Sphere-Tracing/Ray-Marching in C++

C++ User-Treffen in Aachen, 17. Mai 2018

Mirco Müller, macslow@gmail.com

Was Euch erwartet

① Motivation

② Geschichte

③ Grundlagen in 2D

④ Schritt nach 3D

⑤ Kernalgorithmus & Objektdefinitionen

⑥ Beleuchtungsmodelle

⑦ Licht & Schatten

⑧ Reflexionen & Boolsche Operationen

⑨ Landschaften

⑩ Zusammenfassung, Aussicht auf Teil 2 & Fragen

Motivation

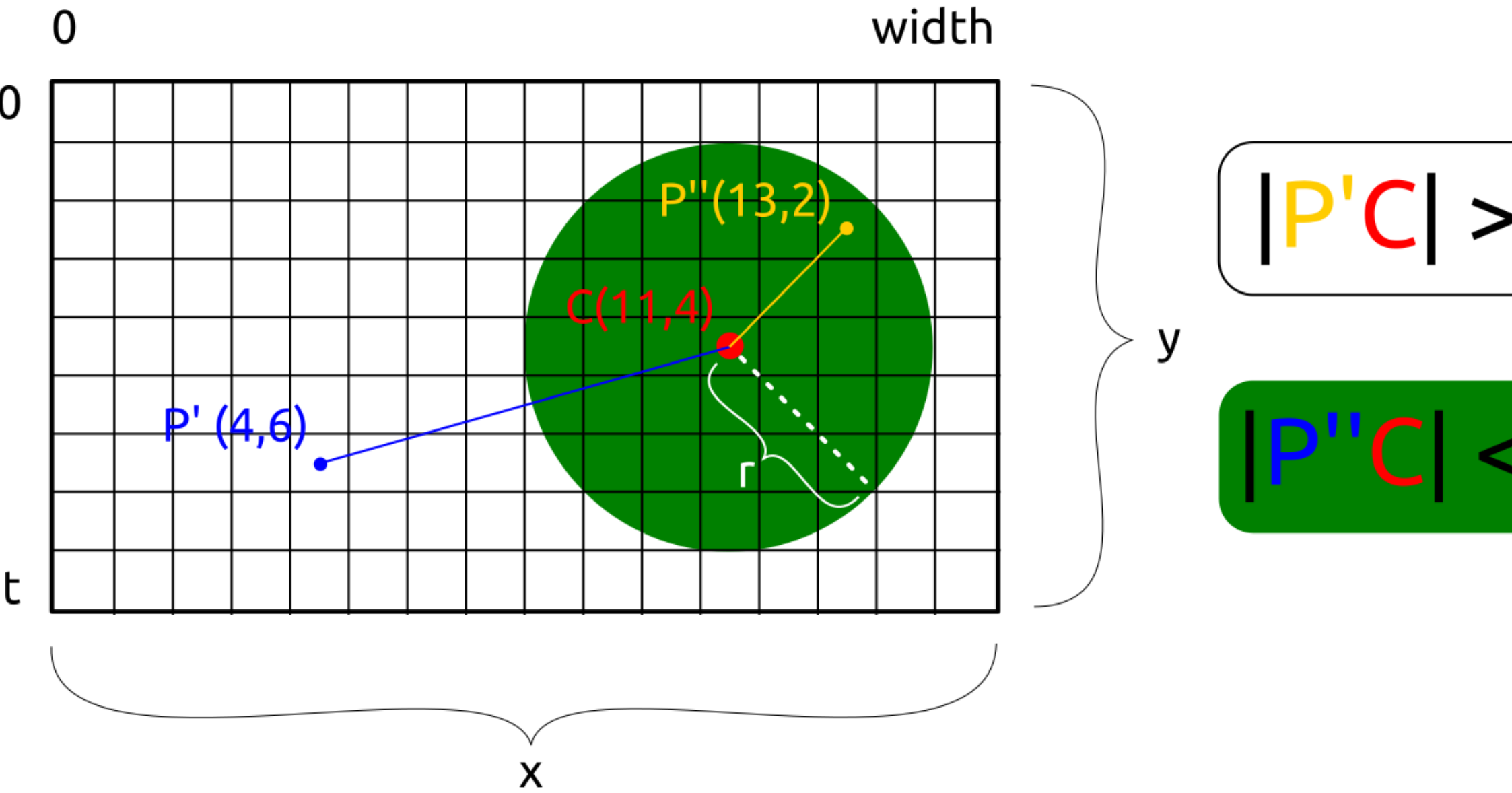
* Vorteile bzw. Abgrenzung zu Polygongrafik
* Analogie: Raymarching : Polygongrafik -> Vektorgrafik : Rastergrafik

Geschichte

* Einführung von Sphere Tracing durch John C. Hart 1996 Paper

Grundlagen in 2D

"Raymarching" in 2D

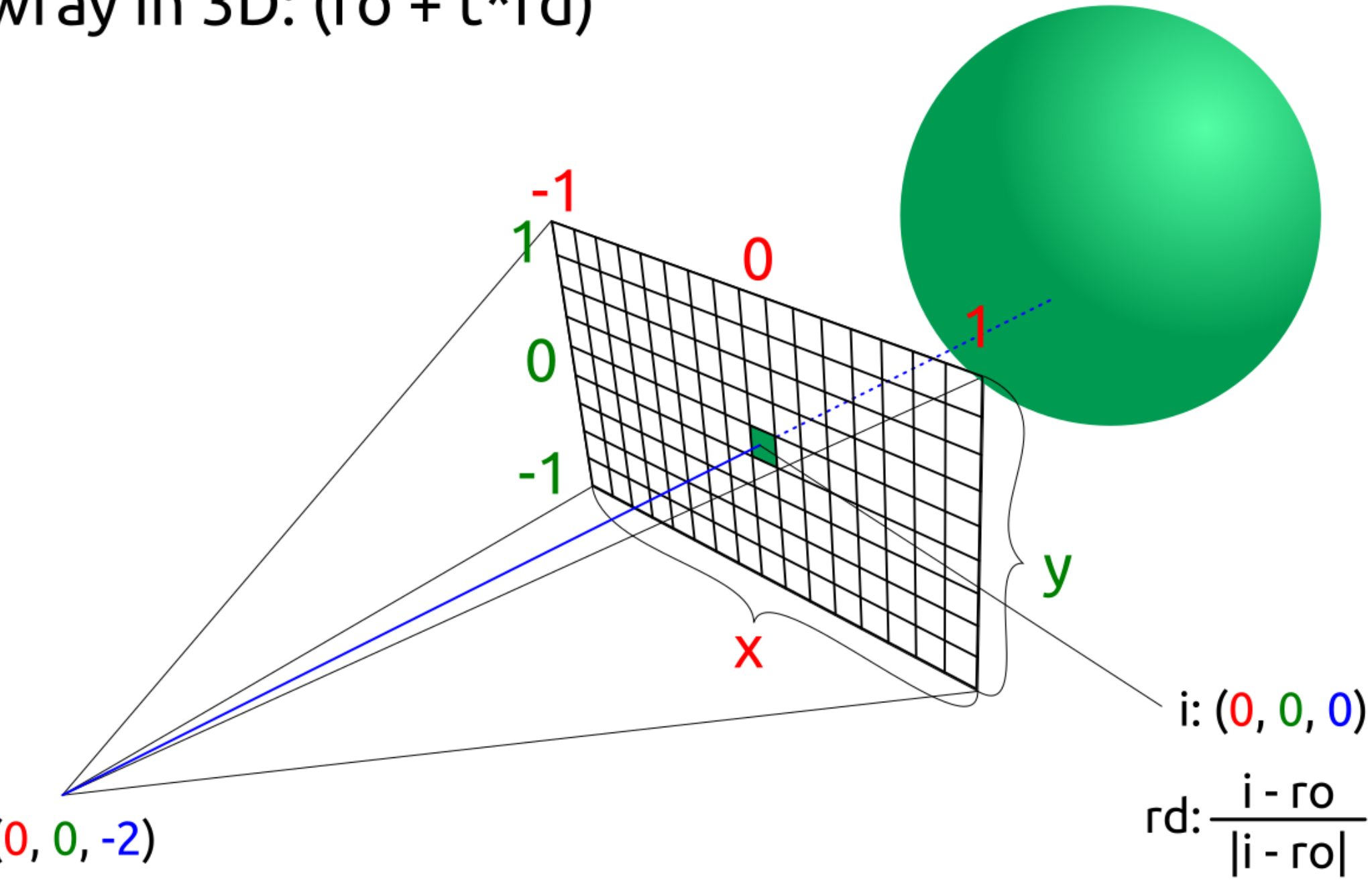


* Signed Distance Functions
* Teilmenge impliziter Funktionen
* 2D-only C++-Beispiel raymarcher-nongl

Demo

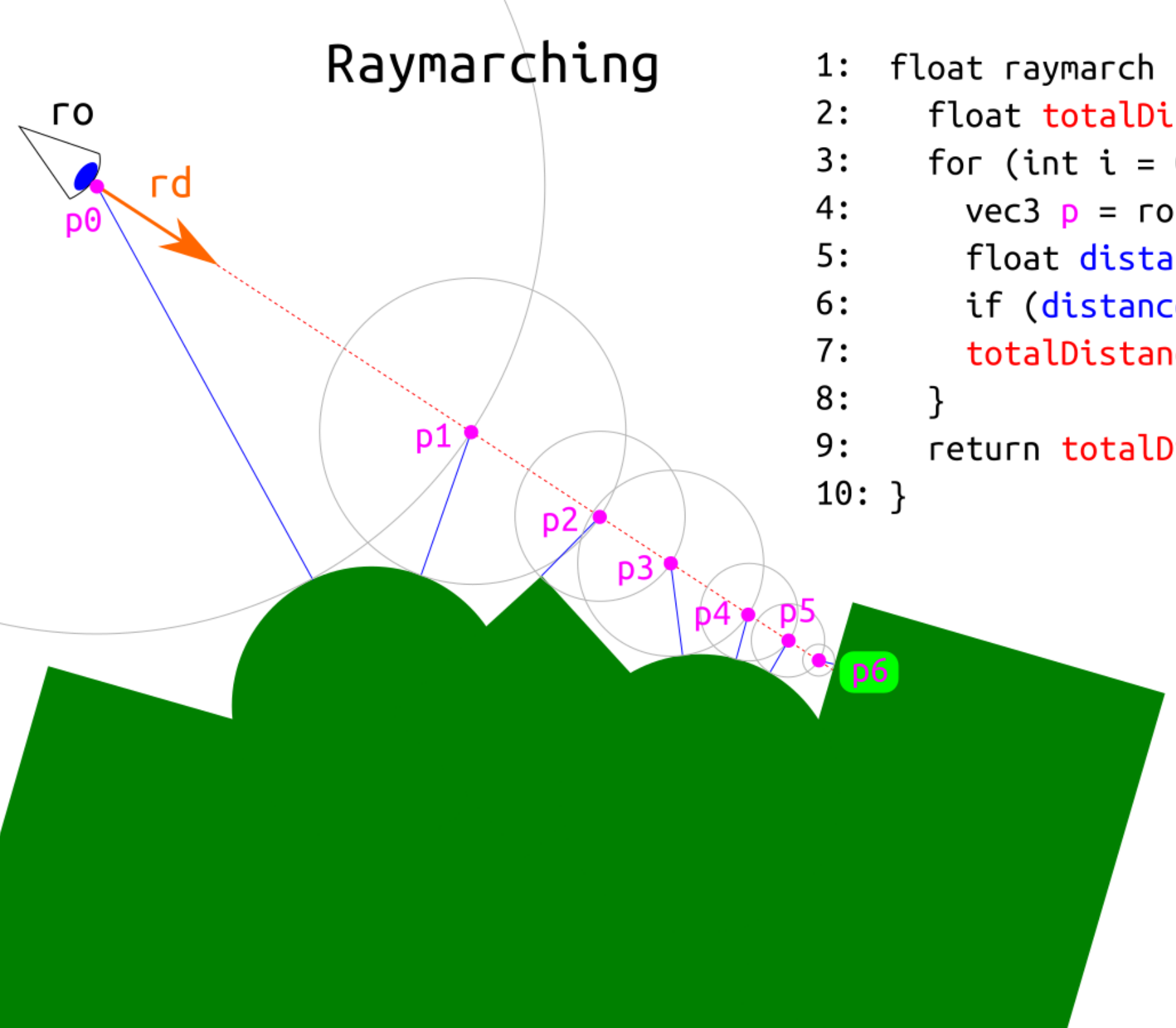
Schritt nach 3D

wray in 3D: (ro + t*rd)



Raymarching-Algorithmus

Raymarching




```
1:  float raymarch (vec3 ro, vec3
2:    float totalDistance = 0.0;
3:    for (int i = 0; i < MAX_ITER
4:      vec3 p = ro + totalDistanc
5:      float distance = scene (p)
6:      if (distance < EPSILON) br
7:      totalDistance += distance;
8:    }
9:    return totalDistance;
10: }
```

Beispiele für Objekte

Basic Object

Plane:

```
point.y - height = .0
```

Sphere:

```
length (point) - radius = .0
```

Cylinder:

```
length (point.xz) - radius = .0
```

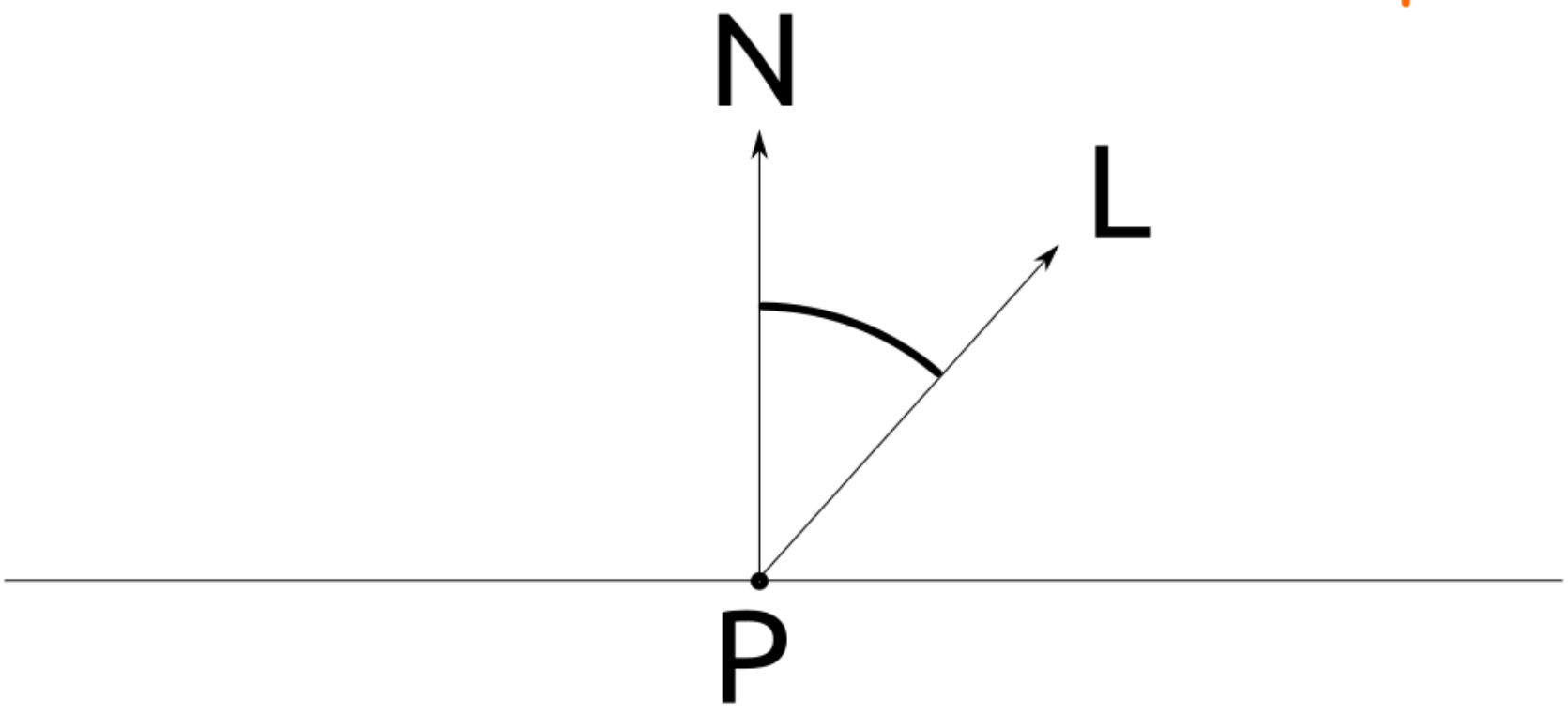
Box:

```
length (max (abs (point) - size, .0)) = .0
```

Demo

Beleuchtungsmodelle

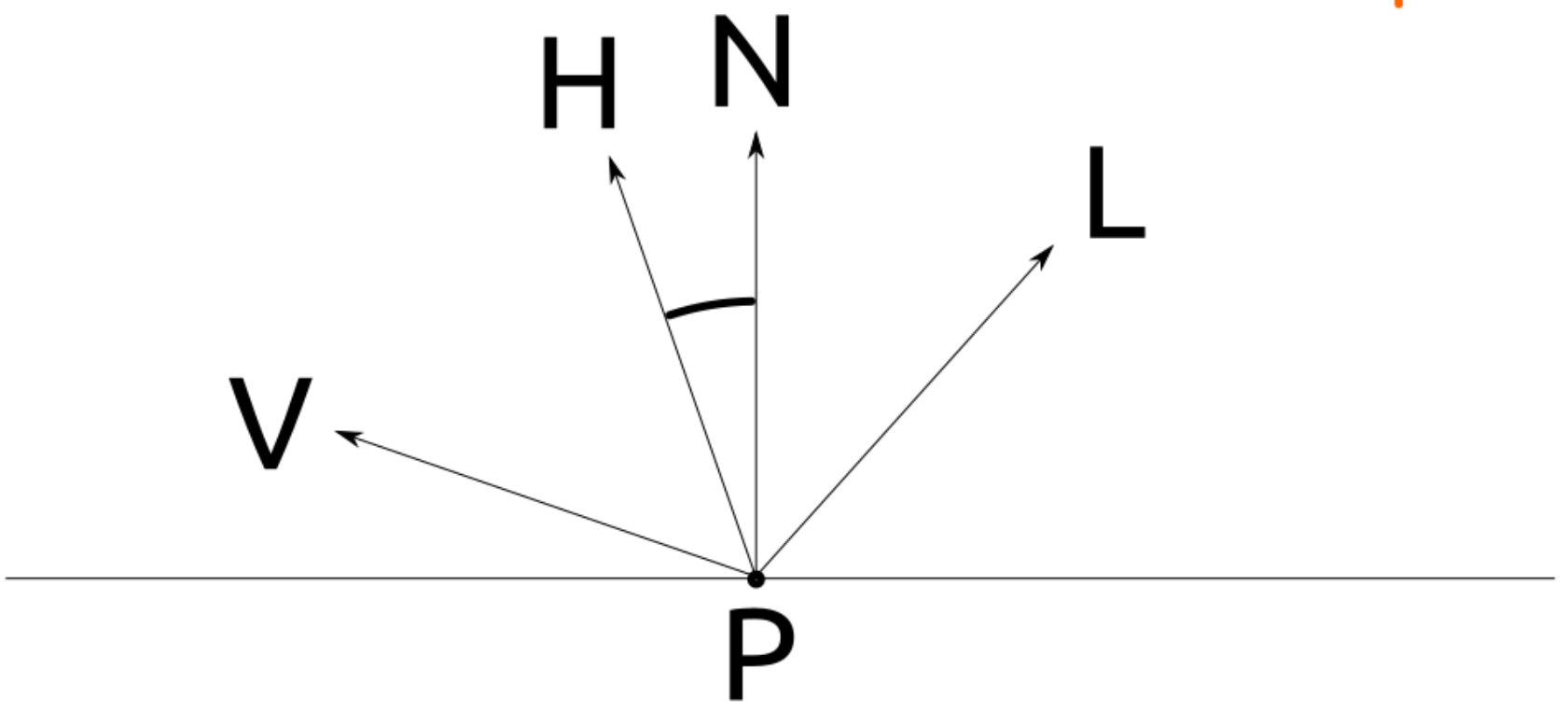
bert Shading Model



$$Color_{diffuse} = N \cdot L * Color_{Surface} * Intensity_{Light}$$

* Johann Heinrich Lambert, schweizer Mathematiker , 1726 - 1777
* ok, aber nicht wirklich komplett in physikalischer Realität verankert

ular Highlights à la Blinn-Phong



V: direction to
H: halfvector o
N: surface norm
L: direction to l

N·L and N·H hav
clamped to [0..

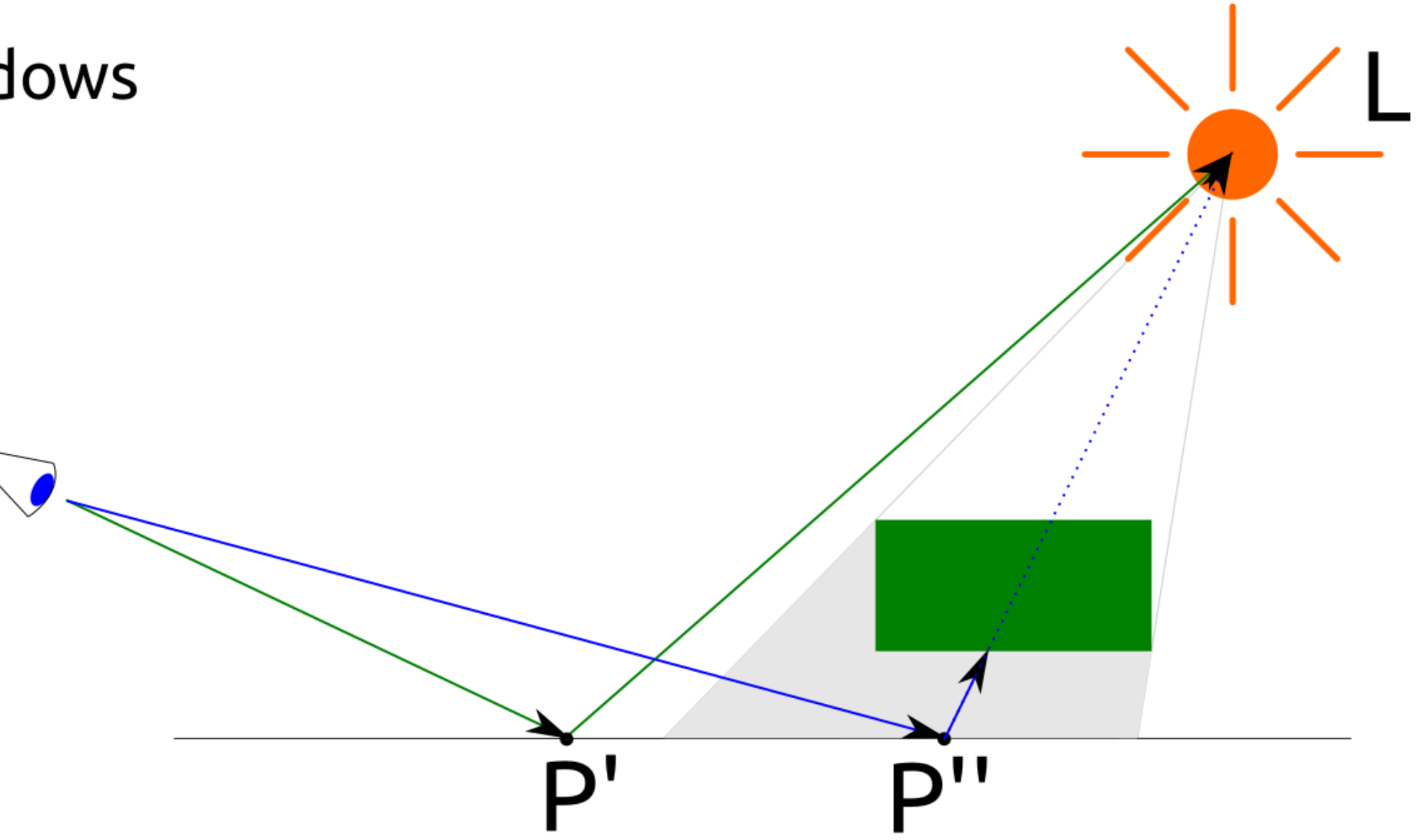
$$Color_{diffuse} = N \cdot L * Color_{Surface} * Intensity_{Light}$$

$$Color_{final} = (N \cdot H)^{shininess} * Color_{specular} + Color_{diffuse}$$

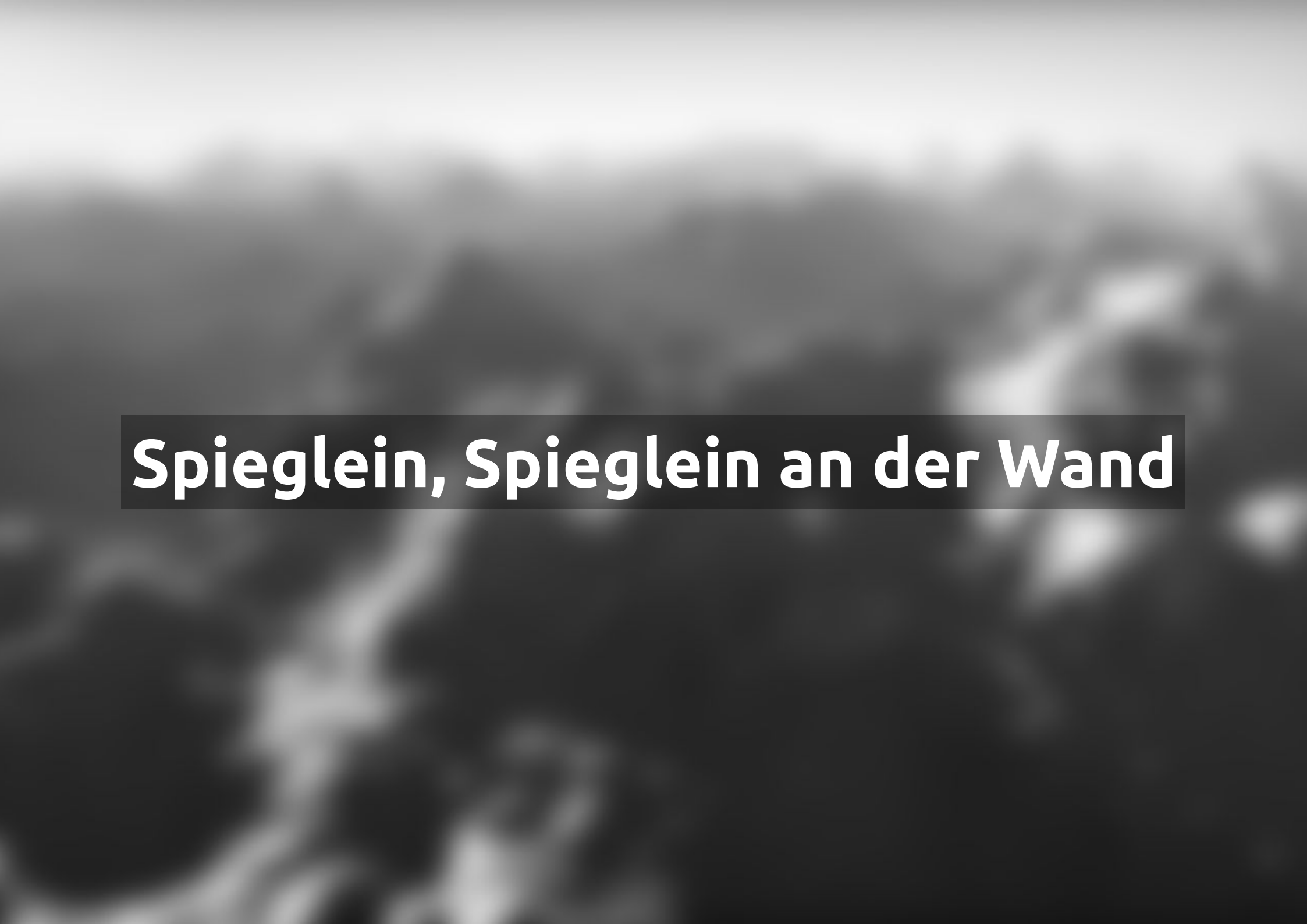
* Bùi Tường Phong, Jim Blinn, ca. 1975-1977
* state-of-the-art ist aber PBR
* PBR zu komplex -> eigener Vortrag
* trotzdem zumindest demonstrieren


Mehr Schatten als Licht

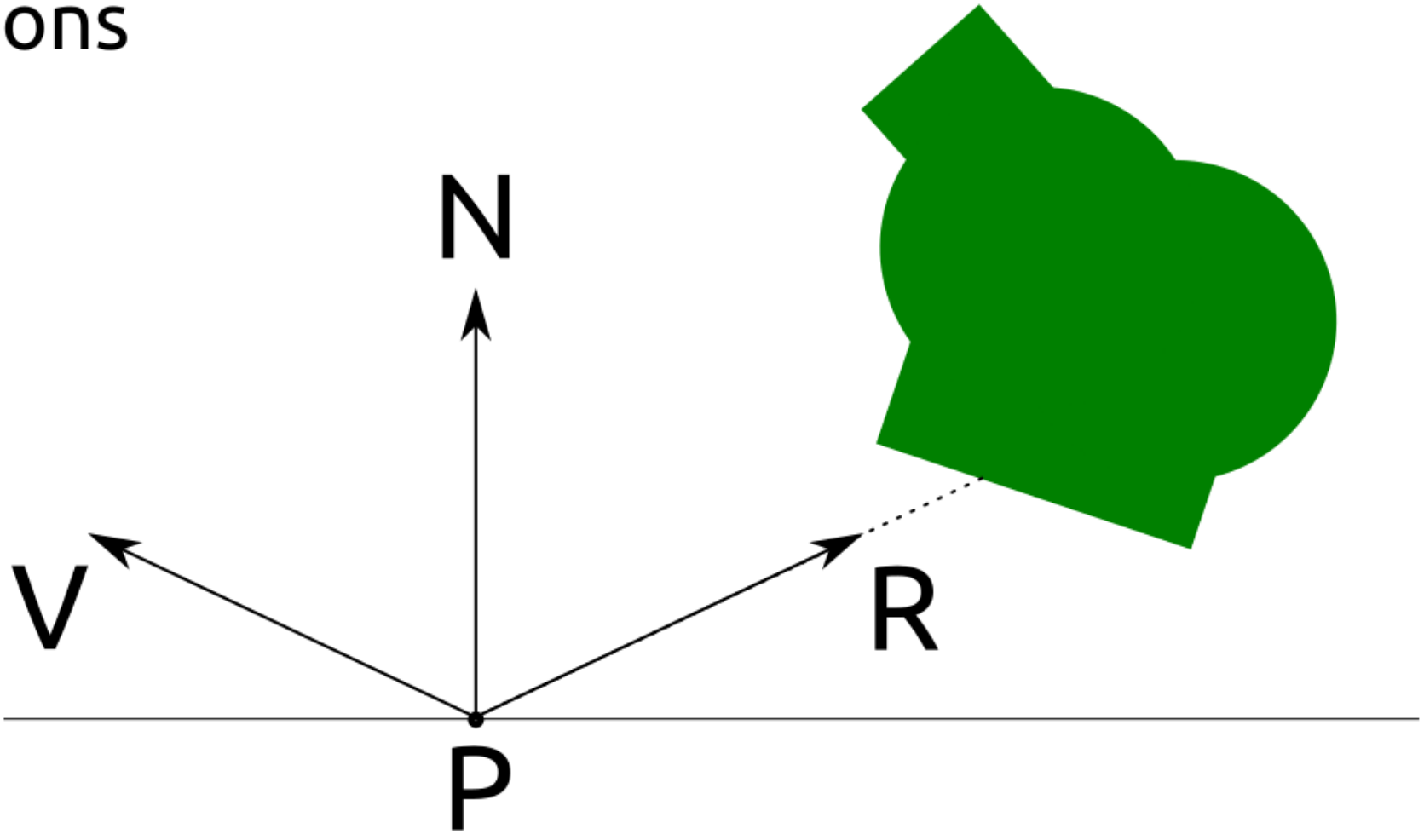
dows




```
distanceToLight = length (L - P)
distanceToObject = raymarch (P, normalize (L - P))
isShadowed = distanceToObject < distanceToLight
finalColor = isShadowed ? darken(shadingColor) : shadingColor
```


Spieglein, Spieglein an der Wand

ections




```
struct HitR
  bool hasH
  float dis
  int mater
}
```

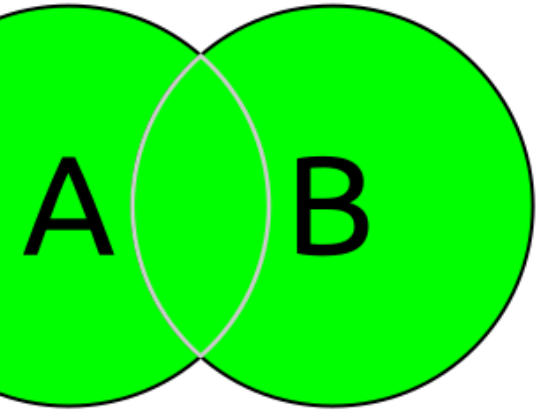
```
rmalize (reflect (V, N));
 = raymarch (P, normalize (R))
olor = result.hasHit ? modulate (shadingColor, result) : shadingColor
```


Demo

Boolsche Operationen

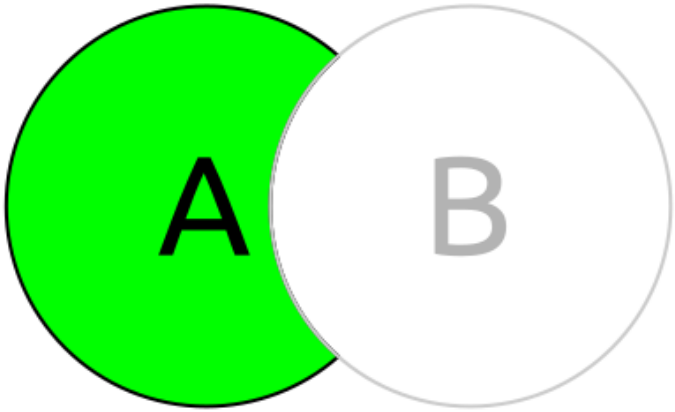
Boolean Operat

pUnion: A + B



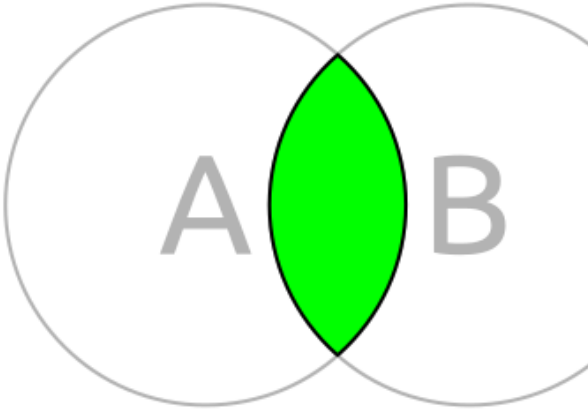
min (A, B)

opSubtract: A - B



max (-A, B)

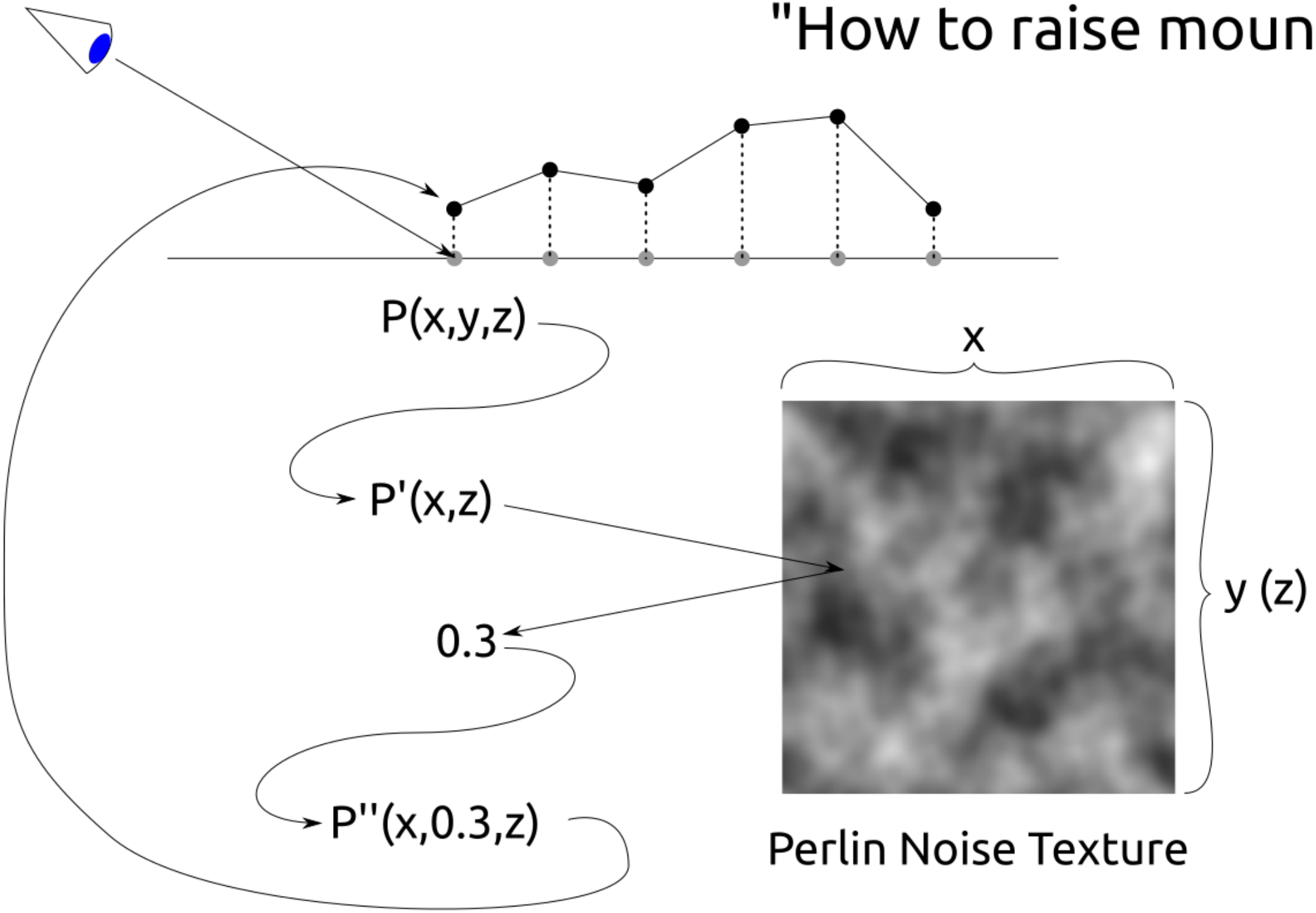
opIntersect:



max (A, B)

Demo


Von der Ebene zur Landschaft


"How to raise mounta
P(x,y,z)
x
P'(x,z)
y (z)
0.3
P''(x,0.3,z)
Perlin Noise Texture


Demo

Zusammenfassung

- Geschichte - 1996 von John C. Hart eingeführt
- Grundlagen in 2D - Signed Distance Functions
- Schritt nach 3D - matrizenlose 3D-Projektion
- Raymarching-Algorithmus
- Beleuchtungsmodelle - Lambert, Blinn-Phong, PBR
- wie man an Schatten kommt
- Reflexionen und Boolsche Operationen
- Erzeugung von Landschaften
- jede Menge Demos

Aussicht auf Teil 2

* mehr über Operatoren
* Krümmung/Faltung/Wiederholung vom Raum
* volumetrische Effekte: z.B. Wolken
* Kontaktschatten/Ambient Occlusion
* Tiefenunschärfe/Depth-of-field
* Überstrahlung/Bloom
* AA-Strategien
* Bau von sehr komplexeren Objekten
* bessere Materialien (Holz, Leder, Marmor, Blech, Plastik...)
* spezielle Wünsche?

Gibt's Fragen?


Besten Dank für
Eure Aufmerksamkeit!

① Vortrag, PDF, Sourcen
https://github.com/MacSlow/raymarching-vortrag

② Leseempfehlung: John C. Hart Paper von 1996
"a geometric method for the antialiased ray tracing of implicit surfaces"

③ Leseempfehlung: iq's Homepage
http://www.iquilezles.org

④ NVScene 2015 Vortrag von cupe
https://www.youtube.com/watch?v=s8nFqwOho-s

⑤ Leseempfehlung: Keinert et. al. Paper von 2014
"Enhanced Sphere Tracing"
http://erleuchtet.org/~cupe/permanent/enhanced_sphere_tracing.pdf

⑥ Leseempfehlung: Pharr, Jakob, Humphreys - Buch von 2016
"Physically Based Rendering - From Theory to Implementation"