichtstbijzijnde object

Ray is ”traced” en snijdt dichtstbijzijnde object



Basic ray tracing algoritme

```
bereken orthonormale basis u,v,w
voor elk pixel
do
 bereken viewing ray
 bepaal dichtstbijzijnde object geraakt door ray en normaal n
 zet pixelkleur berekend met belichtingsvergelijking
```

Berekenen viewing rays

3D parameter voorstelling lijn vanaf $\boldsymbol{e}$ door punt op scherm $\boldsymbol{s}$:

$$\boldsymbol{p}(t) = \boldsymbol{e} + t(\boldsymbol{s} - \boldsymbol{e})$$

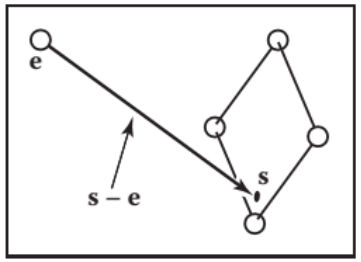


Figure 4.6. The ray from the eye to a point on the image plane.

$\boldsymbol{p}(0) = \boldsymbol{e}$ en $p(1) = \boldsymbol{s}$

- als $t >= 0$ met $t_1 < t_2$ dan $\boldsymbol{p}(t_1)$ dichterbij oog dan $\boldsymbol{p}(t_2)$
- als $t < 0$ dan is $\boldsymbol{p}(t)$ achter oog

s t.o.v. u, v, w

Elke pixel (i, j) op scherm tussen $[-0.5, n_x - 0.5] \times [-0.5, n_y - 0.5]$ correspondeert met punt in viewing window $[l, r] \times [b, t]$

Windowtransformatie die pixel omzet in s t.o.v. (u, v) is:

transleer(l, b) schaal$(\frac{r-l}{n_x}, \frac{t-b}{n_y})$ transleer$(0.5, 0.5)$

$$\begin{pmatrix} 1 & 0 & l \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{r-l}{n_x} & 0 & 0 \\ 0 & \frac{t-b}{n_y} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0.5 \\ 0 & 1 & 0.5 \\ 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} \frac{r-l}{n_x} & 0 & \frac{0.5(r-l)}{n_x} + l \\ 0 & \frac{t-b}{n_y} & \frac{0.5(t-b)}{n_y} + b \\ 0 & 0 & 1 \end{pmatrix}$$

Pixel (i, j) naar (u_s, v_s)

$$\begin{pmatrix} \frac{r-l}{n_x} & 0 & \frac{0.5(r-l)}{n_x} + l \\ 0 & \frac{t-b}{n_y} & \frac{0.5(t-b)}{n_y} + b \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} i \\ j \\ 1 \end{pmatrix} = \begin{pmatrix} u_s \\ v_s \\ 1 \end{pmatrix}$$

$$u_s = l + (r-l)\frac{i+0.5}{n_x}$$

$$v_s = b + (t-b)\frac{j+0.5}{n_y}$$

Punt $\boldsymbol{s}$ kunnen we nu uitdrukken t.o.v. $\boldsymbol{uvw}$:

$$\boldsymbol{s} = u_s\boldsymbol{u} + v_s\boldsymbol{v} + n\boldsymbol{w} = \begin{pmatrix} u_s \\ v_s \\ n \end{pmatrix}_{cameracoordinaten}$$

Ray-driehoek intersectie

- Eerst snijpunt berekenen met vlak gedefinieerd door driehoek
- Dan controleren of snijpunt in driehoek

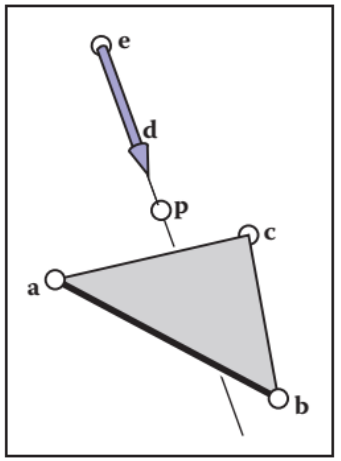


Figure 4.10. The ray hits the plane containing the triangle at point **p**.

Ray-driehoek intersectie

Gegeven: straal $\boldsymbol{p}(t) = \boldsymbol{e} + t\boldsymbol{d}$ en driehoek door punten $\boldsymbol{a}$, $\boldsymbol{b}$ en $\boldsymbol{c}$

Parametervergelijking vlak: $\boldsymbol{p} = \boldsymbol{a} + \beta(\boldsymbol{b} - \boldsymbol{a}) + \gamma(\boldsymbol{c} - \boldsymbol{a})$

($\boldsymbol{p} = \alpha\boldsymbol{a} + \beta\boldsymbol{b} + \gamma\boldsymbol{c}$ met $\alpha = 1 - \beta - \gamma \rightarrow \alpha + \beta + \gamma = 1$)

Snijpunt in driehoek $\Leftrightarrow \beta > 0, \gamma > 0$ en $\beta + \gamma < 1$

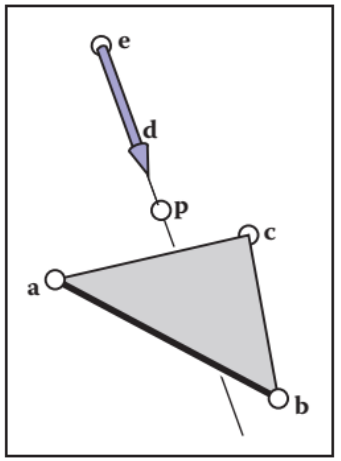


Figure 4.10. The ray hits the plane containing the triangle at point **p**.

Matrixvergelijking snijpunt

Snijpunt tussen lijn en vlak: $\mathbf{e} + t\mathbf{d} = \mathbf{a} + \beta(\mathbf{b} - \mathbf{a}) + \gamma(\mathbf{c} - \mathbf{a})$

$$x_e + tx_d = x_a + \beta(x_b - x_a) + \gamma(x_c - x_a)$$

$$y_e + ty_d = y_a + \beta(y_b - y_a) + \gamma(y_c - y_a)$$

$$z_e + tz_d = z_a + \beta(z_b - z_a) + \gamma(z_c - z_a)$$

Oplossen voor t, β en γ:

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

Oplossen $\mathbf{Ax} = \mathbf{b}$ met $\mathbf{x} = (\beta, \gamma, t)$

Berekenen β, γ en t

Oplossen $\mathbf{Ax} = \mathbf{b}$ met $\mathbf{x} = (\beta, \gamma, t)$

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

1. Regel van Cramer
2. matrix-inverse $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$
3. eliminatie

Oplossing vergelijking

We kunnen t gebruiken om te kijken of snijpunt in view volume

- t in interval (t_0, t_1) dan punt in view volume

We gebruiken β en γ om te kijken of punt in driehoek

- $\beta > 0$
- $\gamma > 0$
- $\beta + \gamma < 1$

Ray-triangle intersection

```
boolean raytri(ray r, vector a, vector b, vector c, t0, t1)
  bereken t
  als (t < t0 of t > t1)
    return false
  bereken gamma
  als (gamma < 0 of gamma > 1)
    return false
  bereken beta
  als (beta < 0 of beta > 1 - gamma)
    return false
  return true
```

Voorbeeld: opgave 2 hoofdstuk 4[1]

Wat zijn de barycentrische coördinaten en t van snijpunt van ray

$$\boldsymbol{p} = \boldsymbol{e} + t\boldsymbol{d} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + t \begin{pmatrix} -1 \\ -1 \\ -1 \end{pmatrix}$$

en driehoek met hoekpunten $(1,0,0)$, $(0,1,0)$, $(0,0,1)$?

Parametervoorstelling van vlak door $(1,0,0)$, $(0,1,0)$, $(0,0,1)$ is

$$\boldsymbol{p} = \boldsymbol{a} + \beta(\boldsymbol{b} - \boldsymbol{a}) + \gamma(\boldsymbol{c} - \boldsymbol{a})$$

$$\boldsymbol{p} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix} + \gamma \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$$

[1]Hoofdstuk 10 in oude edities van het boek.

Antwoord opgave 2 hoofdstuk 4

Snijden lijn en vlak:

$$\mathbf{e} + t\mathbf{d} = \mathbf{a} + \beta(\mathbf{b} - \mathbf{a}) + \gamma(\mathbf{c} - \mathbf{a})$$

$$\begin{pmatrix}1\\1\\1\end{pmatrix} + t\begin{pmatrix}-1\\-1\\-1\end{pmatrix} = \begin{pmatrix}1\\0\\0\end{pmatrix} + \beta\begin{pmatrix}-1\\1\\0\end{pmatrix} + \gamma\begin{pmatrix}-1\\0\\1\end{pmatrix}$$

$$\beta\begin{pmatrix}-1\\1\\0\end{pmatrix} + \gamma\begin{pmatrix}-1\\0\\1\end{pmatrix} + t\begin{pmatrix}1\\1\\1\end{pmatrix} = \begin{pmatrix}0\\1\\1\end{pmatrix}$$

$$\begin{pmatrix}-1 & -1 & 1\\1 & 0 & 1\\0 & 1 & 1\end{pmatrix}\begin{pmatrix}\beta\\\gamma\\t\end{pmatrix} = \begin{pmatrix}0\\1\\1\end{pmatrix}$$

Antwoord opgave 2 hoofdstuk 4

$$\left.\begin{matrix} -\beta - \gamma + t = 0 \\ \beta + t = 1 \\ \gamma + t = 1 \end{matrix}\right\} \Rightarrow \beta = \gamma \Rightarrow \left.\begin{matrix} -2\beta + t = 0 \\ \beta + t = 1 \end{matrix}\right\} \Rightarrow 3\beta = 1 \Rightarrow \beta = \frac{1}{3}$$

$\gamma = 1/3$ en $t = \beta + \gamma = 2/3$ en $\alpha = 1 - \beta - \gamma = 1/3$

Snijpunt in driehoek want $\beta > 0$, $\gamma > 0$ en $\beta + \gamma < 1$

Snijpunt ook in goede kijkrichting, want $t = \frac{2}{3} > 0$

Gevonden snijpunt: $\mathbf{e} + \frac{2}{3}\mathbf{d} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \frac{2}{3}\begin{pmatrix} -1 \\ -1 \\ -1 \end{pmatrix} = \begin{pmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{pmatrix}$

Ray tracing algoritme

```
for each pixel
do
  compute viewing ray
  if (ray hits an object with t in [0, inf])
  then
    compute n
    evaluate lighting equation and set pixel to that color
  else
    set pixel color to background color
```

Schaduwstralen

Licht uit richting $\boldsymbol{l}$ (op oneindig), werp schaduwstraal $\boldsymbol{p} + t\boldsymbol{l}$ van $\boldsymbol{p}$ naar lichtbron

- schaduwstraal $\boldsymbol{q} + t\boldsymbol{l}$ snijdt object, dan $\boldsymbol{q}$ in schaduw
- schaduwstraal $\boldsymbol{p} + t\boldsymbol{l}$ snijdt geen object, dan $\boldsymbol{p}$ niet in schaduw

Stralen die bepalen of er schaduw is heten schaduwstralen

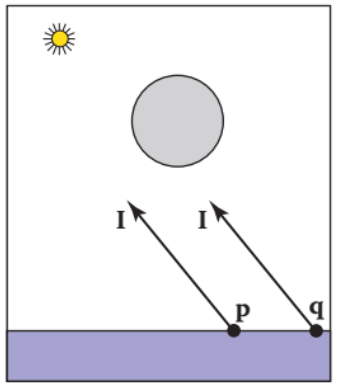


Figure 4.17. The point **p** is not in shadow, while the point **q** is in shadow.

Ray tracing met schaduwen

```
functie raycolor(ray e + td, t0, t1)
  als (ray snijdt object in P)
  dan
    c = cr ca // ambient
    als (P's schaduwstraal geen object snijdt)
    dan
      bereken n.l en h.n
      c = c + cr cl max(0, n.l) + cl (h.n)^p
    return c
  anders
    return achtergrondkleur
```

Oefenopgaven

Opgave 9 hoofstuk 6

Schrijf 3×3 matrix op die 2D punt draait over hoek θ om punt $\boldsymbol{p} = (x_p, y_p)$.

Antwoord opgave 9 hoofdstuk 6

Schrijf 3×3 matrix op die 2D punt draait over hoek θ om punt $\boldsymbol{p} = (x_p, y_p)$.

1. Verschuif $\boldsymbol{p}$ naar oorsprong
2. Roteer
3. Verschuif terug over $\boldsymbol{p}$

$$\boldsymbol{R_p} = \begin{pmatrix} 1 & 0 & x_p \\ 0 & 1 & y_p \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -x_p \\ 0 & 1 & -y_p \\ 0 & 0 & 1 \end{pmatrix} =$$

$$\begin{pmatrix} \cos\theta & -\sin\theta & x_p \\ \sin\theta & \cos\theta & y_p \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -x_p \\ 0 & 1 & -y_p \\ 0 & 0 & 1 \end{pmatrix} =$$

$$\begin{pmatrix} \cos\theta & -\sin\theta & -x_p\cos\theta + y_p\sin\theta + x_p \\ \sin\theta & \cos\theta & -x_p\sin\theta - y_p\cos\theta + y_p \\ 0 & 0 & 1 \end{pmatrix}$$

Opgave 10 hoofdstuk 6

Schrijf de 4×4 rotatiematrix $\boldsymbol{M}$ op die orthonormale basis

$\boldsymbol{u} = (x_u, y_u, z_u)$, $\boldsymbol{v} = (x_v, y_v, z_v)$, $\boldsymbol{w} = (x_w, y_w, z_w)$

overvoert in orthonormale vectoren

$\boldsymbol{a} = (x_a, y_a, z_a)$, $\boldsymbol{b} = (x_b, y_b, z_b)$, $\boldsymbol{c} = (x_c, y_c, z_c)$.

Dus er moet gelden: $\boldsymbol{Mu} = \boldsymbol{a}, \boldsymbol{Mv} = \boldsymbol{b}, \boldsymbol{Mw} = \boldsymbol{c}$.

R_{uvw}, R_{abc} en R^T_{abc}

- $\boldsymbol{R_{uvw}} = \begin{pmatrix} - & \boldsymbol{u} & - \\ - & \boldsymbol{v} & - \\ - & \boldsymbol{w} & - \end{pmatrix} : (\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}) \rightarrow (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$
- $\boldsymbol{R_{abc}} = \begin{pmatrix} - & \boldsymbol{a} & - \\ - & \boldsymbol{b} & - \\ - & \boldsymbol{c} & - \end{pmatrix} : (\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}) \rightarrow (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$
- $\boldsymbol{R^T_{abc}} = \begin{pmatrix} | & | & | \\ \mathbf{a} & \mathbf{b} & \mathbf{c} \\ | & | & | \end{pmatrix} : (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \rightarrow (\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$
- We zoeken $M : (\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}) \rightarrow (\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$

Antwoord opgave 9 hoofdstuk 6

$$\boldsymbol{M = R_{abc}^{T} R_{uvw}}$$

$$\boldsymbol{M} = \begin{pmatrix} | & | & | \\ \mathbf{a} & \mathbf{b} & \mathbf{c} \\ | & | & | \end{pmatrix} \begin{pmatrix} - & \boldsymbol{u} & - \\ - & \boldsymbol{v} & - \\ - & \boldsymbol{w} & - \end{pmatrix}$$

Opgave 11 hoofdstuk 6

Wat is de inverse matrix van M?

Antwoord opgave 11 hoofdstuk 6

Wat is de inverse matrix van M?

$$\boldsymbol{M^{-1} = \left(R^{T}_{abc} R_{uvw}\right)^{-1} = R^{T}_{uvw} R_{abc}}$$