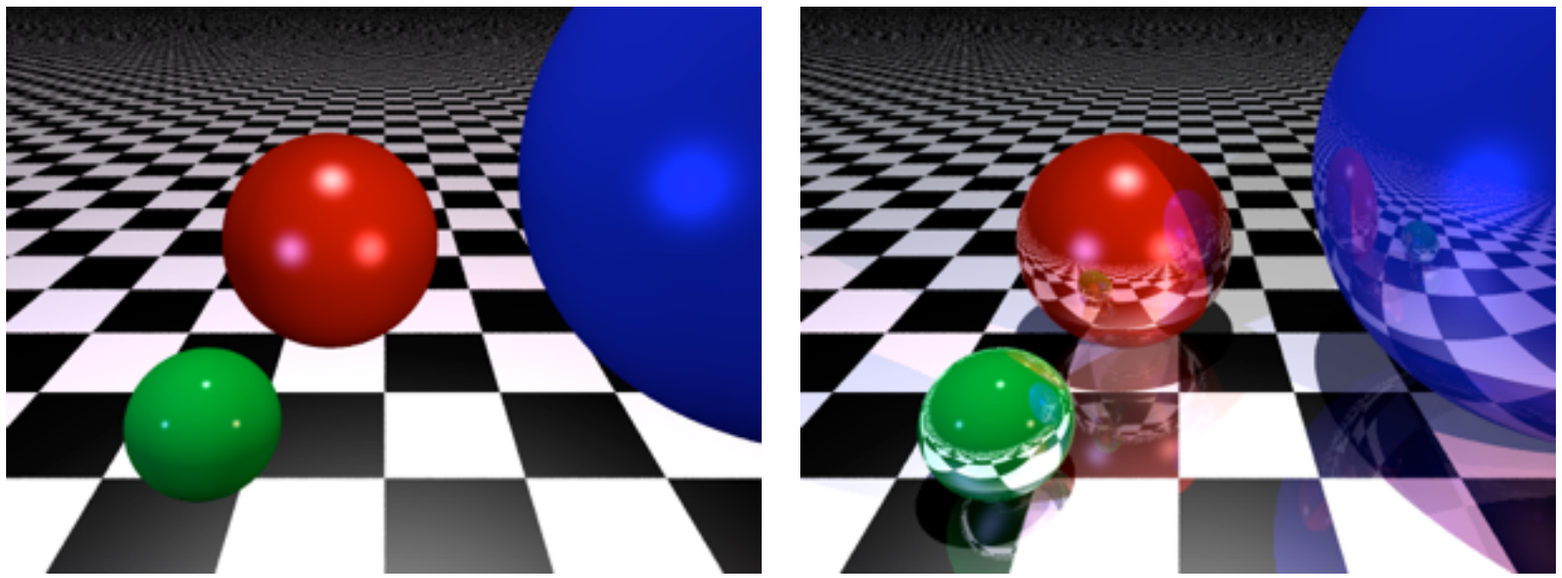
Introduction to Computer Graphics

Colors & Lighting



Prof. Dr. Mario Botsch

Computer Graphics & Geometry Processing

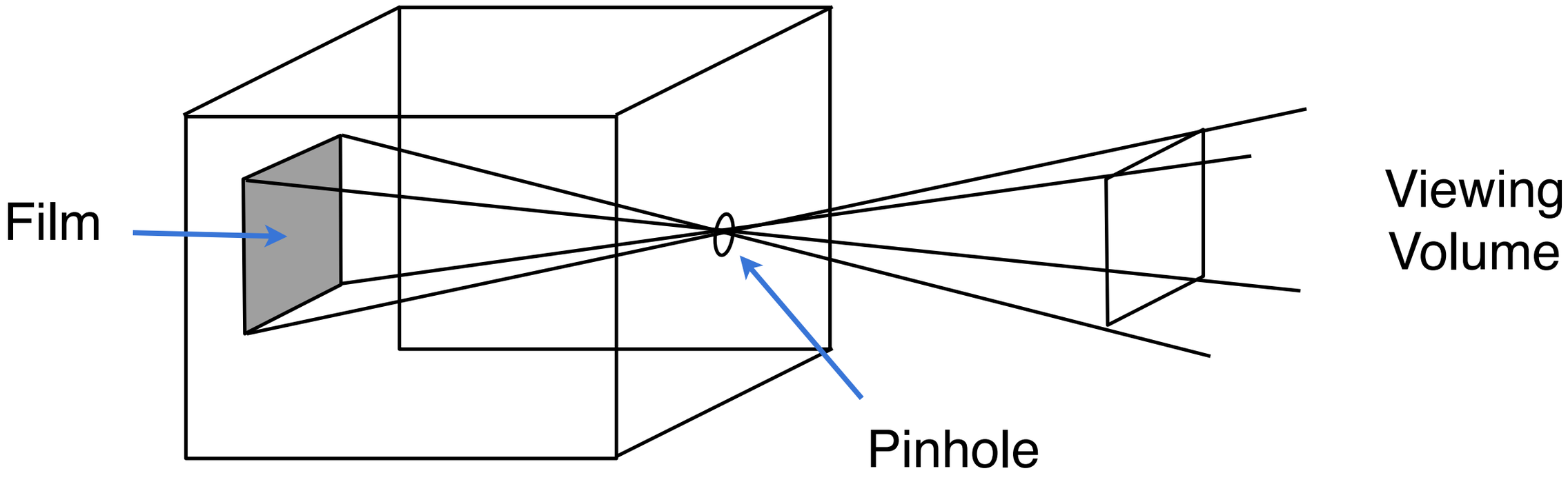
Universität Bielefeld

CITEC

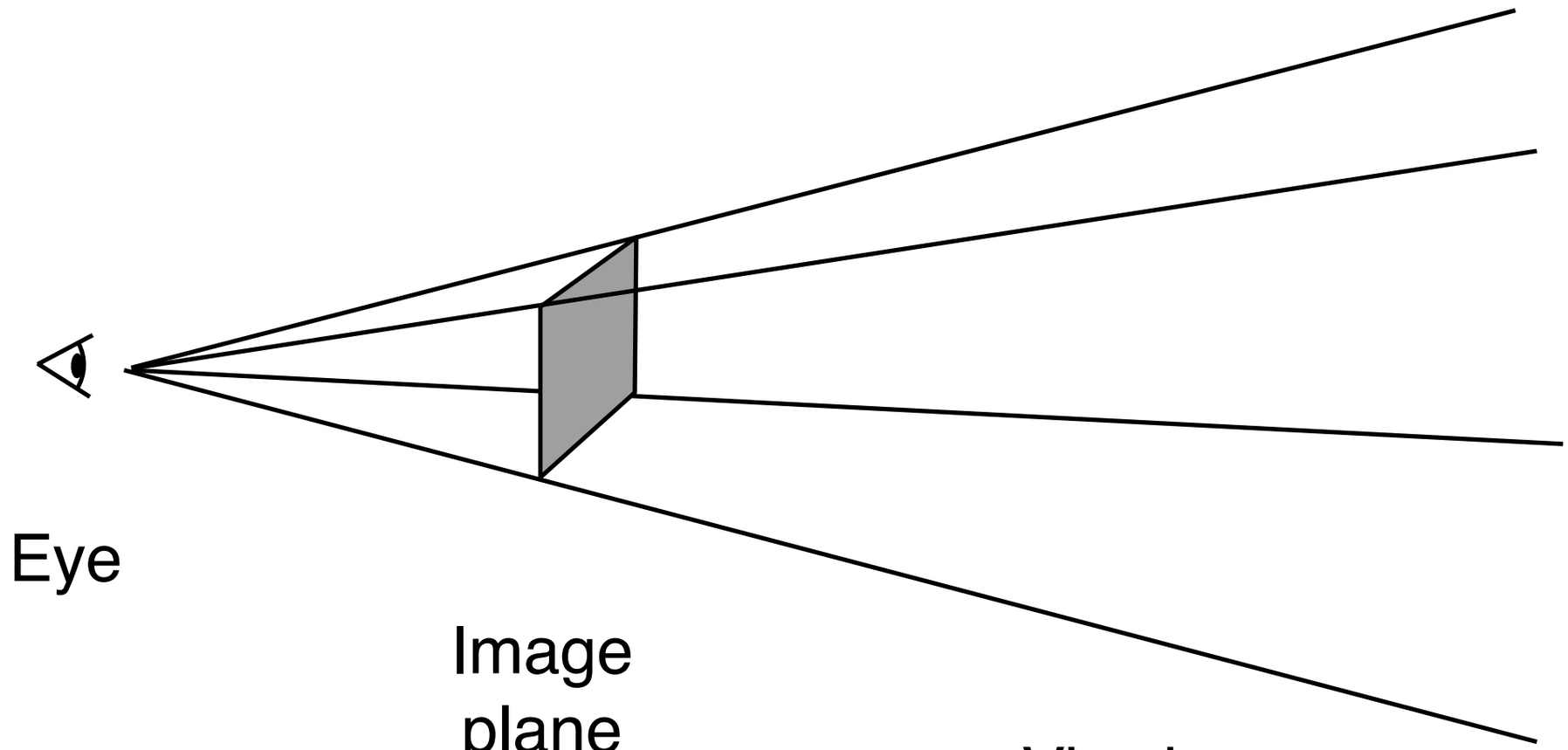
Last time

- What is light?
- How is light propagated?
- How does a camera work?
- How does light interact with objects?
- What is a color?

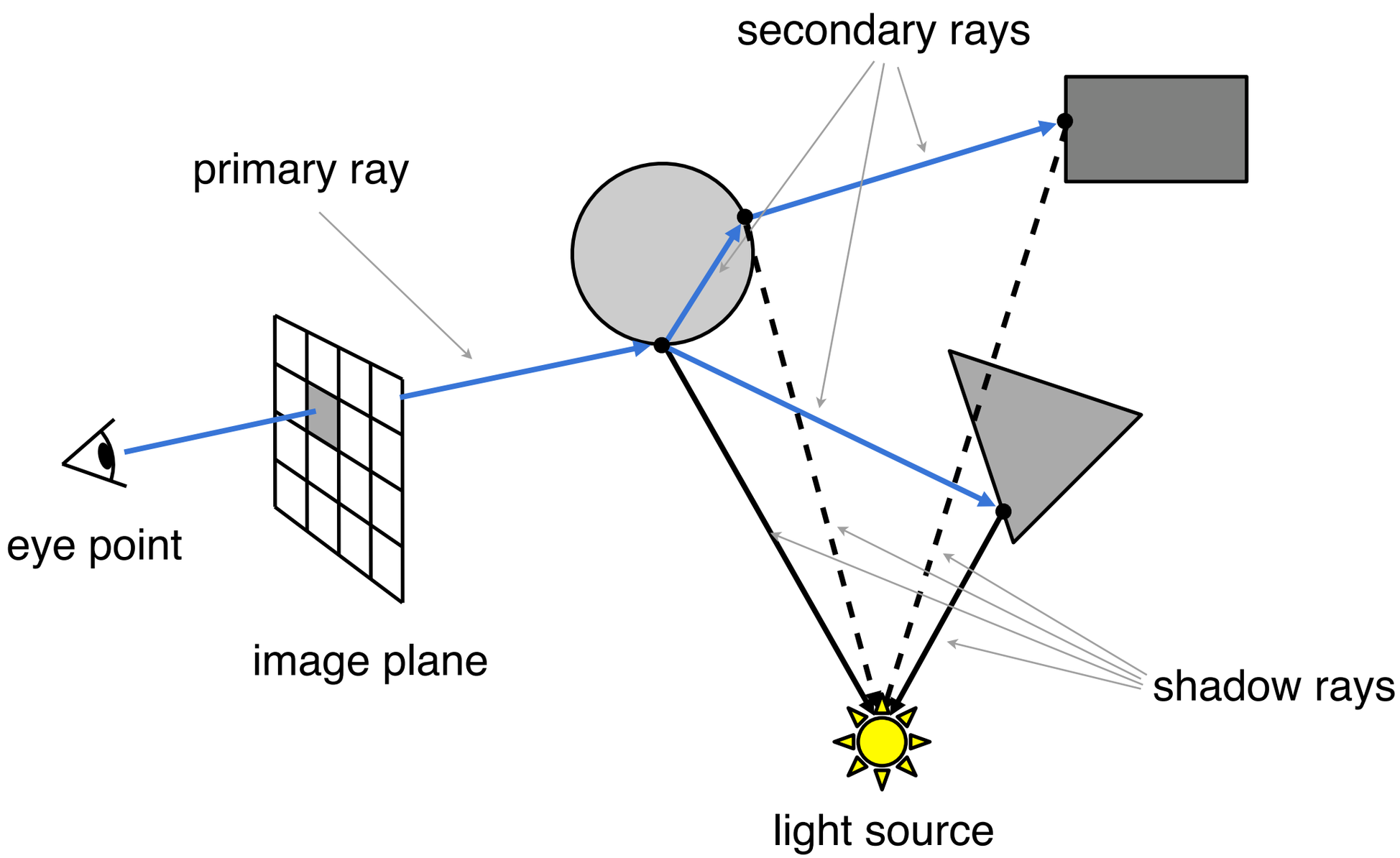
Pinhole Camera



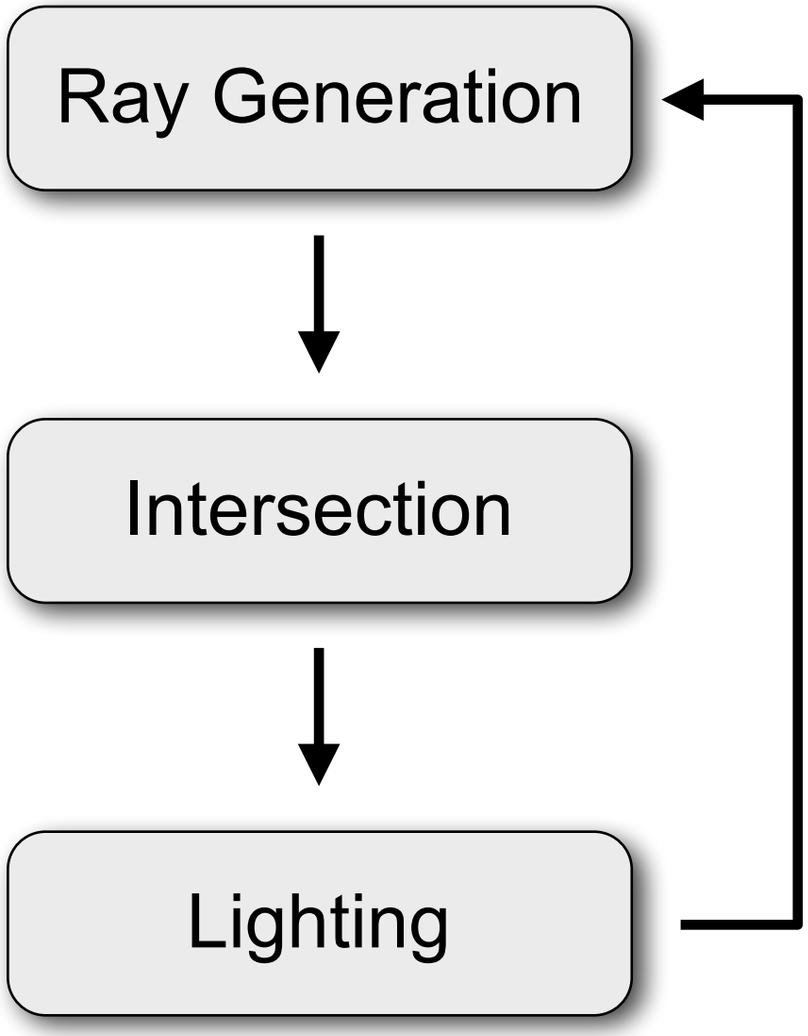
Pinhole Camera

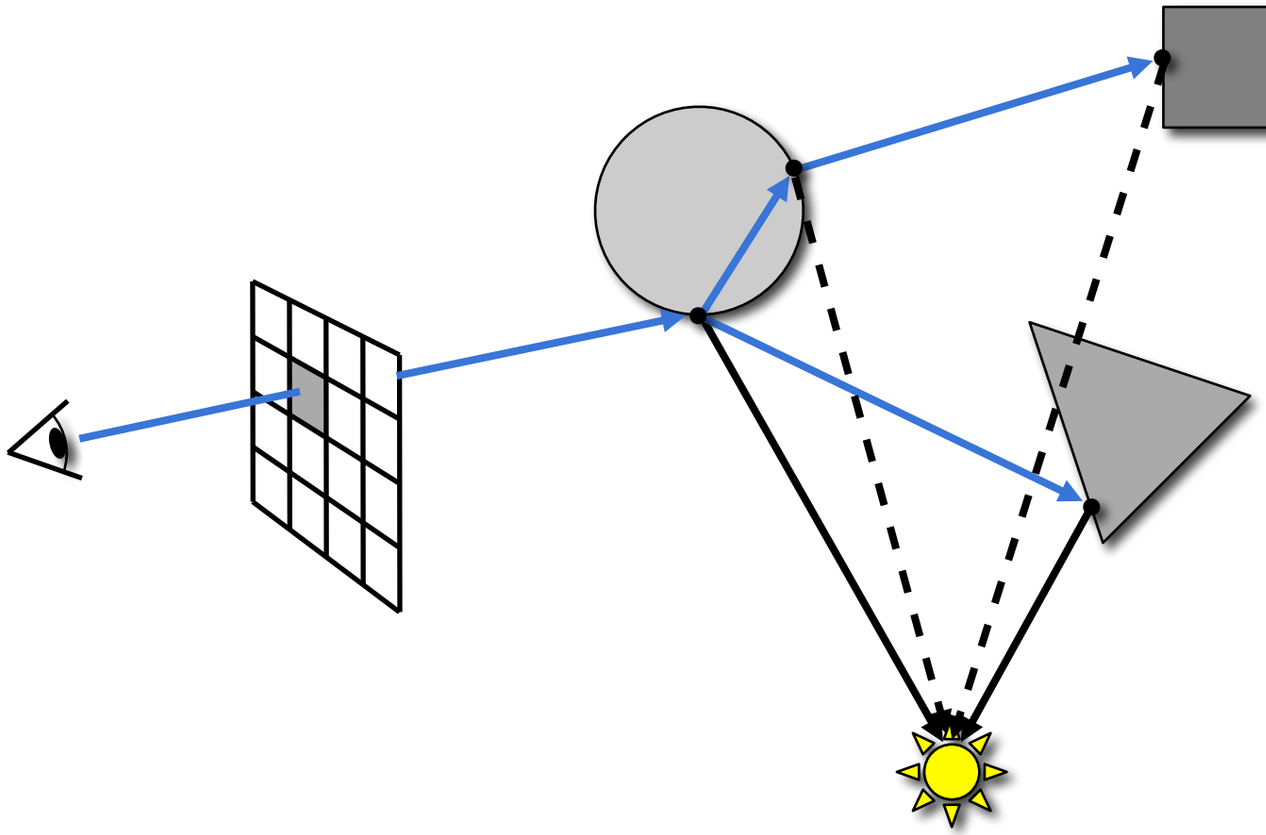


Backward Ray Tracing

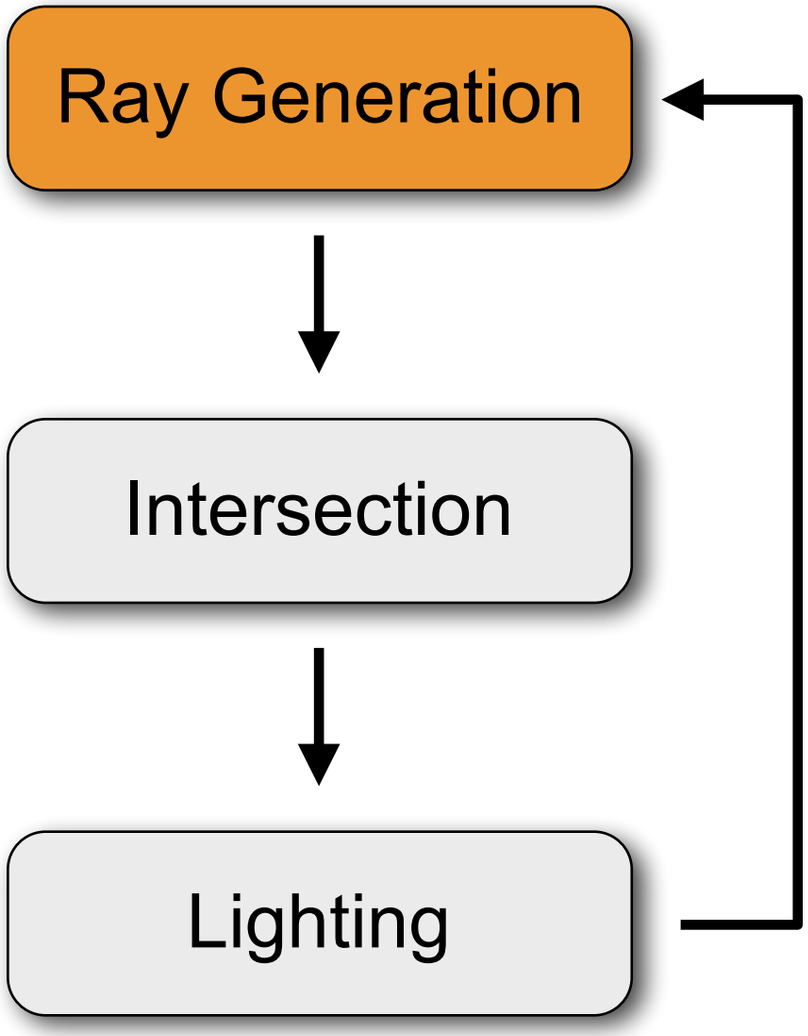


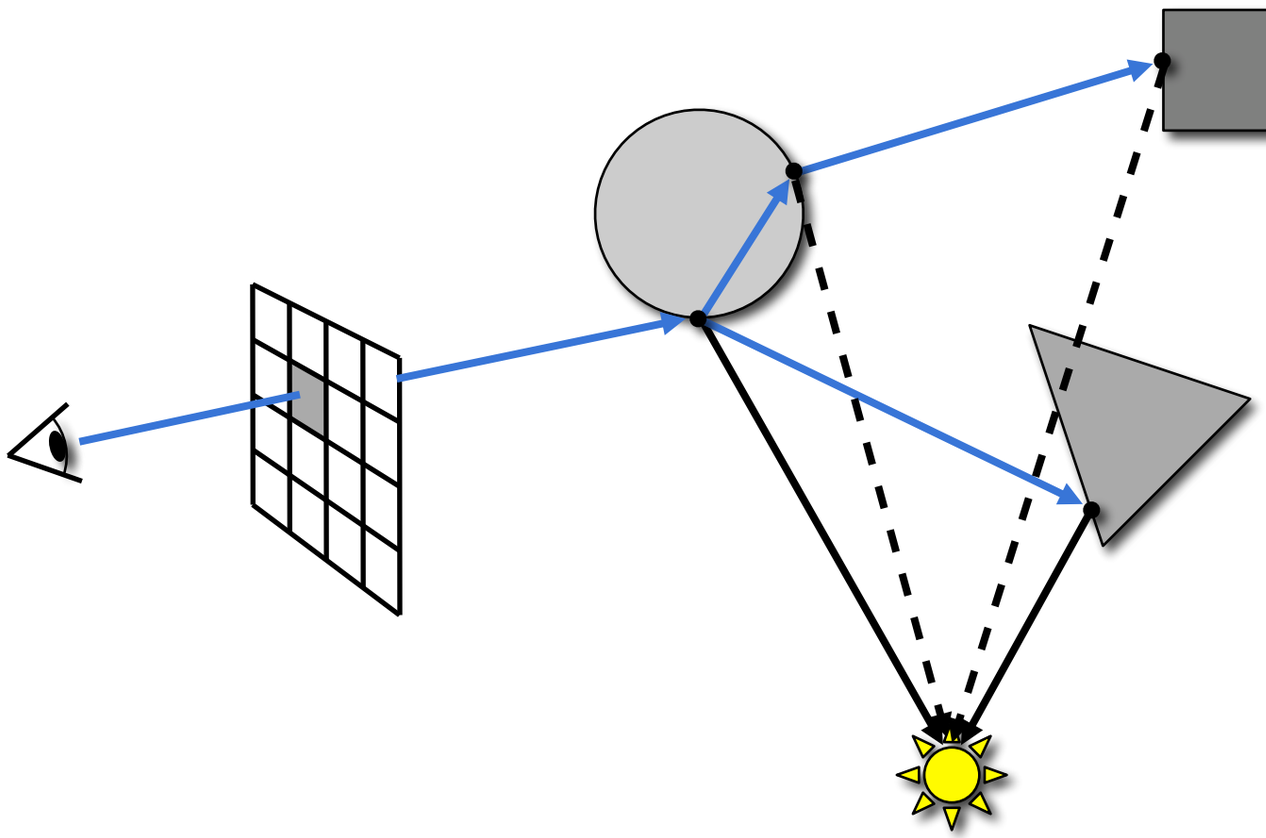
Backward Ray Tracing





Ray Generation



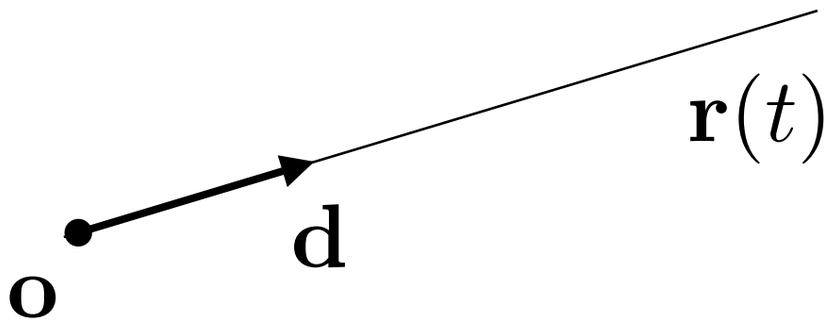


Rays

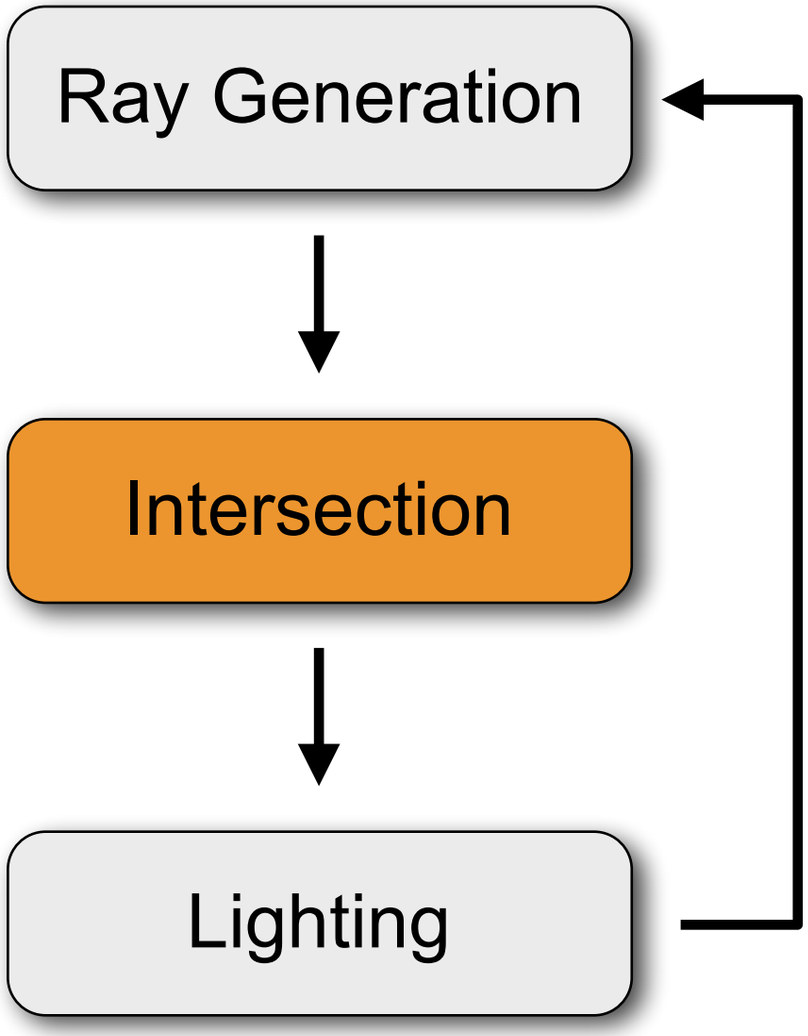
- Ray equation (explicit form)

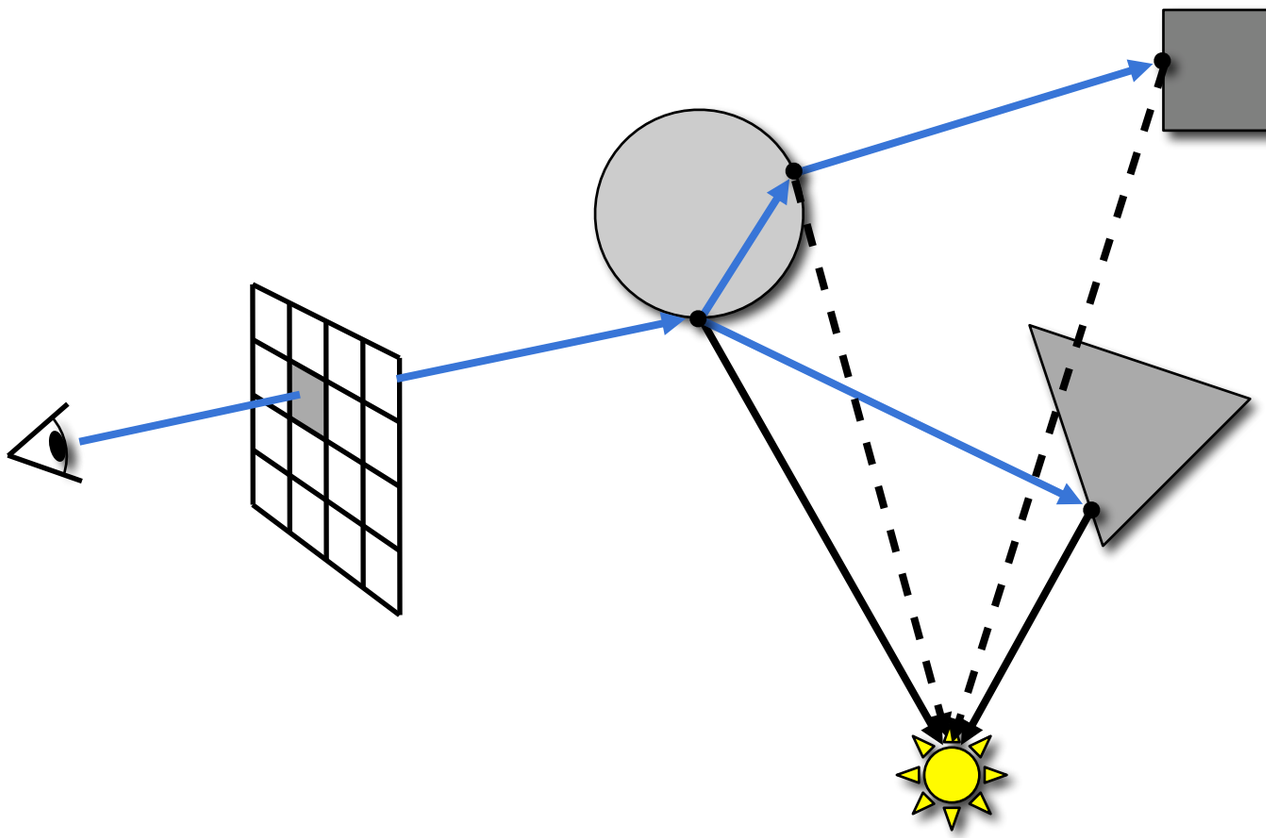
$$\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$$

ray origin direction (normalized)



Ray-Surface Intersections





Ray-Sphere Intersection

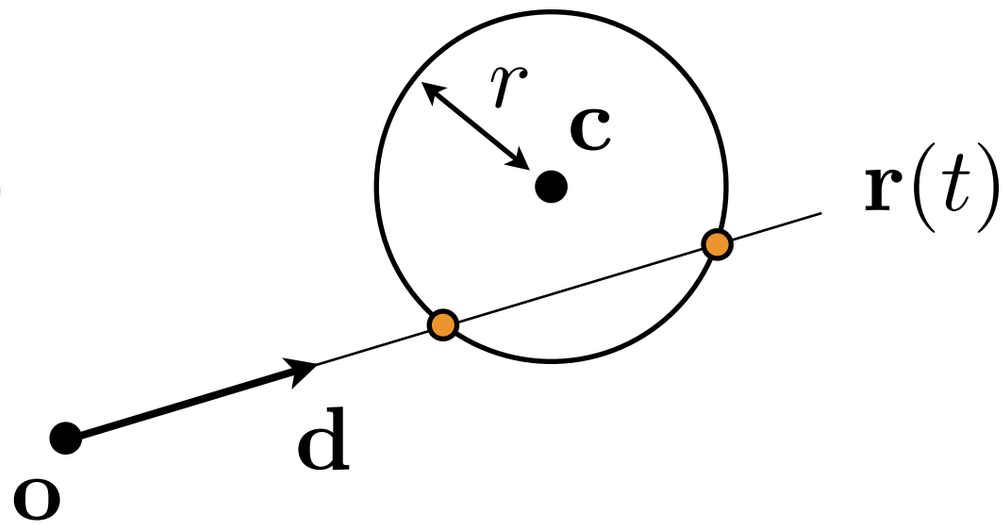
- Sphere equation (implicit form)

$$\|\mathbf{x} - \mathbf{c}\|^2 - r^2 = 0$$

point on sphere — center — radius

- Insert ray equation and solve for *t*

$$\|\mathbf{o} + t\mathbf{d} - \mathbf{c}\|^2 - r^2 = 0$$



Ray-Plane Intersection

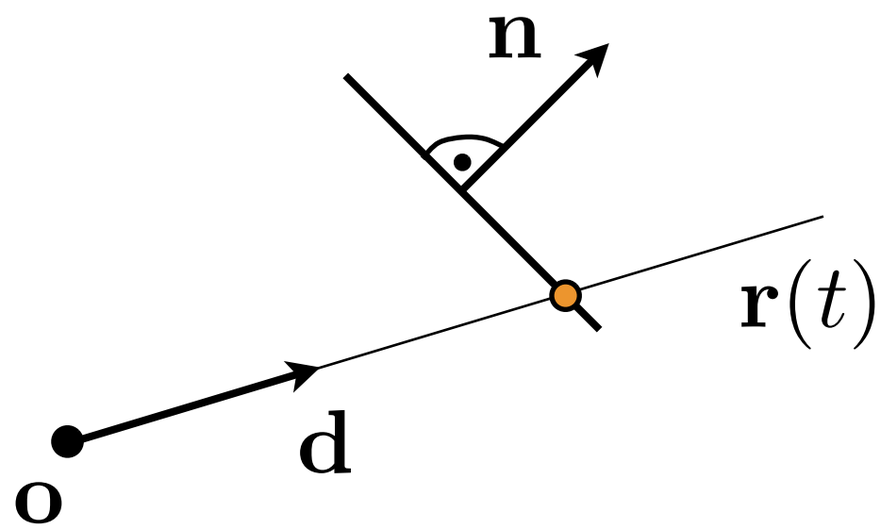
- Plane equation (implicit form)

$$\mathbf{x}^T \mathbf{n} - d = 0$$

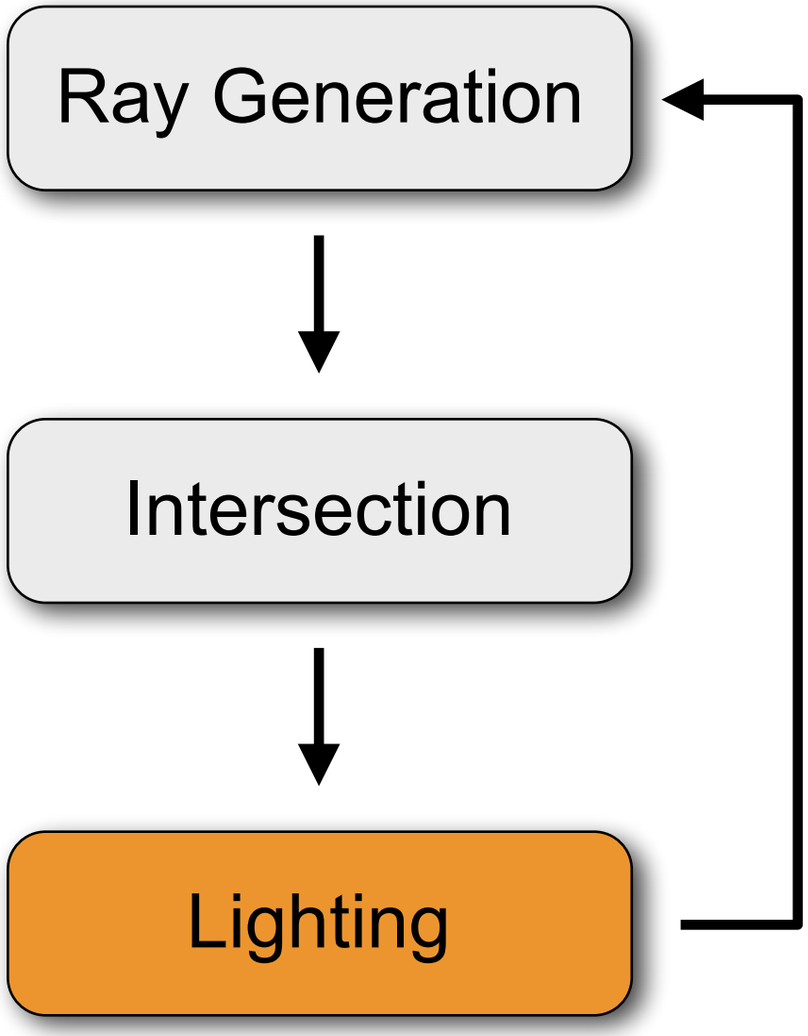
point on plane — plane normal — distance to origin

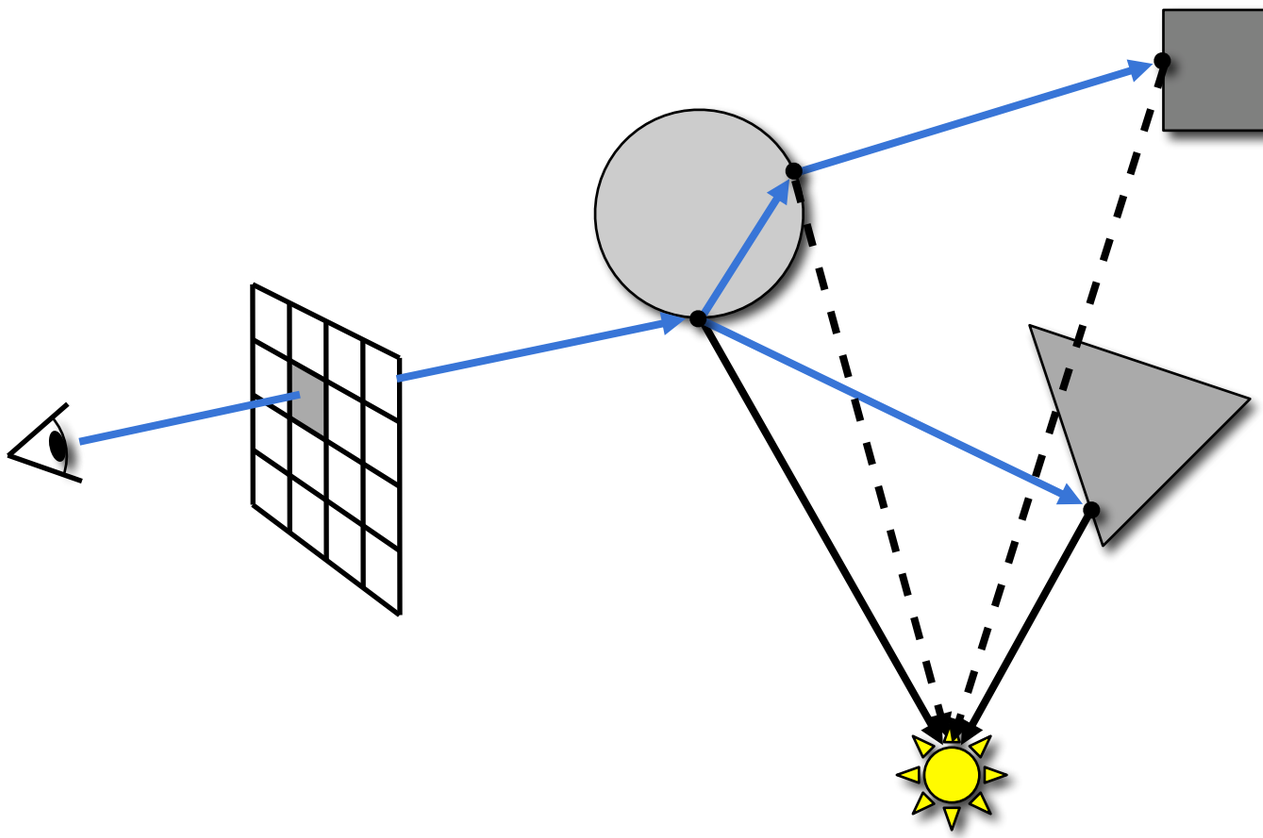
- Insert ray equation and solve for *t*

$$(\mathbf{o} + t\mathbf{d})^T \mathbf{n} - d = 0$$



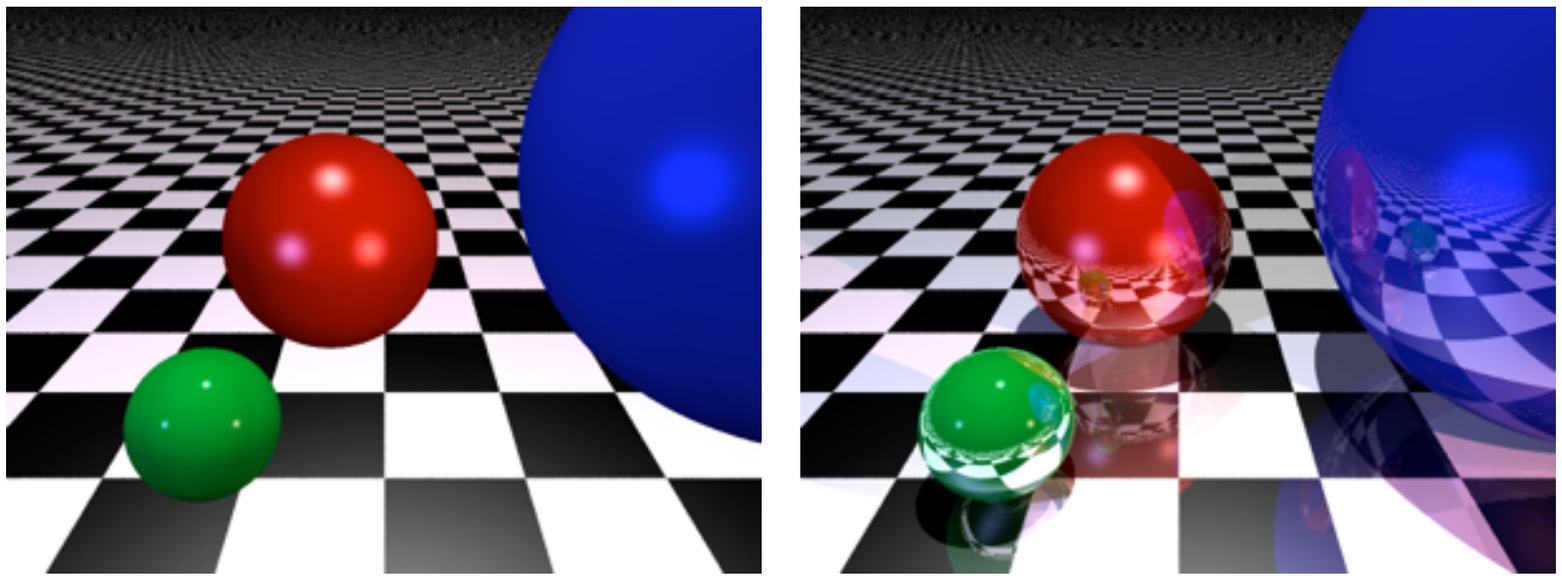
Today: Color & Lighting





Introduction to Computer Graphics

Colors



Prof. Dr. Mario Botsch

Computer Graphics & Geometry Processing

Universität Bielefeld

CITEC

How to describe colors?

How to describe colors?

Subjective?

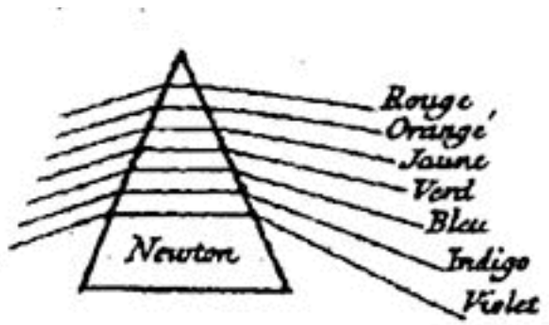
blue & black or white & gold?

Colors

- **Artist's view**
- Physics
- Biology
- Computer Graphics

A *Subjective* Phenomenon?

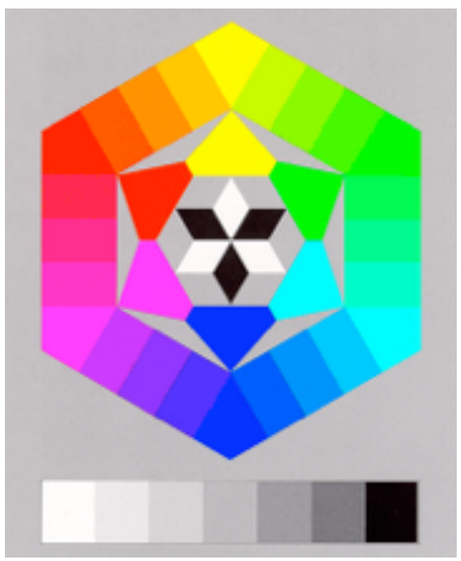
- Newton
- Goethe
- Kandinski
- Itten
- Küppers



Newton

Goethe

Itten

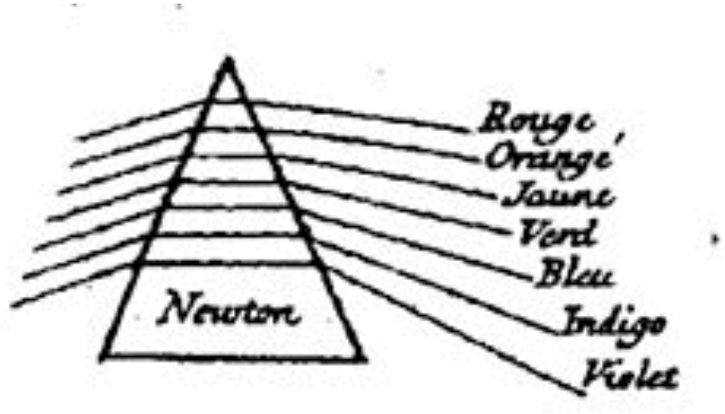


Küppers

A *Subjective* Phenomenon?

Goethe:

“Das Licht ist das einfache, unzerlegteste, homogenste Wesen, das wir kennen. Es ist nicht zusammengesetzt. Am allerwenigsten aus farbigen Lichtern. Jedes Licht, das eine Farbe angenommen hat, ist dunkler als das farblose Licht. Das Helle kann nicht aus Dunkelheit zusammengesetzt sein.”



Newton

Küppers' Color Model

– Primary Colors:

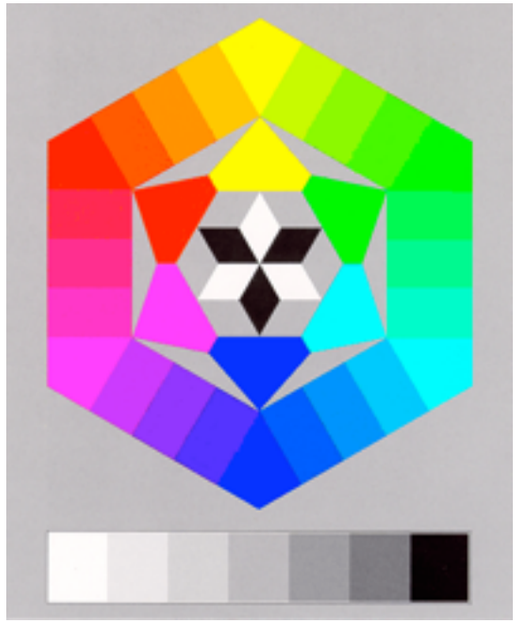
Red Green Blue

– Secondary Colors:

Cyan Magenta Yellow

– Achromatic:

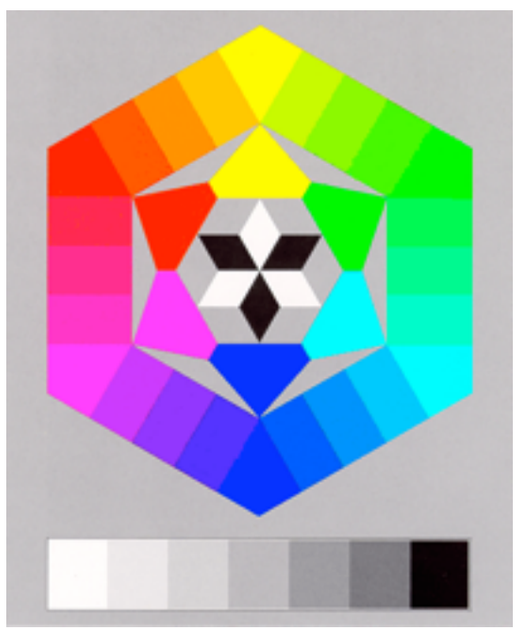
Black White



Harald Küppers

Küppers' Color Model

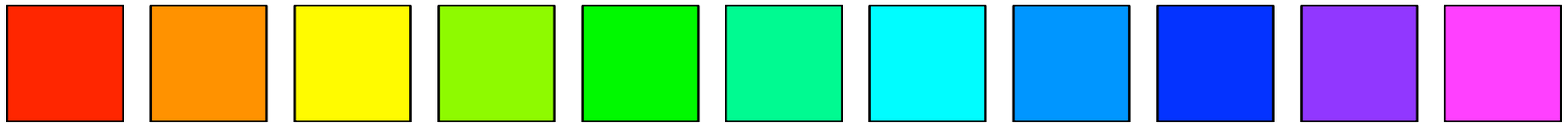
- Characterize RGB(240, 128, 80)
  - White 80: 80, 80, 80
  - Yellow 48: 48, 48, 0
  - Red 112: 112, 0, 0
  - Black 15



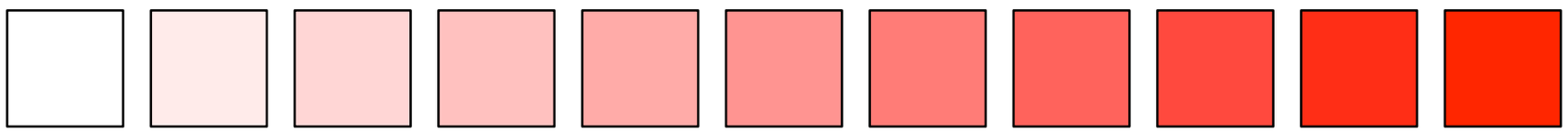
  - *Unbuntart*: W80 / K15 = 5.33
  - *Buntart*: Y48 / R112 = 0.43
  - *Buntgrad*: (Y48+R112) / (W80+K15) = 1.68

Terminology

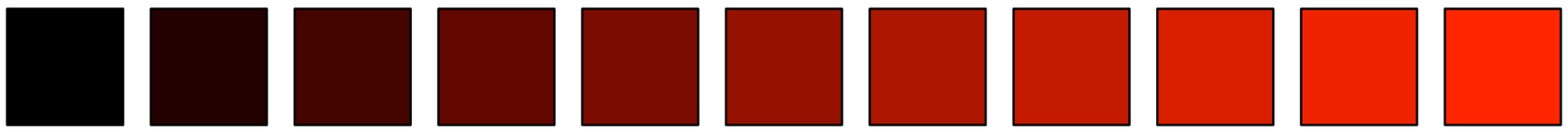
- *Hue* (Farbton)



- *Saturation* (Farbsättigung)



- *Brightness* (Helligkeit)

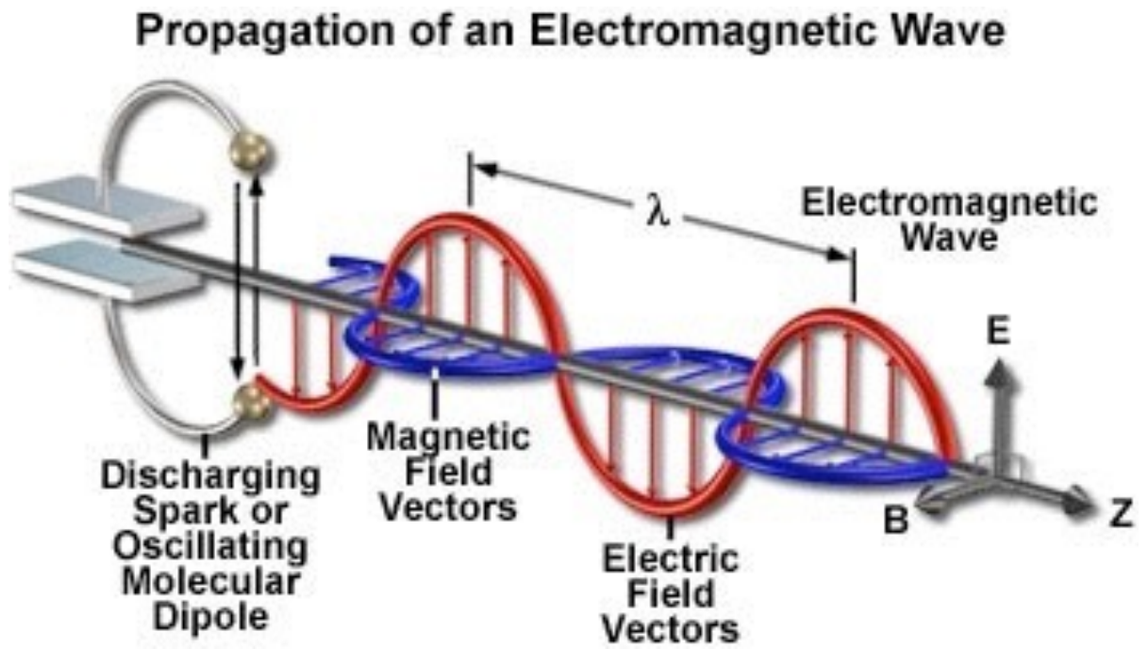


Colors

- Artist's view
- **Physics**
- Biology
- Computer Graphics

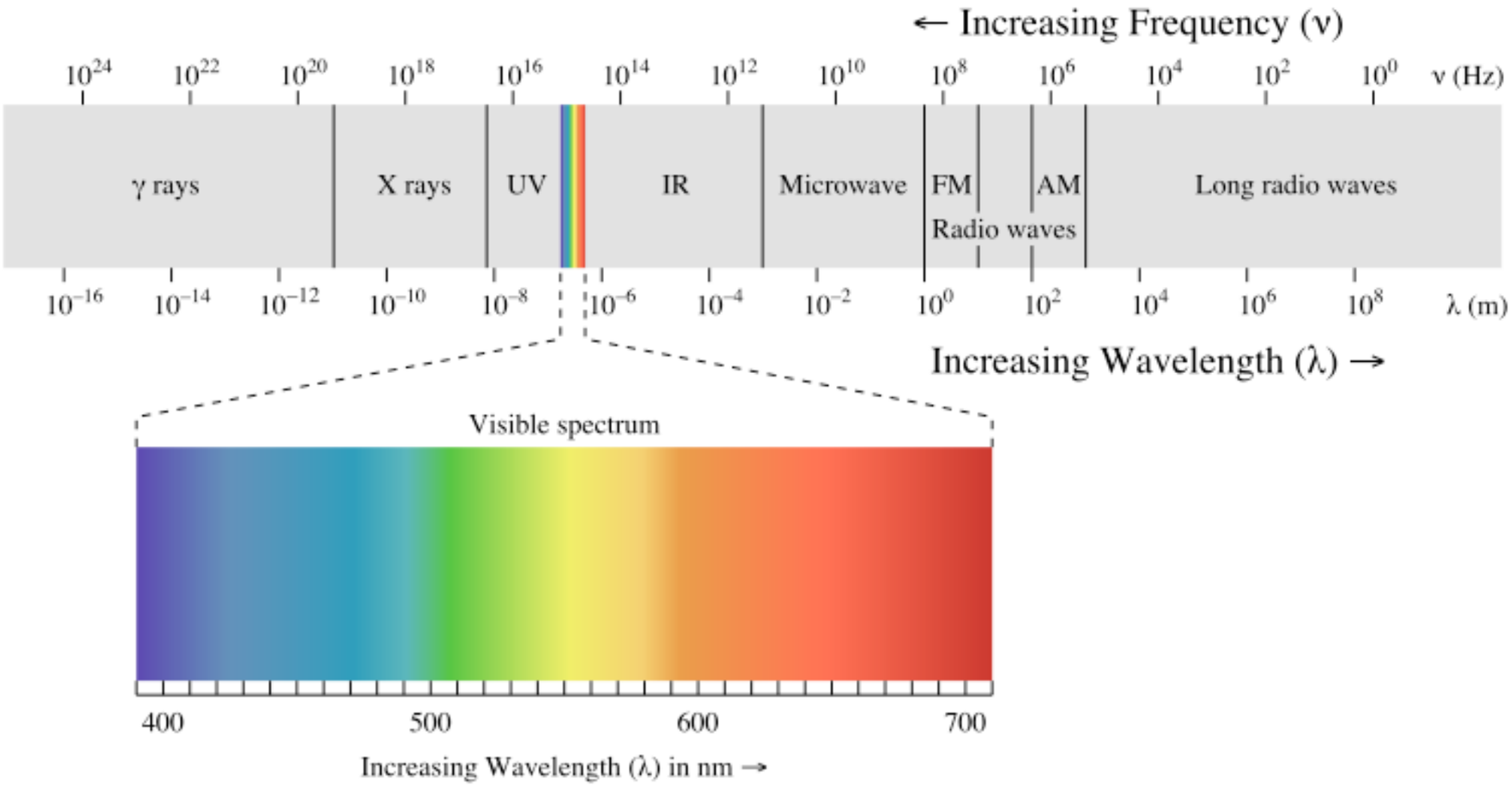
Light in Physics

- A light ray is an electromagnetic wave



- Light spectrum = sum of harmonic waves
  - Combination of different wavelengths λ

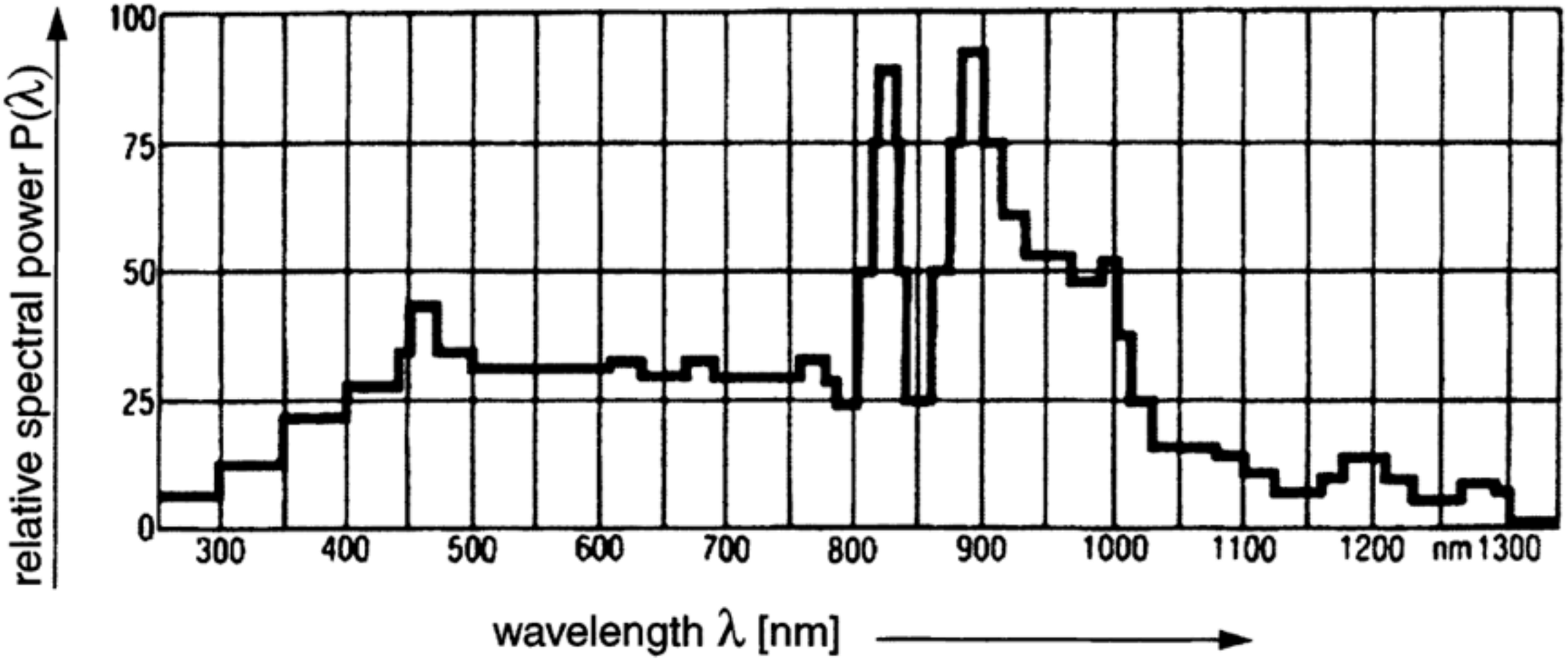
The Visible Spectrum



Visible light: 380nm - 780nm

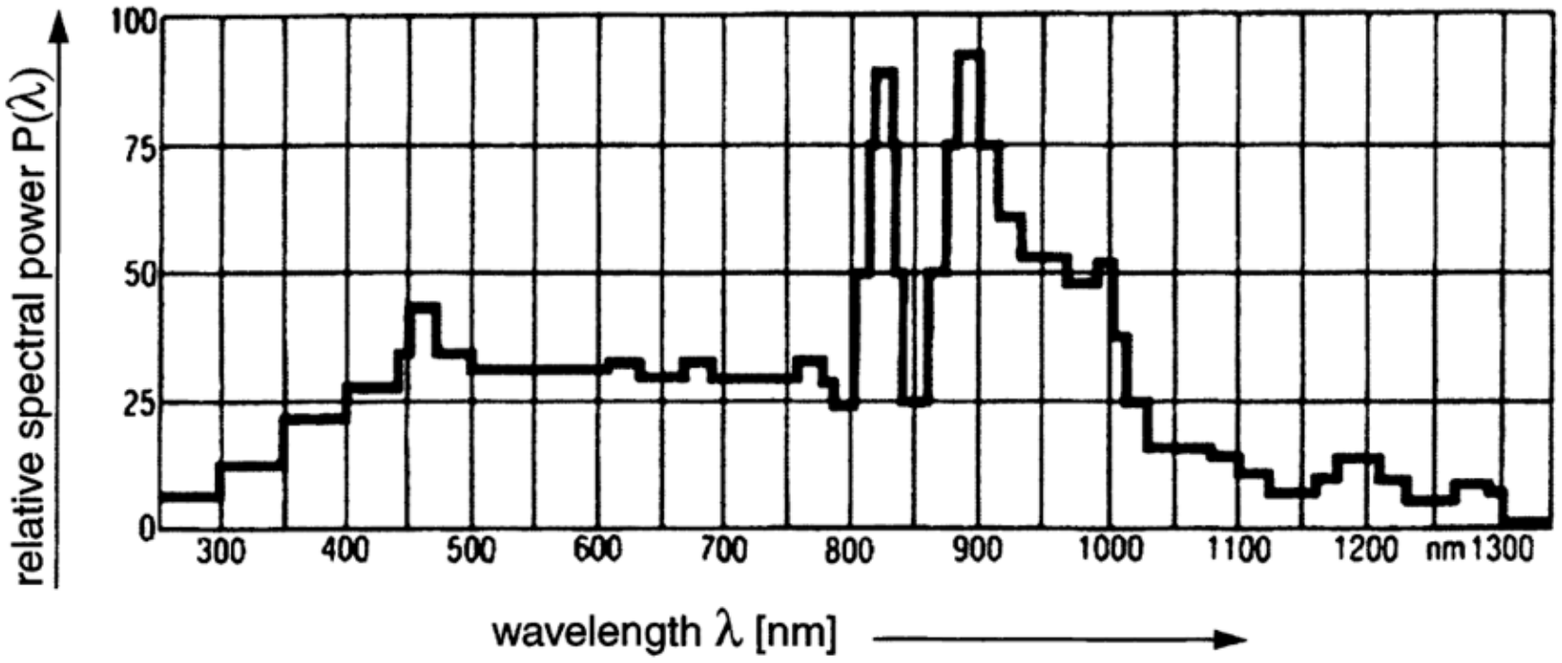
Power Spectrum of Light Sources

- Relative Spectral Power Density $P(\lambda)$



Terminology

- Hue ≈ dominant wavelength
- Saturation ≈ exitation purity
- Brightness ≈ luminance



Colors

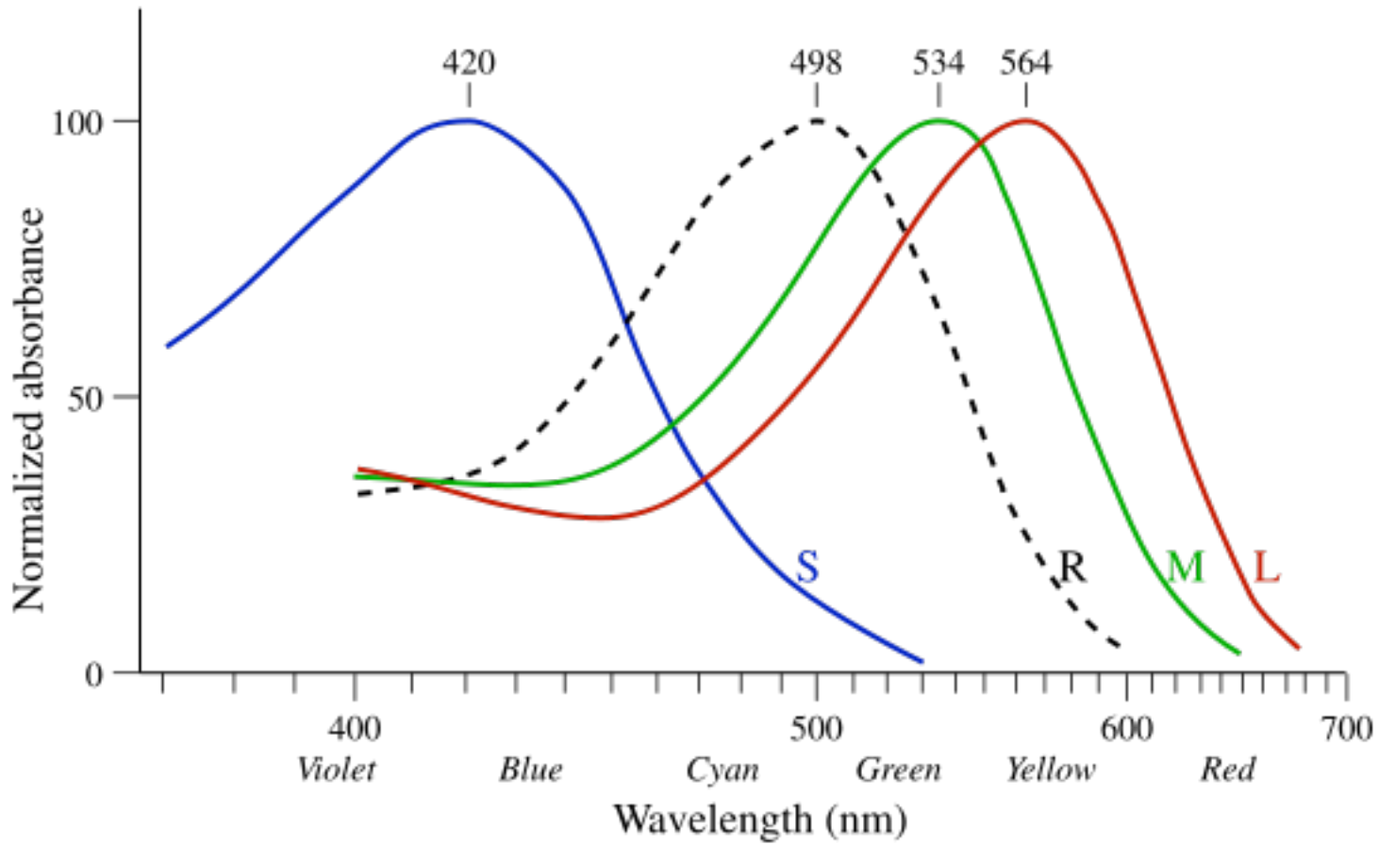
- Artist's view
- Physics
- **Biology**
- Computer Graphics

Perceiving Color

- Each ray carries a spectrum $P(\lambda)$
- $P(\lambda)$ contains more information than humans can and need to process
- How do humans perceive color?

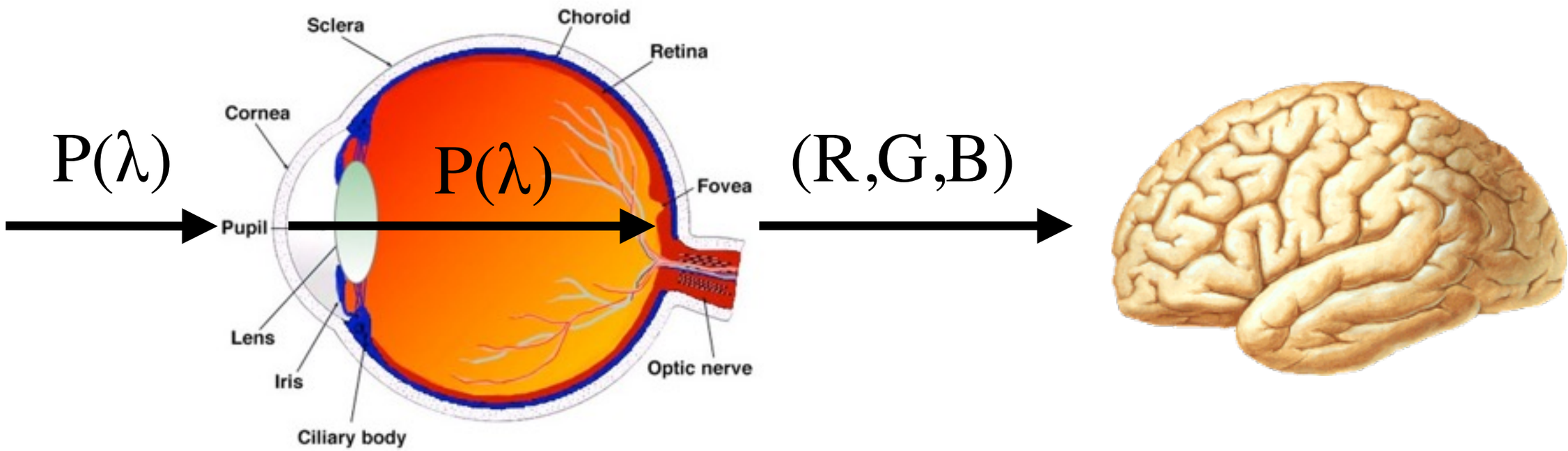
Human Eye

- Rods (*Stäbchen*)
  - very light sensitive
  - achromatic vision
- Cones (*Zapfen*)
  - less light sensitive
  - chromatic vision
  - three types for RGB



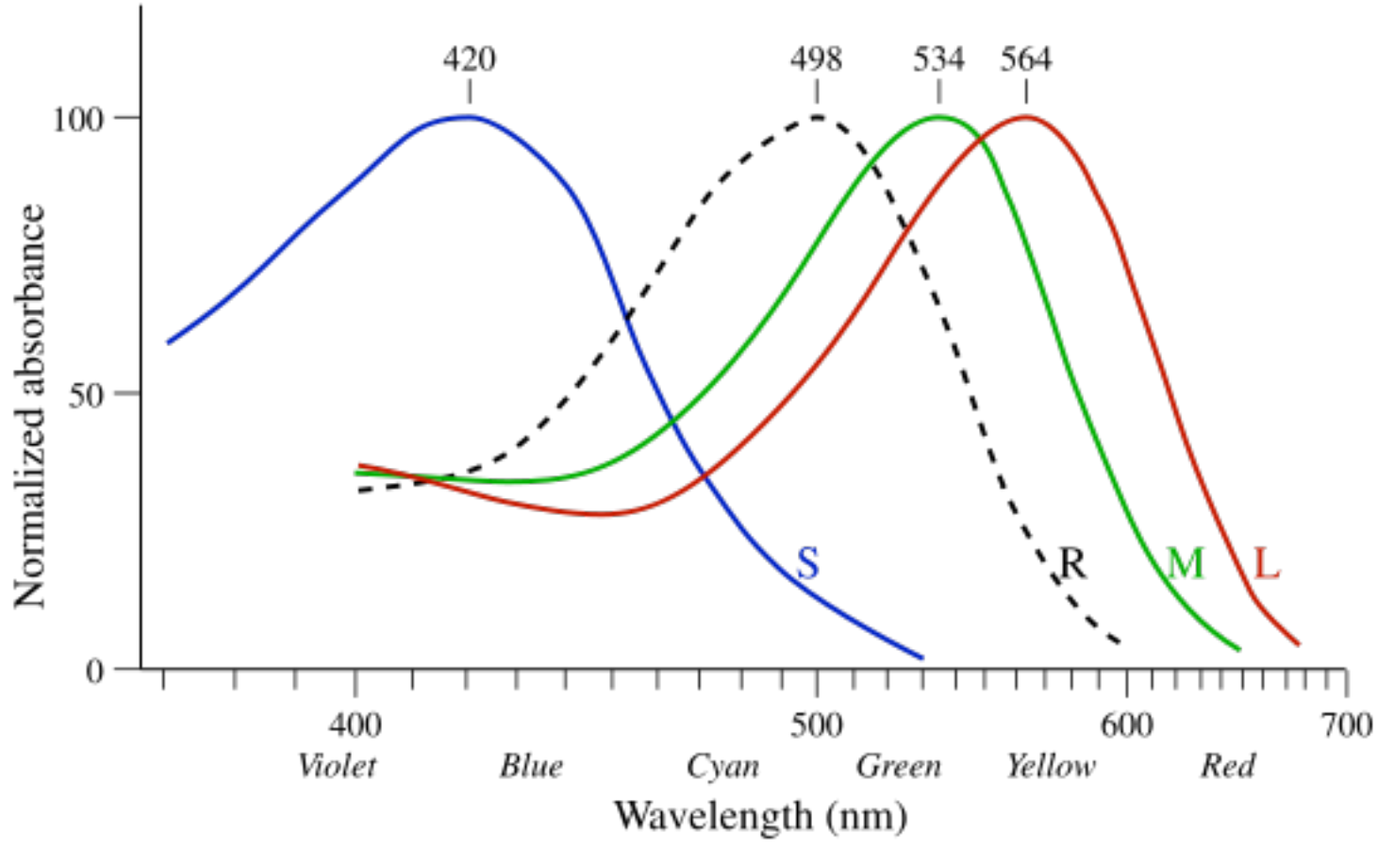
Perceiving Color

- Humans project $P(\lambda)$ into 3D color space



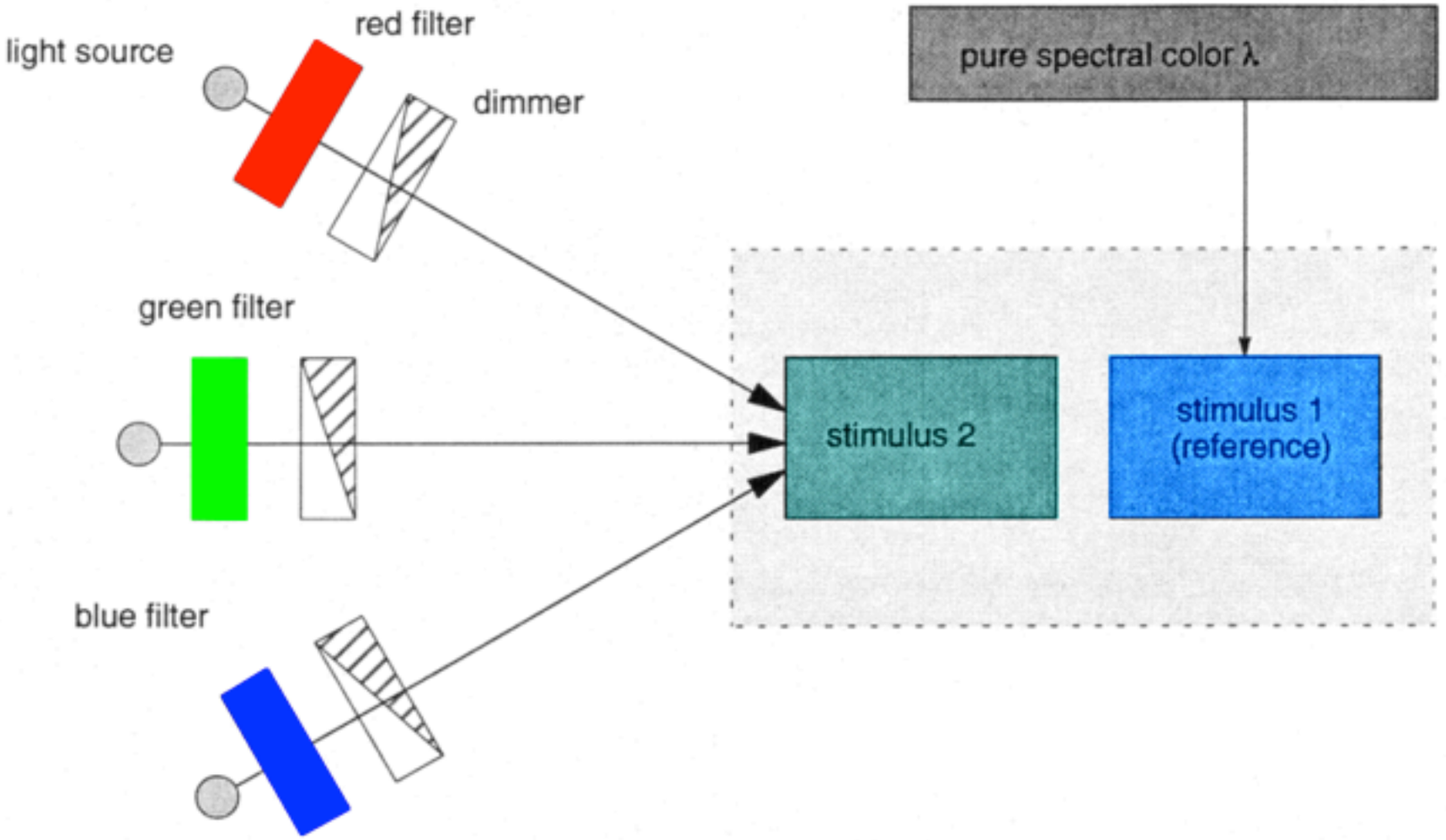
Perceiving Color

- Humans project $P(\lambda)$ into 3D color space
  - Sensitivity of the three types of cones are three basis functions $r(\lambda)$, $g(\lambda)$, $b(\lambda)$



The CIE Primary System (1931)

- **C**ommission **I**nternationale de l'**E**clairage
- Setup for measuring human color sensitivity (436 nm, 546 nm, 700 nm)

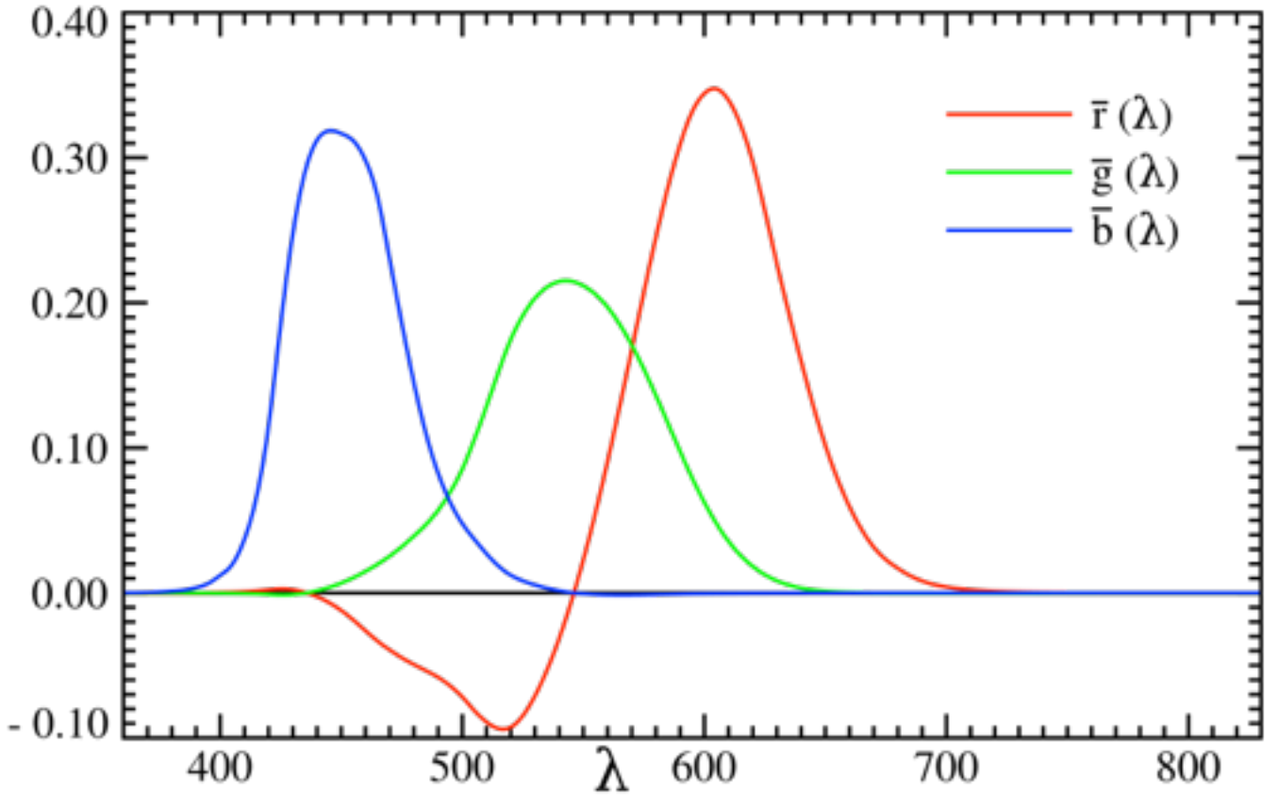


Spectral Sensitivity Functions

- Color matching function with primary colors B=436nm, G=546nm, R=700nm

$$\lambda \approx \bar{r}(\lambda) \cdot R + \bar{g}(\lambda) \cdot G + \bar{b}(\lambda) \cdot B$$

- Bad result: Need negative weights



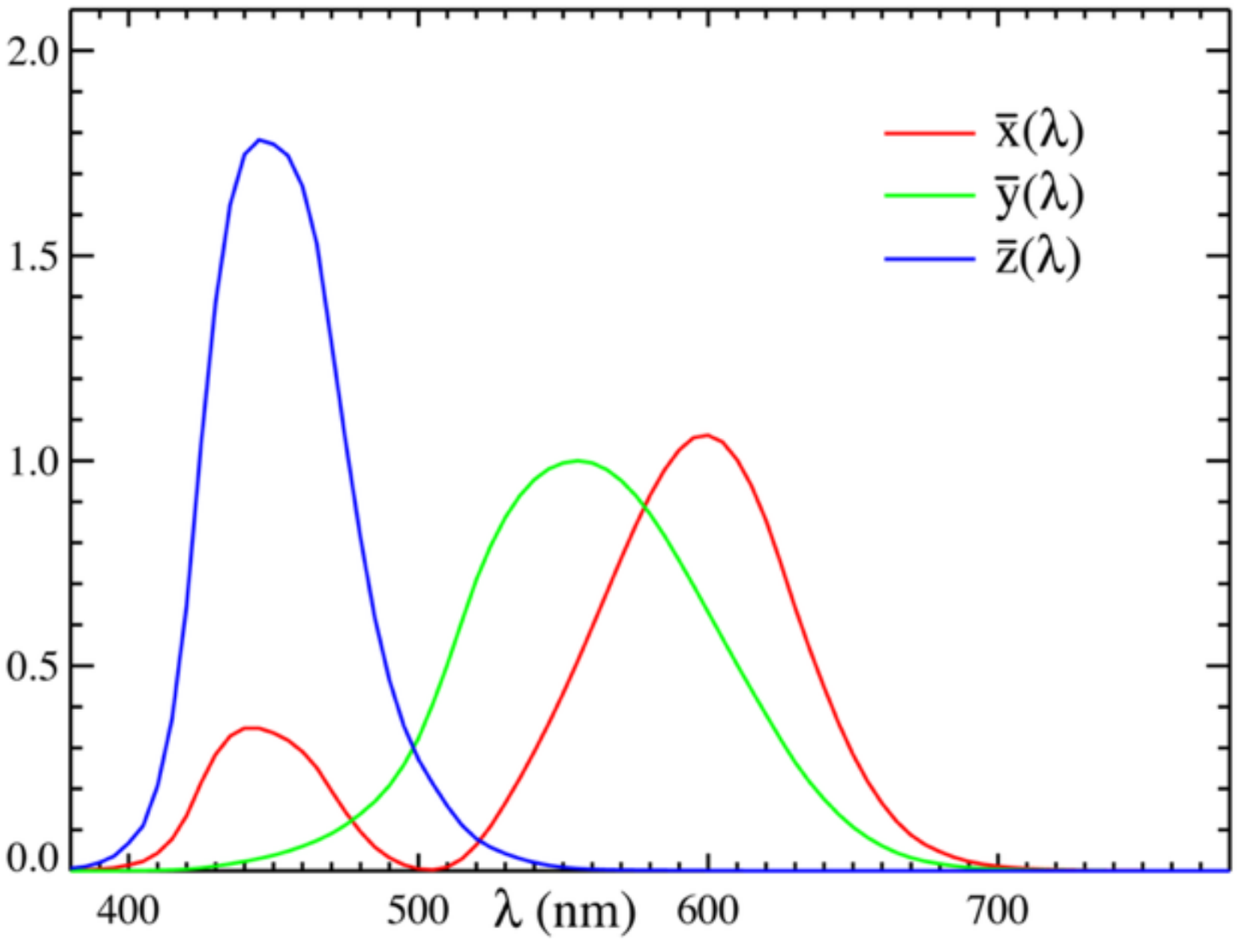
CIE Spectral Response Functions

- We want positive coordinates!
- Define new CIE primary colors **X**, **Y**, **Z**
- Gives new color-matching function $x(\lambda), y(\lambda), z(\lambda)$

$$
\begin{aligned}
\bar{x}(\lambda) &= +2.36\,\bar{r}(\lambda) - 0.515\,\bar{g}(\lambda) + 0.005\,\bar{b}(\lambda) \\
\bar{y}(\lambda) &= -0.89\,\bar{r}(\lambda) + 1.426\,\bar{g}(\lambda) + 0.014\,\bar{b}(\lambda) \\
\bar{z}(\lambda) &= -0.46\,\bar{r}(\lambda) + 0.088\,\bar{g}(\lambda) + 1.009\,\bar{b}(\lambda)
\end{aligned}
$$

CIE Spectral Response Functions

- Now the weights/coordinates are positive!



CIE Primaries of a Color Stimulus

- *Tristimulus vector* (X, Y, Z) provides a quantification of any spectral color stimulus $\mathrm{P}(\lambda)$
- Compute by inner products of x, y, z and P

$$
\begin{aligned}
X &= \int P(\lambda)\ \bar{x}(\lambda)\,\mathrm{d}\lambda \\
Y &= \int P(\lambda)\ \bar{y}(\lambda)\,\mathrm{d}\lambda \\
Z &= \int P(\lambda)\ \bar{z}(\lambda)\,\mathrm{d}\lambda
\end{aligned}
$$

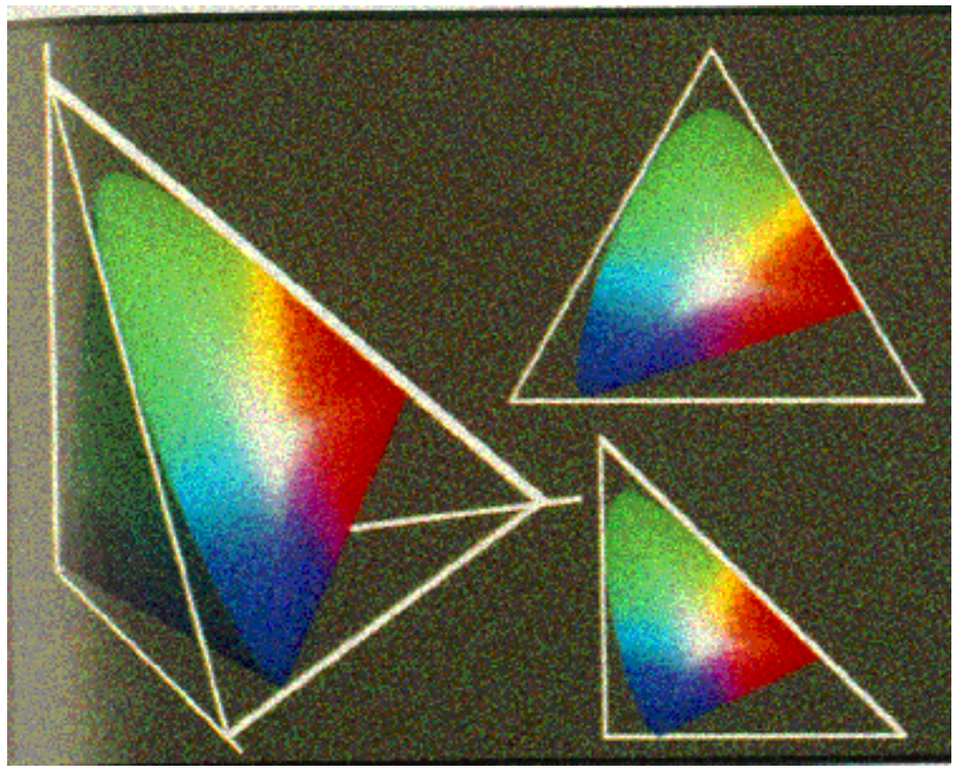
The CIE Chart

- Get 2D chart by projection onto the plane perpendicular to spatial diagonal
  - Plane equation: $\{X + Y + Z = 1\}$
  - Plane normal: $(1,1,1)^{\mathrm{T}}$

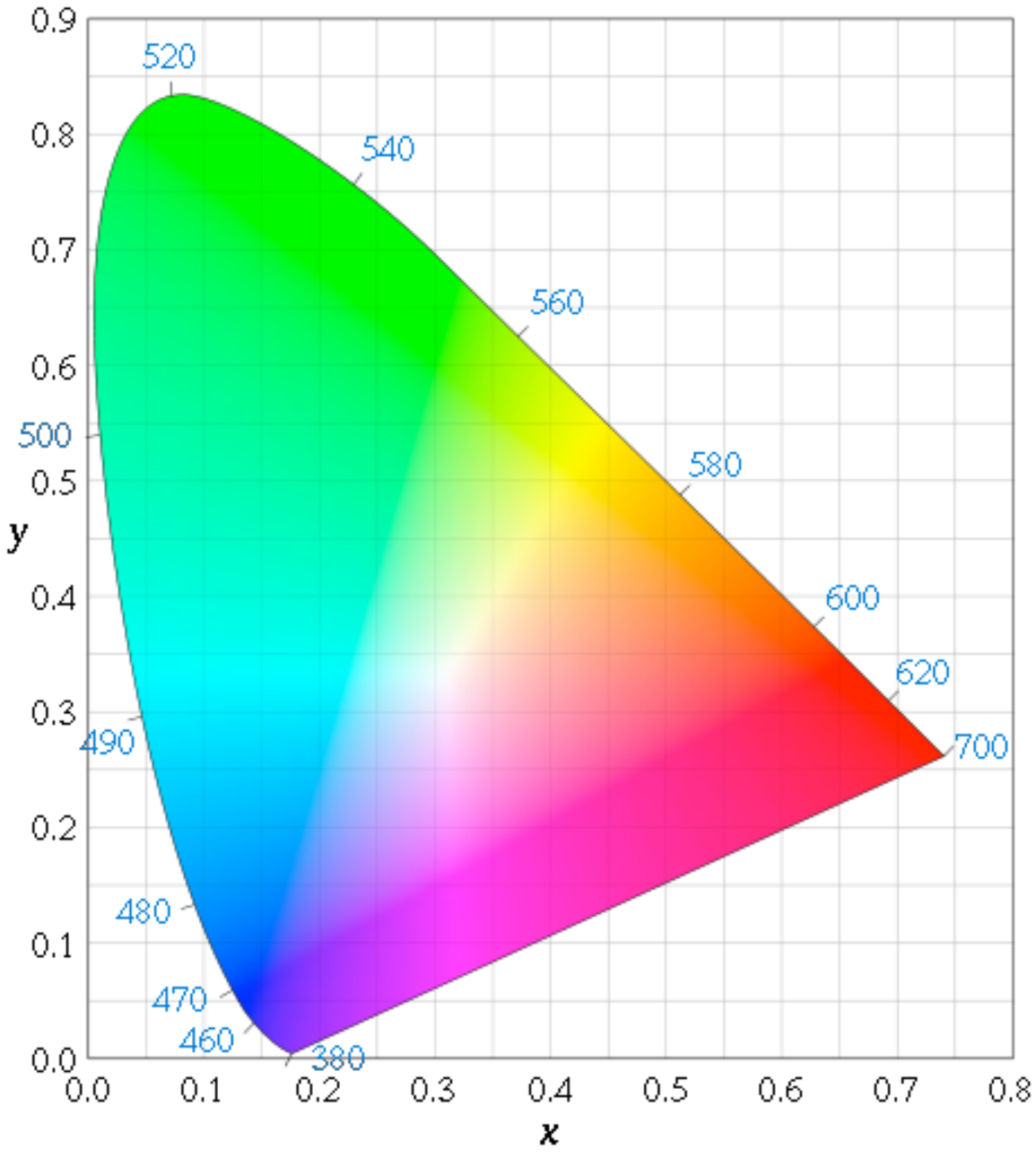
- Resulting (x, y) pair characterizes color

$$
\begin{aligned}
x &= \frac{X}{X+Y+Z} \\
y &= \frac{Y}{X+Y+Z} \\
z &= \frac{Z}{X+Y+Z} = 1 - x - y
\end{aligned}
$$

The CIE Chart

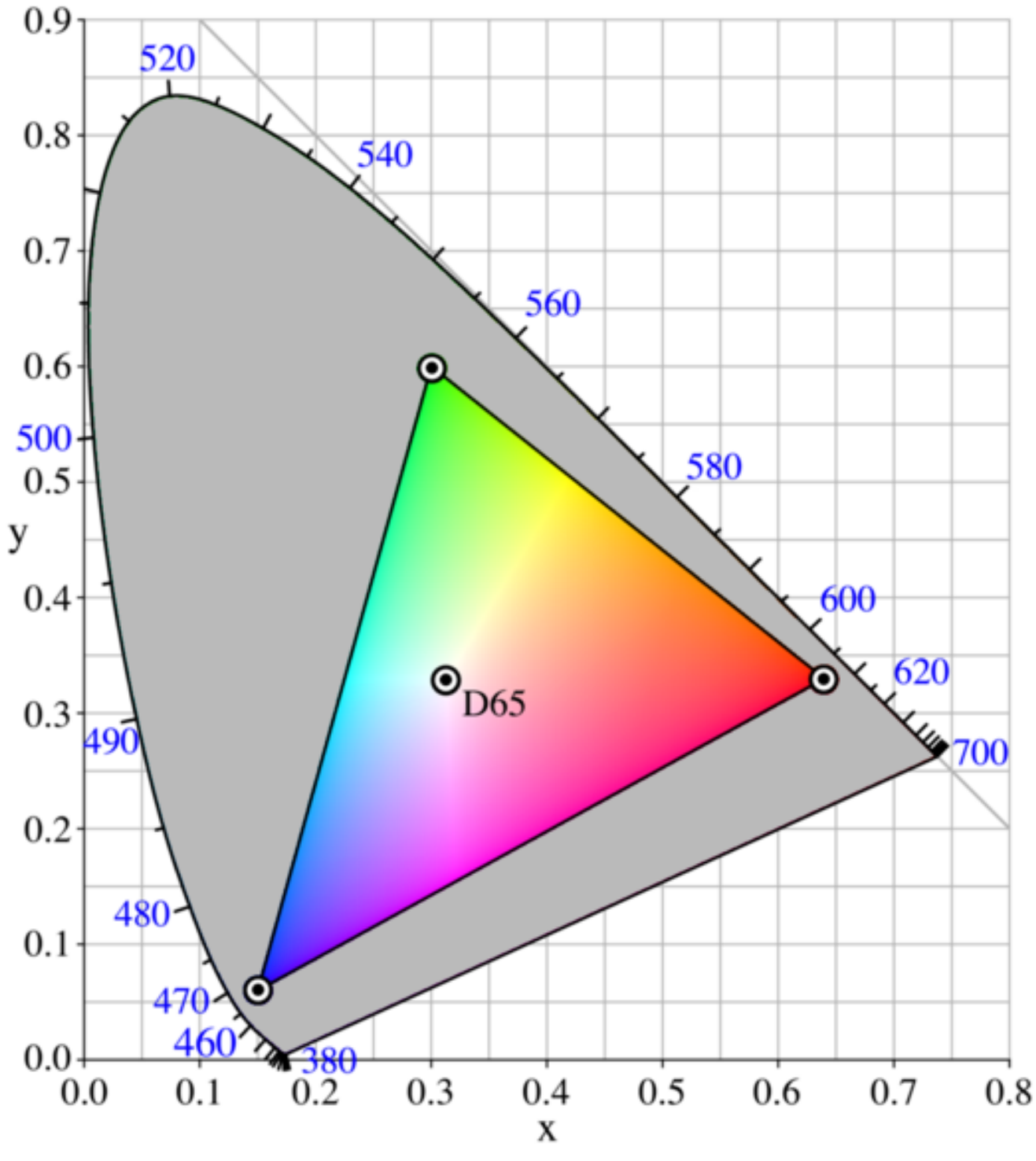


3D (X,Y,Z)



2D (x,y)

sRGB Color Gamut



Colors

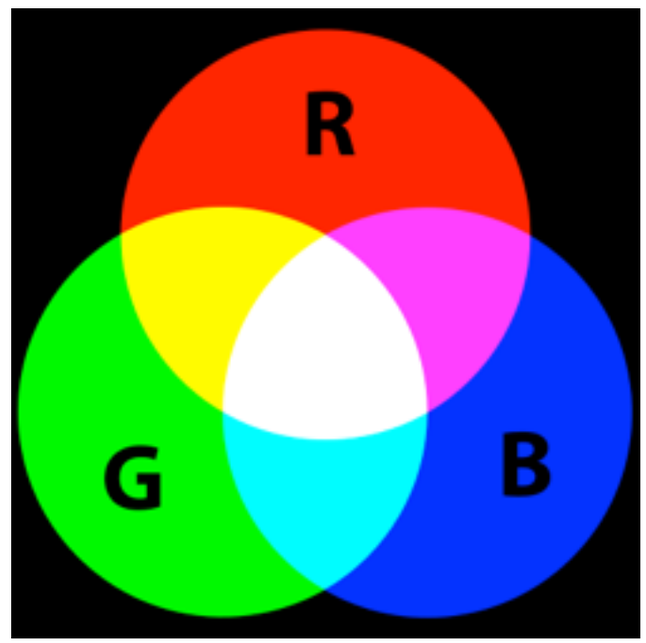
- Artist's view
- Physics
- Biology
- **Computer Graphics**

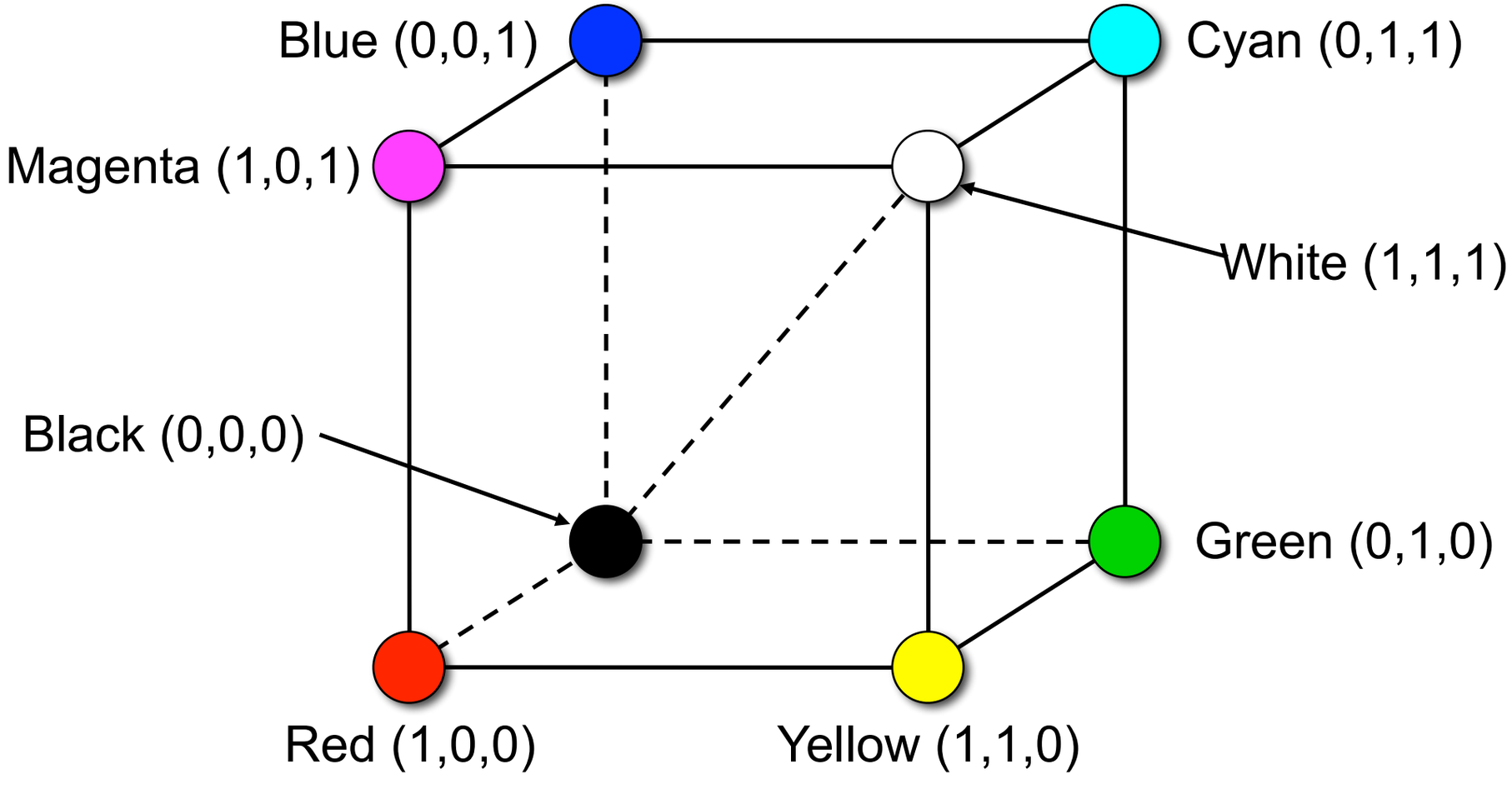
Color Models

- Biology oriented
  - CIE
- Hardware oriented
  - RGB
  - CMY, CMYK
- User oriented
  - HSV

RGB Color Model

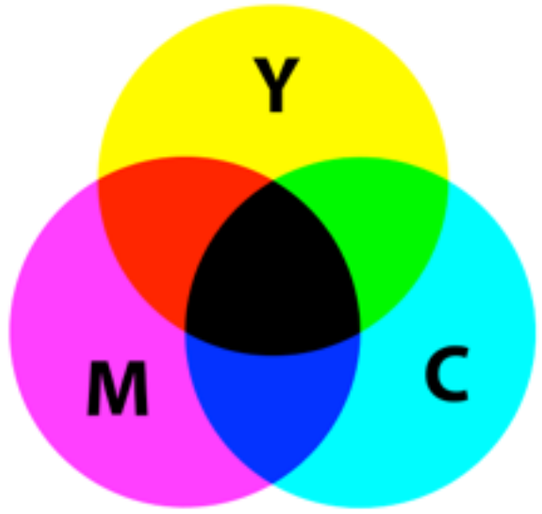
- Additive system: black + (r,g,b)
- Used for color CRTs, LCDs

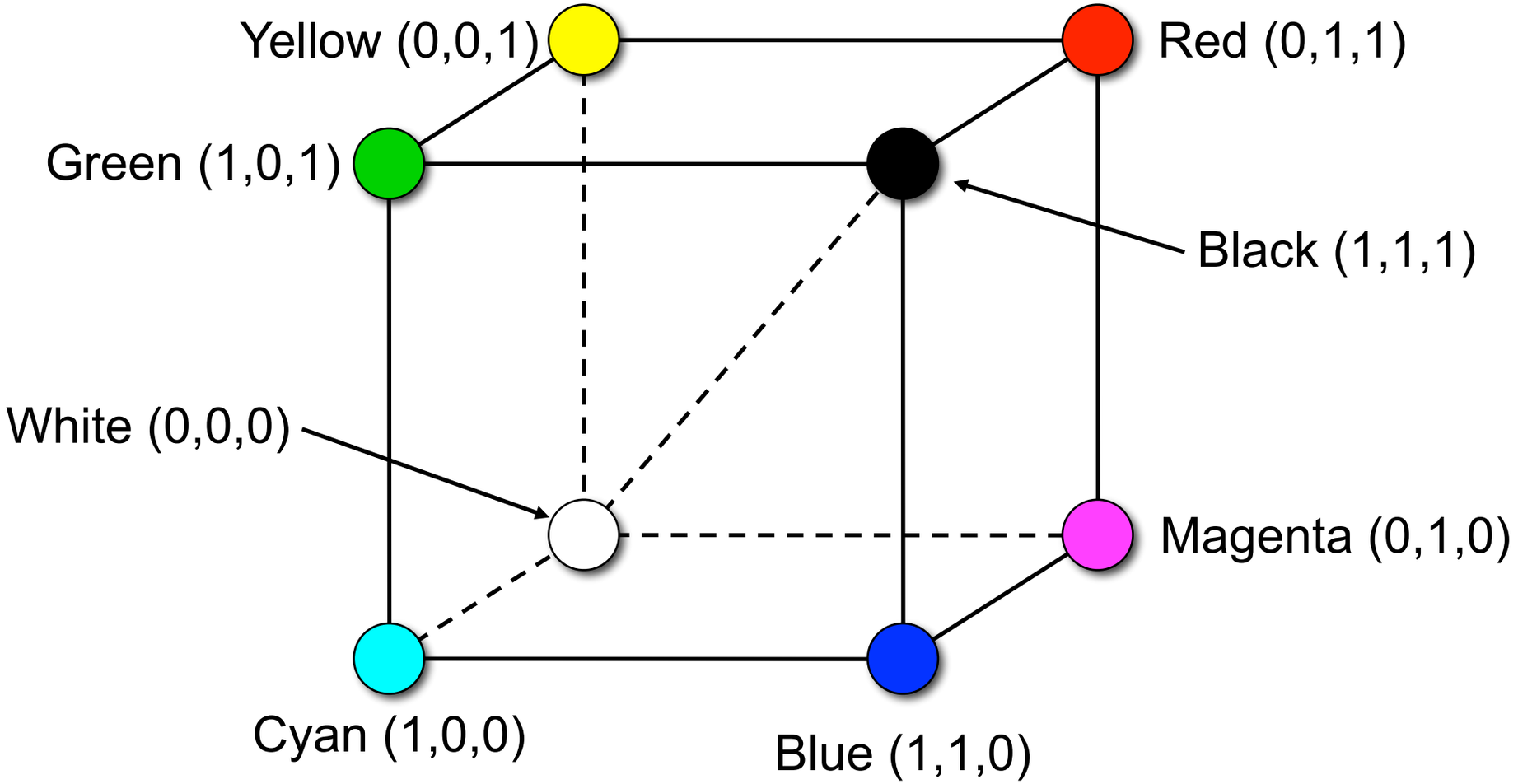




CMY(K) Color Model

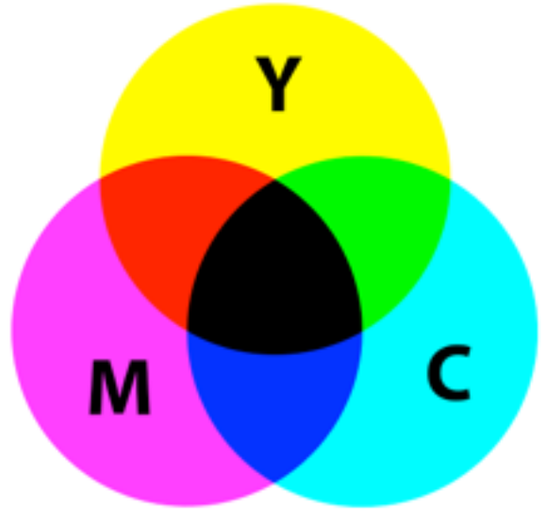
- Complementary colors to RGB
  - Cyan, magenta, yellow
- Subtractive system for color printers





CMY(K) Color Model

- Complementary colors to RGB
  - Cyan, magenta, yellow
- Subtractive system for color printers

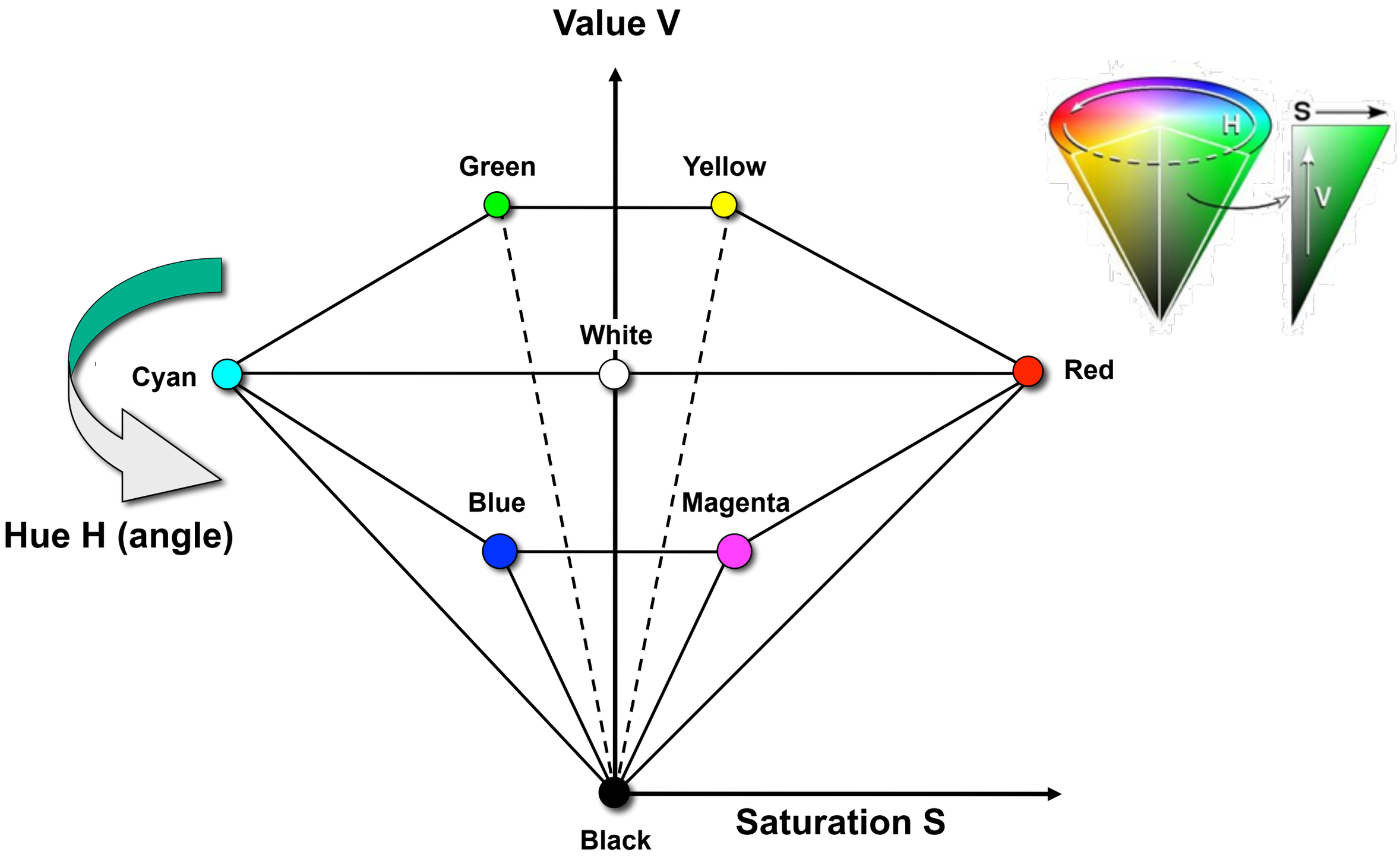


- Conversion
  - (R,G,B) = (1,1,1) - (C,M,Y)
- Extension: CMYK (K=black)
  - K = min{C,M,Y}
  - C=C-K, M=M-K, Y=Y-K

CMY(K) Color Model

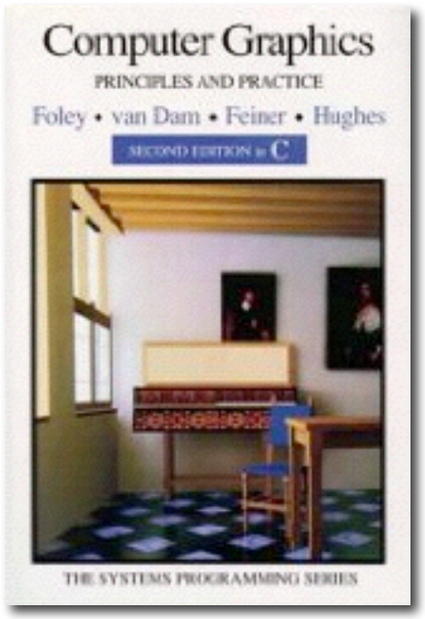
[wikipedia]

HSV Color Model



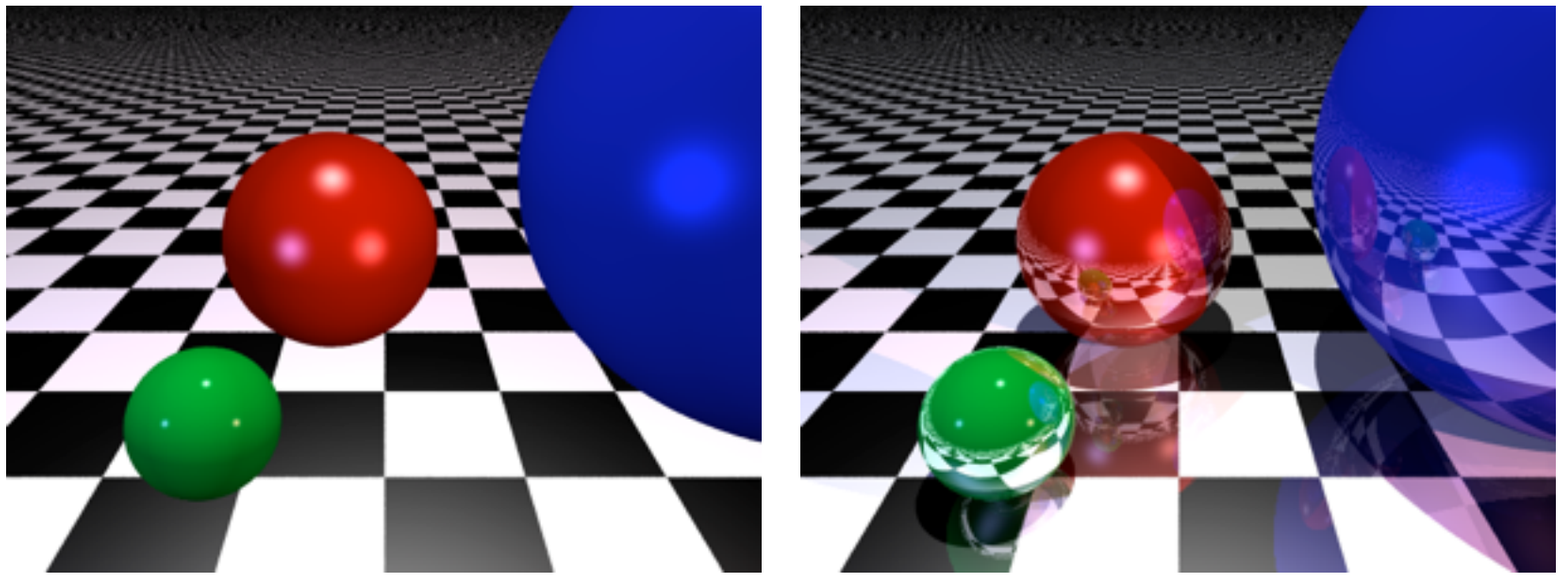
Literature

- Foley, van Dam, Feiner, Hughes: ***Computer Graphics: Principles and Practice***.
  - Chapter 13



Introduction to Computer Graphics

Lighting



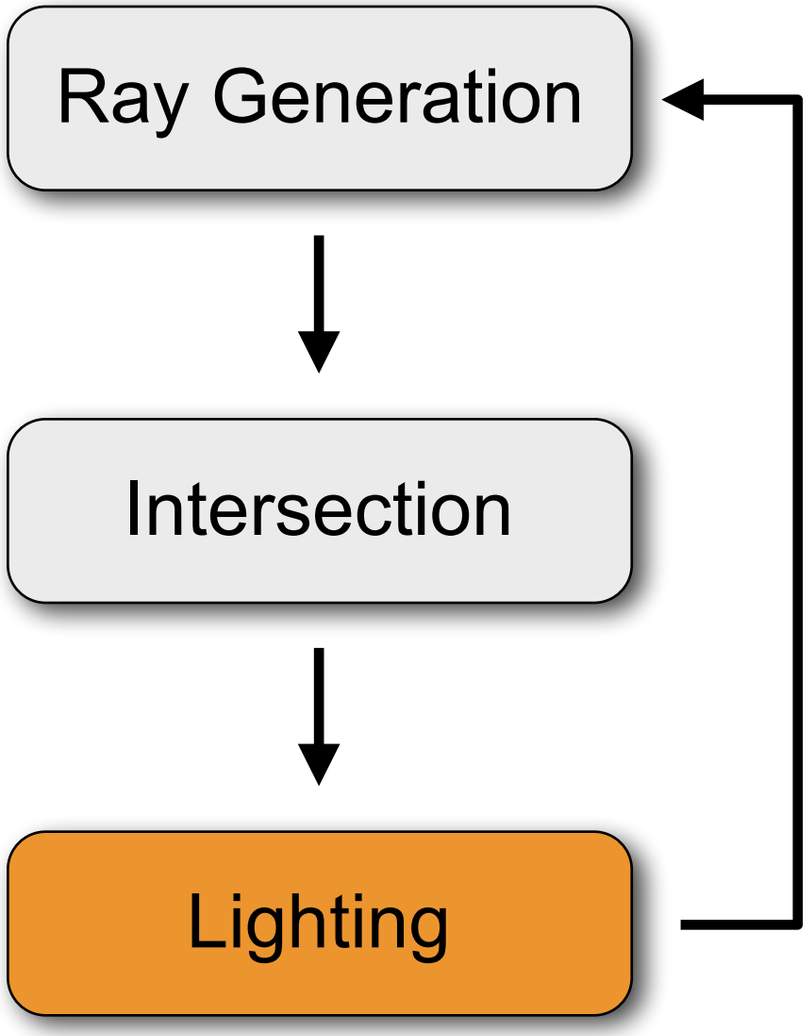
Prof. Dr. Mario Botsch

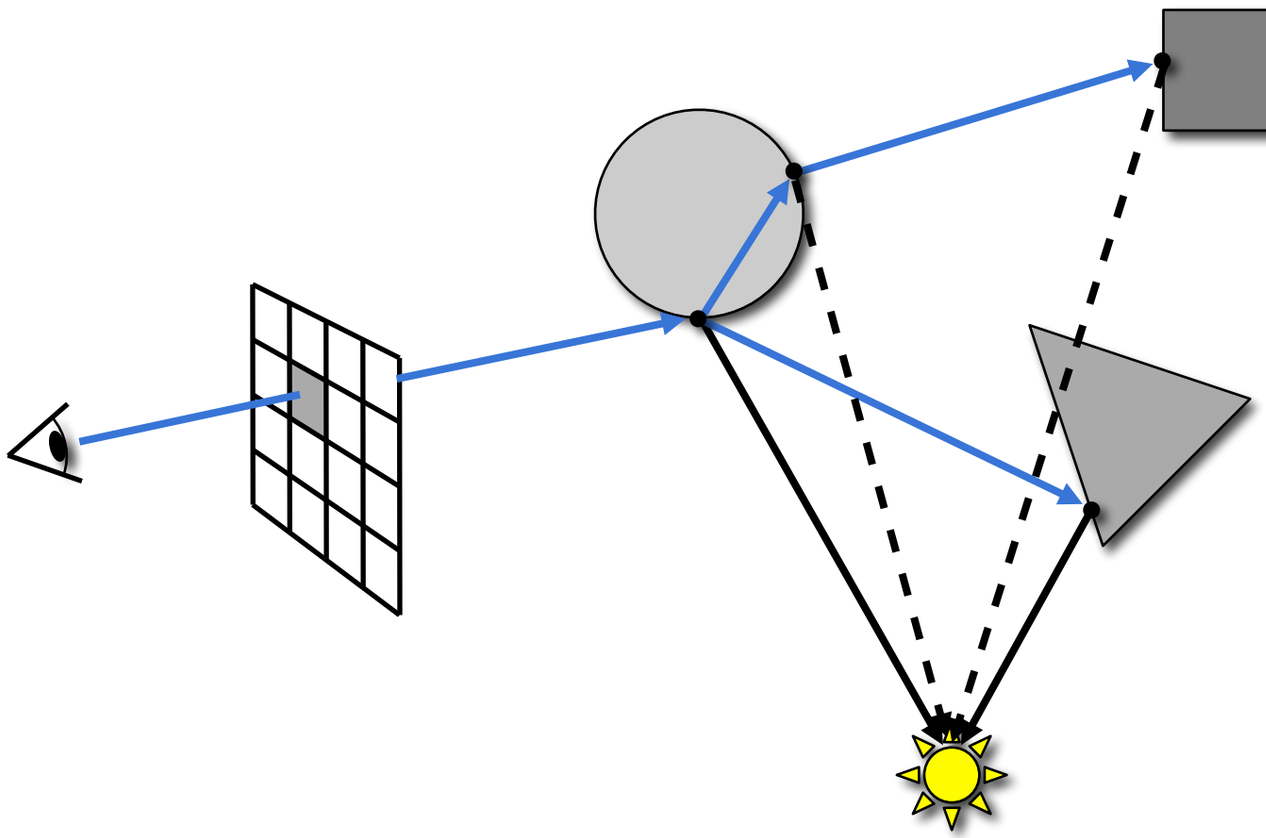
Computer Graphics & Geometry Processing

Universität Bielefeld

CITEC

Lighting



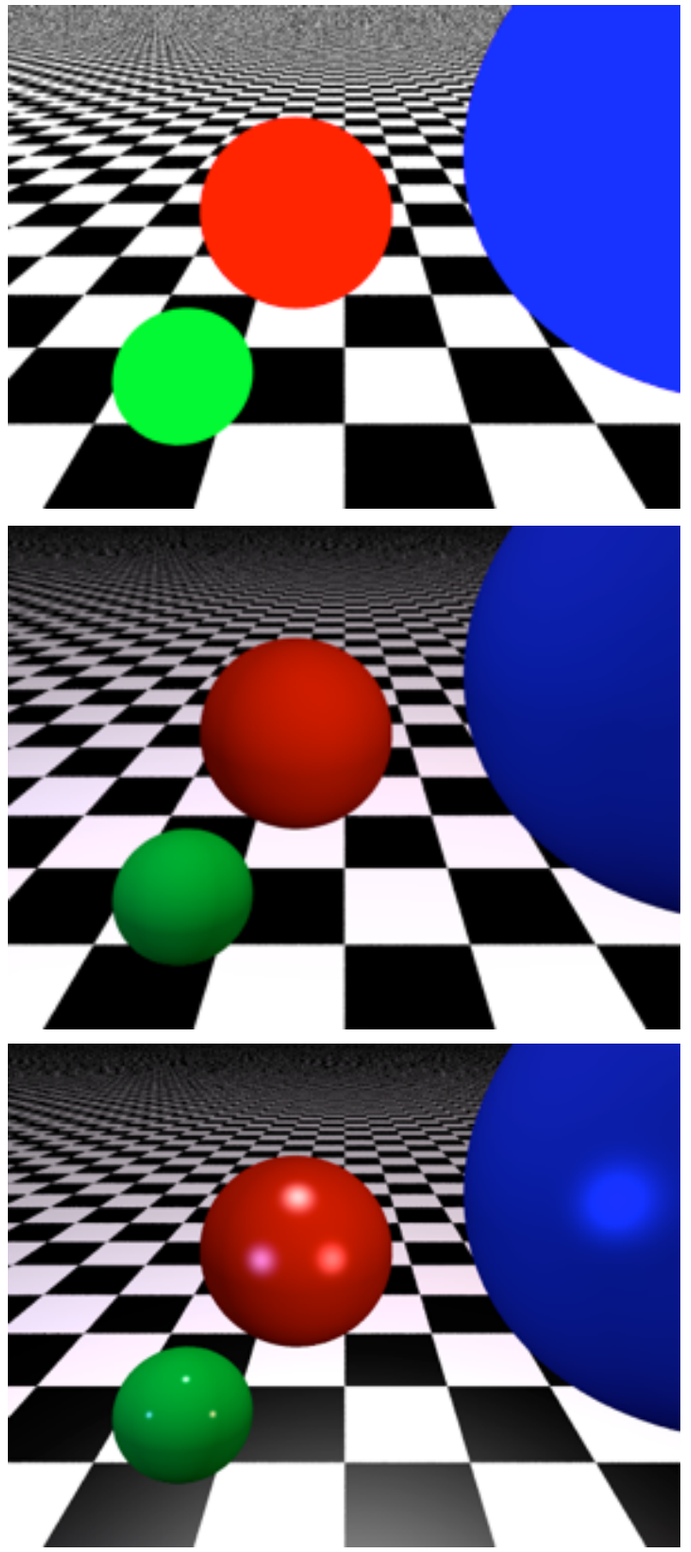


Lighting

- Determines color/brightness of primary ray
  - Gives the color of the corresponding pixel
- Different models for surface reflectance
  - Phong model: ambient, diffuse, specular (today)
  - General BRDF models (later)
- Shadows
  - Shadow rays test whether a point is illuminated

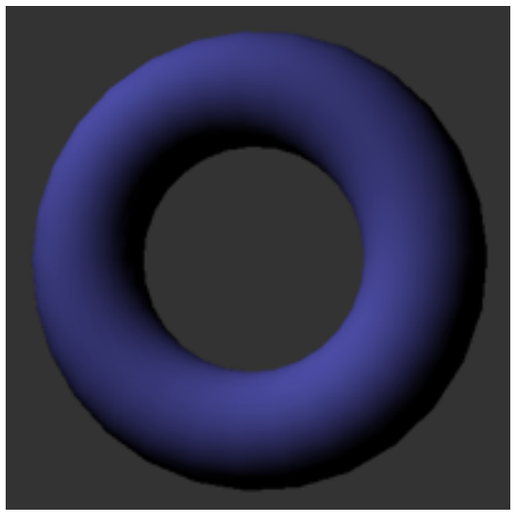
Phong Lighting Model

- **Ambient lighting**
  - approximate global light transport / exchange
- **Diffuse lighting**
  - dull / matt surfaces
- **Specular lighting**
  - shiny surfaces

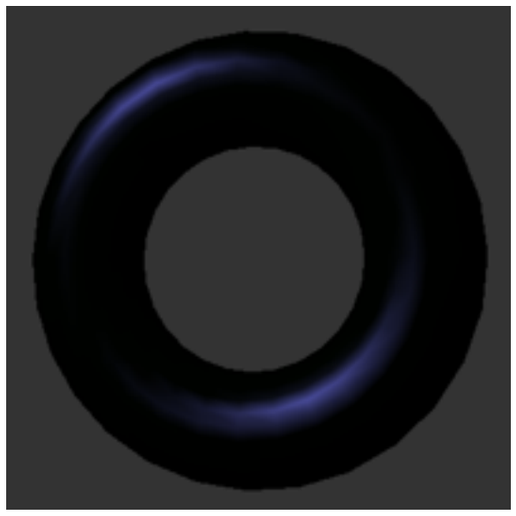


Phong Lighting Model

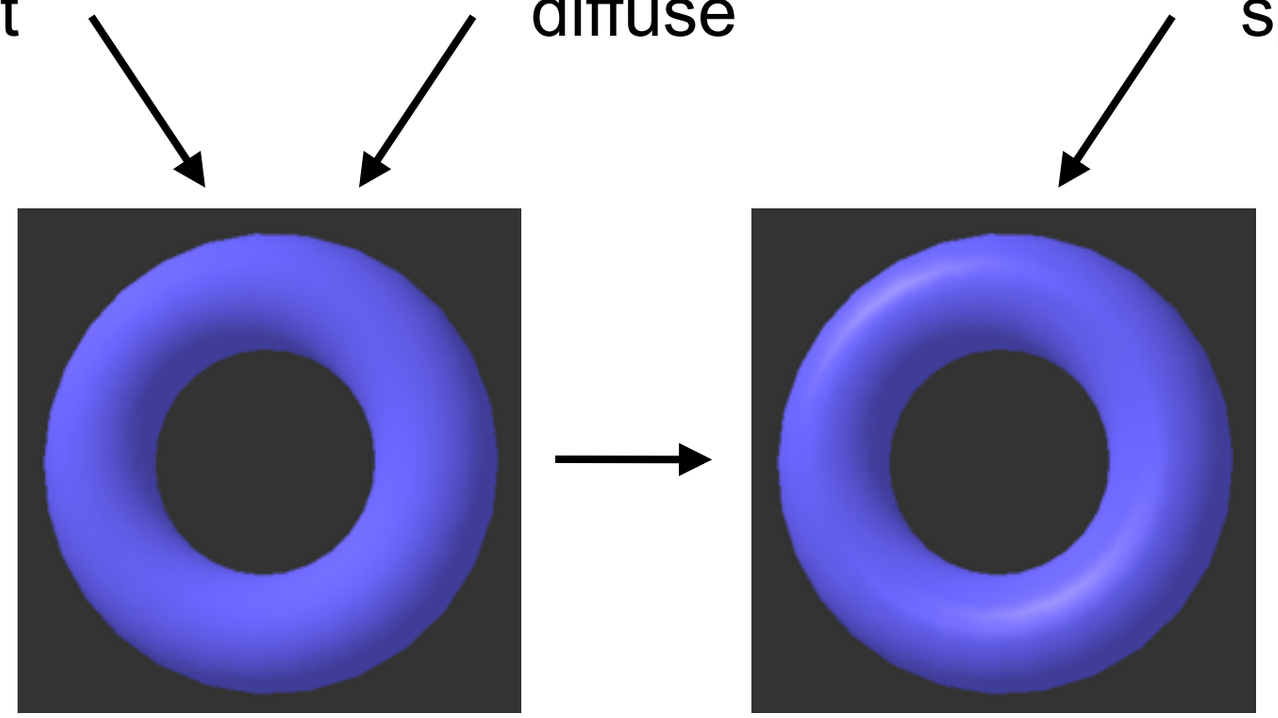
ambient



diffuse



specular



Ambient Light

- Uniform incoming light, uniform reflection

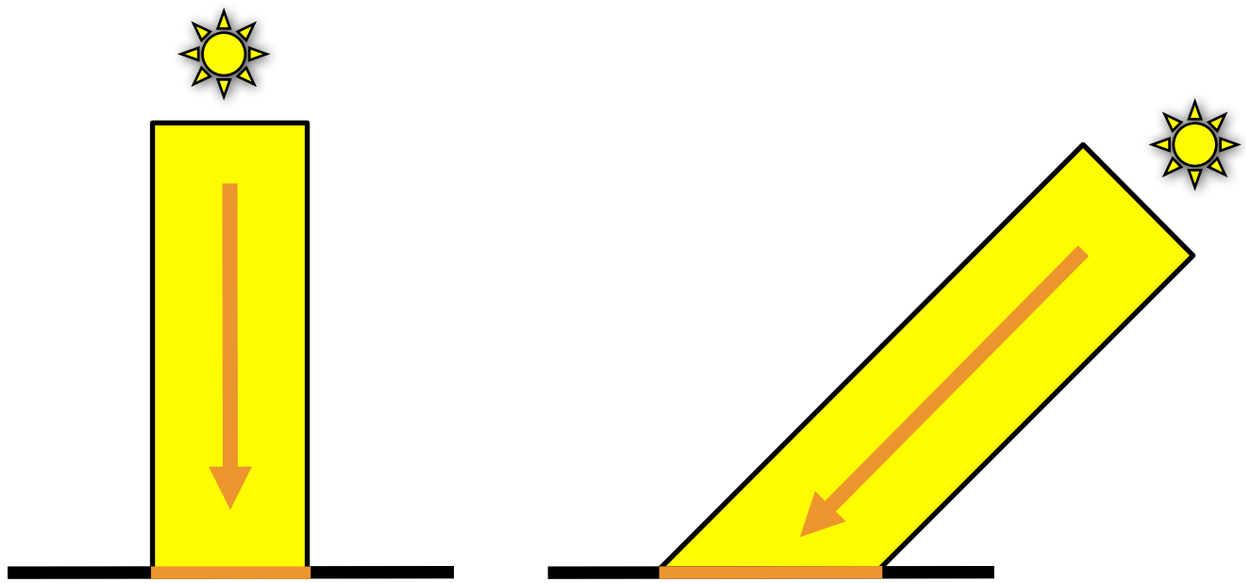
$$I = I_a k_a$$

ambient light in the scene

ambient material parameter

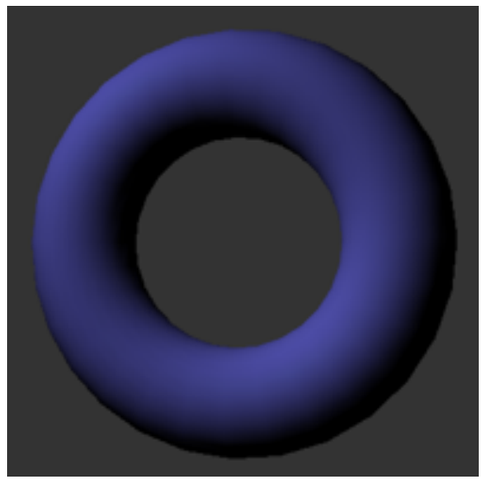
Diffuse Reflection

- Directed incoming light, uniform reflection
- Brightness ~ received energy
  - area ~ $1/\cos\theta$
  - brightness ~ $\cos\theta$
  - *Lambertian reflection*



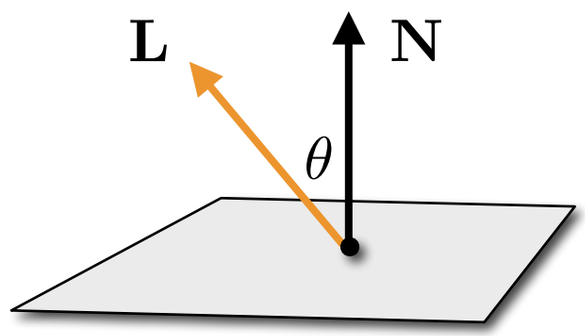
Diffuse Reflection

- Directed incoming light, uniform reflection
- Brightness ~ received energy
  - area ~ $1/\cos\theta$
  - brightness ~ $\cos\theta$
  - *Lambertian reflection*



$$I_d = I_l k_d \cos\theta = I_l k_d (\mathbf{N} \cdot \mathbf{L})$$

light source — material parameter — vectors are normalized!



attn: no diffuse reflection if $(\mathbf{N}\cdot\mathbf{L})<0$

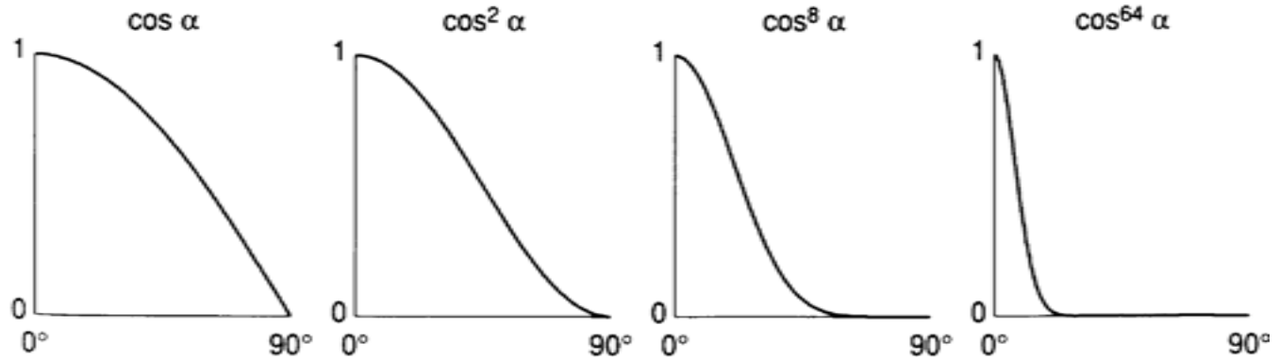
Specular Reflection

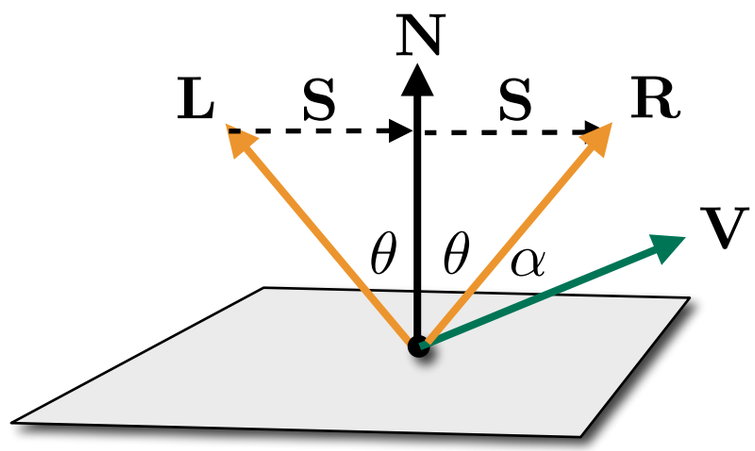
- Directed incoming light, directed reflection
- Reflected light direction $\mathbf{R}$

$$\mathbf{R} = \mathbf{N}\cos\theta + \mathbf{S} = 2\mathbf{N}\cos\theta - \mathbf{L} = 2\mathbf{N}(\mathbf{N}\cdot\mathbf{L}) - \mathbf{L}$$

- Model specular reflection by cosine powers

$$I_s = I_l\, k_s \cos^n \alpha = I_l\, k_s\, (\mathbf{R}\cdot\mathbf{V})^n$$

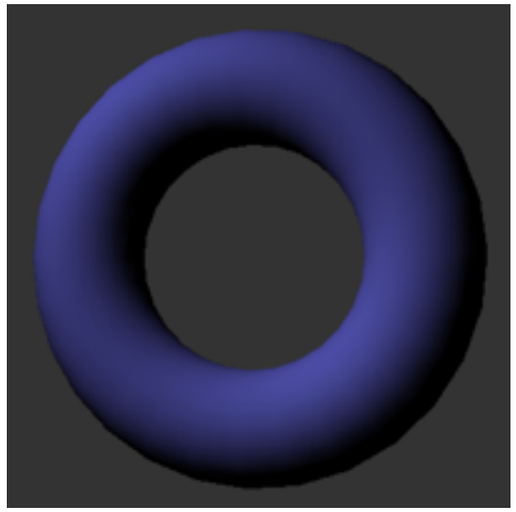




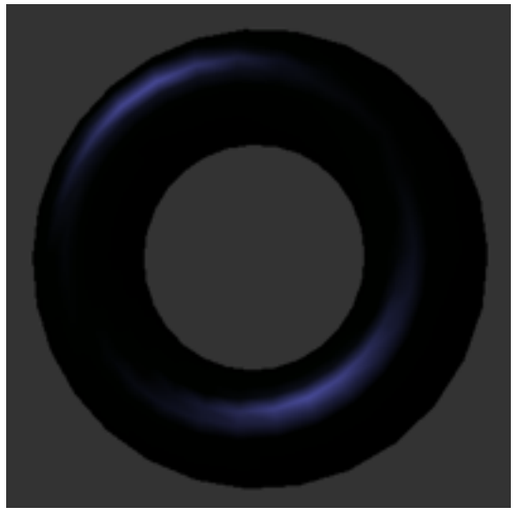
attn: no specular reflection if $(\mathbf{N}\cdot\mathbf{L})<0$ or $(\mathbf{R}\cdot\mathbf{V})<0$

Phong Lighting Model

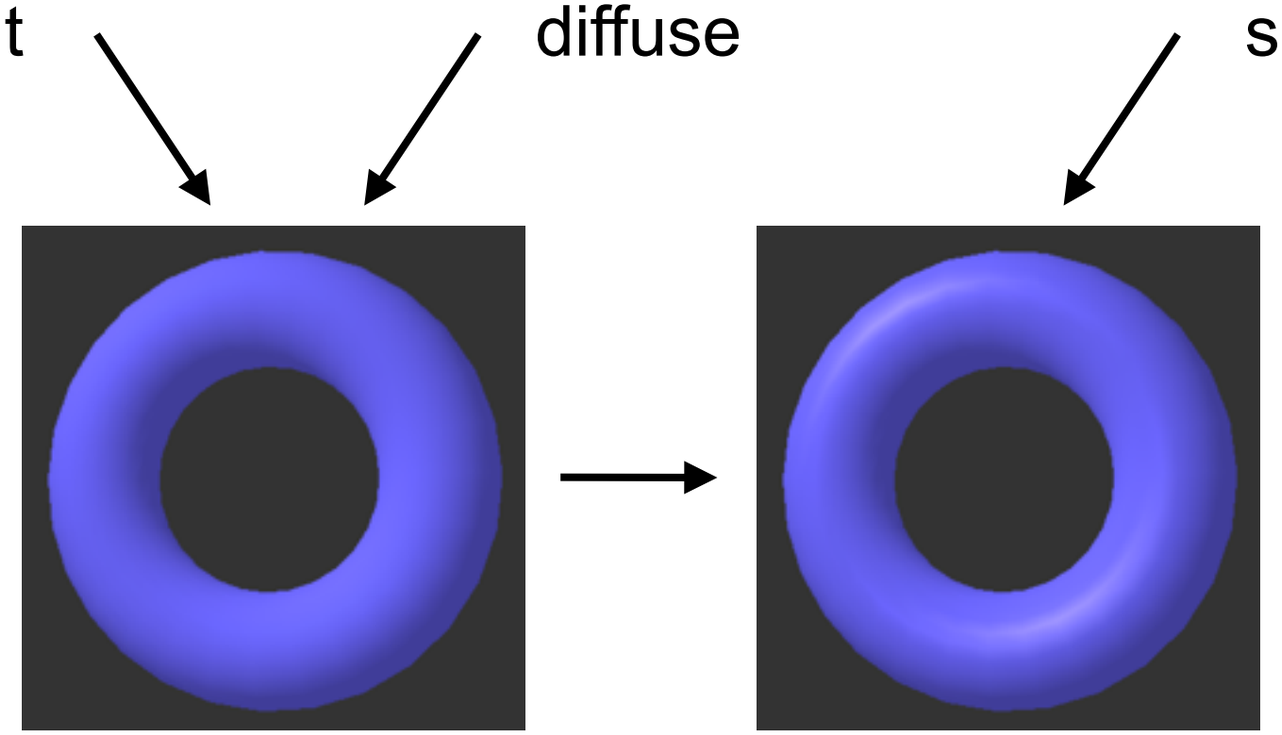
ambient



diffuse



specular

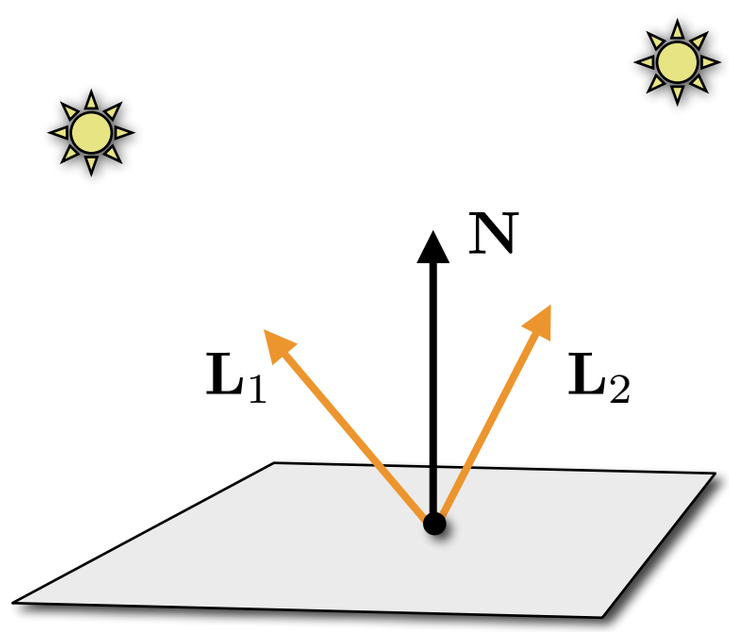


$$I = I_a k_a + I_l (k_d (\mathbf{N} \cdot \mathbf{L}) + k_s (\mathbf{R} \cdot \mathbf{V})^n)$$

Multiple Light Sources

- Sum up contributions of all light sources

$$I = I_a k_a + \sum_l (I_l k_d (\mathbf{N} \cdot \mathbf{L}) + I_l k_s (\mathbf{R} \cdot \mathbf{V})^n)$$



Lighting & Color

- Light and reflection are functions of wavelength

$$I_\lambda = I_{a,\lambda} k_{a,\lambda} + I_{l,\lambda} \left(k_{d,\lambda} \left(\mathbf{N} \cdot \mathbf{L} \right) + k_{s,\lambda} \left(\mathbf{R} \cdot \mathbf{V} \right)^n \right)$$

- We simply restrict to RGB components

$$I_r = I_{a,r} k_{a,r} + I_{l,r} \left(k_{d,r} \left(\mathbf{N} \cdot \mathbf{L} \right) + k_{s,r} \left(\mathbf{R} \cdot \mathbf{V} \right)^n \right)$$
$$I_g = I_{a,g} k_{a,g} + I_{l,g} \left(k_{d,g} \left(\mathbf{N} \cdot \mathbf{L} \right) + k_{s,g} \left(\mathbf{R} \cdot \mathbf{V} \right)^n \right)$$
$$I_b = I_{a,b} k_{a,b} + I_{l,b} \left(k_{d,b} \left(\mathbf{N} \cdot \mathbf{L} \right) + k_{s,b} \left(\mathbf{R} \cdot \mathbf{V} \right)^n \right)$$

Lighting & Color

- Light and reflection are functions of wavelength

$$I_\lambda = I_{a,\lambda} k_{a,\lambda} + I_{l,\lambda} \left(k_{d,\lambda} \left(\mathbf{N} \cdot \mathbf{L} \right) + k_{s,\lambda} \left(\mathbf{R} \cdot \mathbf{V} \right)^n \right)$$

- We simply restrict to RGB components

component-wise product

RGB material colors

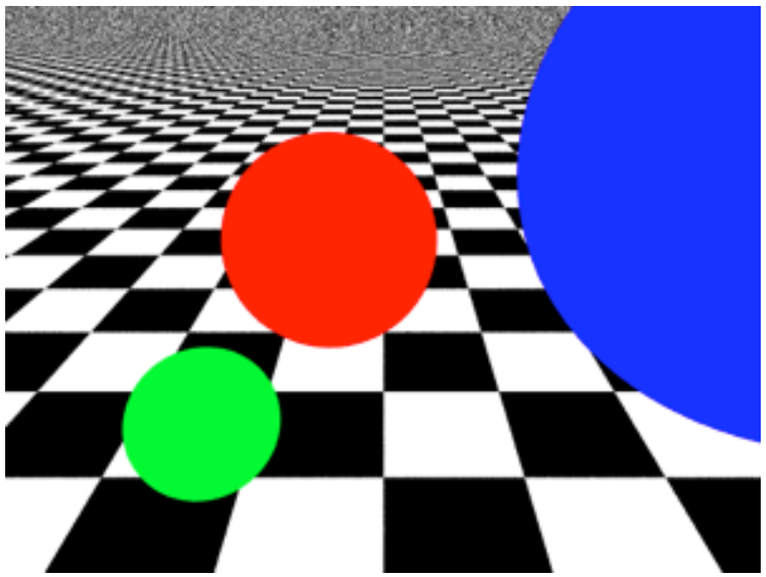
$$\mathbf{I} = \mathbf{I}_a * \mathbf{k}_a + \mathbf{I}_l * \left(\mathbf{k}_d \left(\mathbf{N} \cdot \mathbf{L} \right) + \mathbf{k}_s \left(\mathbf{R} \cdot \mathbf{V} \right)^n \right)$$

RGB ambiance color

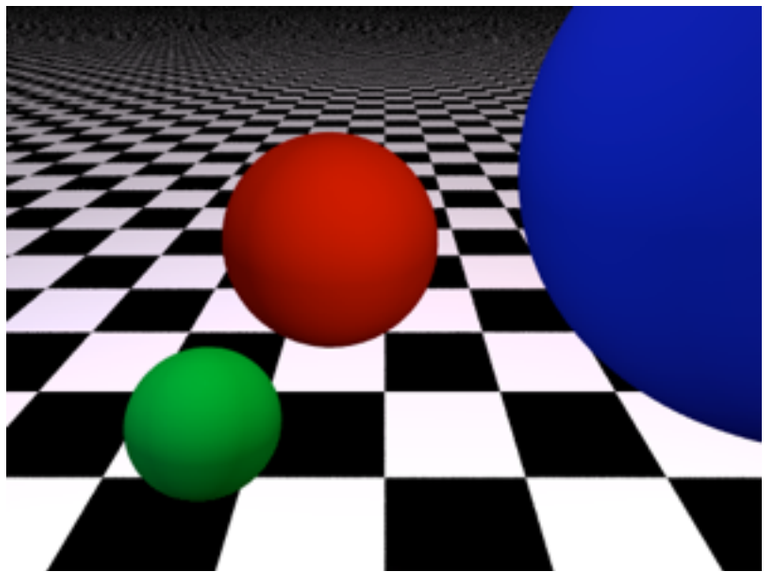
RGB light color

typically $\mathbf{k}_a = \mathbf{k}_d$ and $\mathbf{k}_s = (1,1,1)$

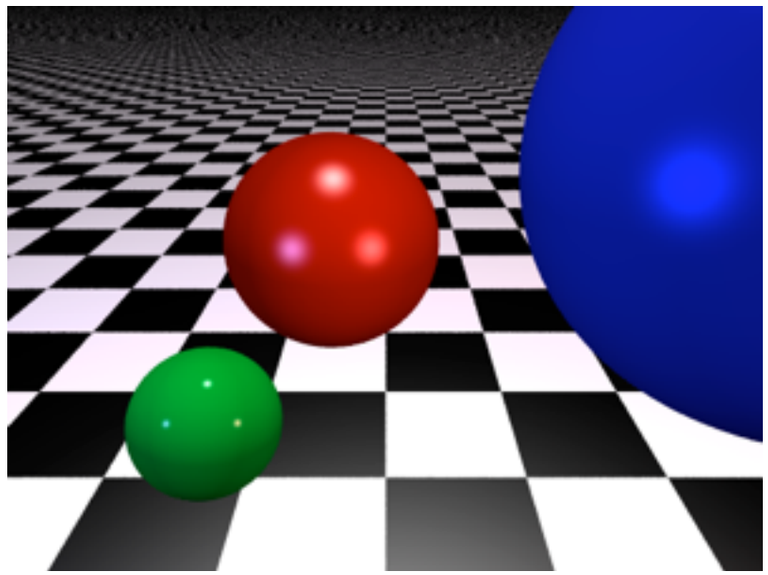
Lighting



Ambient



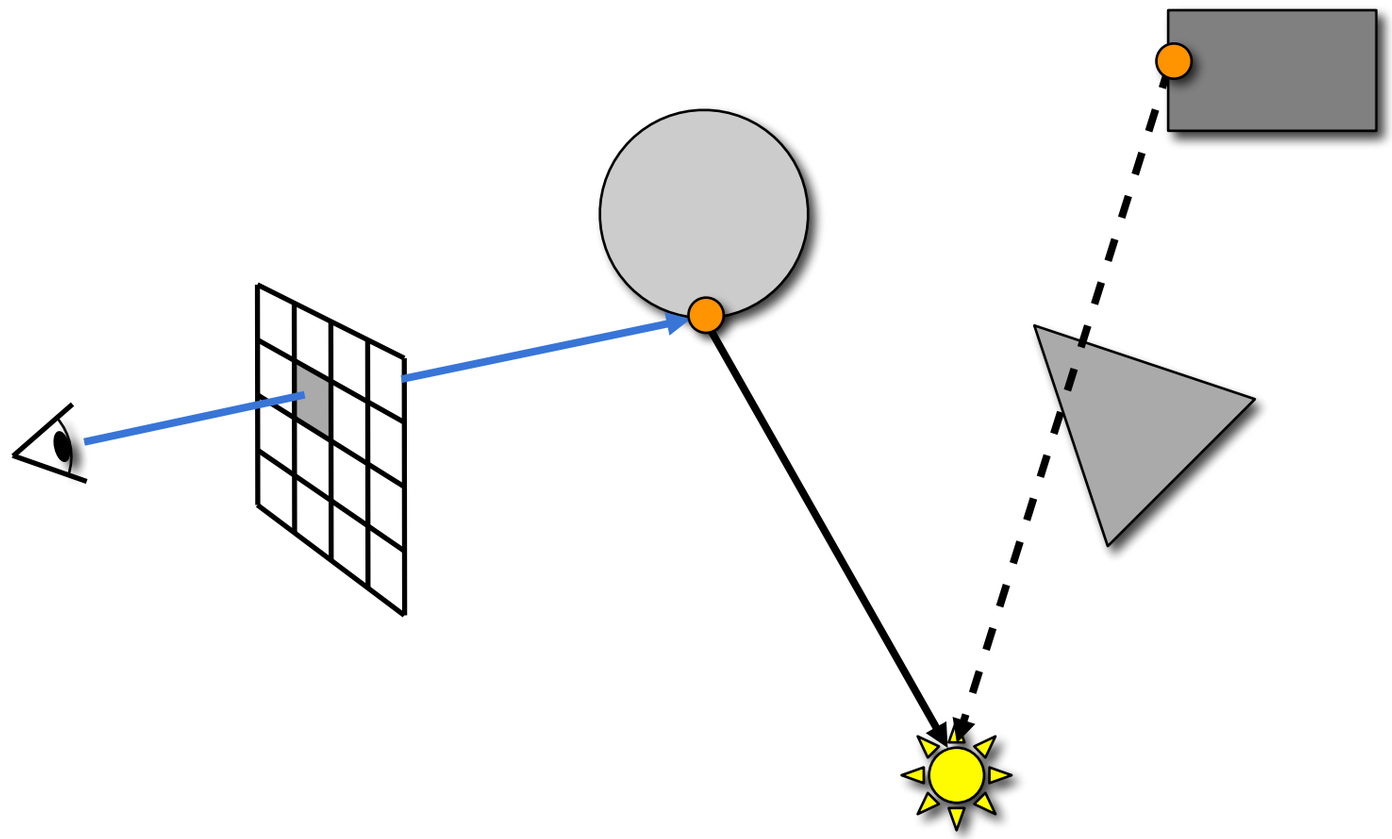
+ Diffuse



+ Specular

Shadows

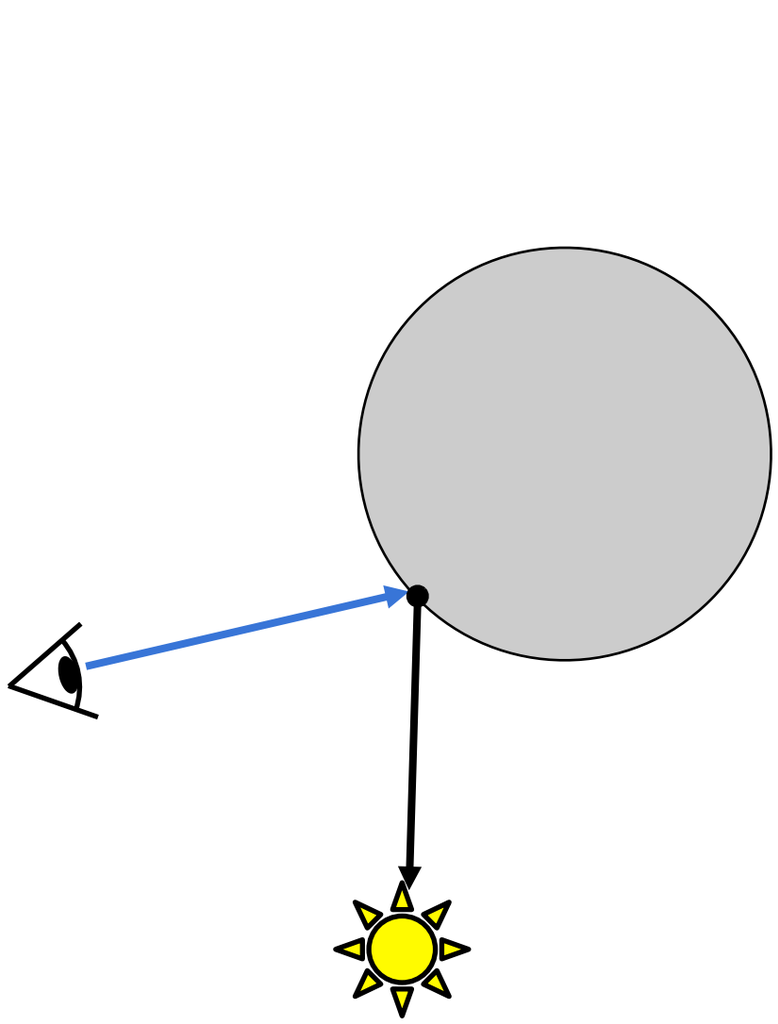
- Send shadow ray from intersection point to light source



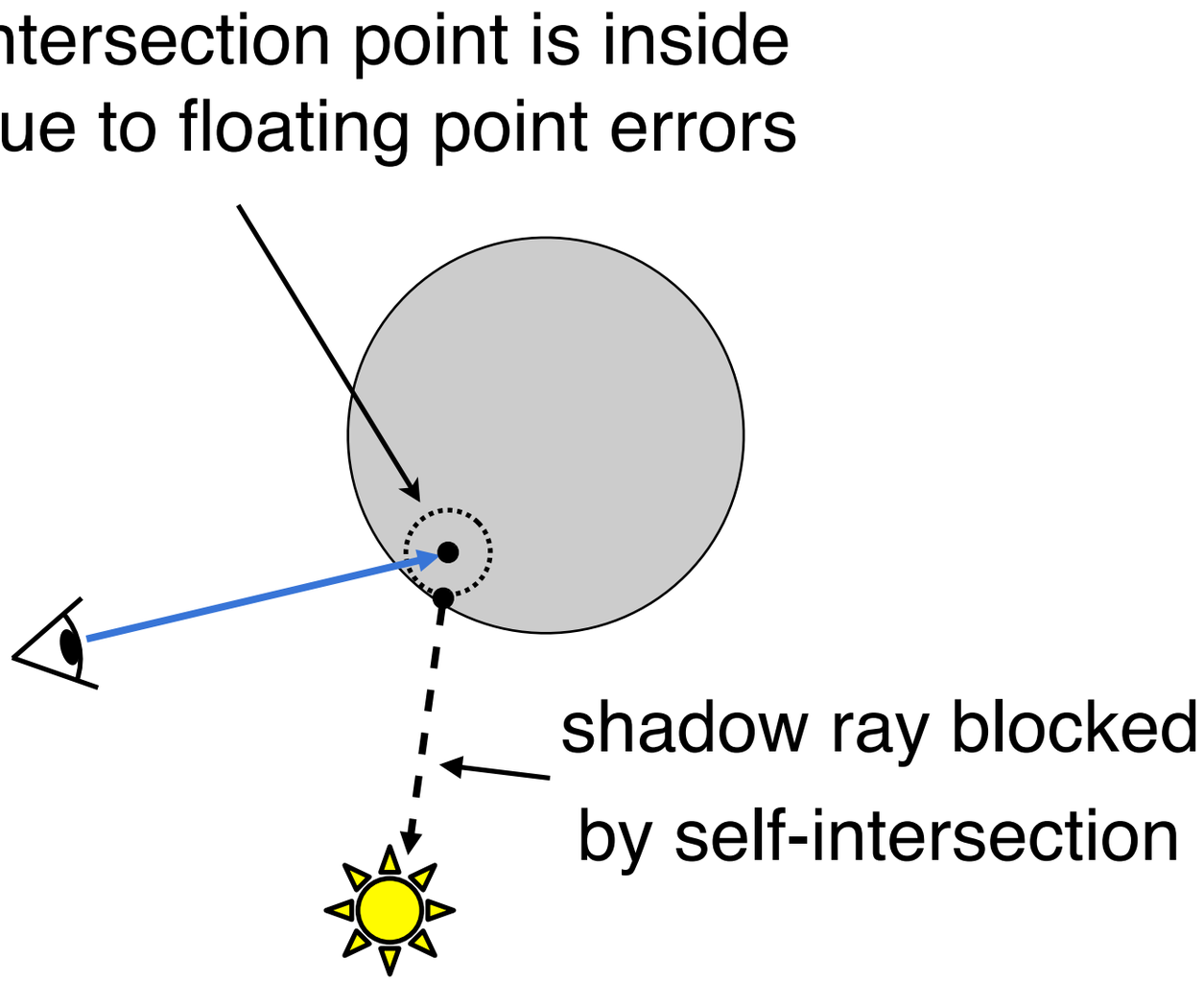
$$I = I_a k_a + I_p (k_d (\mathbf{N} \cdot \mathbf{L}) + k_s (\mathbf{R} \cdot \mathbf{V})^n)$$

Implementation Note

- Numerical precision issues



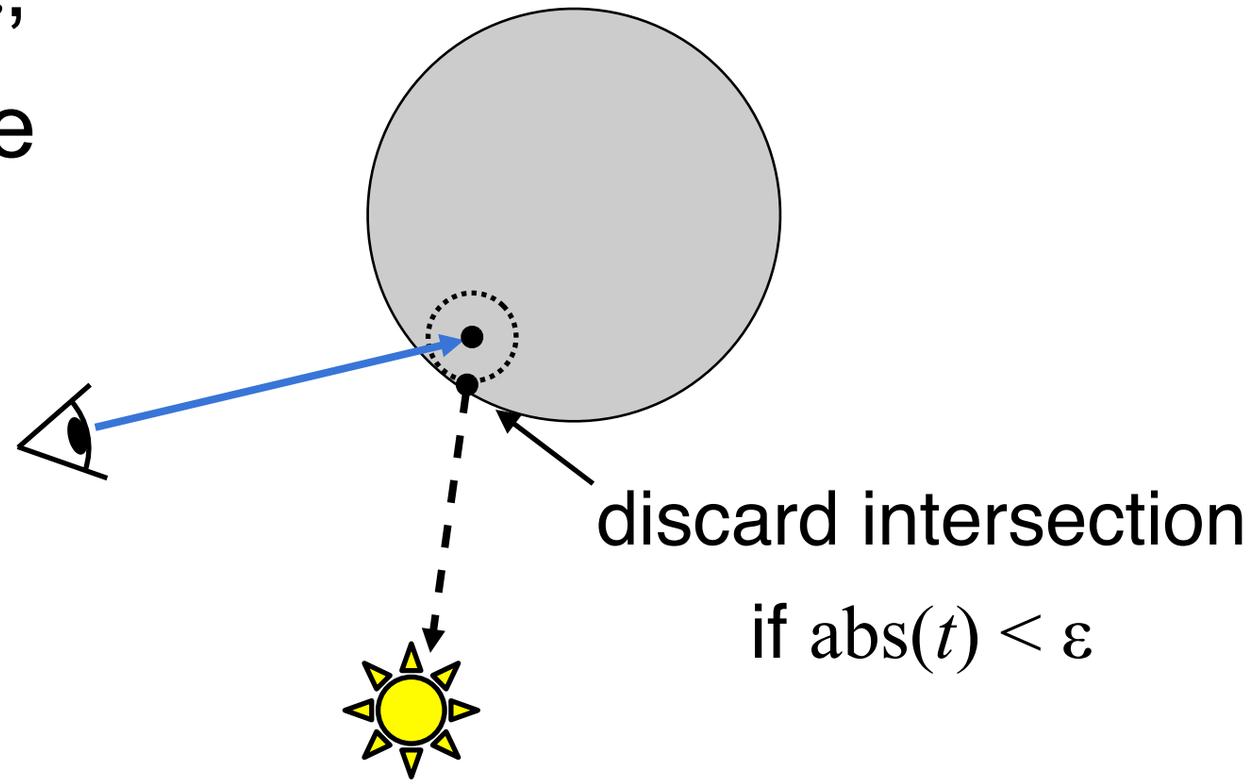
exact computation



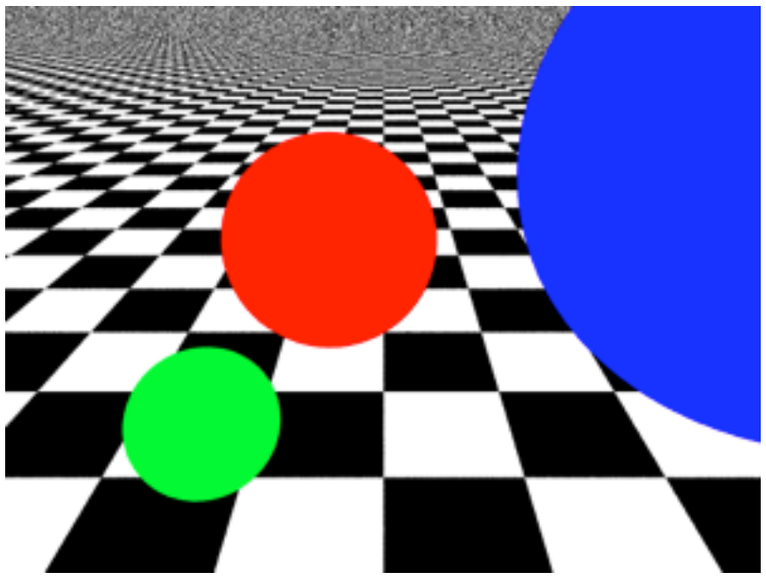
floating point precision

Implementation Note

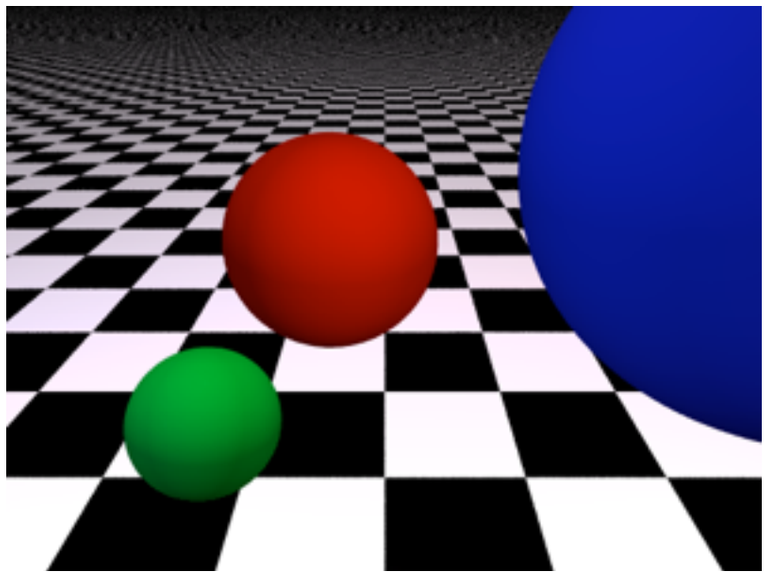
- Numerical precision issues
- Solutions
  - Intersection tolerance ε,
    e.g., 0.1% of scene size
  - Exact arithmetic
    (typically too costly)



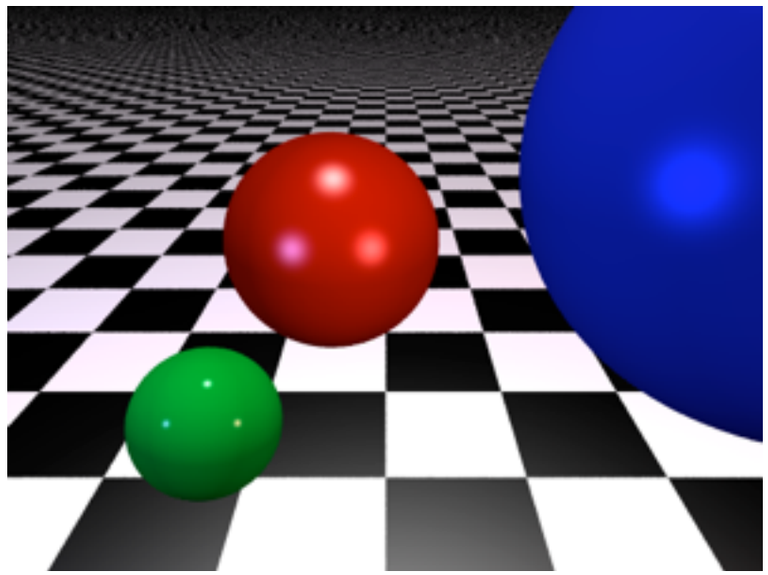
Lighting



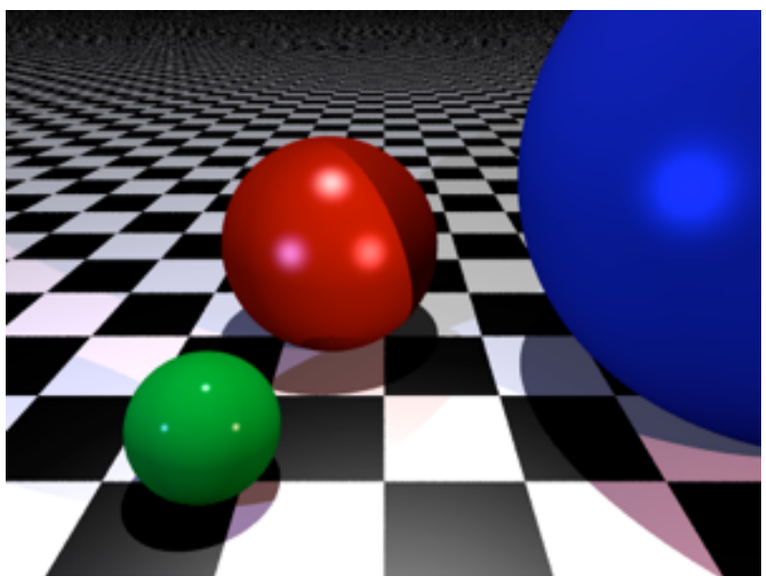
Ambient



+ Diffuse

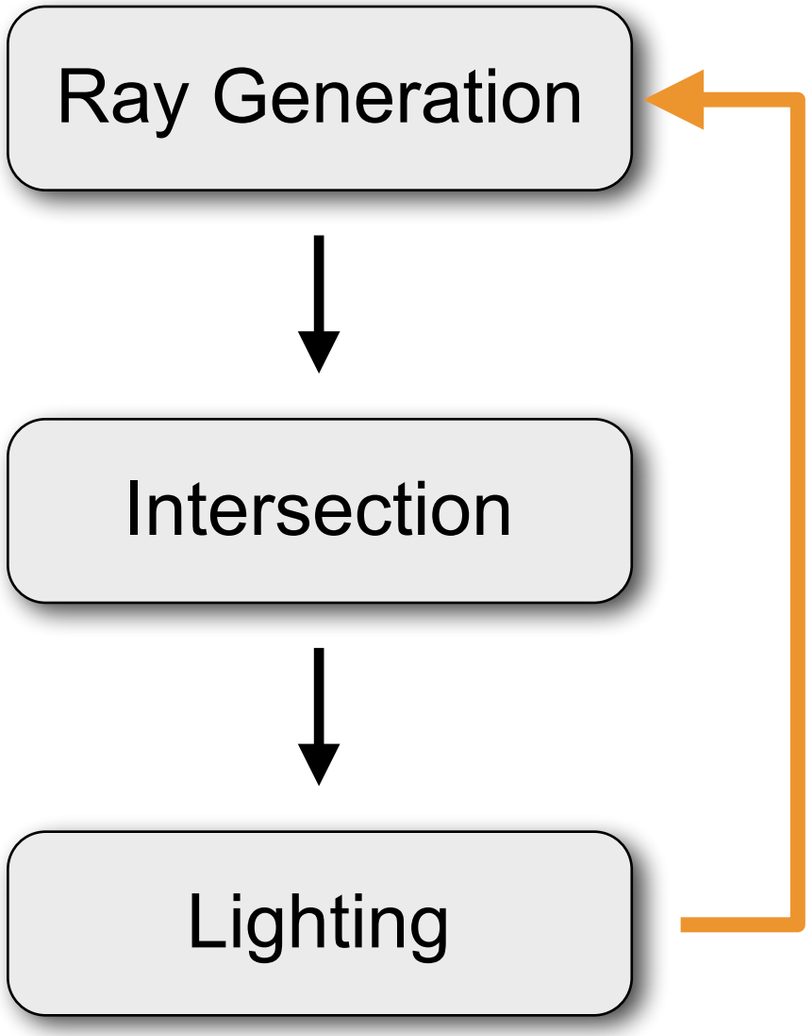


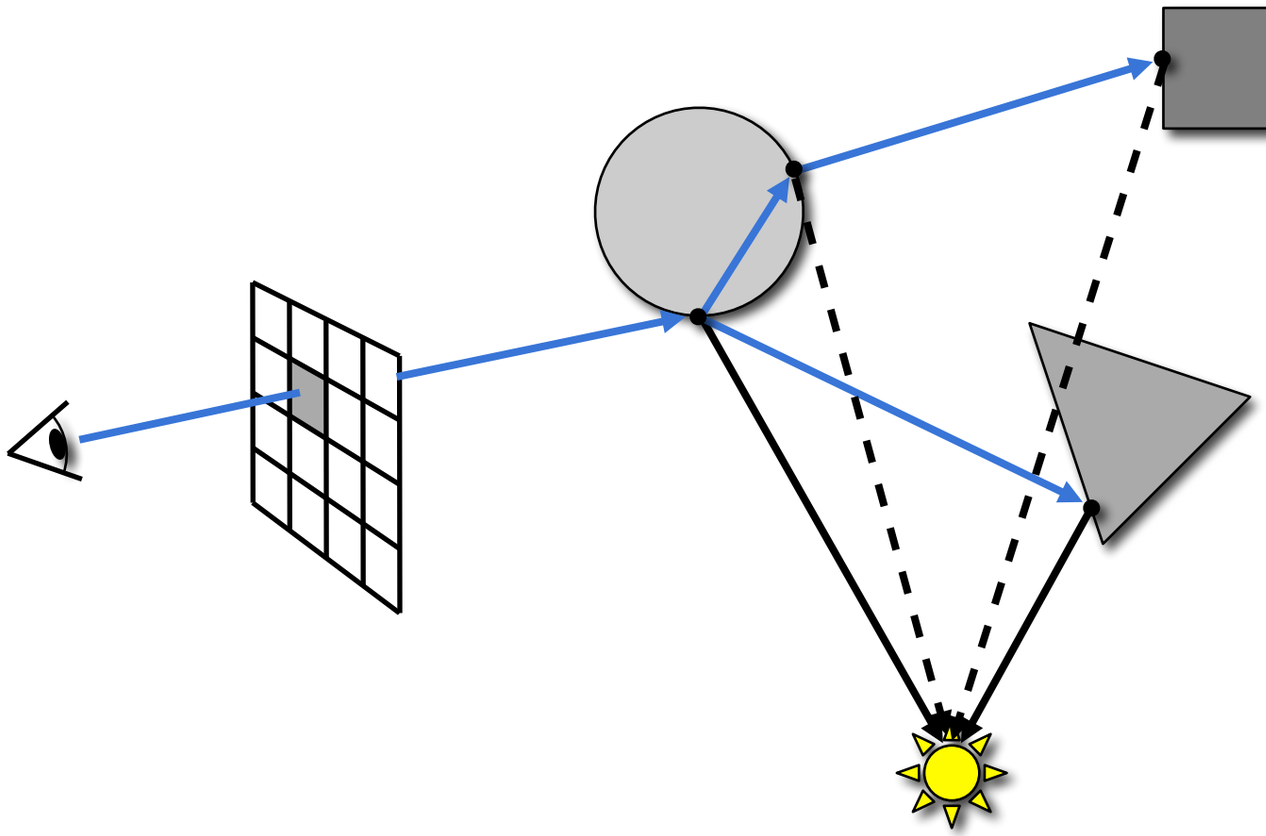
+ Specular



+ Shadows

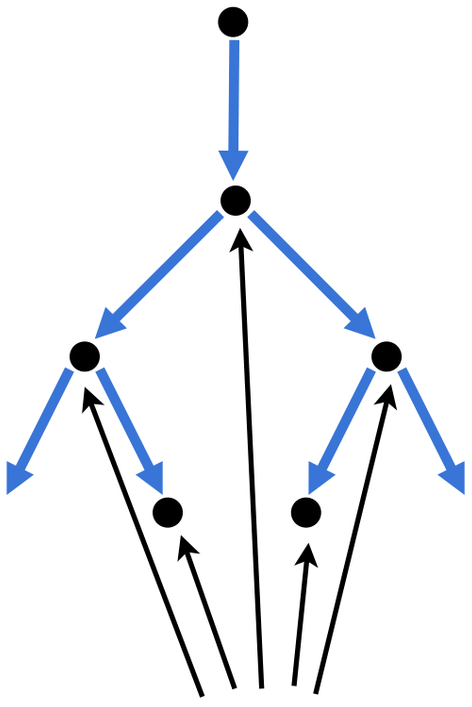
Recursive Ray Tracing



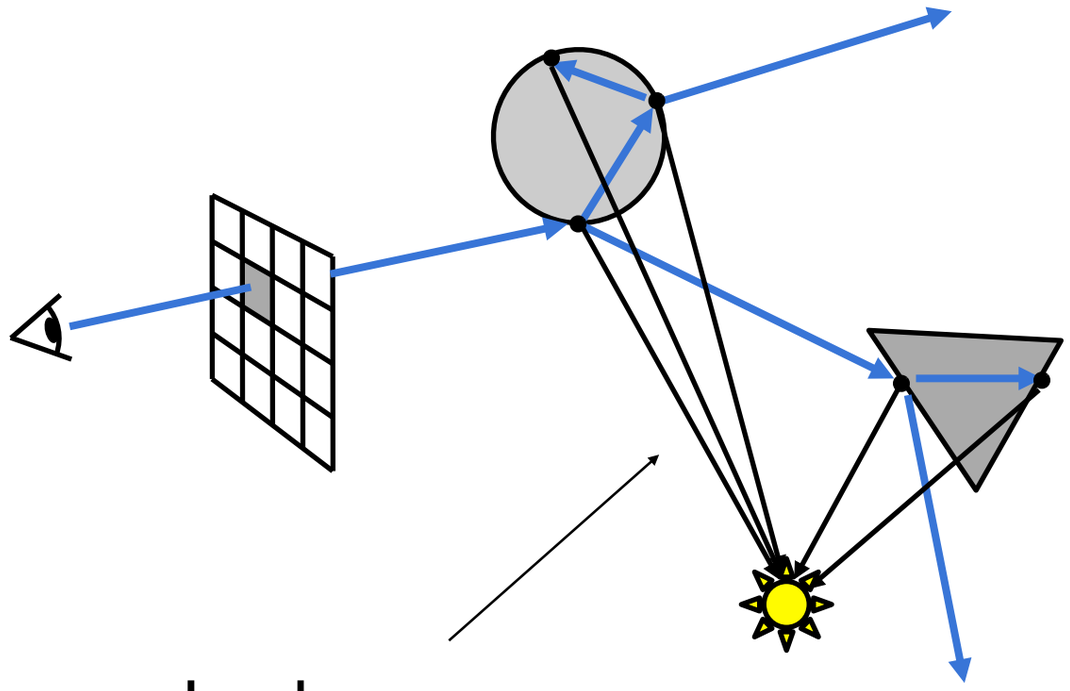


Recursive Ray Tracing

- Recursion tree

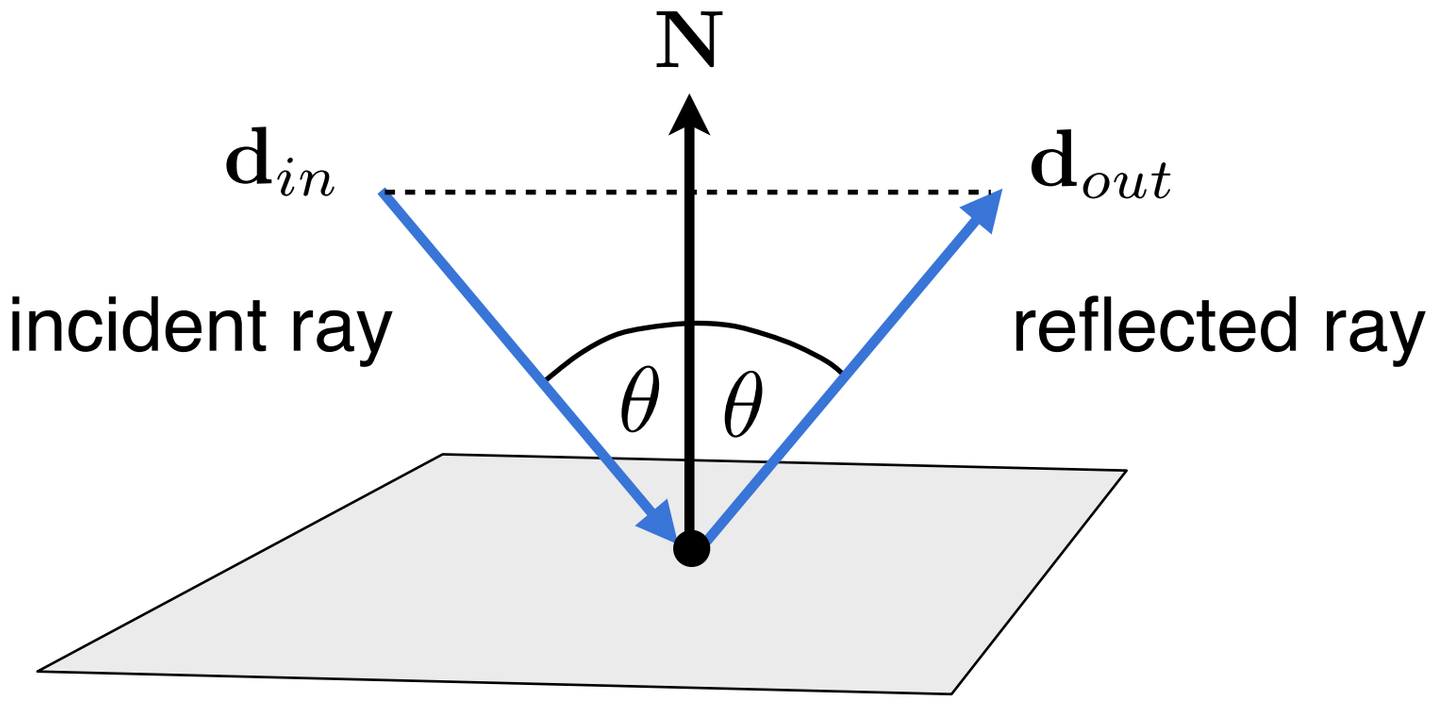


Lighting



shadow rays

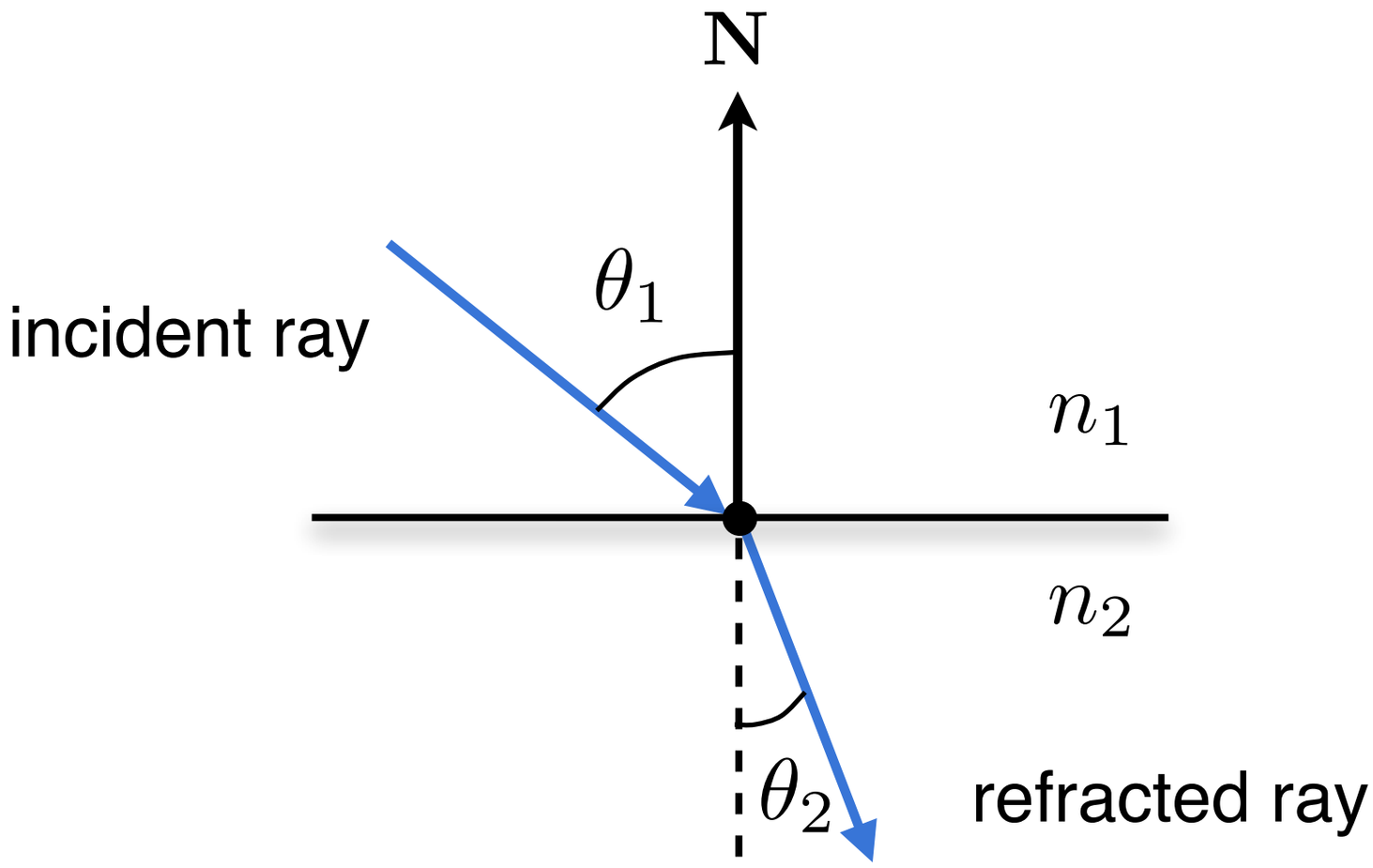
Reflection



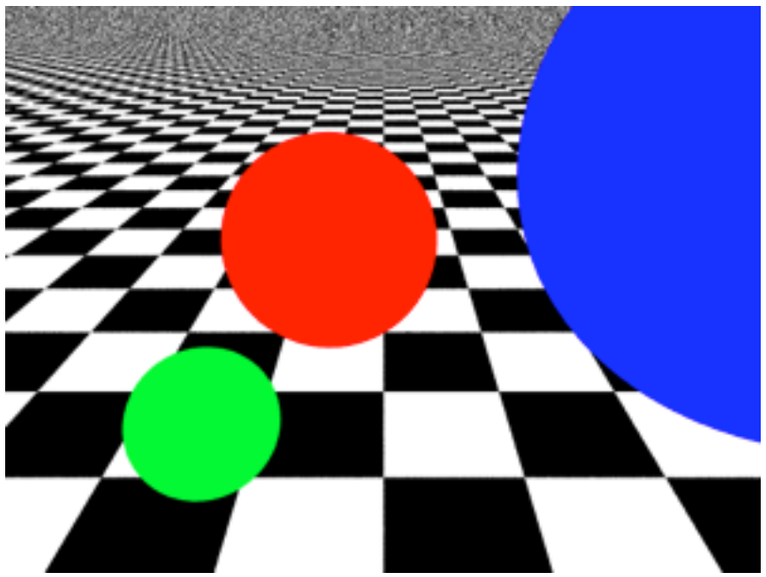
$$\mathbf{d}_{out} = \left(\mathbf{I} - 2\mathbf{N}\mathbf{N}^T\right)\mathbf{d}_{in}$$

Refraction

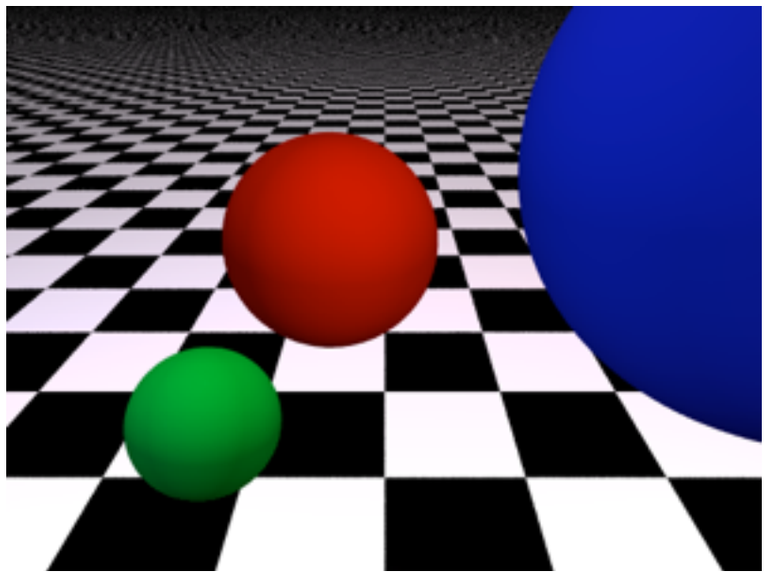
- Snell's law: $n_1 \sin\theta_1 = n_2 \sin\theta_2$



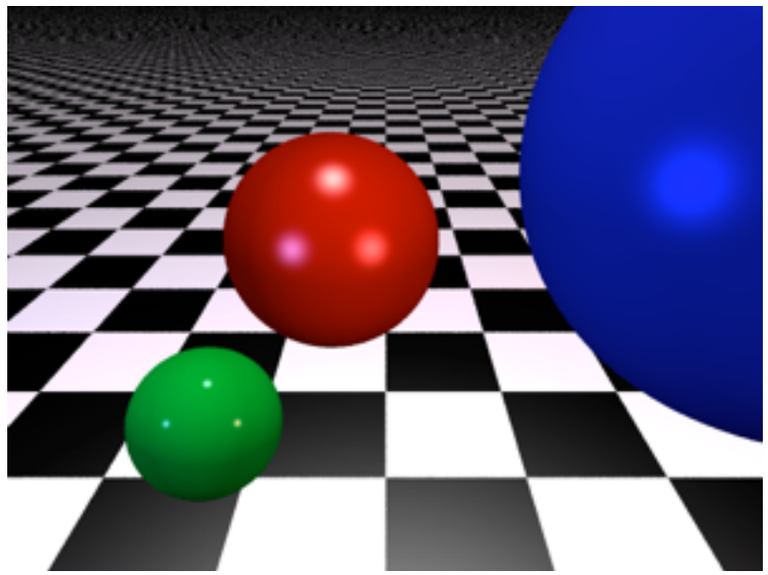
Lighting



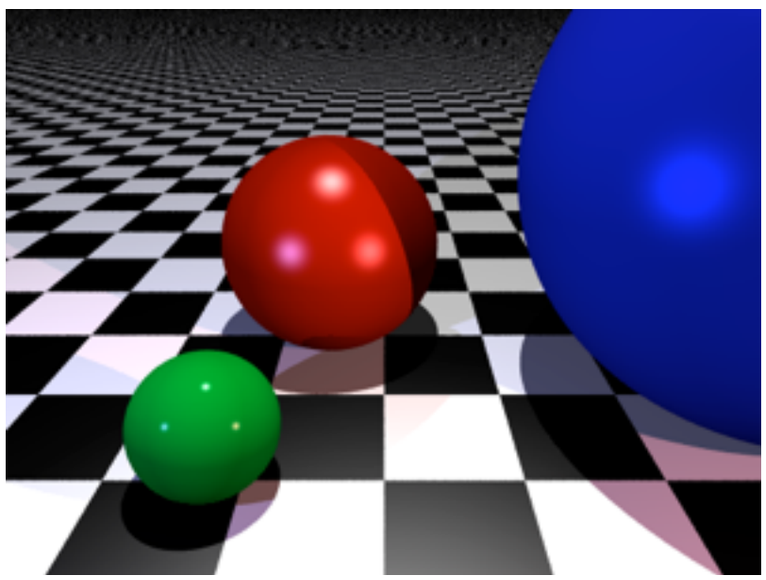
Ambient



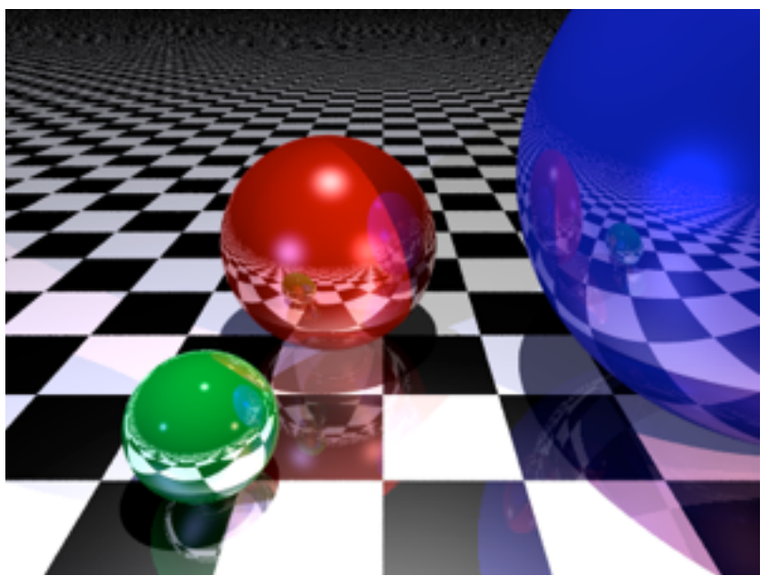
+ Diffuse



+ Specular

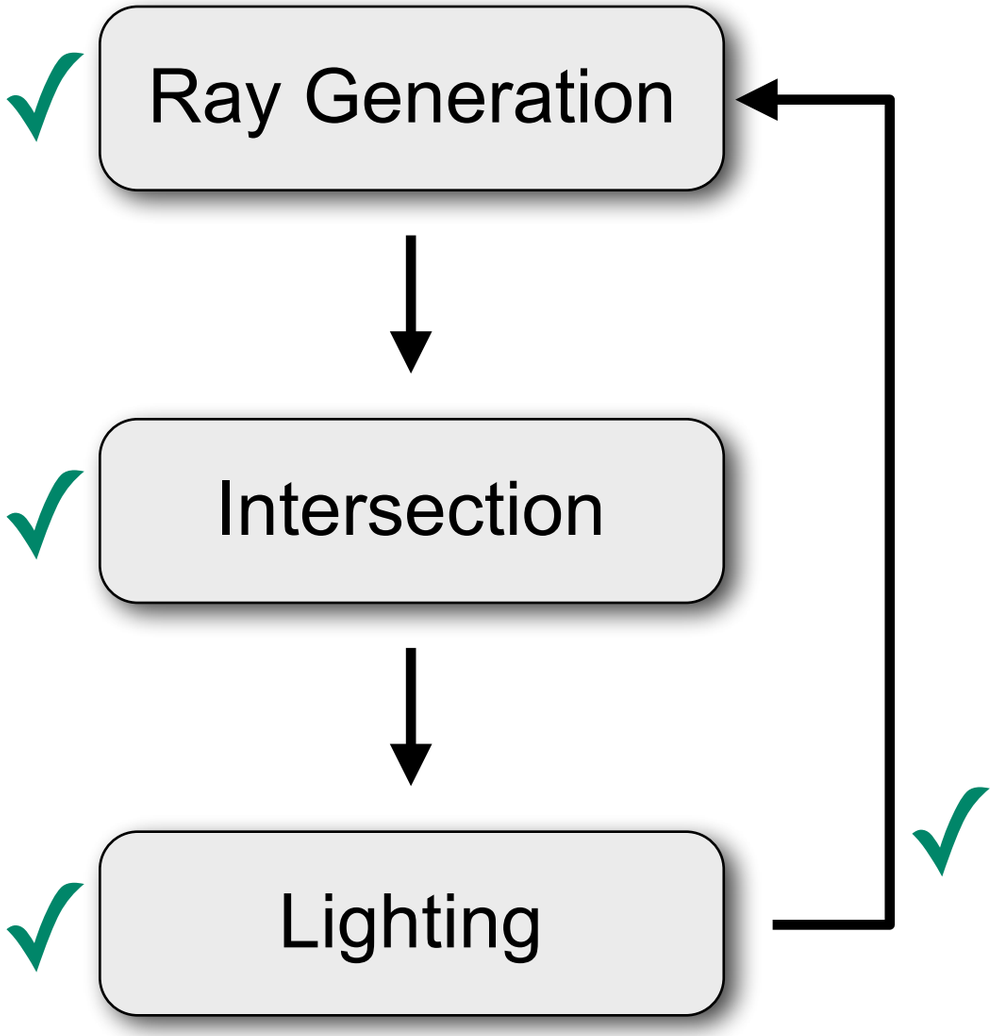


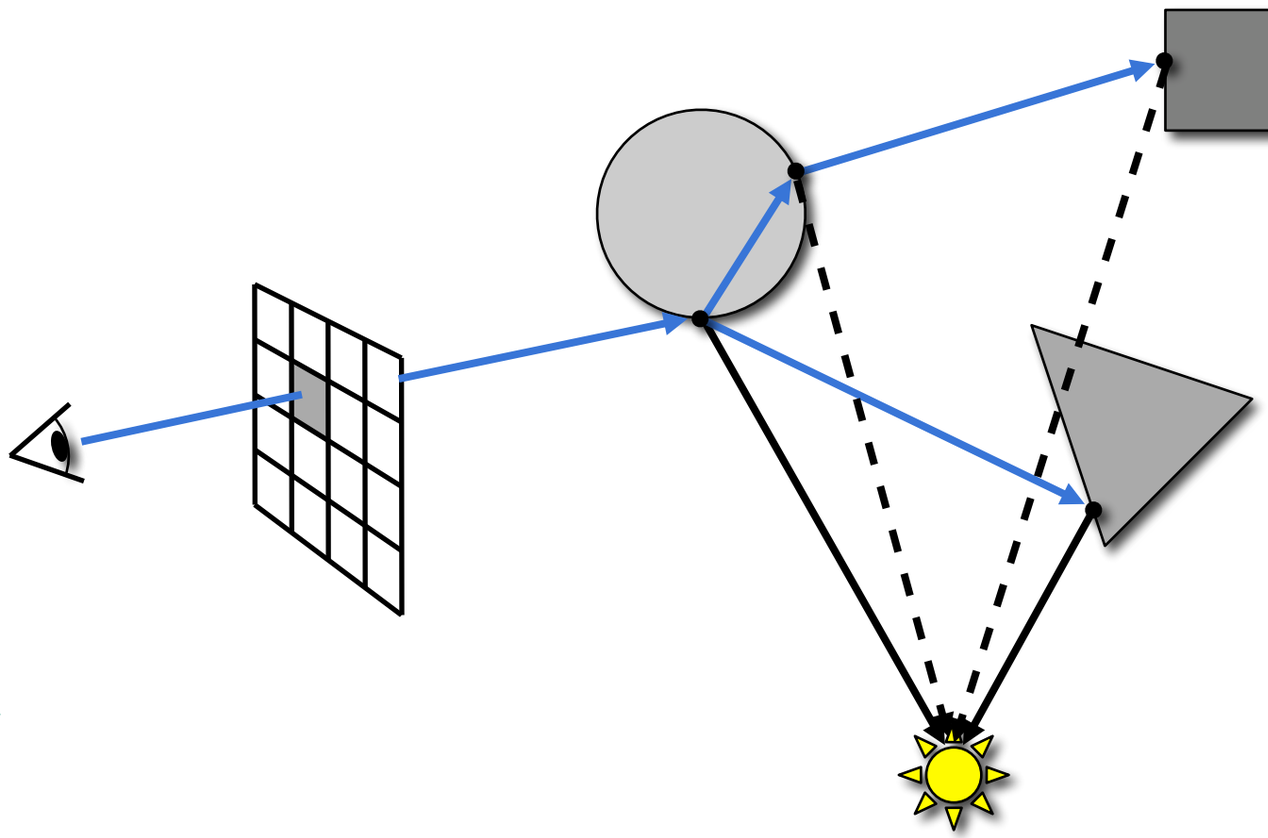
+ Shadows



+ Reflections

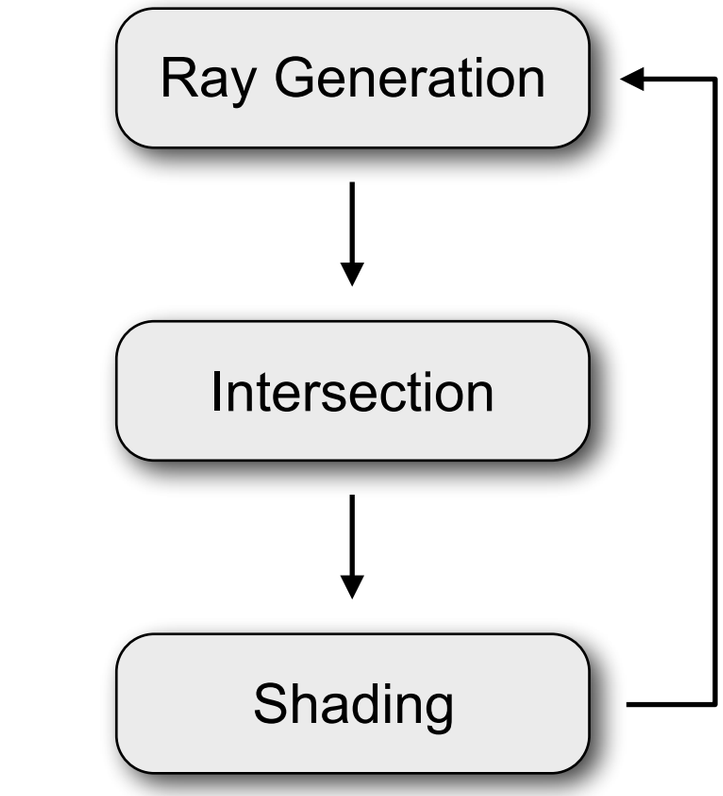
Recursive Ray Tracing

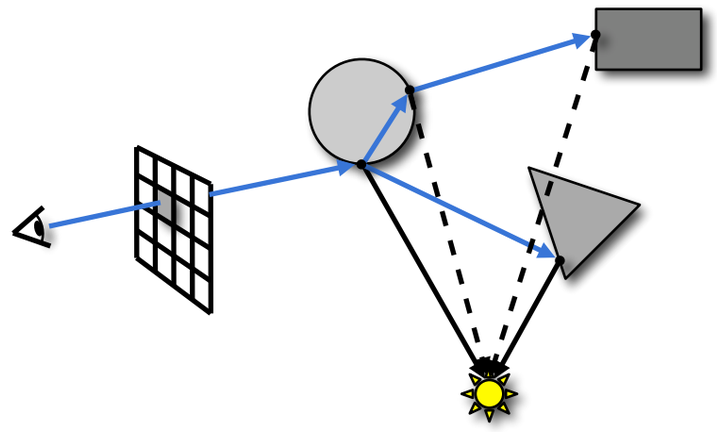




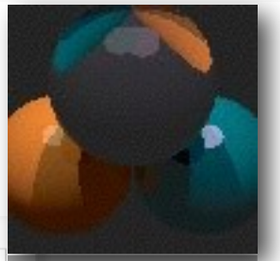
Basic Ray Tracing Pipeline

- You should now be able to build a basic ray tracer!
- More issues to be discussed
  - intersection acceleration, advanced shading, area light sources, soft shadows, etc.





Difficult to code?

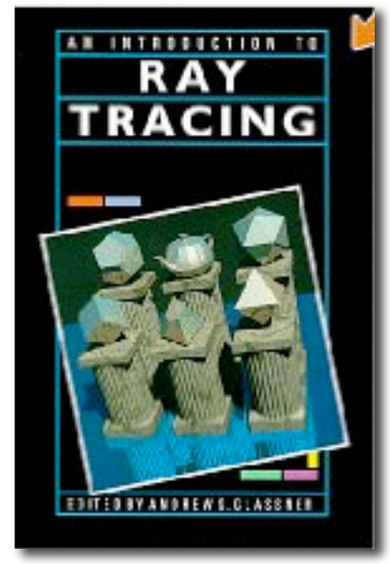


Paul Heckbert's minimal ray tracer

```
typedef struct{double x,y,z;}vec;vec U,black,amb={.1,.1,.1};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("P3\n99 99\n255\n");while(yx<99*99)
U.x=yx%99-99/2,U.z=99/2-yx++/99,U.y=99/2/tan(25/114.5915590261),U=vcomb(255.,
trace(3,black,vunit(U)),black),printf("%.0f %.0f %.0f\n",U.x,U.y,U.z);}/*minray!*/
```

Literature

- Glassner: ***An Introduction to Ray Tracing***, Academic Press, 1989.
  - Chapters 2 & 4



- Pharr, Humphreys: ***Physically Based Rendering***, Morgan Kaufmann, 2004.
  - Chapters 1-3