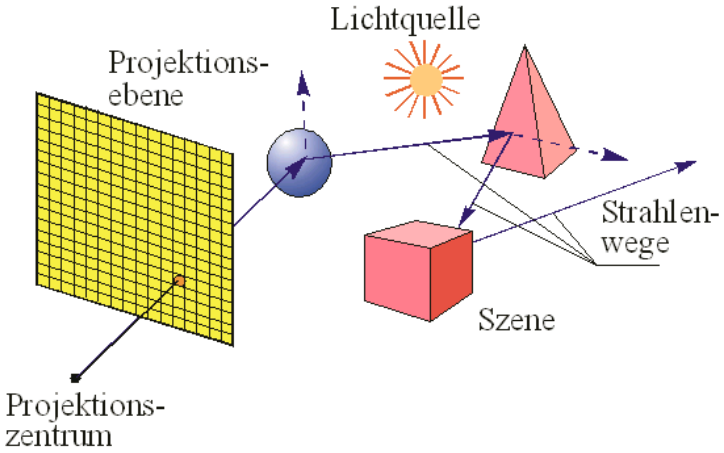


Raytracing

```
typedef struct{double x,y,z}vec;vec U,black,amb={.02,.02,.02};struct sphere{ vec
cen,color;double rad,kd,ks,kt,kl,ir}*s,*best,sph[]={0.,6.,.5,1.,1.,1.,.9,
.05,.2,.85,0.,1.7,-1.,8.,-.5,1.,.5,.2,1.,.7,.3,0.,.05,1.2,1.,8.,-.5,.1,.8,.8,
1.,.3,.7,0.,0.,1.2,3.,-6.,15.,1.,.8,1.,7.,0.,0.,0.,.6,1.5,-3.,-3.,12.,.8,1.,
1.,5.,0.,0.,0.,.5,1.5,};yx;double u,b,tmin,sqrt(),tan();double vdot(A,B)vec A
,B;{return A.x*B.x+A.y*B.y+A.z*B.z;}vec vcomb(a,A,B)double a;vec A,B;{B.x+=a*
A.x;B.y+=a*A.y;B.z+=a*A.z;return B;}vec vunit(A)vec A;{return vcomb(1./sqrt(
vdot(A,A)),A,black);}struct sphere*intersect(P,D)vec P,D;{best=0;tmin=1e30;s=
sph+5;while(s-->sph)b=vdot(D,U=vcomb(-1.,P,s->cen)),u=b*b-vdot(U,U)+s->rad*s -
>rad,u=u>0?sqrt(u):1e31,u=b-u>1e-7?b-u:b+u,tmin=u>=1e-7&&u<tmin?best=s,u:
tmin;return best;}vec trace(level,P,D)vec P,D;{double d,eta,e;vec N,color; struct
sphere*s,*l;if(!level--)return black;if(s=intersect(P,D));else return
amb;color=amb;eta=s->ir;d= -vdot(D,N=vunit(vcomb(-1.,P=vcomb(tmin,D,P),s->cen
)));if(d<0)N=vcomb(-1.,N,black),eta=1/eta,d= -d;l=sph+5;while(l-->sph)if((e=l -
>kl*vdot(N,U=vunit(vcomb(-1.,P,l->cen))))>0&&intersect(P,U)==l)color=vcomb(e ,l-
>color,color);U=s->color;color.x*=U.x;color.y*=U.y;color.z*=U.z;e=1-eta* eta*(1-
d*d);return vcomb(s->kt,e>0?trace(level,P,vcomb(eta,D,vcomb(eta*d-sqrt
(e),N,black))):black,vcomb(s->ks,trace(level,P,vcomb(2*d,N,D)),vcomb(s->kd,
color,vcomb(s->kl,U,black))));}main(){printf("%d %d\n",32,32);while(yx<32*32)
U.x=yx%32-32/2,U.z=32/2-yx++/32,U.y=32/2/tan(25/114.5915590261),U=vcomb(255.,
trace(3,black,vunit(U)),black),printf("%.0f %.0f %.0f\n",U);}/*minray!*/
```

© Paul Heckbert

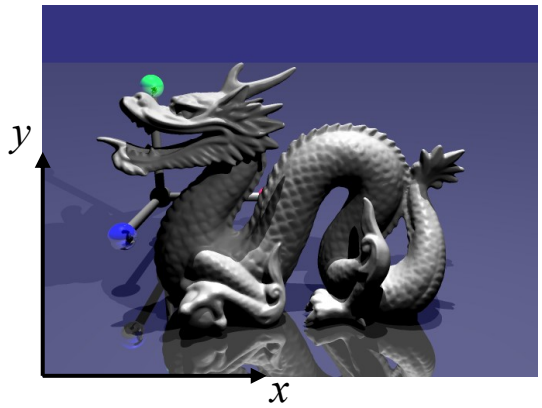
Inhalt

- Rekursives Raytracing
- Strahlverfolgung
  - Primärstrahlen (Supersampling)
  - Sekundärstrahlen (Epsilonproblematik)
  - Schattenstrahlen (Transparenzproblematik)
  - weiche Schatten
- Strahlschnitte
  - Ebene
  - Kugel
  - Achsenparallele Box
  - Dreieck
- Beschleunigungsdatenstrukturen
  - Gitter
  - KD-Baum
  - Hüllvolumen

Recursive Raytracing

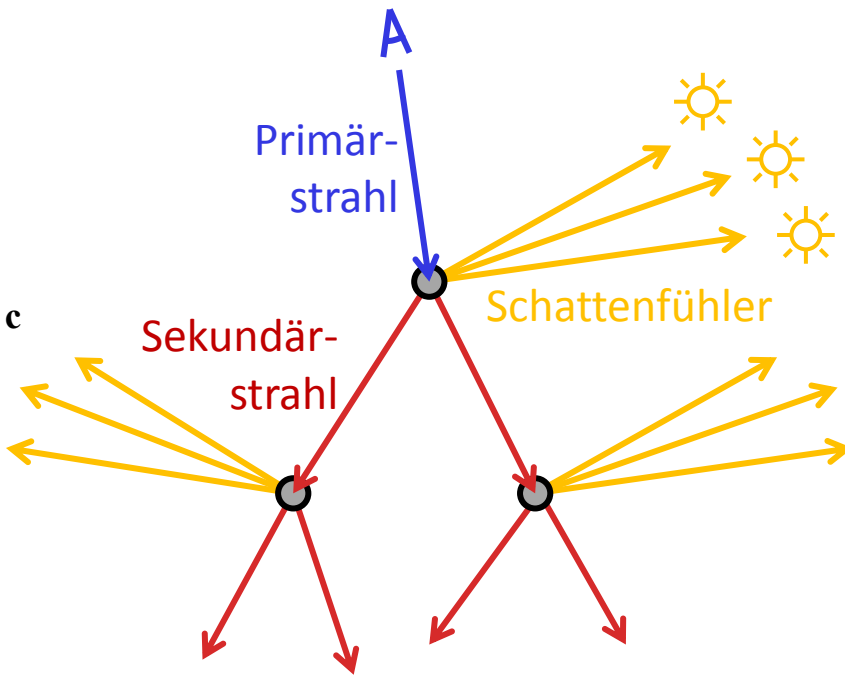
Äußere Schleife des Algorithmus:

for each pixel (x,y) of camera **do**
    generate a primary ray $R = (\underline{\boldsymbol{p}},v)$
    set image pixel to color of incoming light:
        $I[x,y] = trace(R)$



Rekursive Berechnung der Pixelfarbe:

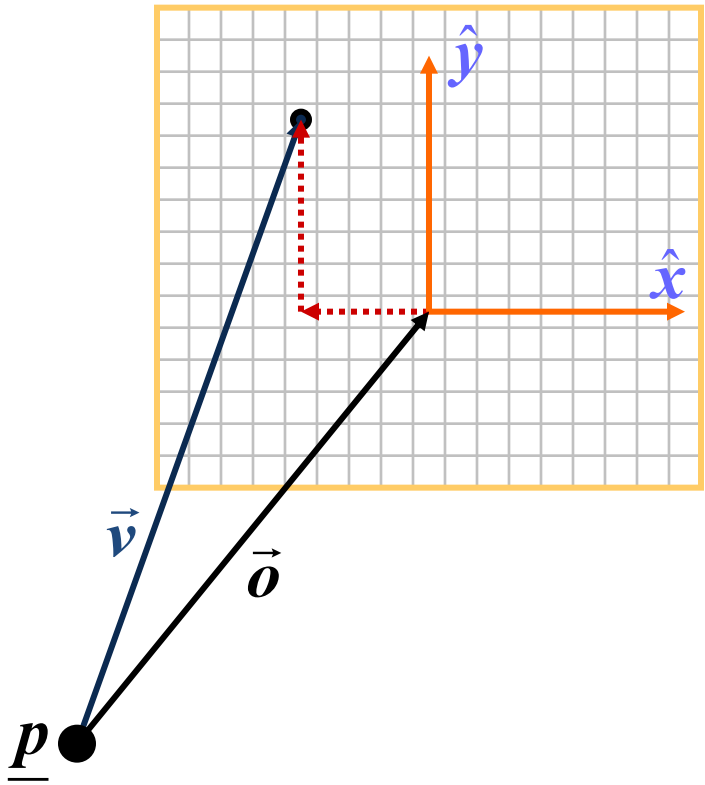
$trace(R = (\underline{\boldsymbol{p}},v))$
    determine first intersection $\underline{\boldsymbol{q}}$ of R and scene
    if no intersection exists
        return background color
    set resulting color $\mathbf{c}$ to black
    for each light source at $\underline{\boldsymbol{l}}$ **do**
        if scene intersects segment $(\underline{\boldsymbol{q}},\underline{\boldsymbol{l}})$ **continue**
        add light from $\underline{\boldsymbol{l}}$ reflected at $\underline{\boldsymbol{q}}$ in direction of $-v$ to $\mathbf{c}$
    if surface at $\underline{\boldsymbol{q}}$ has [partial] mirror reflection
        generate reflected secondary ray R_{refl}
        contribute $trace(R_{refl})$ to $\mathbf{c}$
    if surface at $\underline{\boldsymbol{q}}$ has [partial] transmission
        generate transmitted secondary ray R_{trans}
        contribute $trace(R_{trans})$ to $\mathbf{c}$
    return $\mathbf{c}$



Strahlverfolgung
Primärstrahlen

- Inkrementelle Berechnung der Strahlen von Augpunkt durch Pixel der virtuellen Kamera:

```
for (x= 0; x < xres; x++)
  for (y= 0; y < yres; y++)
  {
    v = o + 2(x/xres - 0.5)·x
          + 2(y/yres - 0.5)·y;
    v = v/|v|; // Normalize
    col= trace(p, v);
    write_pixel(x,y,col);
  }
```

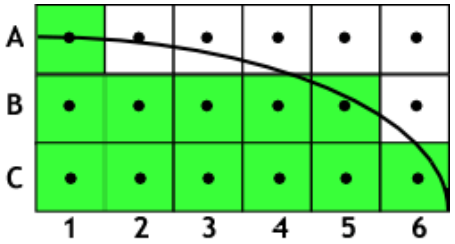


parametrisierter Strahl:

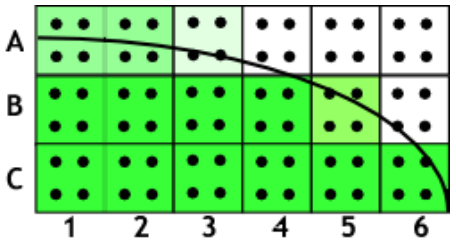
$$\underline{x}(t) = \underline{p} + t\vec{v}$$

Primärstrahlen
Supersampling

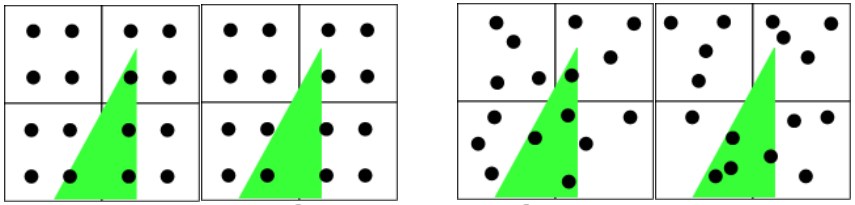
- Um Treppenstufenartefakte an Kanten und Schatten zu vermeiden, bietet sich der Einsatz von Supersampling an
- Im einfachsten Fall wird das Bild mit einer höheren Auflösung berechnet und danach die Auflösung mit einem Filter reduziert
- Meist wird aber pro Pixel ein Supersampling durchgeführt:
  - Gitter … regelmäßig
  - Zufall … ganz zufällig – neigt zur Verklumpung
  - Poisson … zufällig ohne Verklumpung
  - Jitter … zufällig pro Gitterzelle

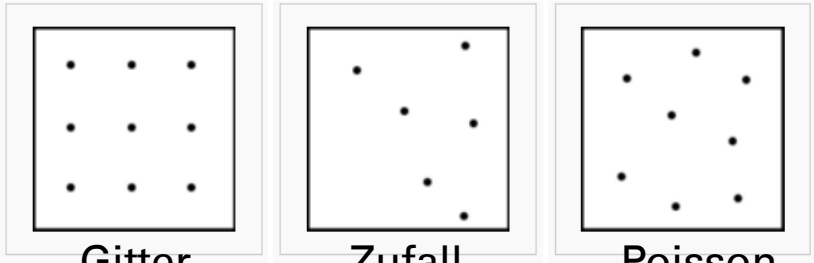


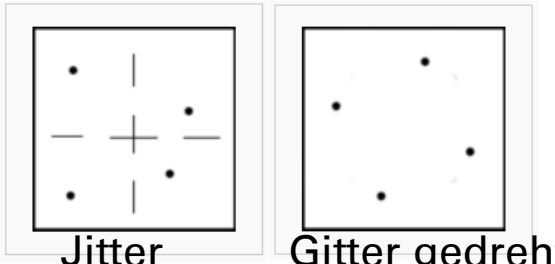
Treppenstufen



Antialiasing mit Gitter

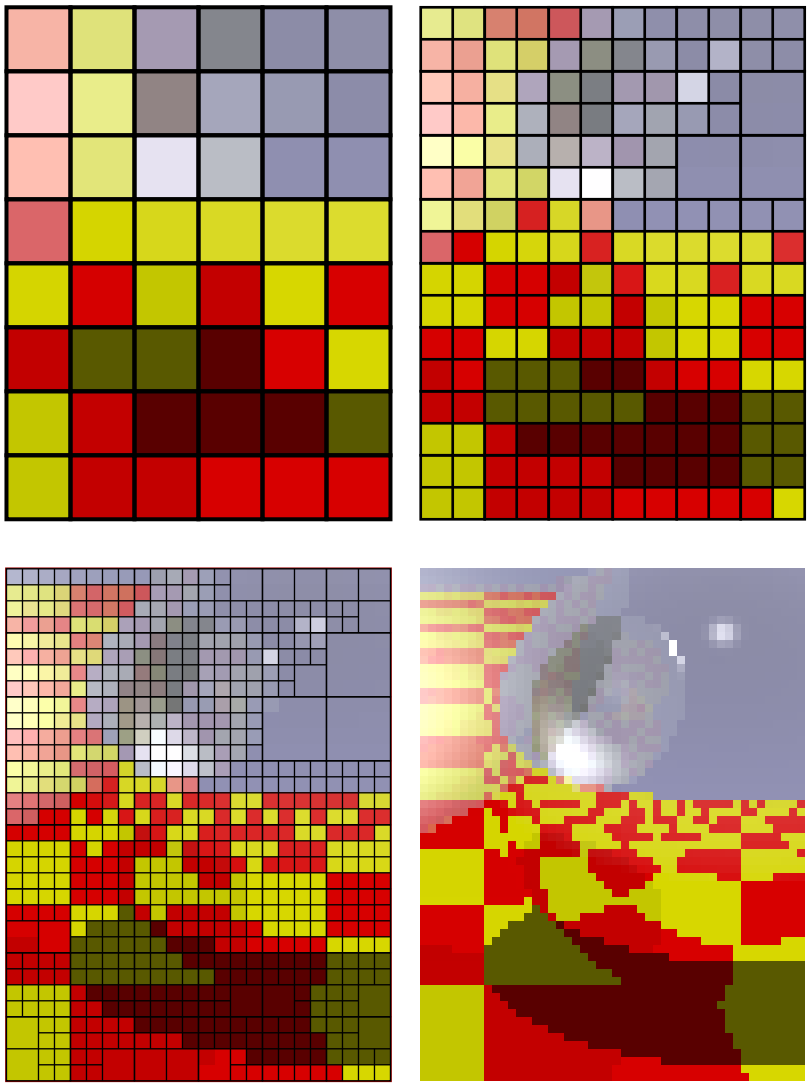

Gitter vs Zufallssampling
@www.andymeneely.com/prog/raytracer/supersampling/index.htm


Gitter Zufall Poisson


Jitter Gitter gedreht
@http://en.wikipedia.org/wiki/Supersampling

Primärstrahlen
Adaptives Sampling

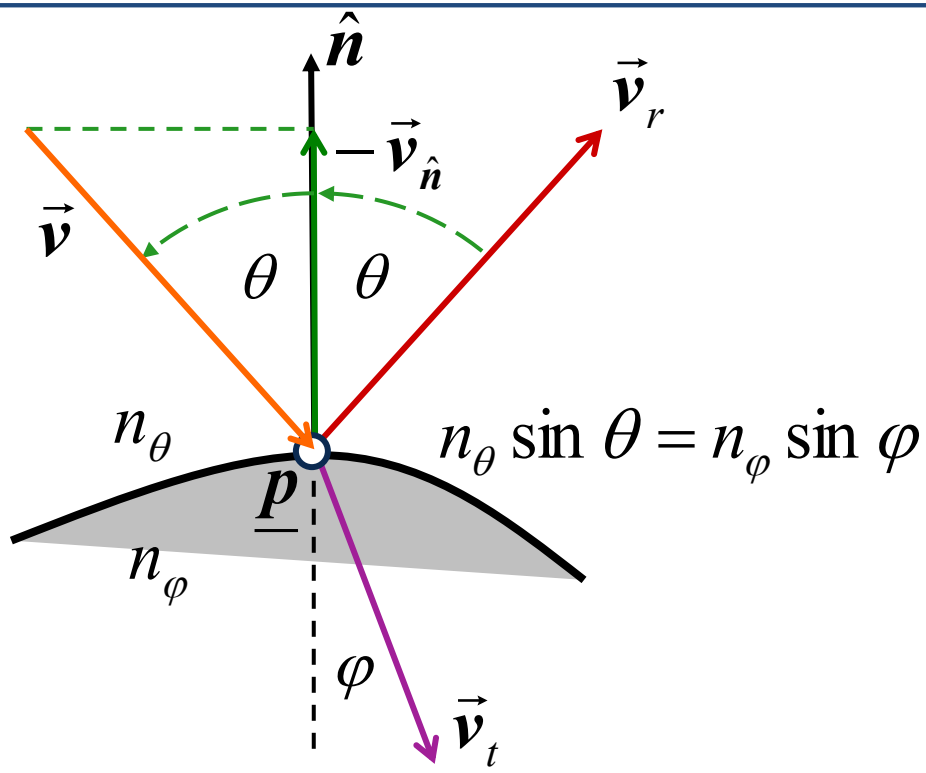
- Idee: generiere Bild in grober Auflösung und verfeinere nur in den Bereichen, in denen sich der Farbwert stark ändert
- Betrachte dazu quadratische Zellen und generiere Strahlen durch die Eckpunkte der Zellen nicht durch die Mittelpunkte
- unterteile rekursiv alle Zellen in 4 gleich große Zellen, deren Farbwerte an den Eckpunkten deutlich unterschiedlich sind
- Zu beachten ist, dass die Anfangsauflösung fein genug sein muss.



Strahlverfolgung
Sekundärstrahlen

- Trifft ein Strahl auf eine Oberfläche, so ergibt sich der Oberflächenpunkt $\underline{\boldsymbol{p}}$ aus dem Strahl-Flächenschnitt.
- Zusätzlich erhält man die Oberflächennormale $\boldsymbol{n}$
- Sekundärstrahlen werden erzeugt wenn das Material spiegelnd oder transparent ist.
- Die Sekundärstrahlen beginnen beide bei $\underline{\boldsymbol{p}}$ und haben die zu berechnenden Richtungen $\boldsymbol{v}_r$ und $\boldsymbol{v}_t$
- Zur Berechnung von $\boldsymbol{v}_t$ werden zusätzlich die Brechnungs-indizes benötigt. Die Herleitung der Formel basiert auf der Identität:

$$\sin^2\theta+\cos^2\theta=1$$



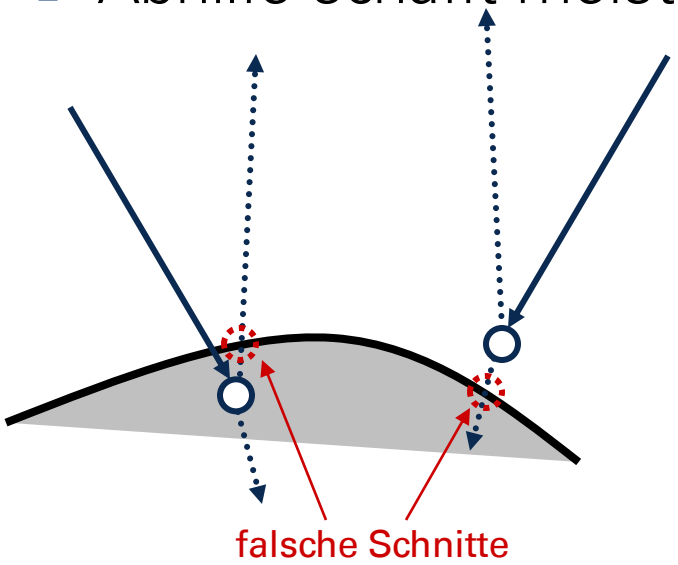
Berechnung der Sekundärstrahlen

$$\vec{v}_r = \vec{v} - 2\langle \hat{n}, \vec{v} \rangle \hat{n}$$

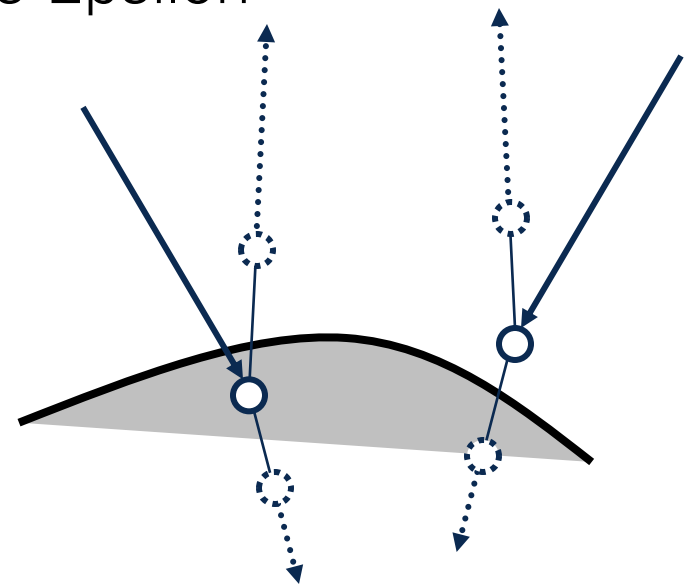
$$\vec{v}_t = \frac{n_\theta}{n_\varphi}\vec{v} - \left(\frac{n_\theta}{n_\varphi}\langle \hat{n}, \vec{v} \rangle + \sqrt{1 - \left(\frac{n_\theta}{n_\varphi}\right)^2 \left(1 - \langle \hat{n}, \vec{v} \rangle^2\right)} \right) \hat{n}$$

Sekundärstrahlen
Numerische Ungenauigkeiten

- Beim Verfolgen von Sekundärstrahl ausgehend von Schnitten mit Oberflächen, kann es durch numerische Ungenauigkeiten zu erneutem Schnitt mit derselben Oberfläche kommen.
- Abhilfe schafft meist ein globales Epsilon



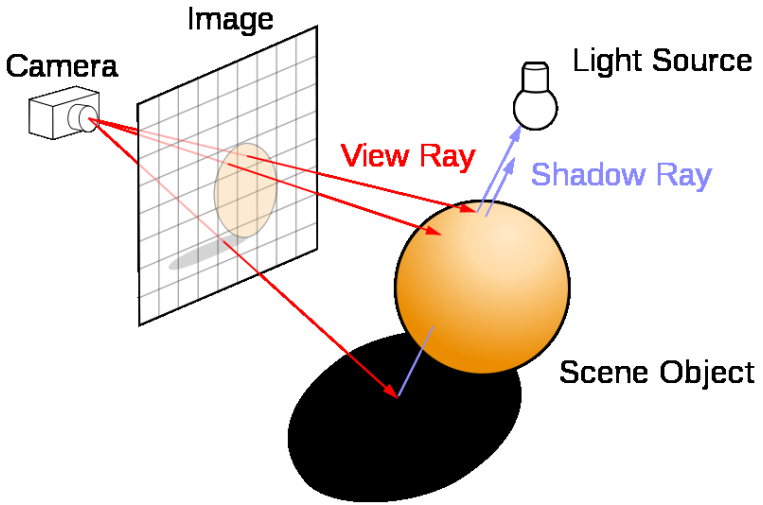
Problem: numerische Ungenauigkeiten



Abhilfe: richtungsabhängiges Offset im Sekundärstrahl

Strahlverfolgung
Schattenfühler

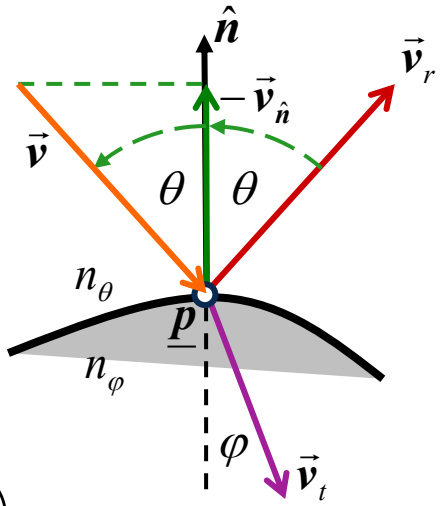
- Bei jedem Schnitt $\underline{\boldsymbol{p}}$ mit einer Oberfläche wird zu jeder Lichtquelle $\underline{\boldsymbol{l}}_i$ ein Schattenstrahl geschickt und geprüft, ob ein anderes Objekt zwischen aktuell betrachtetem Oberflächenpunkt und Lichtquelle ist.
- Hierzu wird nur ein beliebiger Schnitt benötigt, nicht unbedingt der erste. Deshalb werden meist zwei Schnittfunktionen implementiert:
  - `find_first_intersection` ($\underline{\boldsymbol{p}}$,$\boldsymbol{v}$) … erster Schnitt vor $\underline{\boldsymbol{p}}$ in Richtung $\boldsymbol{v}$
  - `is_blocked` ($\underline{\boldsymbol{p}}$,$\underline{\boldsymbol{q}}$) … Test, ob zwischen $\underline{\boldsymbol{p}}$ & $\underline{\boldsymbol{q}}$ irgendein Objekt liegt



Schattenfühler – geblockt und ungeblockt

Strahlverfolgung
Berechnung der Leuchtdichte

- Typischerweise werden die Materialparameter um einen skalaren Reflexions- , einen skalaren Refraktionskoeffizienten und einen Brechungsindex erweitert:
  - r_a … ambienter Reflektions-farbkoeffizient
  - r_d … diffuser Reflektions-farbkoeffizient
  - r_s … spekularer Reflektions-farbkoeffizient
  - s … shininess
  - r_r … Prozentsatz des ideal spiegelnden Lichts
  - r_t … Prozentsatz des refraktierten Lichts
  - n … Brechungsindex
  - n_{scene} … Brechungsindex der Szene
  - n_{ray} … Brechungsindex des Strahls



$$\ddot{\underline{L}}_{\text{out}}\left(\underline{p},-\vec{v}\right)=$$

$$\left(\ddot{\underline{L}}_{\text{scene},a}+\sum_{i=1}^{m}\ddot{\underline{L}}_{i,a}\right)\otimes\ddot{r}_a+$$

Sekundärstrahlen

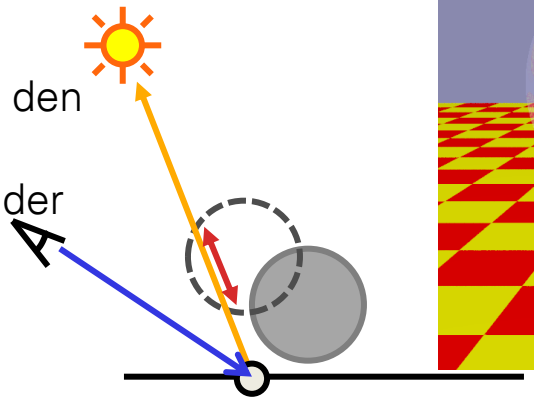
$$r_r\cdot\ddot{\underline{L}}_{\text{in}}\left(\underline{p},\vec{v}_r\right)+r_t\cdot\ddot{\underline{L}}_{\text{in}}\left(\underline{p},\vec{v}_t,n,n_{\text{ray}}\right)+$$

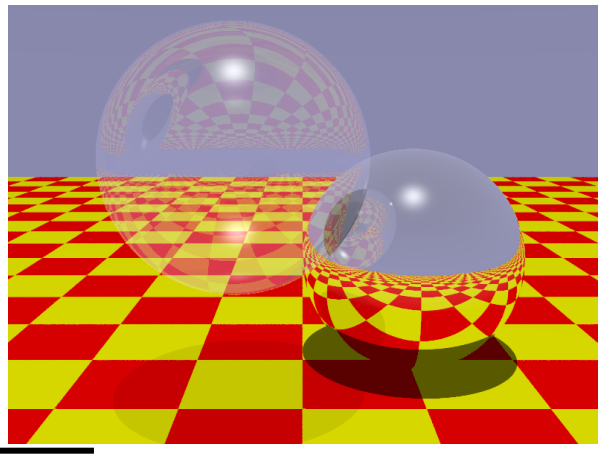
$$\sum_{i=1}^{m}V\left(\underline{p},\underline{l}_i\right)\ddot{\underline{L}}_{\text{refl},d\&s}\left(\ddot{\underline{L}}_{i,d\&s},\ddot{r}_{d\&s},s\right)$$

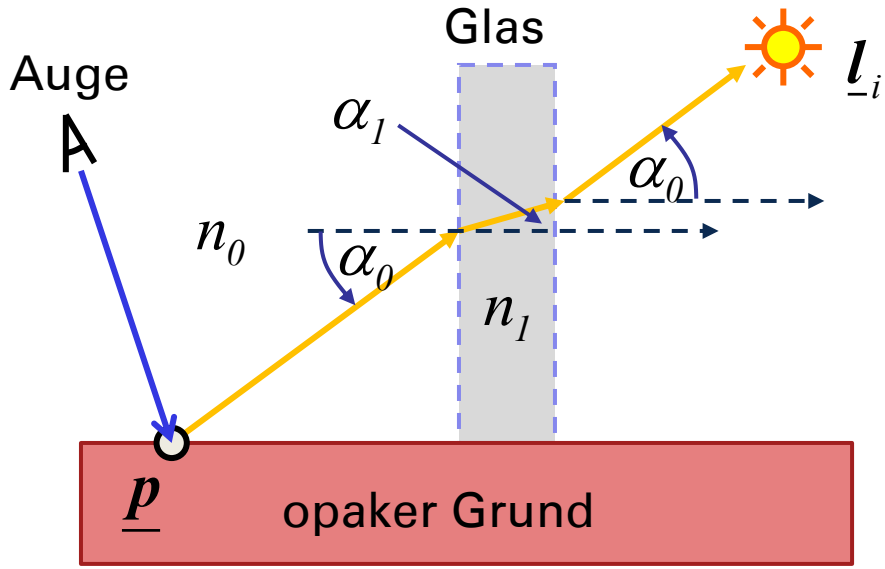
Visibilität — Lokales Beleuchtungsmodell

Strahlverfolgung
Schatten & Transparenz

- Der Visibilitätsfaktor ist 0 oder 1 je nachdem, ob der Schattenfühler geblockt wird oder nicht
- Blockiert ein transparentes Objekt den Schattenfühler, so kann man auch einen Visibilitätsfaktor berechnen, der angibt wie viel Prozent des Lichts durch das transparent Objekt hindurchkommt.
- Dazu wird die Länge des Durchganges durch das transparente Objekt mit einem vom Objekt abhängigen Absorpotionsfaktor multiplizert. Zusätzlich wird zweimal der Transmissionskoeffizient multipliziert.
- Das ist allerdings nur eine Approximation, weil der Schattenfühler an der Oberfläche des transparenten Objekts zwei mal gebrochen wird. Die Brechung ist zu kompliziert um berücksichtigt zu werden







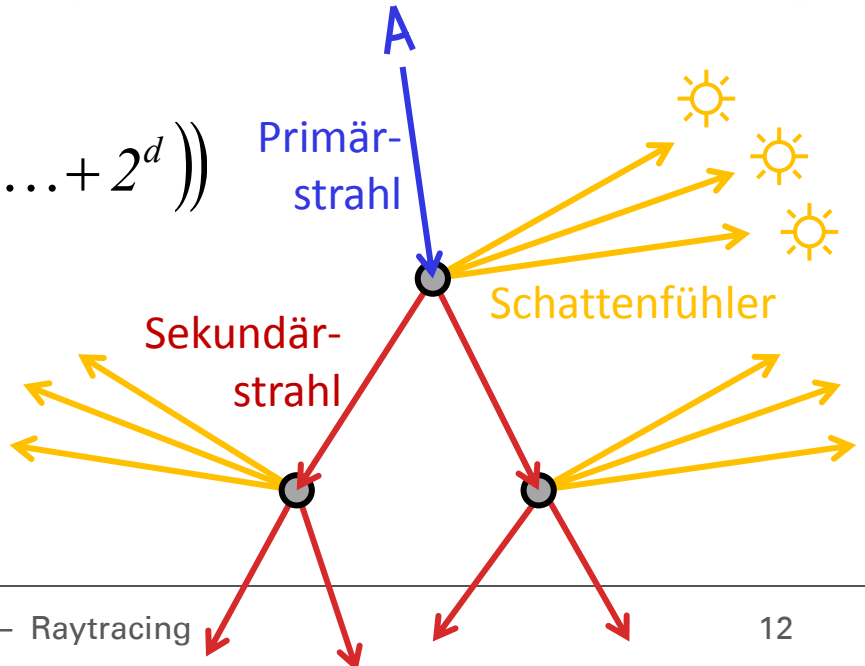
Strahlverfolgung
Komplexität

- Kenngrößen
  - w, h … Breite und Höhe des Ausgabebildes in Pixel
  - s … supersampling Faktor
  - n … Anzahl der Objekte in der Szene
  - m … Anzahl der Lichtquellen
  - d … maximale Rekursionstiefe
- worst case
  - Anzahl Strahlen:

$$\#_{\text{rays}} = \mathrm{O}\left(w \cdot h \cdot s \cdot (1+m) \cdot \left(1+2+4+\ldots+2^{d}\right)\right)$$
$$= \mathrm{O}\left(w \cdot h \cdot s \cdot (1+m) \cdot 2^{d+1}\right)$$

  - Anzahl Berechnungen:

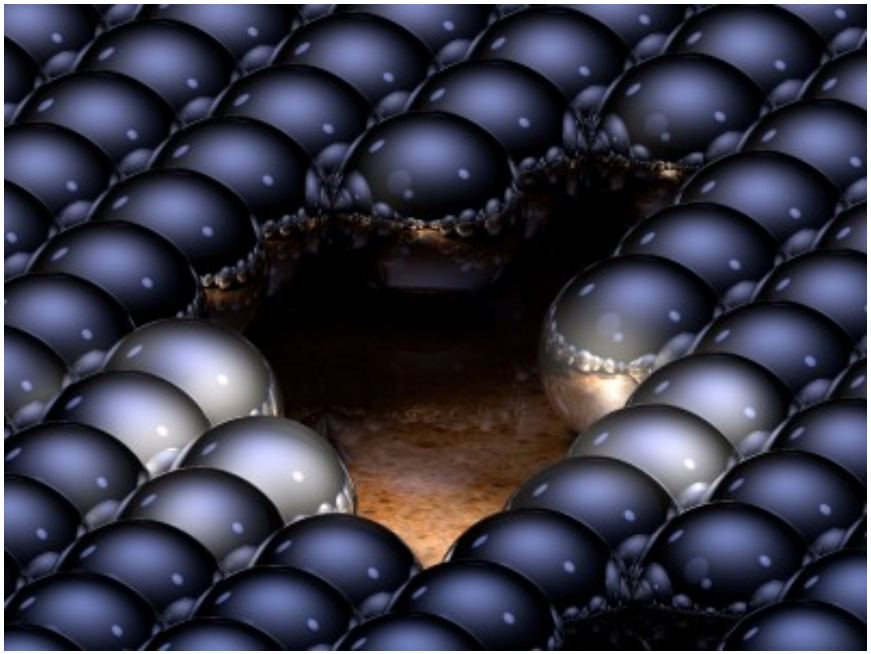
$$\#_{\text{comp}} = \mathrm{O}\left(n \cdot \#_{\text{rays}}\right)$$



Strahlverfolgung
Komplexität – Reduktion

- Zur Reduktion der Komplexität beschränkt man typischerweise die maximale Rekursionstiefe
- Außerdem werden Beschleunigungsdatenstrukturen verwendet, um die Schnittberechnung zwischen einem Strahl und allen Objekten in der Szene zu Beschleunigen. Damit kann die Schnittberechnung Strahl-Szene auf $\log n$ reduziert werden
- Resultierende Laufzeit:

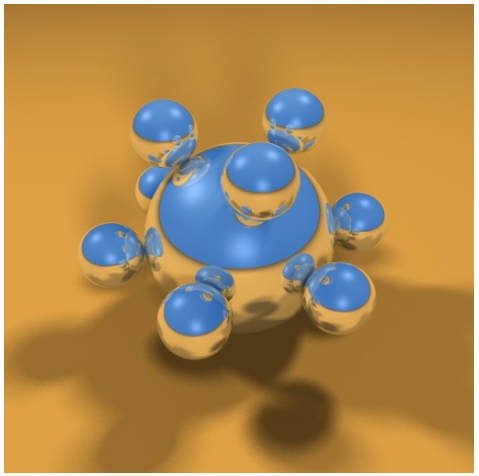
$$\#_{\text{comp}} = \mathrm{O}\left(w \cdot h \cdot s \cdot m \cdot \log n\right)$$

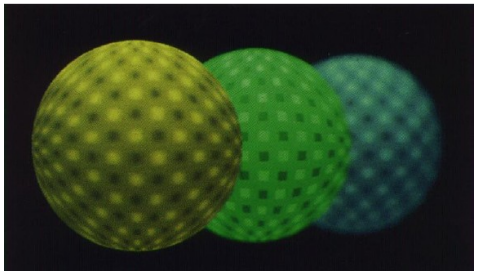


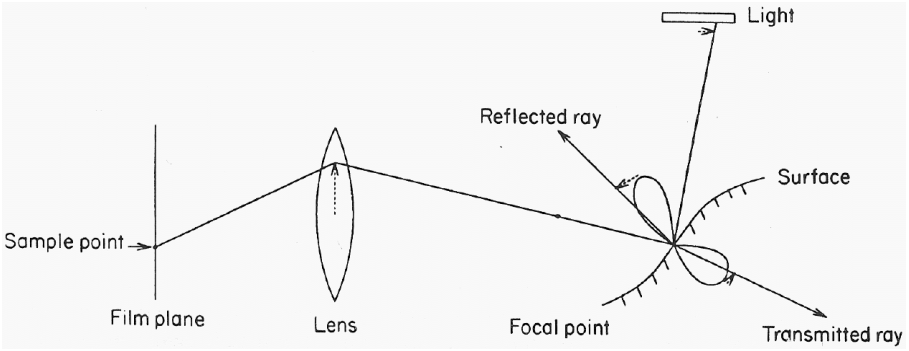
Rekursionstiefe auf 16 eingeschränkt

Strahlverfolgung
Distribution Raytracing

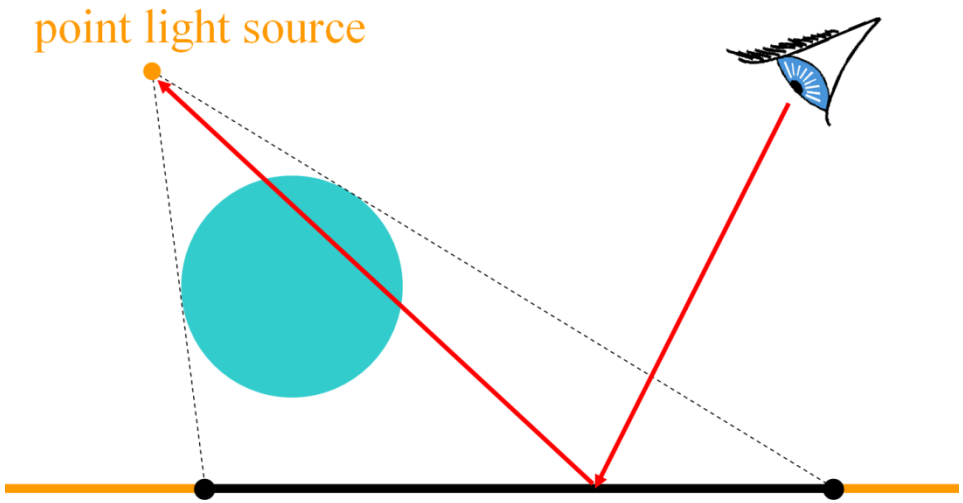
- Das Supersampling eines Pixels kann auch verwendet werden, um weiche Schatten, Motion Blur und Tiefenunschärfe zu modellieren
- Dazu wird für jedes Sample eines Pixels zufällig die jeweiligen Parameter ausgewählt:
  - weiche Schatten … Position auf der Flächenlichtquelle
  - Motion Blur … Zeit aus dem zu integrierenden Zeitintervall
  - Tiefenunschärfe … Position auf der ausgedehnten Blendenöffnung
- Dieses Verfahren nennt man Distribution Raytracing

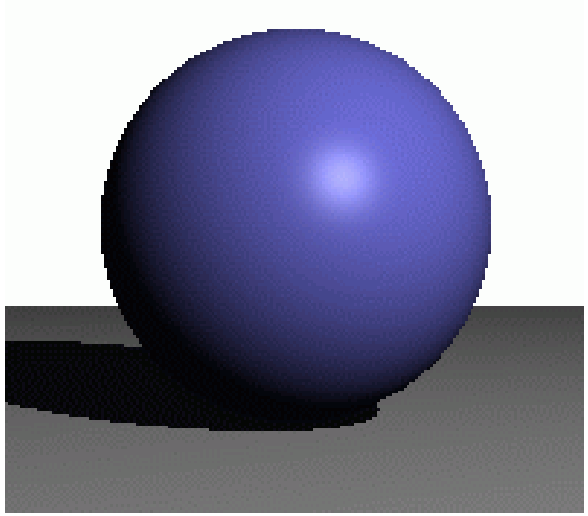


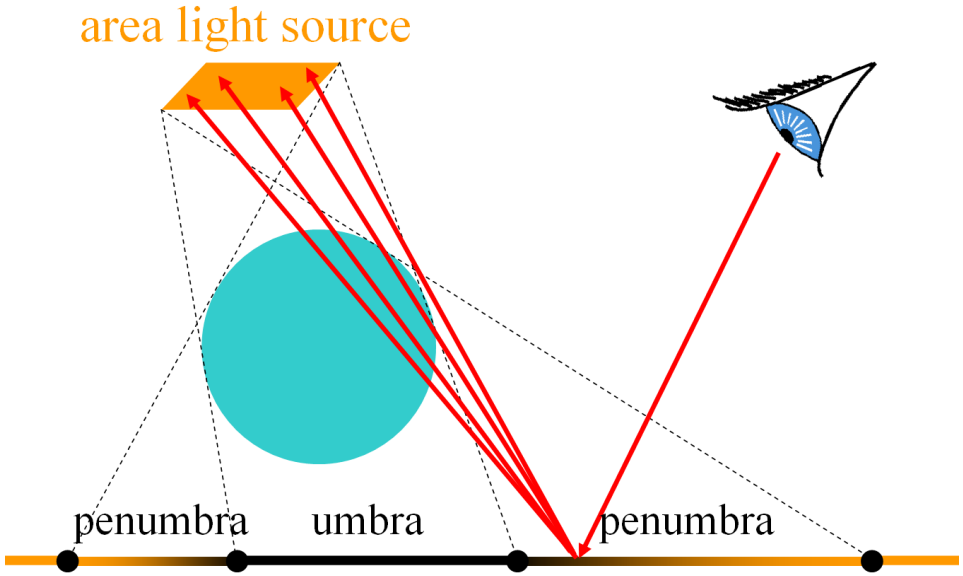


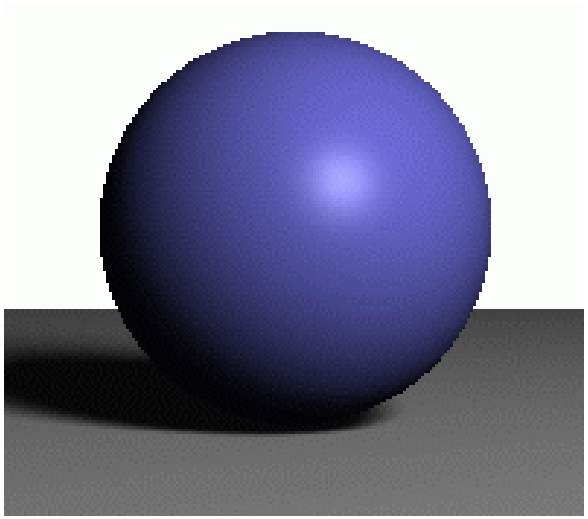


Distribution Raytracing
Weiche Schatten



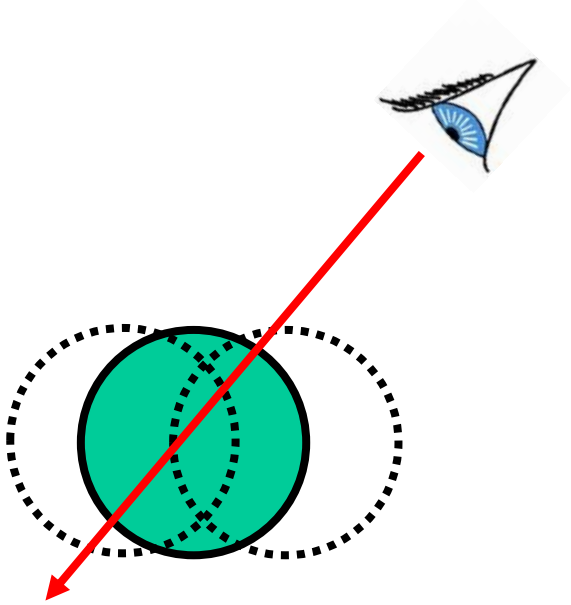






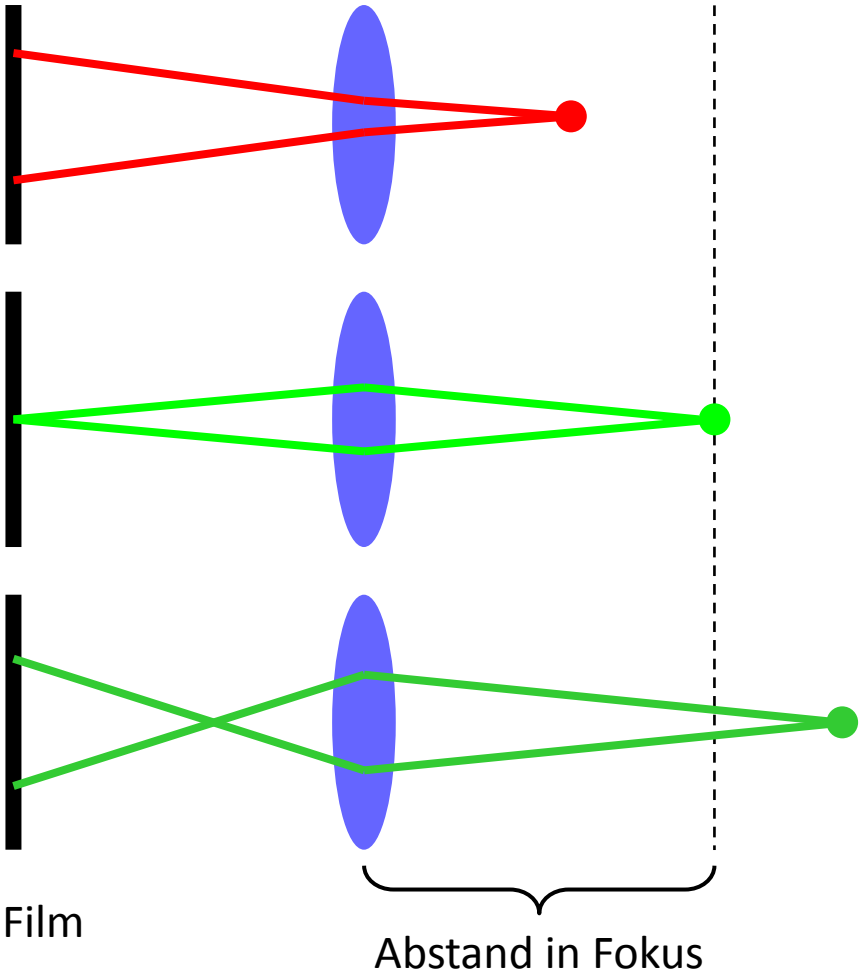
Distribution Raytracing
Motion Blur

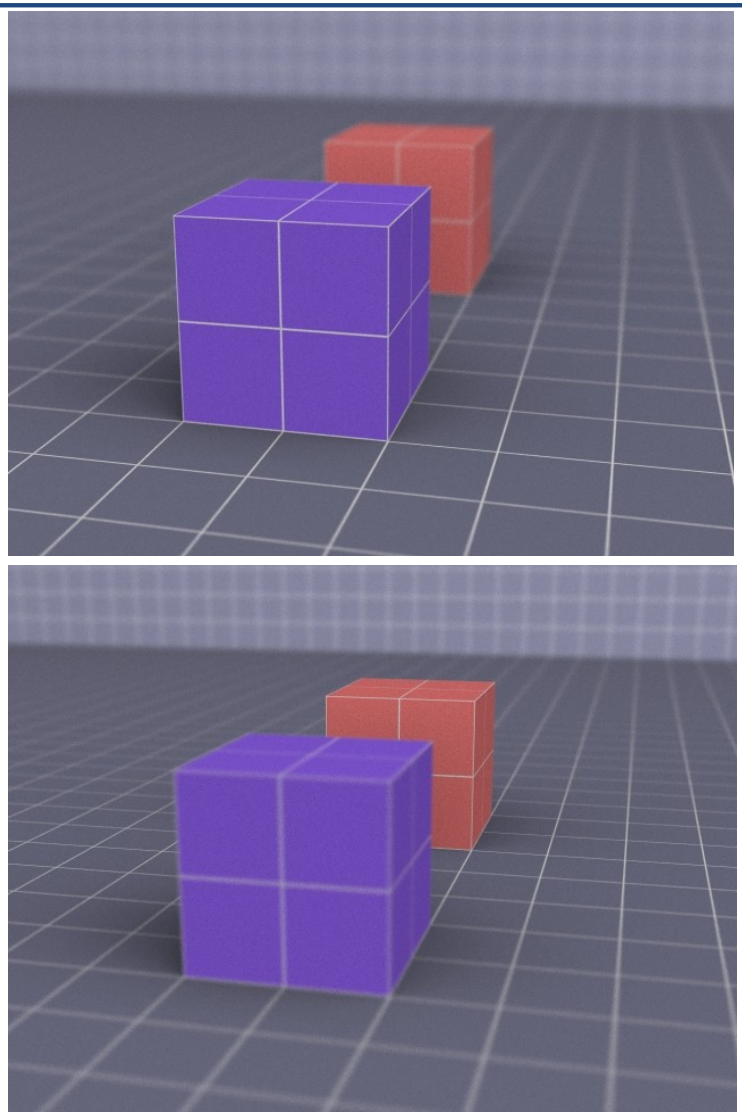
- Für jeden Strahl wird zufällig ein Zeitpunkt in dem darzustellenden Zeitintervall gewählt.
- Vorsicht: hier müssen auch die bewegten Objekte pro Strahl neu positioniert werden.



Rob Cook

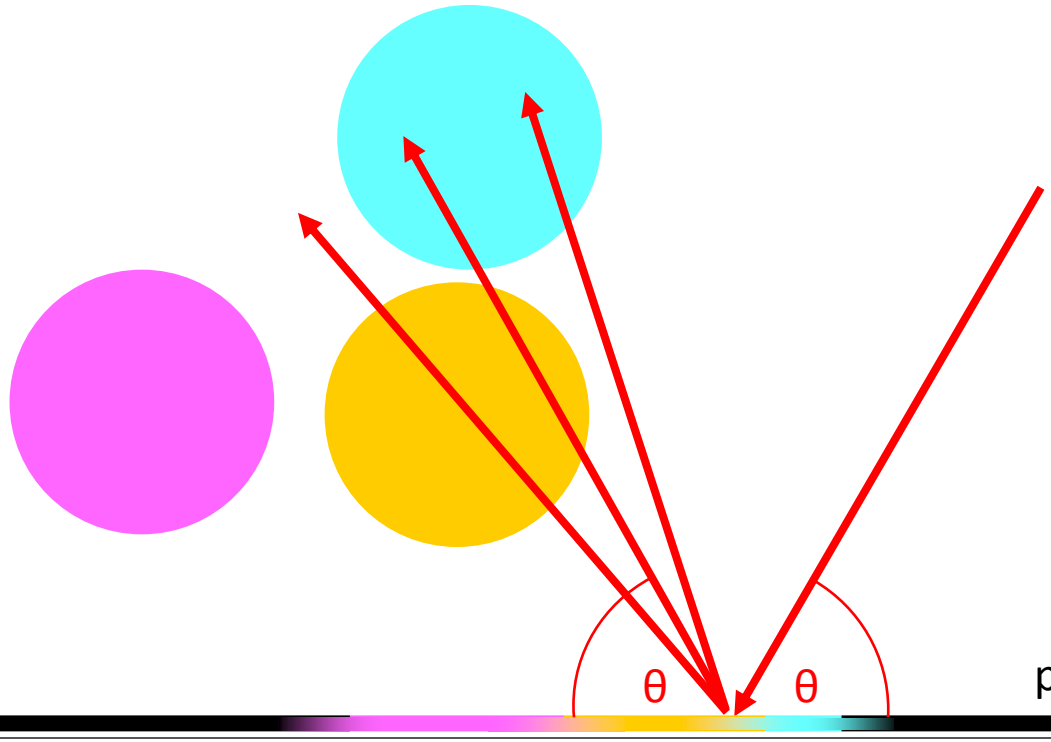
Distribution Raytracing
Tiefenunschärfe



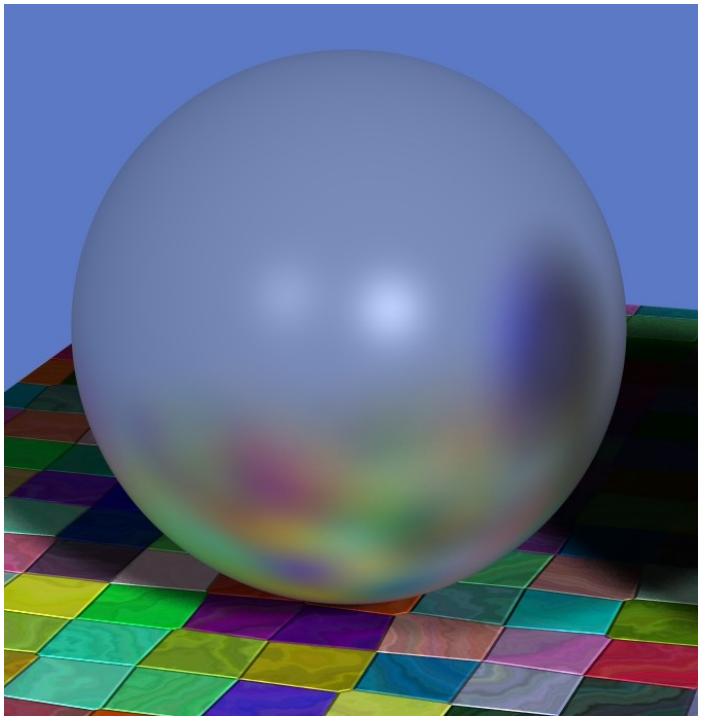


© Justin Legakis

Distribution Raytracing diffuse Spiegelung



polished surface



© Justin Legakis

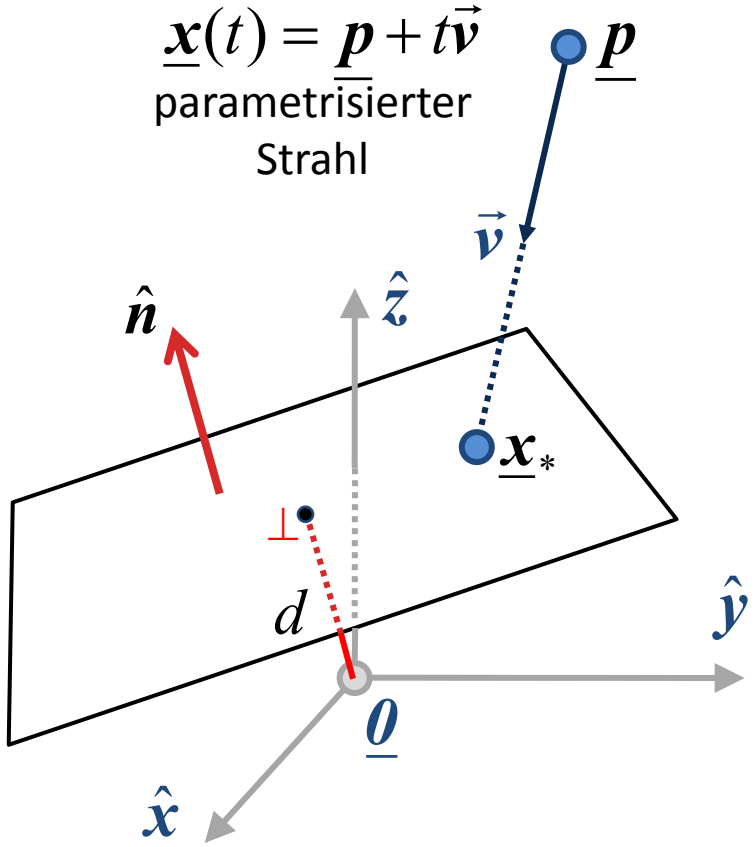
Schnittberechnung Ebene

- Bei vielen Strahl-Primitiv-Schnittberechnungen setzt man die parametrische Form des Strahls in eine implizite Darstellung des Primitives und berechnet den Strahlparameter:

$$\left\langle \hat{\boldsymbol{n}}, \underline{\boldsymbol{p}} + t_* \vec{\boldsymbol{v}} \right\rangle = d \Rightarrow t_* = \frac{d - \left\langle \hat{\boldsymbol{n}}, \underline{\boldsymbol{p}} \right\rangle}{\left\langle \hat{\boldsymbol{n}}, \vec{\boldsymbol{v}} \right\rangle}$$

- Als Ergebnis erhält man keinen, einen oder mehrere Schnittparameter (bei Ebene maximal einen). Daraus sucht man den mit dem kleinsten positiven t_{fst}
- Gibt es ein t_{fst}, berechnet man den Schnittpunkt und die Normale des Primitives am Schnittpunkt. Im Falle der Ebene:

$$\underline{\boldsymbol{x}}_{\text{fst}} = \underline{\boldsymbol{p}} + t_{\text{fst}} \vec{\boldsymbol{v}}, \quad \hat{\boldsymbol{n}}_{\text{fst}} = \hat{\boldsymbol{n}}$$



$$\left\langle \hat{\boldsymbol{n}}, \underline{\boldsymbol{x}} \right\rangle = d$$

Hesse'sche Normalform

Schnittberechnung Kugel

- Beim Schnitt von Strahl mit Kugel geht man genauso vor:

$$\left\|\left(\underline{p}+t_*\vec{v}\right)-\underline{c}\right\|^2 = r^2 \quad \Rightarrow$$

$$\left\|\vec{v}\right\|^2 t_*^2 + 2\left\langle \vec{v}, \underline{p}-\underline{c}\right\rangle t_* + \left\|\underline{p}-\underline{c}\right\|^2 - r^2 = 0$$

$$at_*^2 + bt_* + c = 0$$

- Numerisch kann dies mit folgender Fallunterscheidung stabiler gelöst werden:

$$t_1 = \frac{q}{a}, t_2 = \frac{c}{q}, \text{ mit}$$

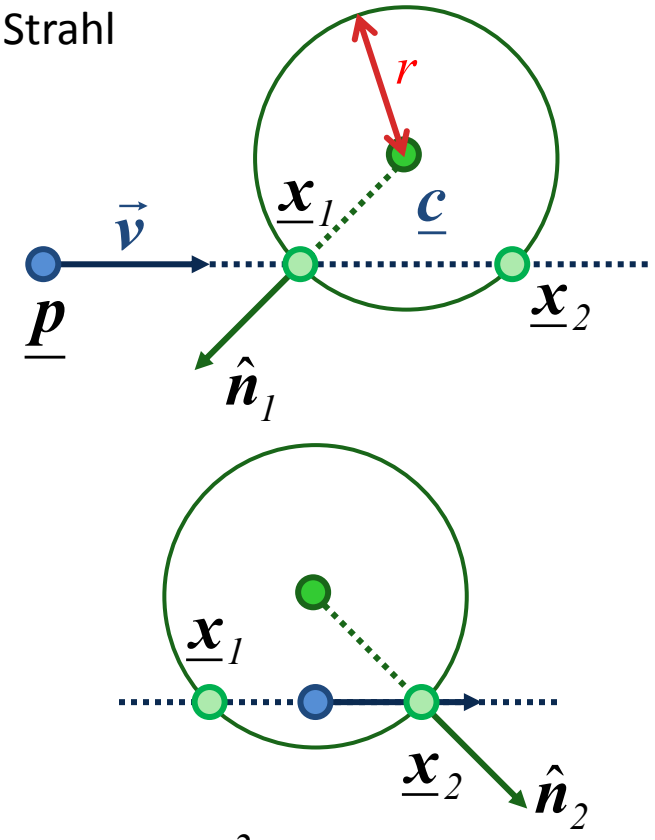
$$q = \begin{cases} -\frac{1}{2}\left(b - \sqrt{b^2-4ac}\right) \ldots b < 0 \\ -\frac{1}{2}\left(b + \sqrt{b^2-4ac}\right) \ldots \text{sonst} \end{cases}$$

- Nach Auswahl von t_{fst} ergibt sich die Normale zu

$$\hat{n} = \left(\underline{x}_{\text{fst}} - \underline{c}\right) / \left\|\left(\underline{x}_{\text{fst}} - \underline{c}\right)\right\|$$

$$\underline{x}(t) = \underline{p} + t\vec{v}$$

parametrisierter Strahl

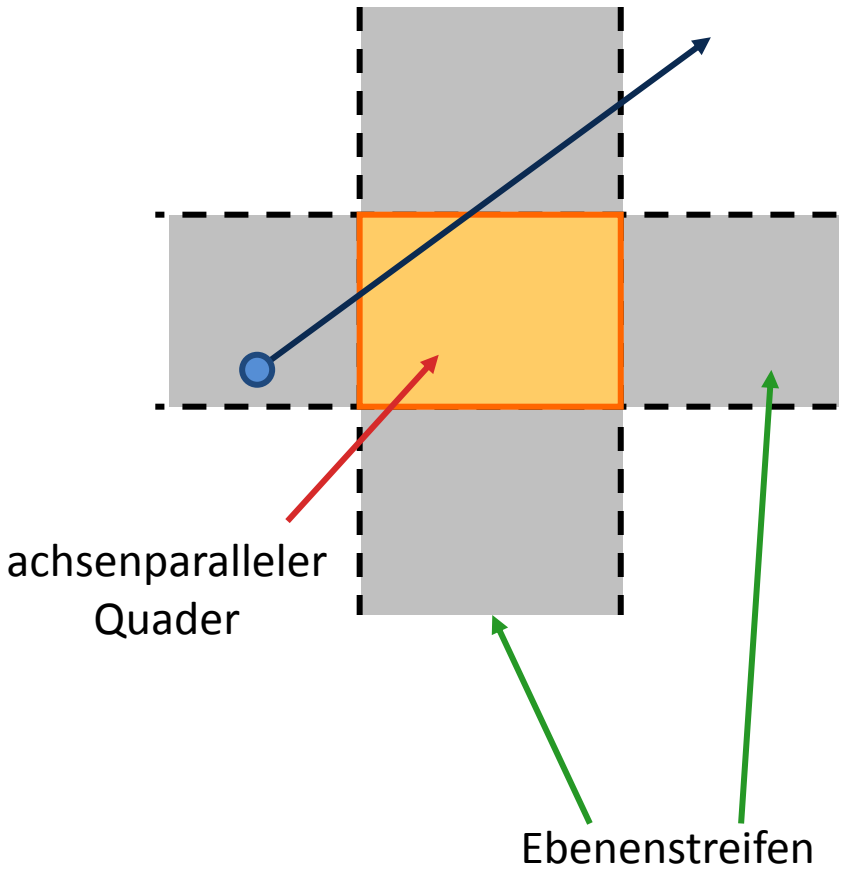


$$\left\|\underline{x}-\underline{c}\right\|^2 = r^2$$

implizite Darstellung der Kugel

Schnittberechnung Achsenparalleler Quader

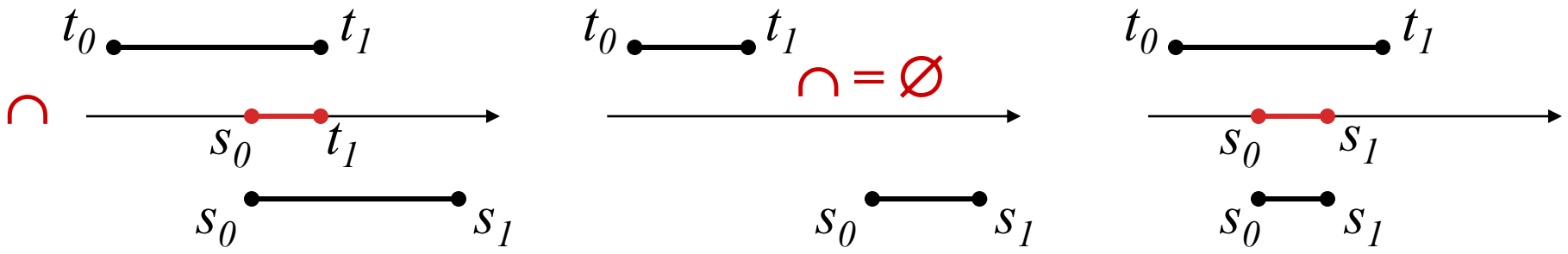
- Quader sind besonders wichtig für Hierarchien von Hüllvolumen.
- Idee: spalte Berechnung in Schnitt mit Ebenenstreifen und diese wiederum in Schnitt mit achsenparallelen Ebenen.
- Kombiniere Ergebnisse durch Mengendurchschnitt mittels Intervallarithmetik auf Intervallen des Strahlparameters t.



Schnittberechnung
Intervallarithmetik

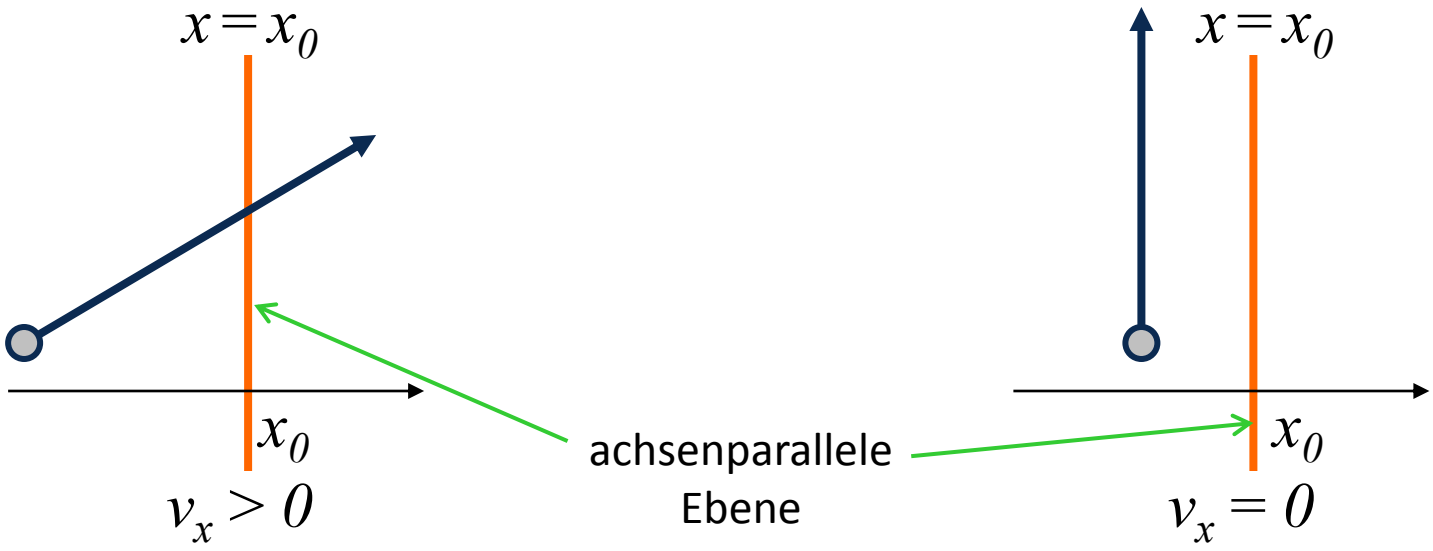
- Motivation: der Schnitt zwischen einem Strahl und einem beliebigen Objekt führt auf Intervalle im Parameter t.
- Parameterintervall: $T = [t_0, t_1]$ mit $t_0 \le t_1$ oder $T = \varnothing$
- Schnittoperation:

$$[t_0, t_1] \cap [s_0, s_1] = \begin{cases} \emptyset, & \text{wenn } \max\{t_0, s_0\} > \min\{t_1, s_1\} \\ [\max\{t_0, s_0\}, \min\{t_1, s_1\}], & \text{sonst} \end{cases}$$



Schnittberechnung
Achsenparallele Ebene

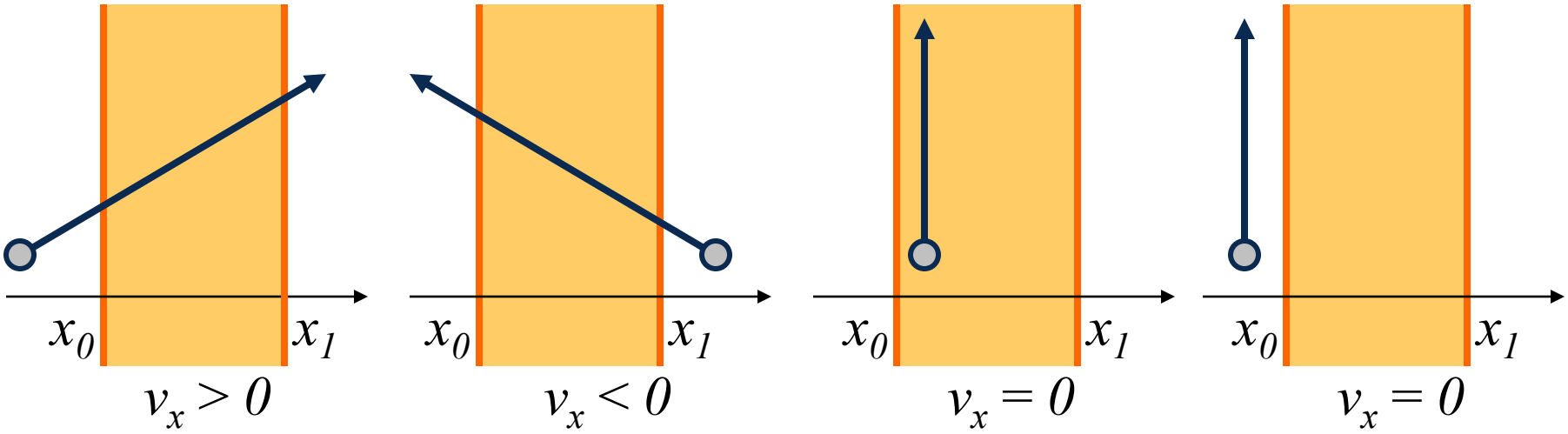
- Das Schnittproblem lässt sich auf eine Dimension reduzieren: der Strahl beginnt bei $x(t=0)=p_x$ und die x-Komponente wächst mit der Geschwindigkeit $\partial x/\partial t = v_x$
- Aus $x_0 = p_x + t_* \cdot v_x$ ergibt sich der Schnittparameter t_* zu: $t_* = \begin{cases} \dfrac{x_0 - p_x}{v_x} : v_x \neq 0 \\ \text{undef} : v_x = 0 \end{cases}$



Schnittberechnung Achsenparalleler Ebenenstreifen

- Intervall wird abhängig vom Vorzeichen von v_x berechnet
- Weitere Fallunterscheidung bei Parallelität

$$[t_0,t_1]=\begin{cases}\left[\dfrac{x_0-p_x}{v_x},\dfrac{x_1-p_x}{v_x}\right] & : v_x>0\\ \left[\dfrac{x_1-p_x}{v_x},\dfrac{x_0-p_x}{v_x}\right] & : v_x<0\\ [-\infty,+\infty] & : v_x=0 \wedge p_x\in[x_0,x_1]\\ \emptyset & : \text{sonst}\end{cases}$$

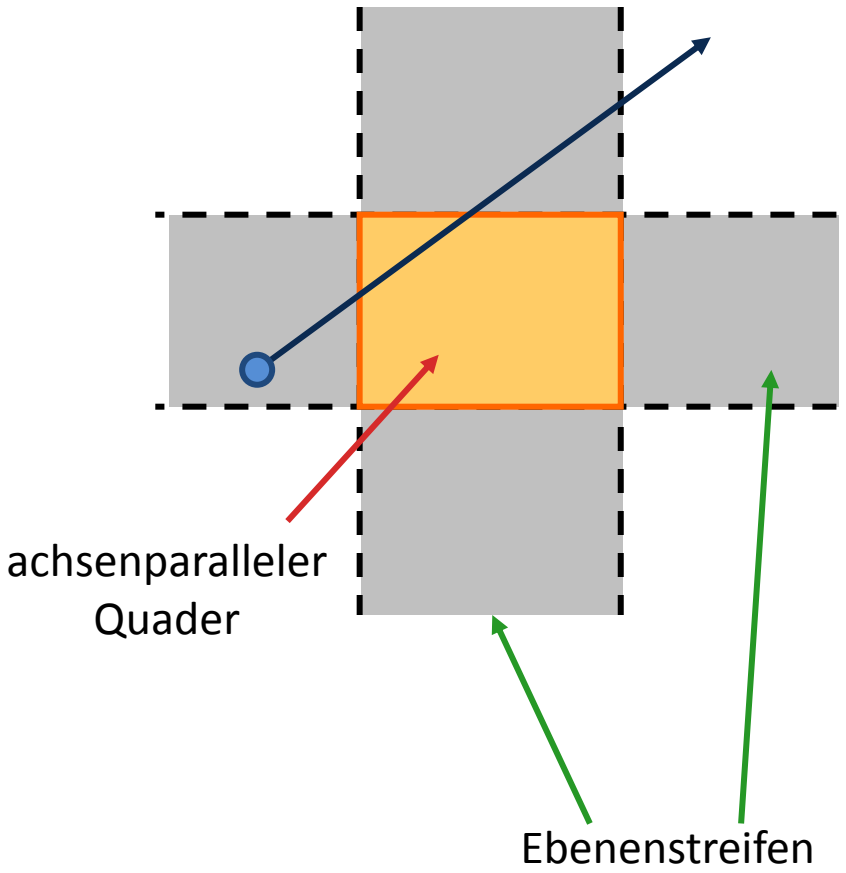


Schnittberechnung Achsenparalleler Quader

- Gegeben: Quader, definiert durch zwei Punkte aus den minimalen und maximalen Koordinaten: $\underline{\boldsymbol{x}}_{\min}, \underline{\boldsymbol{x}}_{\max}$
- Berechne Schnittintervall $T_{x/y/z}$ mit den drei Ebenenstreifen gemäß voriger Folie.
- Schnittintervall T_Q mit Quader ergibt sich aus dem Schnitt über alle drei Intervalle:

$$T_Q = T_x \cap T_y \cap T_z$$

- prüfe abschließend ob es ein $0 < t_{\text{fst}} \in T_Q$ gibt.
- die Normale ergibt sich entlang einer Koordinatenachse



Schnittberechnung Dreieck

- Setzt man $\sigma_2 = 1 - \sigma_0 - \sigma_1$ kann man den Strahlparameter t_* sowie σ_{0*} und σ_{1*} durch Gleichsetzen der parametrischen Strahldarstellung und der baryzentrischen Dreiecksdarstellung berechnen und anschließend testen, ob alle $\sigma_i > 0$ sind. Abschließend wird geprüft, ob $t_* > 0$ und falls ja, $\underline{\boldsymbol{x}}_{\text{fst}}$ und $\underline{\boldsymbol{n}}_{\text{fst}}$ berechnet:

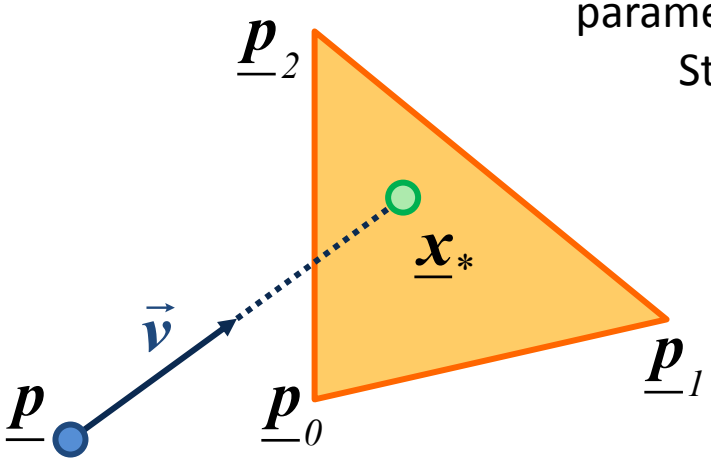
$$\hat{\boldsymbol{n}}_{\text{fst}} = \vec{\boldsymbol{n}}(\underline{\boldsymbol{p}}_0 \underline{\boldsymbol{p}}_1 \underline{\boldsymbol{p}}_2) / \left\| \vec{\boldsymbol{n}}(\underline{\boldsymbol{p}}_0 \underline{\boldsymbol{p}}_1 \underline{\boldsymbol{p}}_2) \right\|$$

mit der Definition

$$\vec{\boldsymbol{n}}(\underline{\boldsymbol{p}}_0 \underline{\boldsymbol{p}}_1 \underline{\boldsymbol{p}}_2) = (\underline{\boldsymbol{p}}_1 - \underline{\boldsymbol{p}}_0) \times (\underline{\boldsymbol{p}}_2 - \underline{\boldsymbol{p}}_0)$$

$$\underline{\boldsymbol{x}}(t) = \underline{\boldsymbol{p}} + t\vec{\boldsymbol{v}}$$

parametrisierter Strahl



$$\underline{\boldsymbol{x}} = \sigma_0 \underline{\boldsymbol{p}}_0 + \sigma_1 \underline{\boldsymbol{p}}_1 + \sigma_2 \underline{\boldsymbol{p}}_2, \forall i : \sigma_i \geq 0$$

Dreieck in baryzentrischer Darstellung

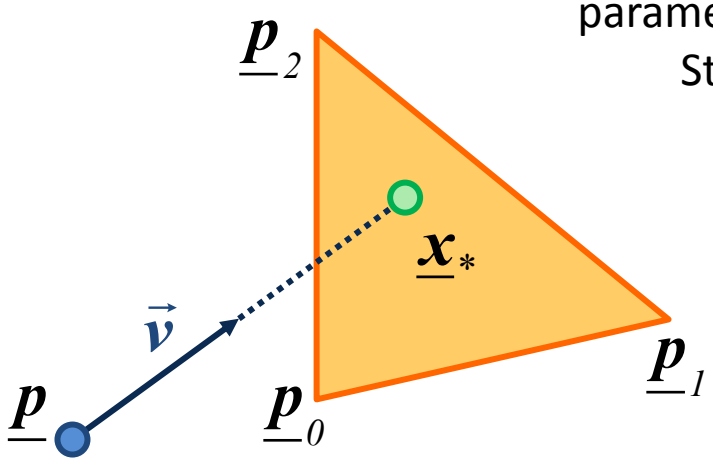
Schnittberechnung Dreieck II

- Man kann optional auch den Schnittparameter t_* wie beim Schnitt mit einer Ebene bestimmen und die σ_{0*} direkt aus $\underline{\boldsymbol{x}}_*$ und den $\underline{\boldsymbol{p}}_i$ berechnen:

$$t_* = \frac{\left\langle \vec{n}(\underline{p}_0\,\underline{p}_1\,\underline{p}_2), \underline{p}_0 - \underline{p} \right\rangle}{\left\langle \vec{n}(\underline{p}_0\,\underline{p}_1\,\underline{p}_2), \vec{v} \right\rangle}$$

$$\underline{\boldsymbol{x}}(t) = \underline{\boldsymbol{p}} + t\vec{\boldsymbol{v}}$$

parametrisierter Strahl



$$\sigma_0 = f \cdot \vec{n}(\underline{p}_0\,\underline{p}_1\,\underline{p}_2) \cdot \vec{n}(\underline{x}_*\,\underline{p}_1\,\underline{p}_2)$$

$$\sigma_1 = f \cdot \vec{n}(\underline{p}_0\,\underline{p}_1\,\underline{p}_2) \cdot \vec{n}(\underline{p}_0\,\underline{x}_*\,\underline{p}_2)$$

$$\sigma_2 = f \cdot \vec{n}(\underline{p}_0\,\underline{p}_1\,\underline{p}_2) \cdot \vec{n}(\underline{p}_0\,\underline{p}_1\,\underline{x}_*)$$

$$\underline{\boldsymbol{x}} = \sigma_0\,\underline{\boldsymbol{p}}_0 + \sigma_1\,\underline{\boldsymbol{p}}_1 + \sigma_2\,\underline{\boldsymbol{p}}_2, \forall i : \sigma_i \geq 0$$

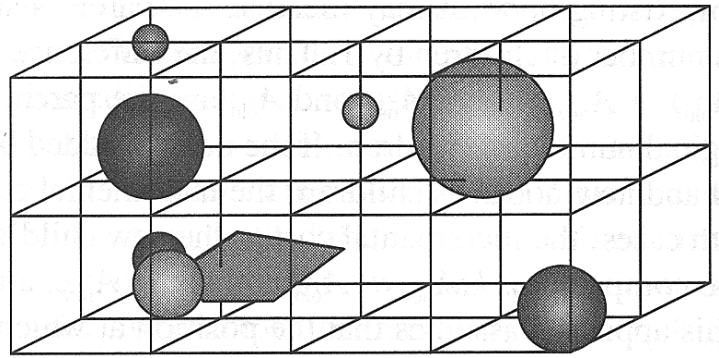
Dreieck in baryzentrischer Darstellung

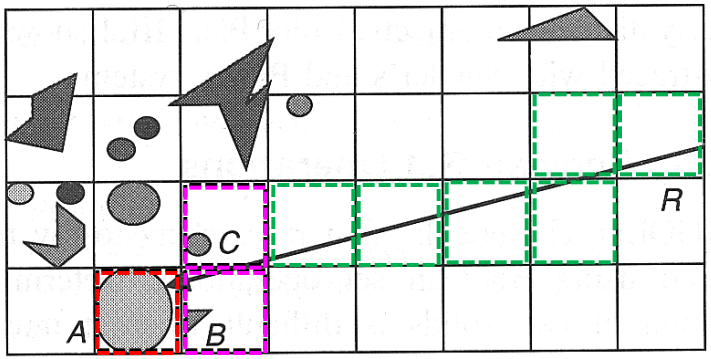
$$\text{mit } f = \frac{1}{\left\| \vec{n}(\underline{p}_0\,\underline{p}_1\,\underline{p}_2) \right\|^2}$$

- Beim Vergleich $\sigma_i \geq 0$ kann f auch ignoriert werden

Beschleunigungstechniken
Gitter

- Unterteile Bounding Box in regelmäßiges Voxel
- Auflösung: oft $\sqrt[3]{n}$
- Objekte einfügen
  - füge in alle Voxel ein, die das Objekt überlappen
  - leicht optimierbar
- Traversierung
  - verfolge Voxel entlang des Strahls
  - schneide jeweils mit enthaltenen Objekten
  - terminiere wenn erster Schnitt gefunden wurde





Gitter
Strahlverfolgung

- verfolge Zeitpunkte der Voxelübergänge für jede Achse getrennt
- berechne Startzelle,

$$i = \text{floor}\left(\frac{p_x - x_{\min}}{\Delta x}\right), \Delta x = \frac{x_{\max} - x_{\min}}{n}$$

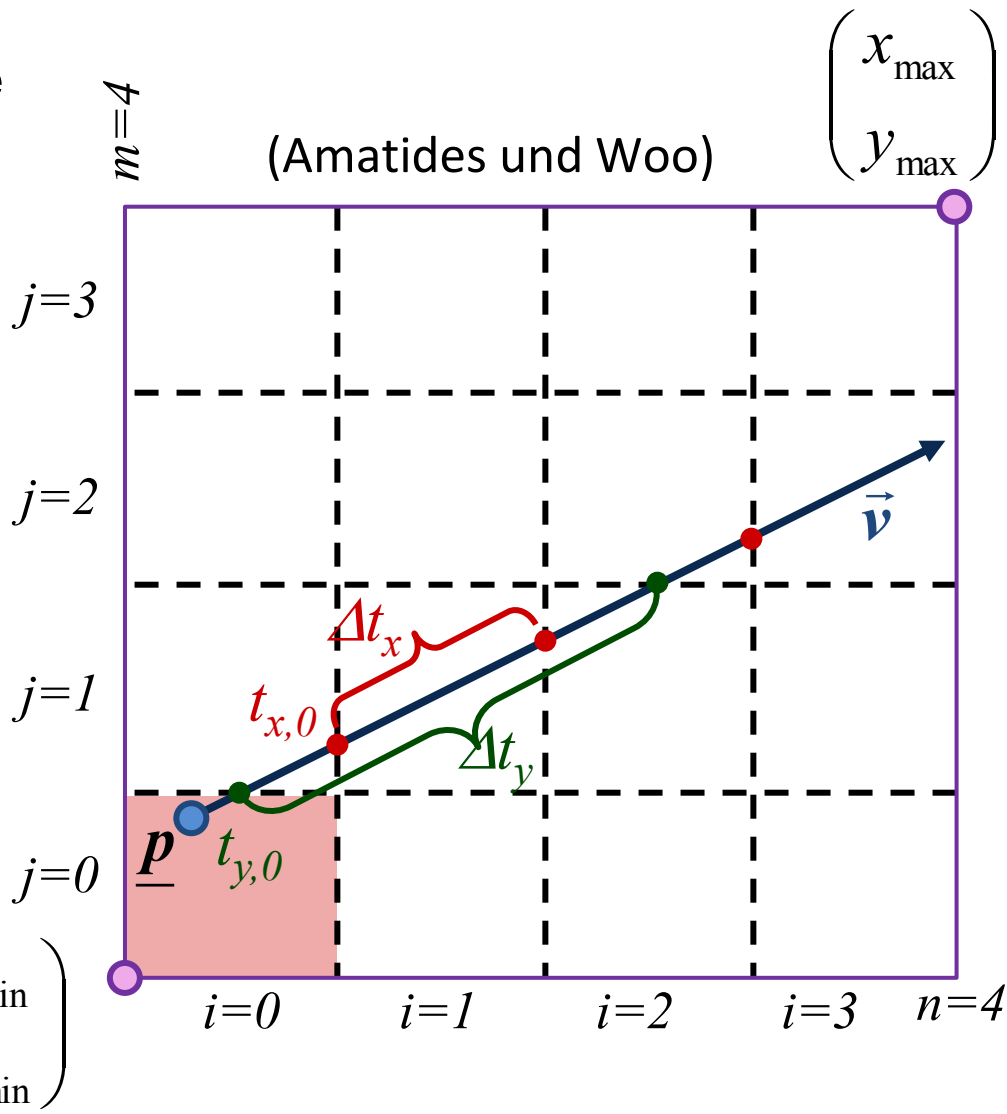
- erste Schnittzeiten $t_{\alpha,0}$ sowie

$$x_{\min} + (i+1)\Delta x - p_x = v_x t_{x,0}$$

- Zeitschritte Δt_α zum nächsten Wechsel für alle $\alpha \in \{x,y,z\}$

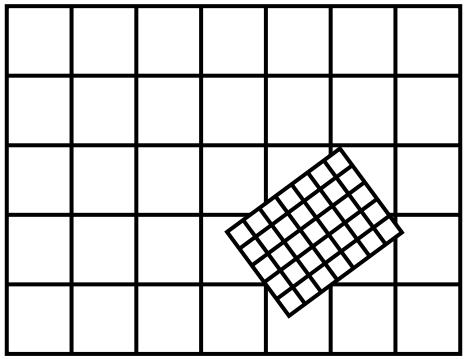
$$\Delta x = v_x \Delta t_x$$

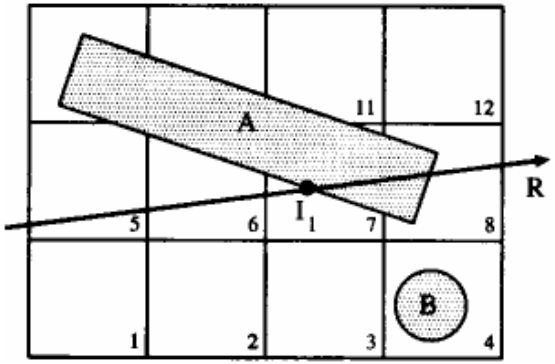
- verfolge Strahl entlang t und führe nächsten Wechsel durch

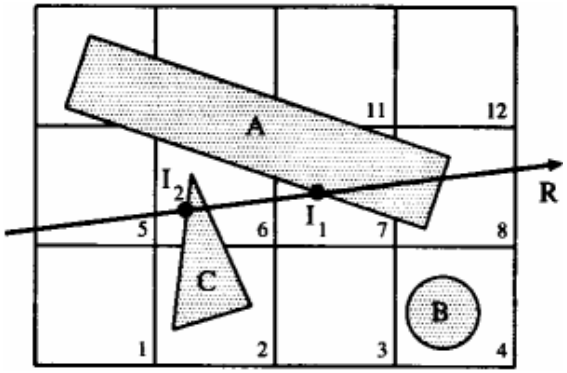


Gitter
Diskussion

- Gitterauflösung
  - hängt stark von Szene ab
  - keine Adaption an einen Fußball in einem Fußballstadium
  - mögliche Abhilfe: geschachtelte Gitter
- **Mailbox Technik**
  - Einsatz: wenn Objekte von mehreren umgebenden Volumen referenziert werden können, wie z.B. beim Gitter
  - Gebe jedem Strahl eine eindeutige ID und jedem Objekt eine Mailbox für letzte Schnittberechnung (ID, Schnittinfo)
  - Prüfe vor Schnittberechnung Mailbox des Objektes und speichere nach evtl. Neuberechnung Ergebnis in Mailbox

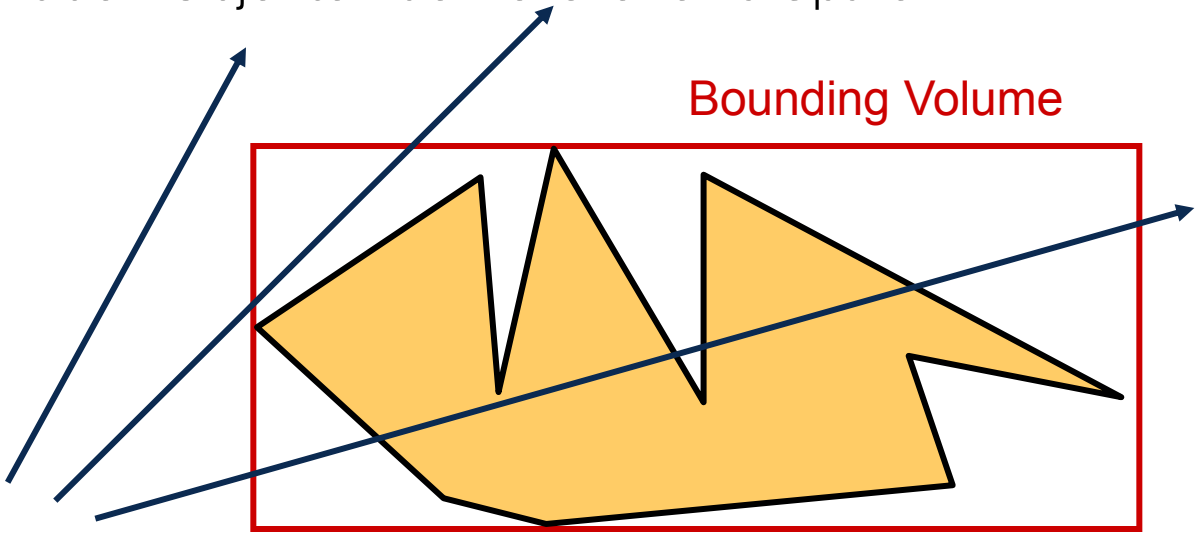






Beschleunigungstechniken Hüllvolumen

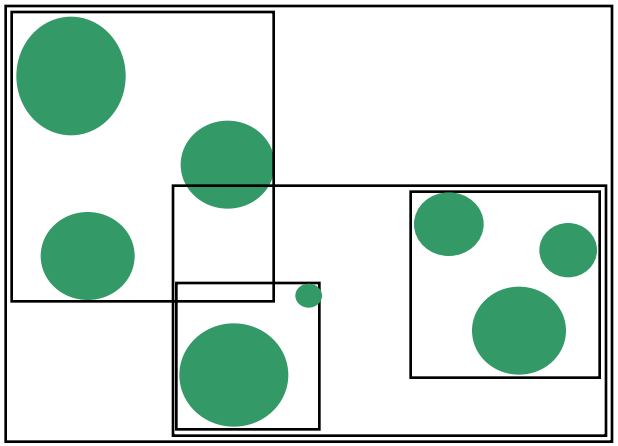
- Grundidee: Das Ziel besteht darin, Kosten bei den Schnittpunkttests eines Strahls mit den Objekten der Teilszene zu Sparen.

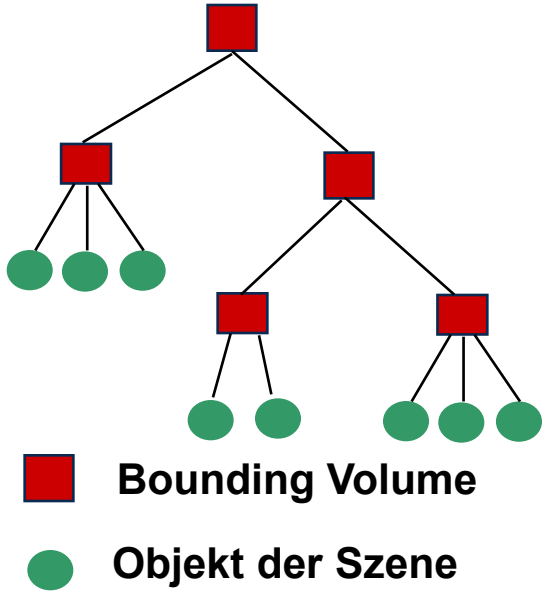


- Verfehlt der Strahl das Hüllvolume, so kann man auf den Schnitt mit der Teilszene verzichten
- Oft bietet es sich an, aus der Szene alle geometrischen Primitive (z.B. alles in Dreiecke zerlegen) in einem globalen Koordinatensystem zu extrahieren und darauf die Beschleunigungstechniken anzuwenden

Hüllvolumen Hierarchie

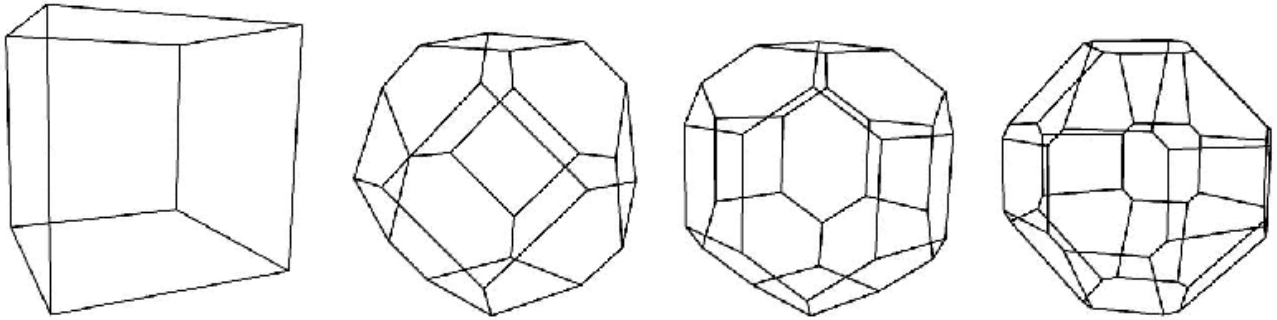
- Idee: Umschließe die Objekte in einer Hierarchie von sich zum Teil überlappenden Hüllvolumen (BVH)
- Teilbäume: Objekte nahe in der Szene beieinander
- minimiere Oberflächeninhalte der Hüllvolumen
- Fokus auf obere Knoten, weil hier das Abschneiden eines Teilbaumes mehr spart
- Berechnungszeit für Bild durch Raytracing sehr viel geringer trotz zusätzlicher Vorverarbeitung





Hüllvolumen
Entwurfskriterien

- Hüllvolumen werden so konstruiert, dass Teilobjekte komplett umschlossen werden ⇒ keine Mailbox-Techniken nötig
- Man muss abwägen zwischen
  - einfache Hüllvolumen (z.B. Kugel, Ellipsoid, Quader) mit kleinen Schnittkosten haben relativ hohe Strahltrefferzahlen
  - enganliegende Hüllvolumen (z.B. konvexe Hülle) mit kleiner Strahltrefferzahl haben hohe Schnittkosten.
- Einen guten Kompromiss ergeben k-Dops:

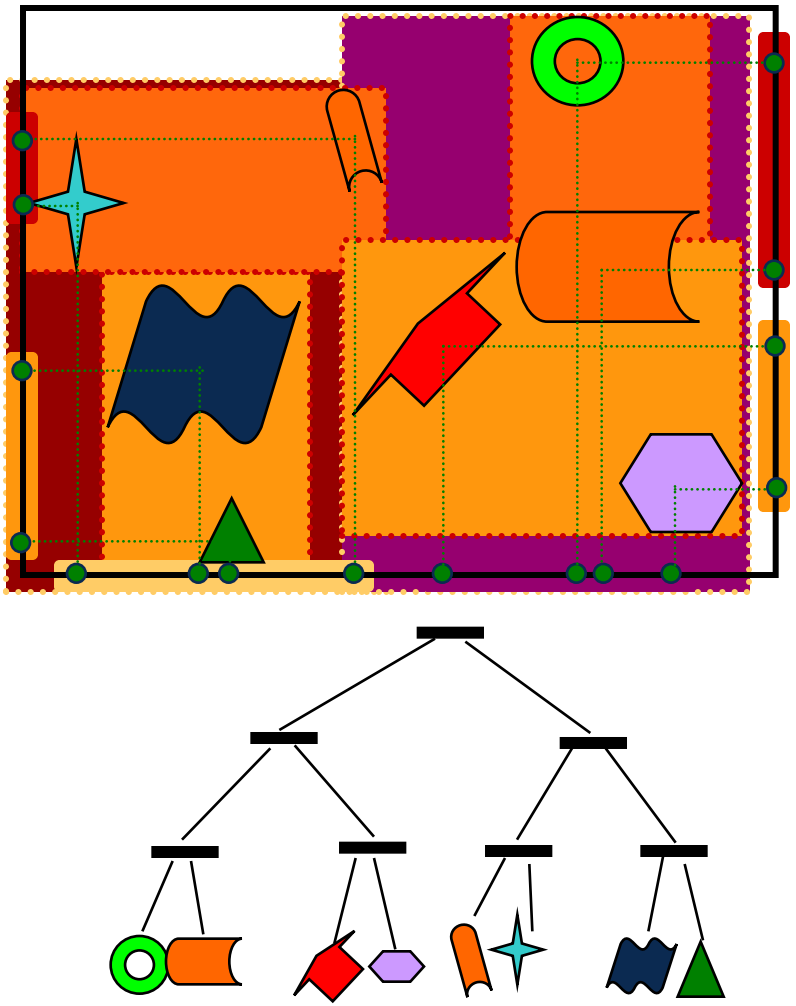


6-DOP 14-DOP 18-DOP 26-DOP

Hüllvolumen
AABB

Aufbau (top-down):

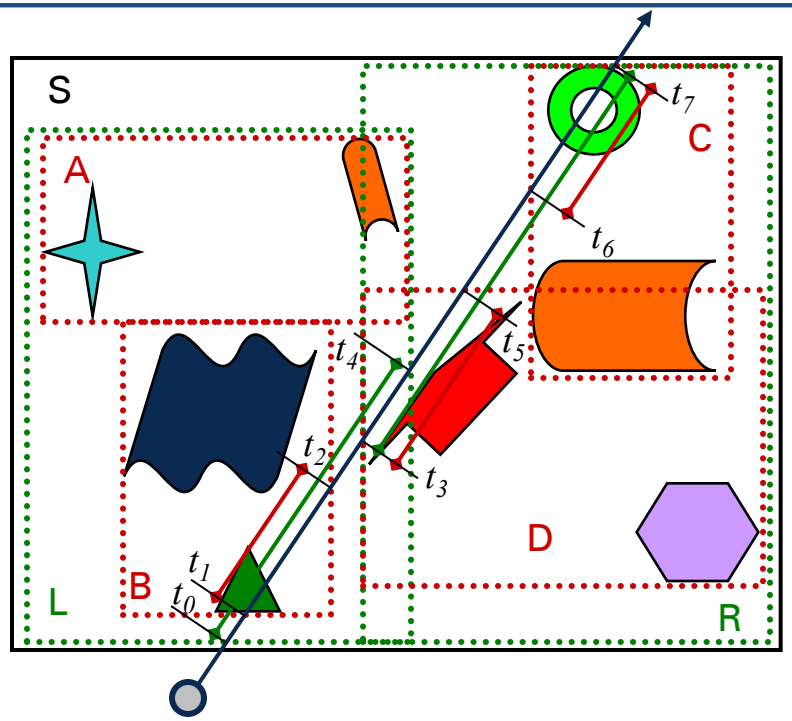
- Umschließe alle Objekte mit axis aligned bounding box (AABB) der Szene
- Spalte Box entlang erster Koordinatenrichtung und verteile Objekte gleichmäßig
- berechne Kindboxen so, dass sie die enthaltenen Objekte ganz umschließen
- dadurch können sich die Kind-boxen gegenseitig überlagern
- Rekursion mit nächster Koordinatenrichtung bis die Anzahl der Objekte in den Kindboxen klein genug



AABB Traversierung

Traversierung nach Kay et al.:

- initialisiere Heap von zu testenden AABBs mit Wurzel und setze gesuchten Schnittparameter auf $t_{fst}=\infty$
- sortiere Heap nach Schnittparameter t_{in} so, dass zuerst die AABB geprüft wird, die als erstes getroffen wird
- AABB ohne Schnitt werden ignoriert
- Test der nächstgelegen AABB
  - wenn $\min t_{in} > t_{fst}$ ist kein weiterer Test nötig
  - bei Blattknoten werden Schnitte mit enthaltenen Objekte berechnet und evtl. t_{fst} erneuert
  - sonst beide Kind-BVs in Heap eingetragen



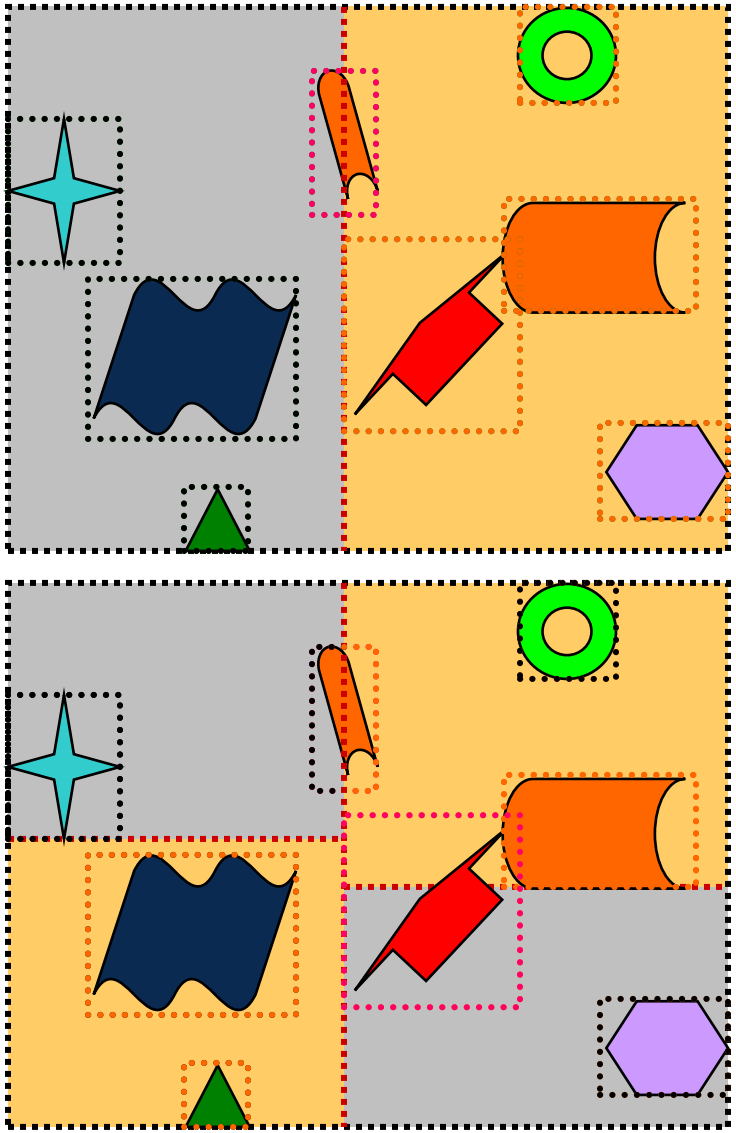
Beispieltraversierung

- push(S,t_0,t_7), $t_{fst}=\infty$
- pop S, $t_0 \le t_{fst}$, push(L,t_0,t_4), push(R,t_3,t_7)
- pop L, $t_0 \le t_{fst}$, A is skipped, push(B,t_1,t_2)
- pop B, $t_1 \le t_{fst}$, intersect objects, $t_{fst} := t_1$
- pop R, $t_3 > t_{fst}$, skip
- terminate

Beschleunigungstechniken
KD-Baum

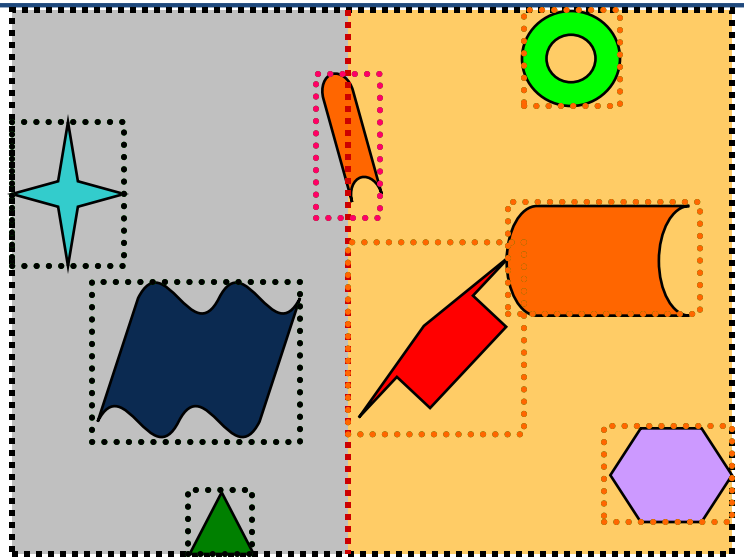
Aufbau

- Berechne für jedes Primitiv eine AABB und daraus die AABB der Szene
- Splitte AABB der Szene rekursiv an einer achsenparallelen Ebene in Binärbaum
- Optimierungspotential pro Split:
  - eine von drei Hauptachse
  - Position der Splittebene (man kann zeigen, dass es sich hier um eine Grenzen der enthaltenen AABBs handeln muss)
- Optimiere nach minimalen Kosten für den Schnitt mit einem zufällig gewählten Strahl



KD-Baum Kostenfunktion

- Pro Knoten zu entscheiden: wird gesplittet oder nicht, und wenn ja, dann in welcher Richtung und wo.
- Berechnungsmodell pro Knoten:
  - N Primitive mit Schnittkosten t_i
  - Kosten für Traversierung eines inneren Knotens: t_t
  - Wahrscheinlichkeit, dass ein zufälliger Strahl das Kind A, B schneidet, unter der Bedingung, dass der Elternknoten geschnitten wurde: p_A, p_B
- Kosten ohne Split: $$T_{leaf} = \sum_{i=1}^{N} t_i$$



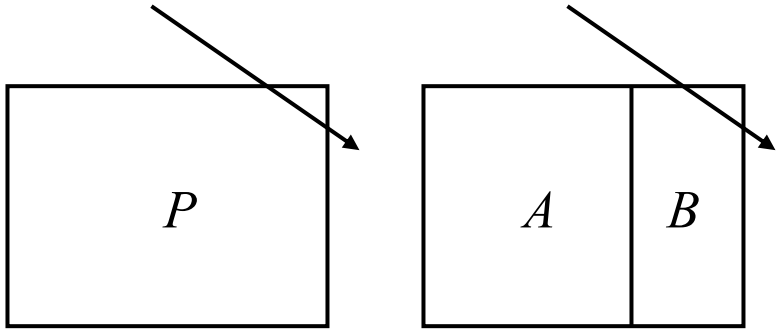
- Für alle 3 Hauptachsen werden die Knoten nach der minimalen Position sortiert.
- Gehen N_A Primitive in Kind A und überlappen N_O Primitive, so sind die Kosten bei einem Split:

$$T_{split} = t_t + p_A \sum_{i=1}^{N_A} t_i + p_B \sum_{i=N_A-N_O+1}^{N} t_i$$

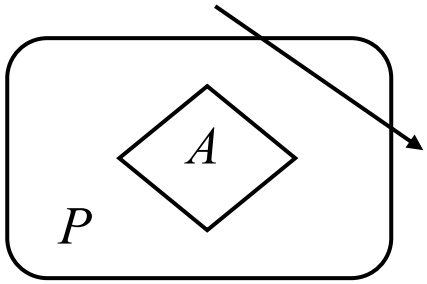
KD-Baum
Optimierung mit SAH

- Optimiert wird mit Hilfe einer vollständigen Suche über alle Hauptachsen und alle Grenzpositionen der AABBs der Primitive.
- Die noch fehlenden Wahrscheinlichkeiten p_A und p_B ergeben sich aus dem Quotient der Oberflächenverhältnisse von Kind durch Elternknoten (**S**urface **A**rea **H**euristic), wegen:
- **Satz:** Gegeben sei ein konvexes Volumen P mit Oberfläche s_P und ein darin enthaltenes konvexes Volumen A mit s_A. Ein zufällig gewählter Strahl, der P schneidet, schneidet A mit der bedingten Wahrscheinlichkeit

$$p(A \mid P) = \frac{s_A}{s_P}$$

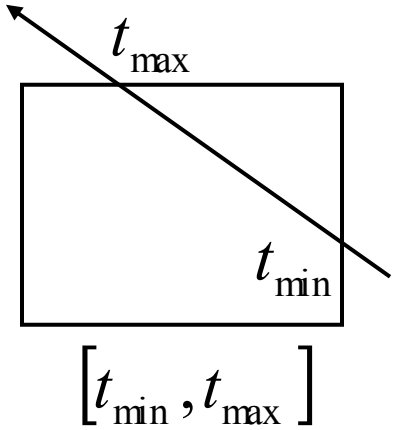


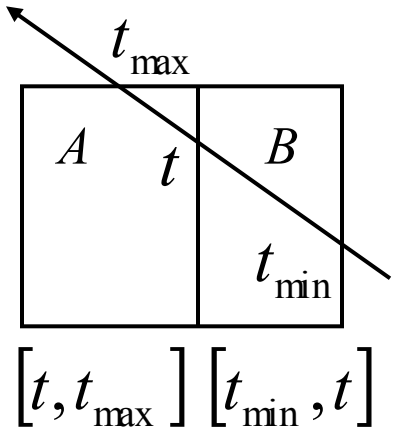
$$p_A = \frac{s_A}{s_P} \qquad p_B = \frac{s_B}{s_P}$$



KD-Baum
Traversierung beim Schnitttest

- Bei der Traversierung müssen die Blattknoten nach aufsteigend sortiertem Strahlparameter geprüft werden, bis der erste Schnitt gefunden ist.
- Anfangs wird das Schnittintervall mit der Szenen-AABB berechnet und auf Abbruch geprüft.
- In den Blättern werden Tests mit den Primitiven durchgeführt und die Mailboxtechnik angewandt
- In inneren Knoten wird der Schnittparameter der Trennfläche berechnet und mit linkem und rechten Teilintervall rekursiv die Kinder traversiert.





- Das zum linken Intervall gehörige Kind, ist dasjenige auf der selben Seite der Splittebene wie der Strahlstartpunkt