matfyz

Computer Graphics Charles University

Distributed Ray Tracing

CGG MFF UK Praha

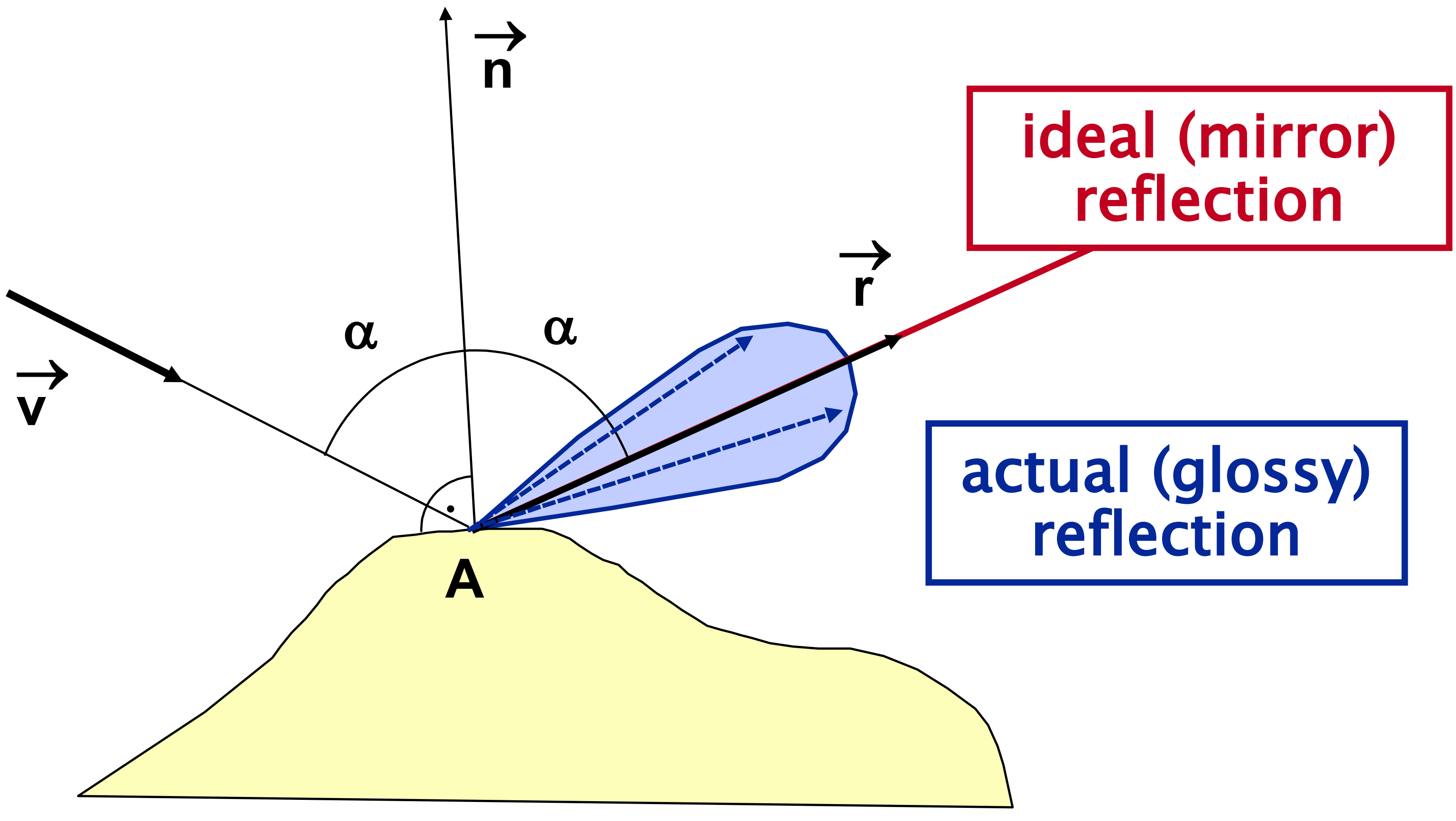
pepca@cgg.mff.cuni.cz

http://cgg.mff.cuni.cz/~pepca/

Distributed ray tracing

- **better image quality** (fidelity)
  - soft shadows, glossy reflections, soft refractions
  - motion blur
  - depth of field imitation
  - light dispersion (index of refraction depends on λ)
- introducing **new variables** to an image function
  - reflection or refraction angle, wavelength, light source point, lens entry point, time, ..

Glossy reflection



Glossy reflection computation

Sharp reflection:

$$I(V) = I(R)$$

(one reflected ray)

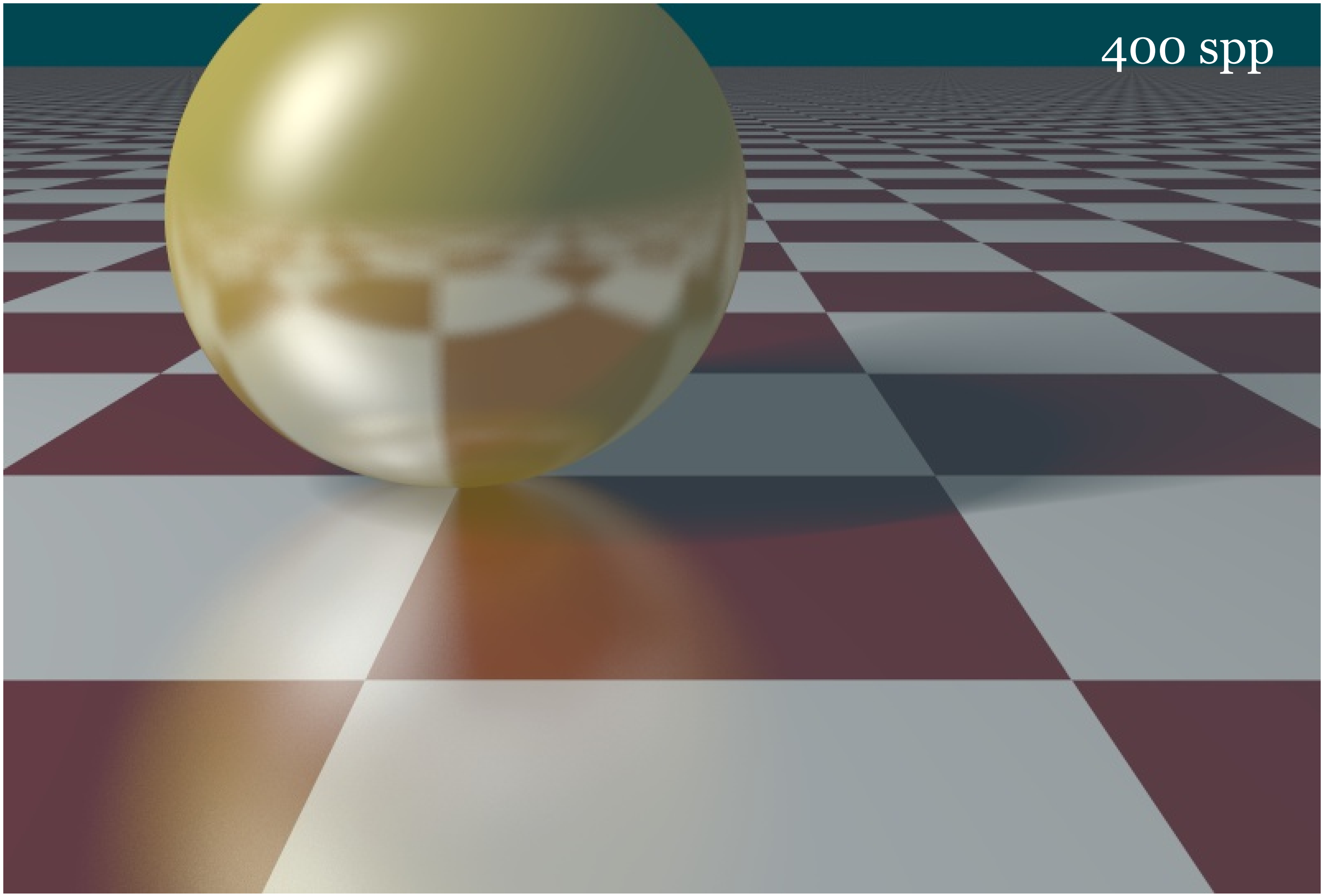
Glossy reflection:

$$I(V) = \iint_{sphere} I(R(\varphi,\theta)) \cdot \underline{BRDF(\alpha,\beta,\varphi,\theta)}\, d\varphi\, d\theta$$

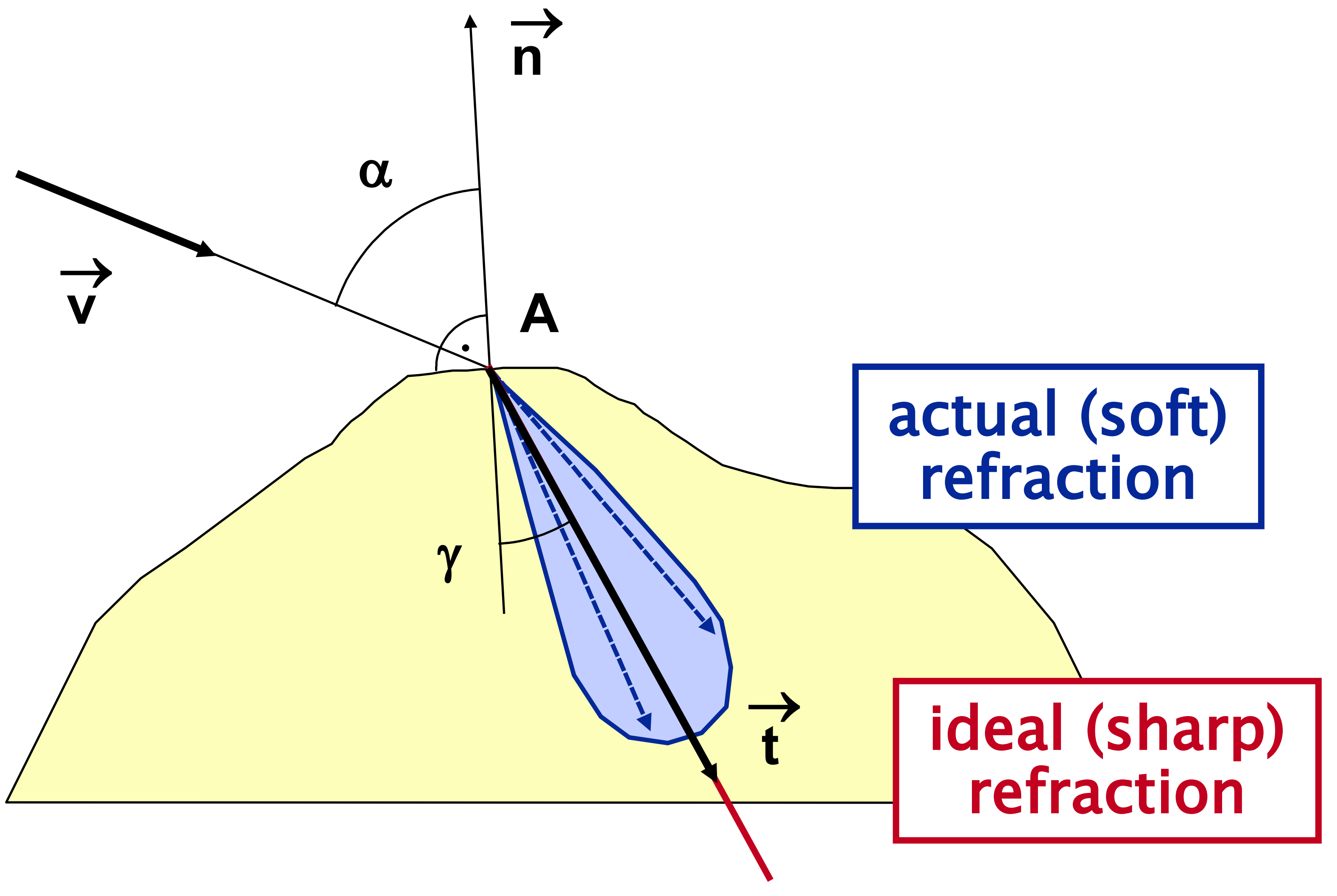
reflectance function

(weighted integral average .. over all reflection angles)

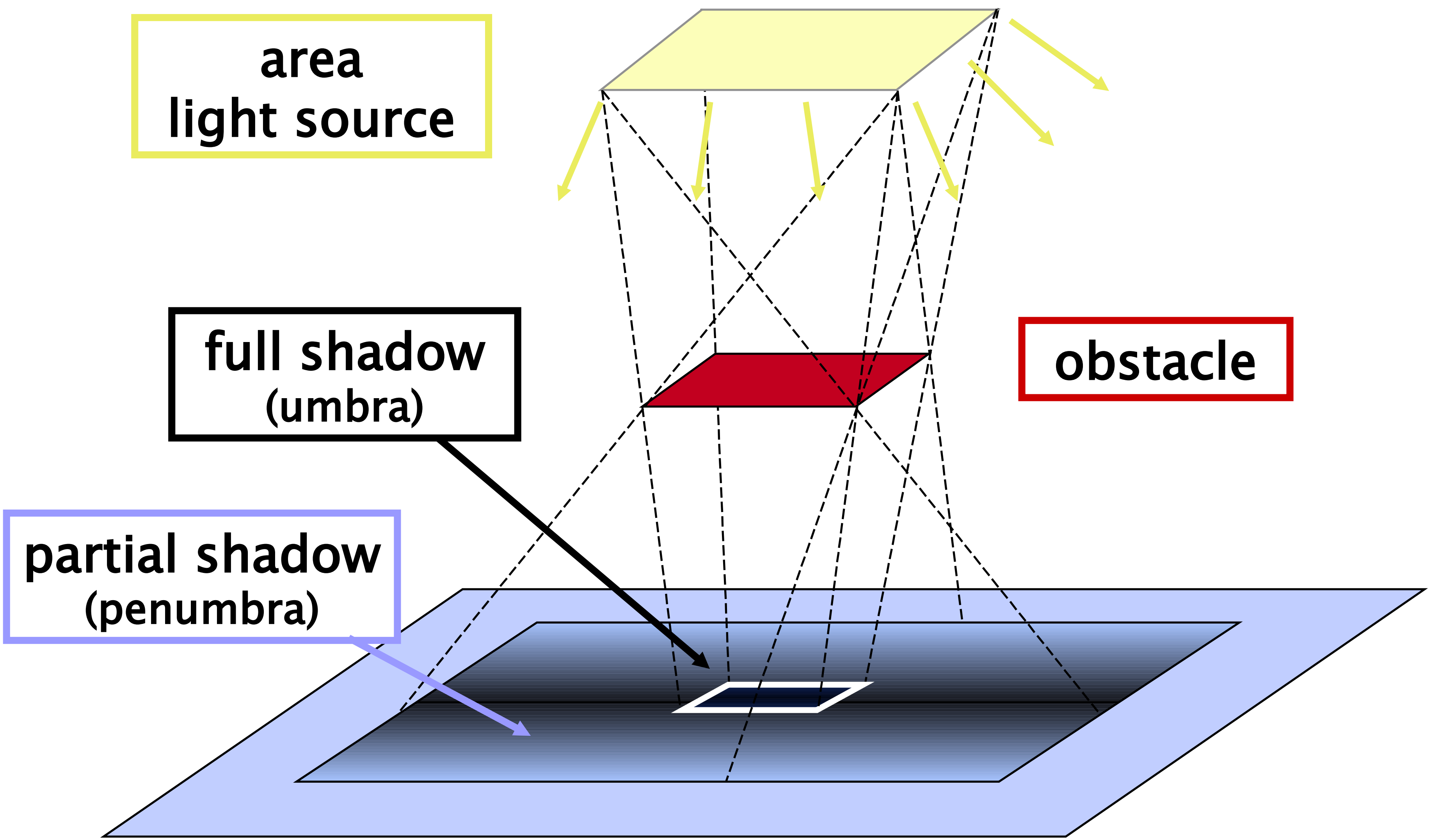
Glossy reflection – example



Soft refraction



Soft shadow


area
light source
full shadow
(umbra)
obstacle
partial shadow
(penumbra)

Soft shadow computation

Contribution of a point light source:

$$I(A) = \begin{cases} I_L & \text{if source is visible from } \mathbf{A} \\ 0 & \text{else} \end{cases}$$

Contribution of an area light source:

$$I(A) = I_L \cdot S[\%]$$

visible portion of a light source

Soft shadow computation

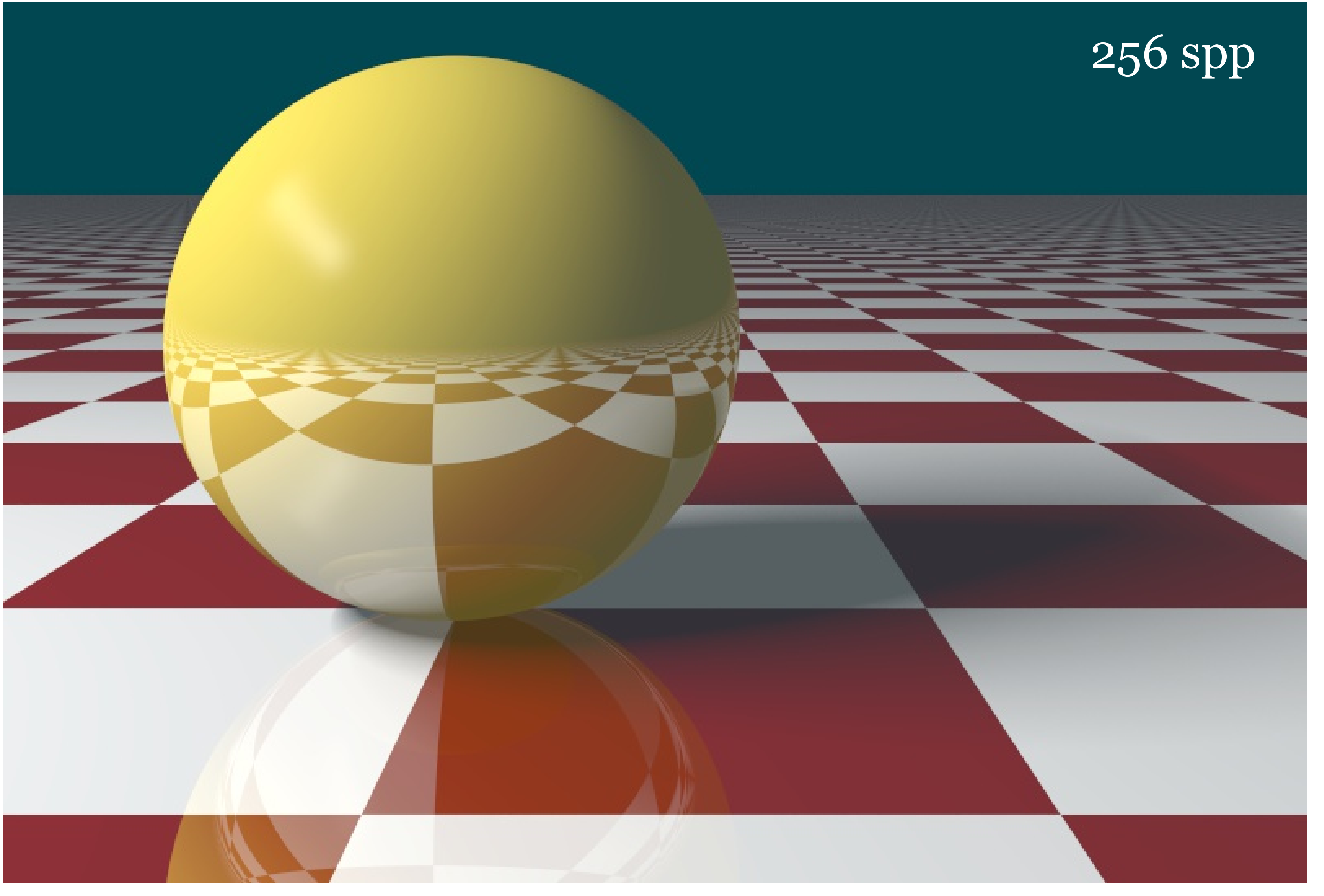
Contribution of an inhomogenous light source:

$$\mathbf{I}(\mathbf{A}) = \iint_{\text{light source area}} \mathbf{I_L}(\mathbf{u},\mathbf{v}) \cdot \mathbf{vis}(\mathbf{A},\mathbf{u},\mathbf{v})\ \mathbf{du}\,\mathbf{dv}$$

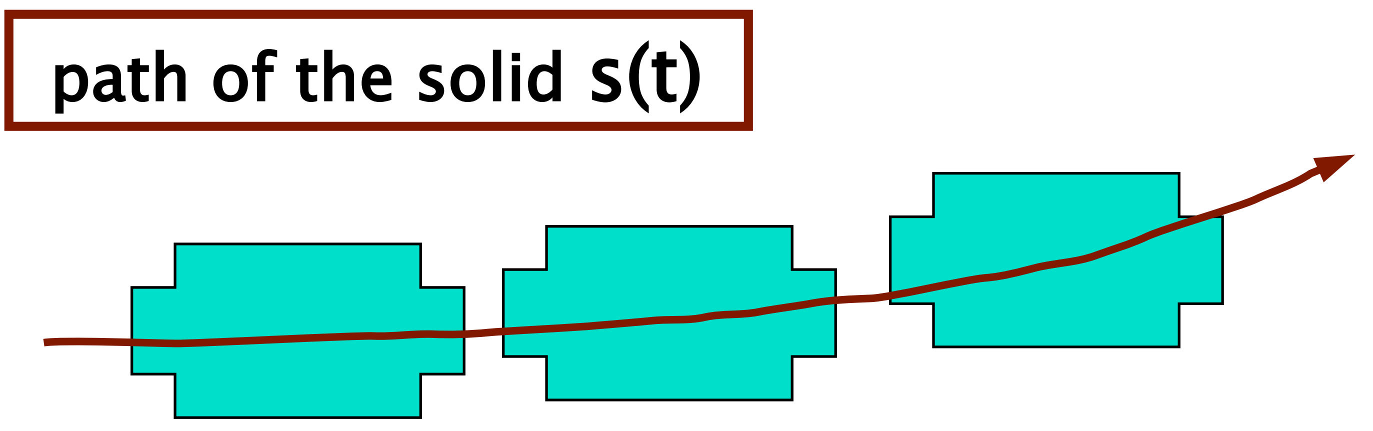
visibility function

$$\mathbf{vis}(\mathbf{A},\mathbf{u},\mathbf{v}) = \begin{cases} \mathbf{1} & \text{if } \mathbf{S(u,v)} \text{ is visible from } \mathbf{A} \\ \mathbf{0} & \text{else} \end{cases}$$

Soft shadow – example



Motion blur



rendered interval:
(shutter open time)

$[\ t_1,\ t_2\ ]$

scene rendering in time t:

$f(t) = f(x,y,t)$

Motion blur

General motion blur:

$$f_{blurr} = \int_{t_1}^{t_2} f(t)\, dt$$

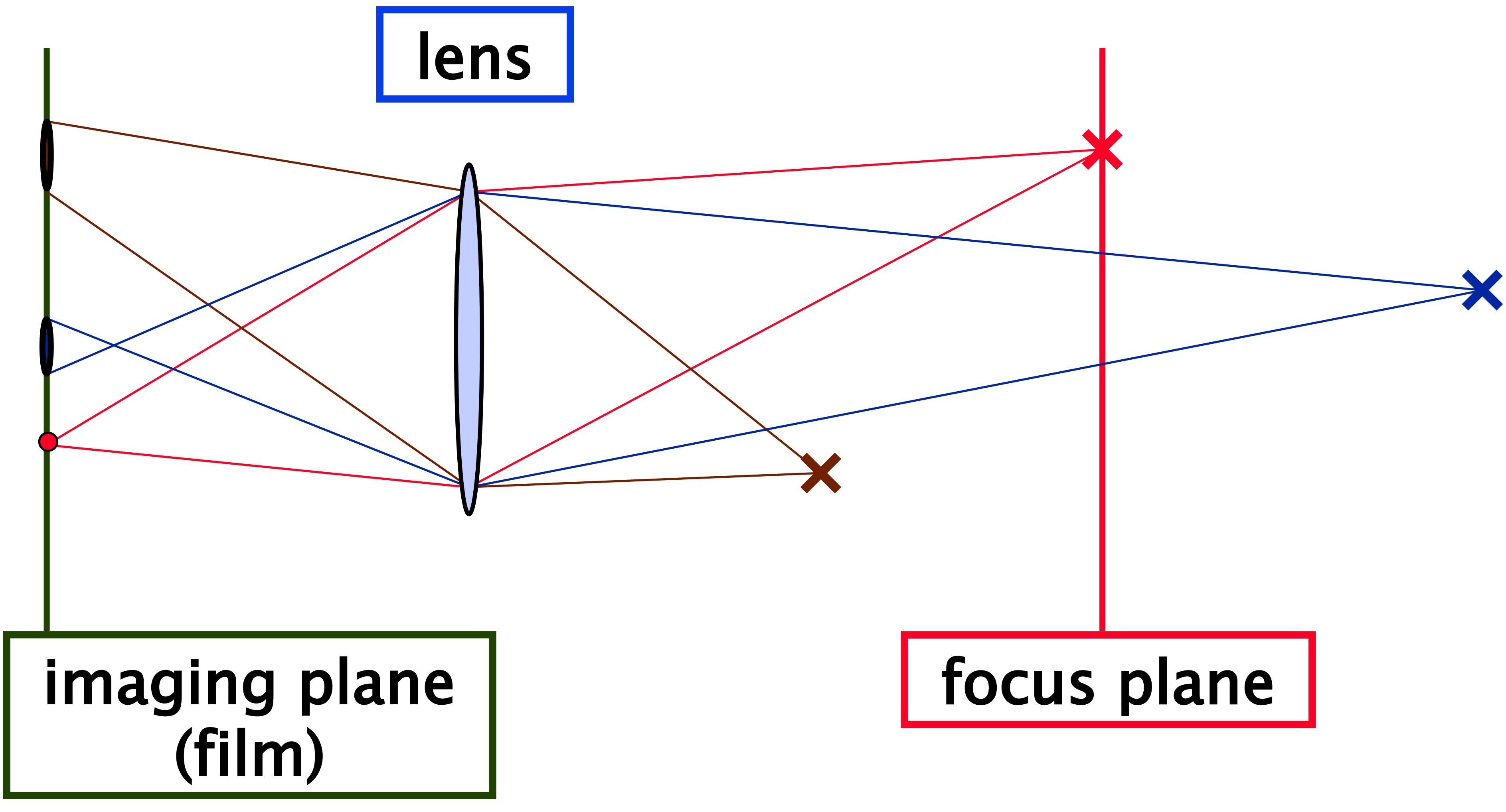
Motion blur

Scene with single moving object:

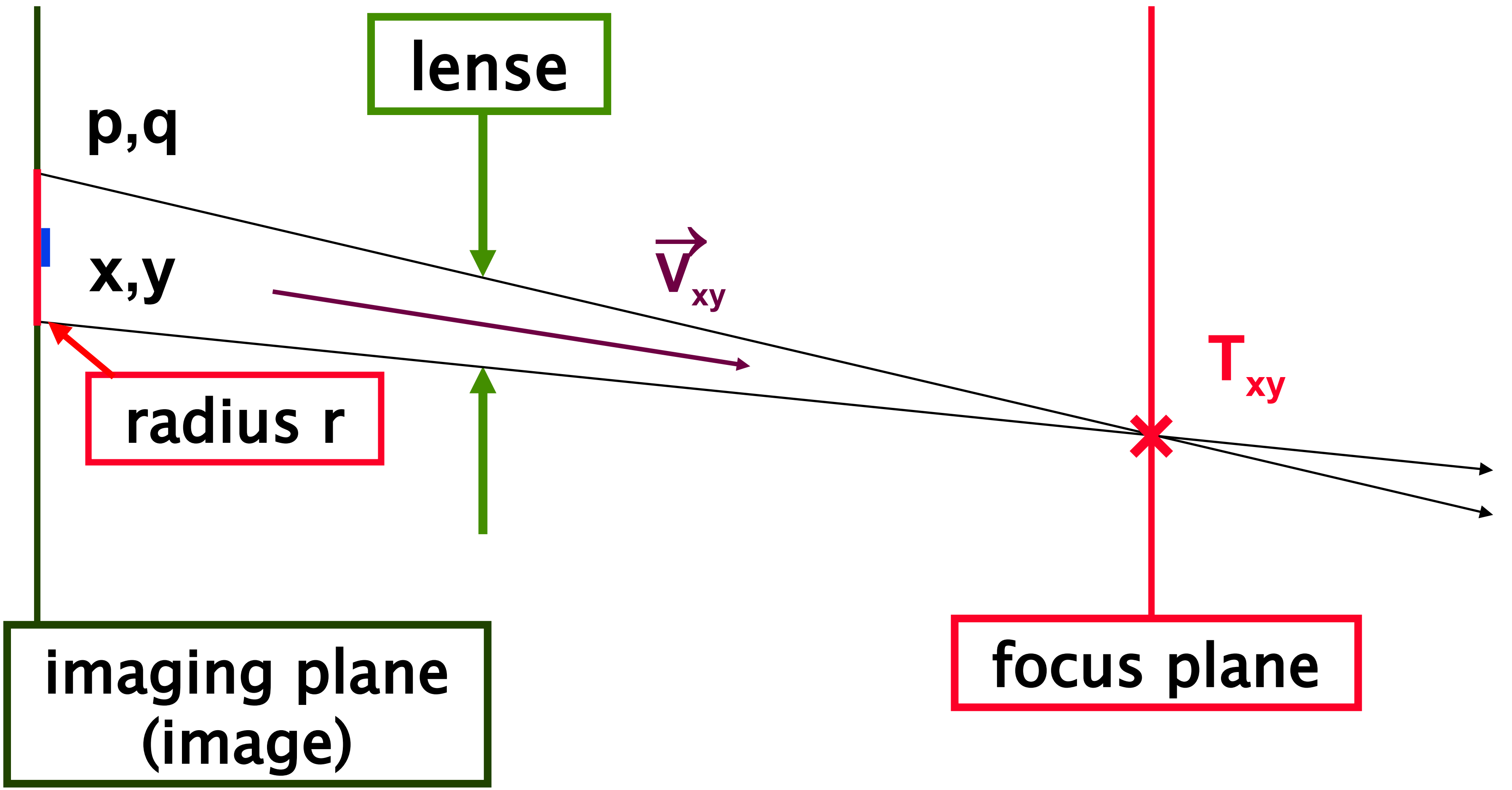
$$f_{blurr} = \frac{\int_{t_1}^{t_2} f(t) \cdot \left| s'(t) \right|^{-1} dt}{\int_{t_1}^{t_2} \left| s'(t) \right|^{-1} dt}$$

$$\left| s'(t) \right| \neq 0 \text{ on } \langle t_1, t_2 \rangle$$

Depth of field



Geometric simplification



Depth of field computation

Pinhole camera model:

$$f(x, y) = I(V_{xy})$$

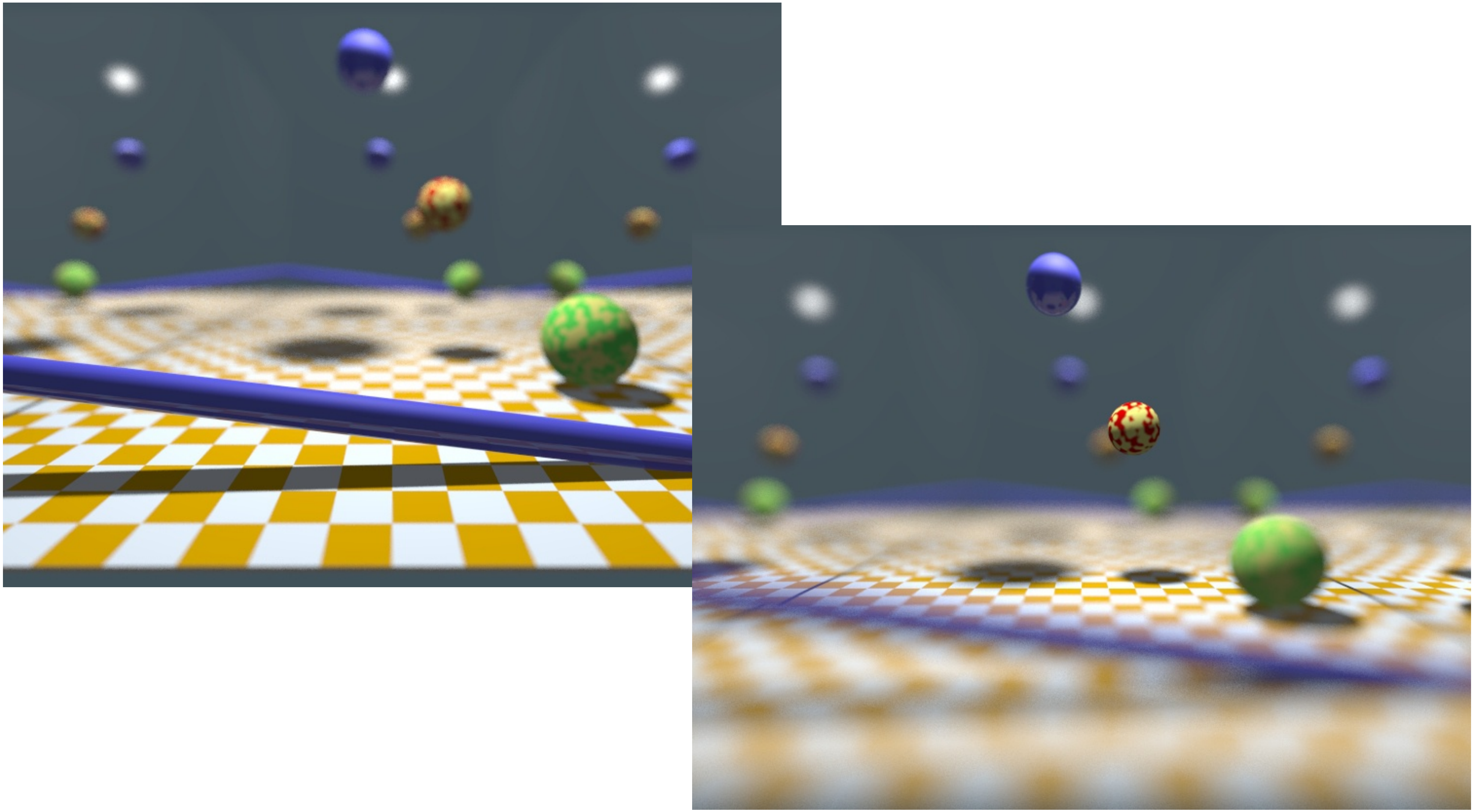
$$V_{xy} = T_{xy} - [x, y, 0]$$

$$V_{pq} = T_{xy} - [p, q, 0]$$

Lens with finite aperture:

$$f(x, y) = \int_{\text{circle around } [x, y]} I(V_{pq}) \, dp \, dq$$

Depth of field – examples



Light dispersion

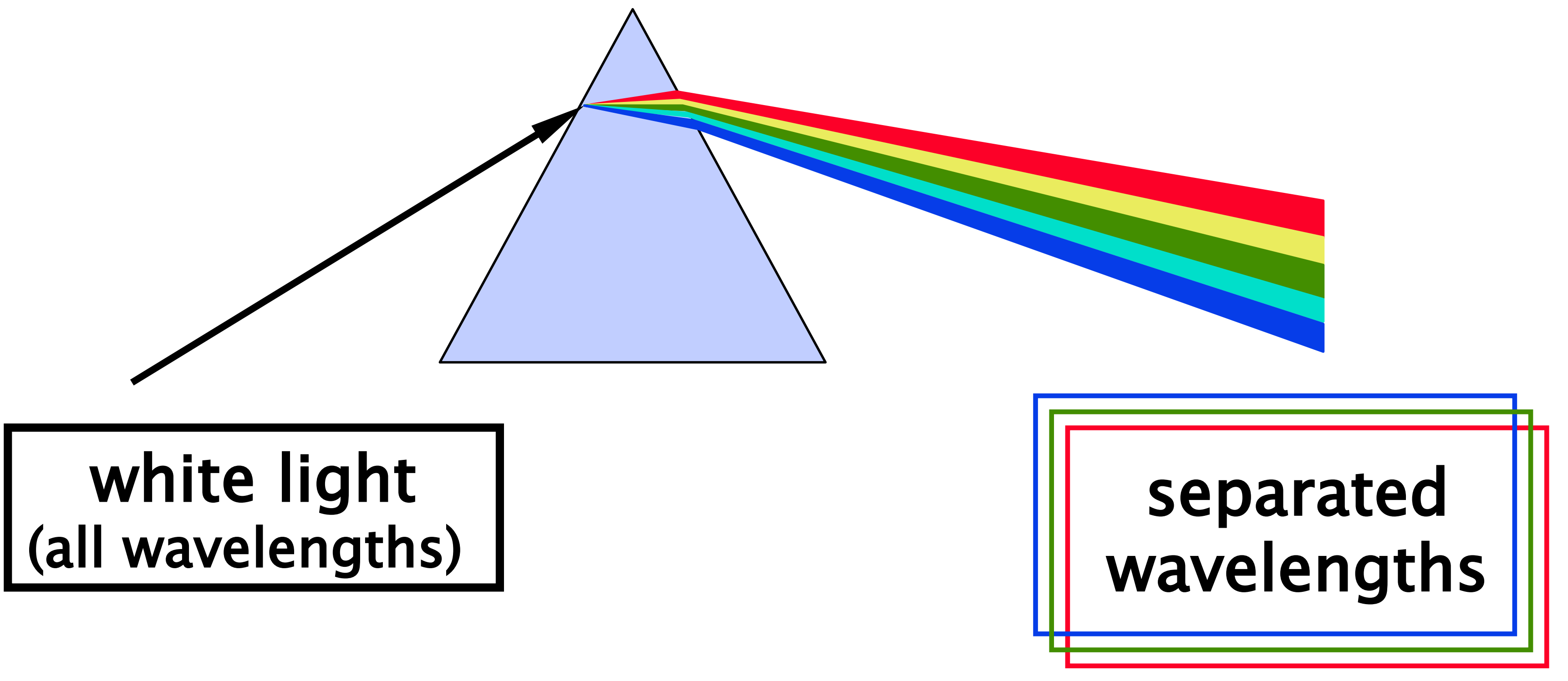


image function: $f(\lambda) = f(x,y,\lambda)$

Light dispersion computation

Pixel RGB color from spectral dristribution:

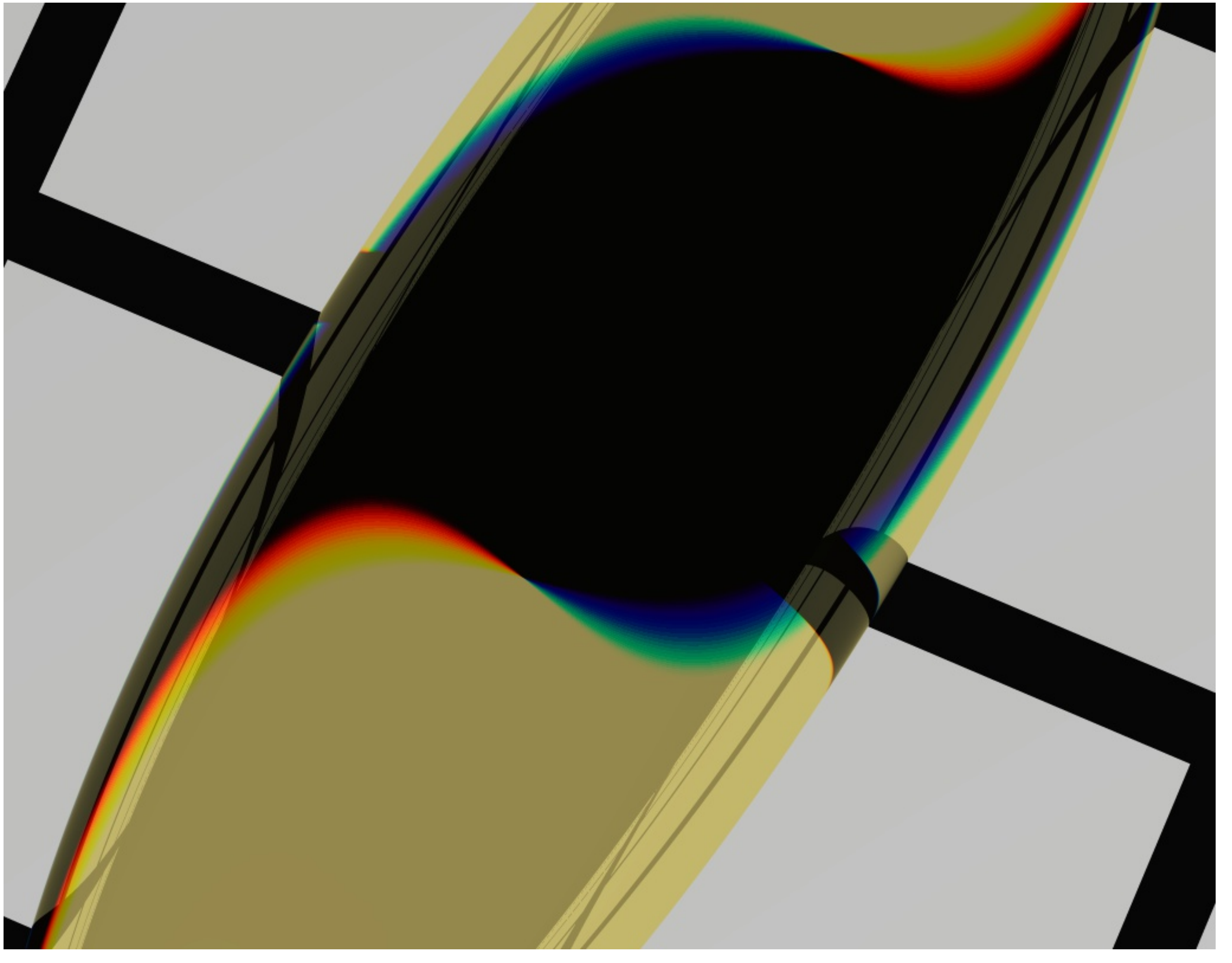
$$R(x,y) = \int_{\text{spectrum}} f(x,y,\lambda) \cdot R(\lambda)\, d\lambda$$

$$G(x,y) = \int_{\text{spectrum}} f(x,y,\lambda) \cdot G(\lambda)\, d\lambda$$

$$B(x,y) = \int_{\text{spectrum}} f(x,y,\lambda) \cdot B(\lambda)\, d\lambda$$

trichromatic spectral coefficients

Light dispersion – example



Implementation

- integral averaging is done **stochastically** (Monte-Carlo methods)
  – finite number of point samples (rays)
  – integral is estimated by a [weighted] sum
- **weighted** integral average
  – uniform sampling and appropriate weight function
  – nonuniform sampling (using the right density/PDF)

Combining methods

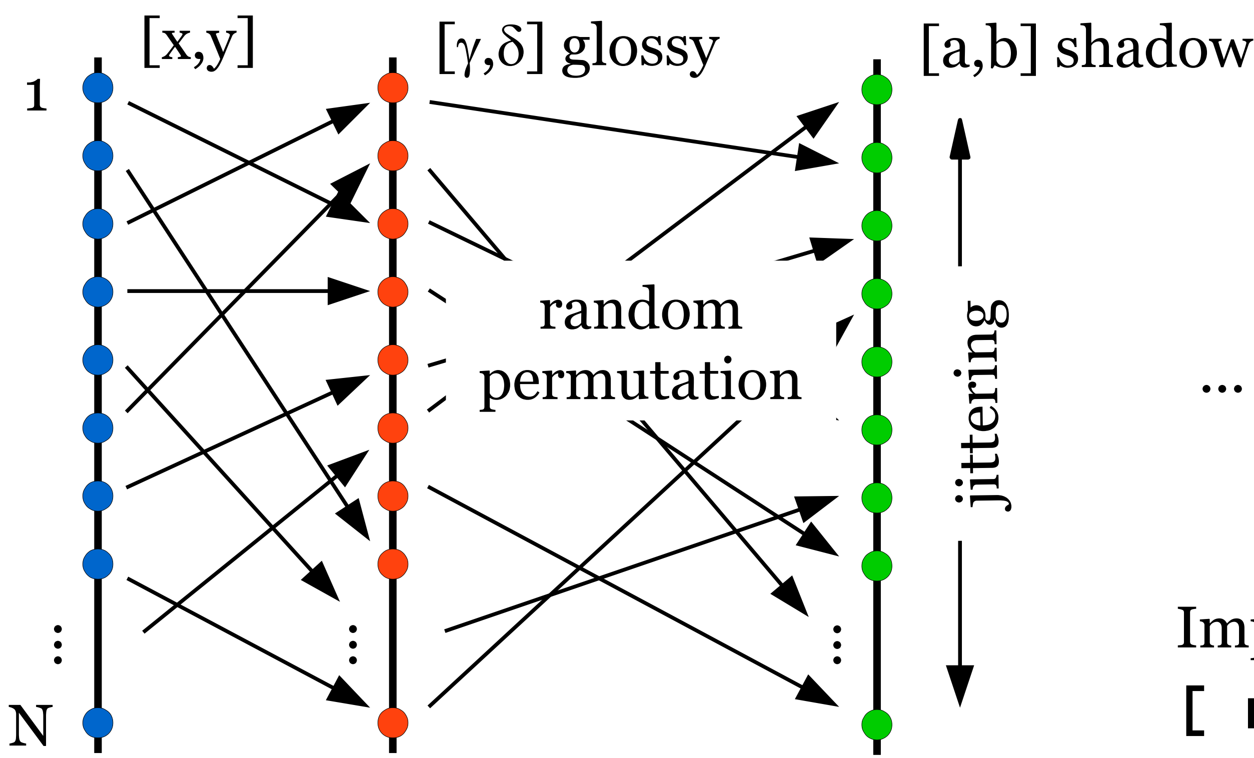
- any methods can be **combined**
  - with anti-aliasing as well
  - higher order integrals – e.g. dimension 10:

    anti-aliasing (2), depth of field (2), glossy reflection (2), soft shadows (2), motion blur (1), light dispersion (1)

- **sampling method**:
  - jittering
  - independent jittering (“N rooks”) in form of **hidden sampling**
  - adaptive sampling

Hidden sampling

- number of **samples per pix** (primary rays) is defined
  - every inner component is sampling independently
  - arbitrary number of additional (sampled) dimensions



Implementation:
`[ rank, total ]`

References

A. Glassner: ***An Introduction to Ray Tracing***, Academic Press, London 1989, 171-199

A. Watt, M. Watt: ***Advanced Animation and Rendering Techniques***, Addison-Wesley, Wokingham 1992, 262-265