Course Updates

- Tutorial this week
  - Assignment 3 and Assignment 4

- Assignment 3
  - **Programming** is due Friday
  - **Theory** is due Monday

- Assignment 4 (**only programming,** can be done in groups of 2)
  - Send me e-mail with the names of people in your group
  - **Due date:** December 3rd
  - **Demo Day:** December 3rd

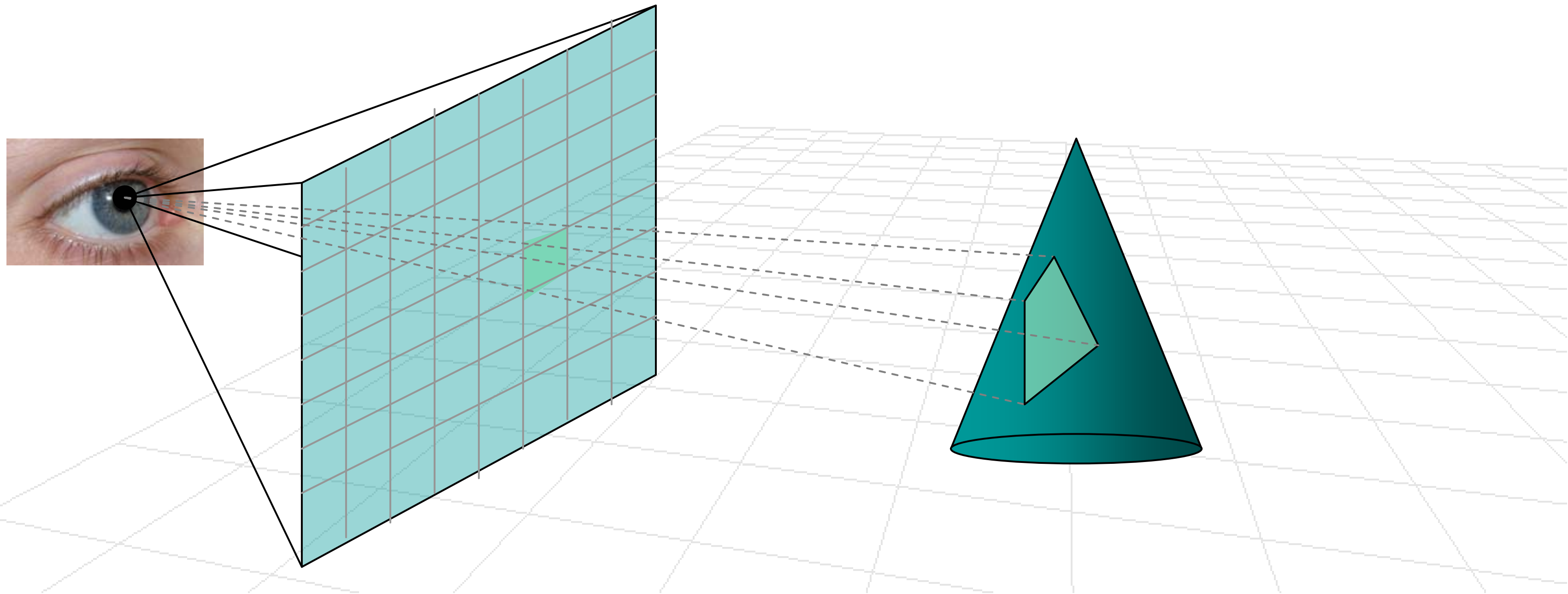
Short Review

- Ray casting
  - Generate a ray through each pixel (x,y) in the image plane
- Ray-surface intersection
  - Triangles, planar patches, spheres, etc.
- Computing normal at the "hit point"
- Lighting at the point
  - Whitted Model (Phong lighting + recursive global lighting term)
- Radiometry
  - Theory of lighting
  - Bidirectional Reflectance Distribution Functions (BRDFs)
  - How to compute radiance (by integrating over the incoming directions to compute irradiance)

How will all of this help in Ray Tracing?

- We will consider a more accurate (and much more expensive) approximation to the radiance at the "hit point" based on the integral of the BRDF and incident irradiance

- What do we integrate over?
  - We integrate over area of a pixel



Distribution Ray Tracing

Computer Graphics, CSCD18

Fall 2007

Instructor: Leonid Sigal

Distribution Ray Tracing

- In Basic Ray Tracing we computed lighting very crudely
  - Phong + specular global lighting

- In Distributed Ray Tracing we want to compute the lighting as accurately as possible
  - Use the formalism of Radiometry introduced in last class
  - Compute irradiance at each pixel (by integrating all the incoming light)
  - Since integrals are can not be done analytically, we will employ numeric approximations

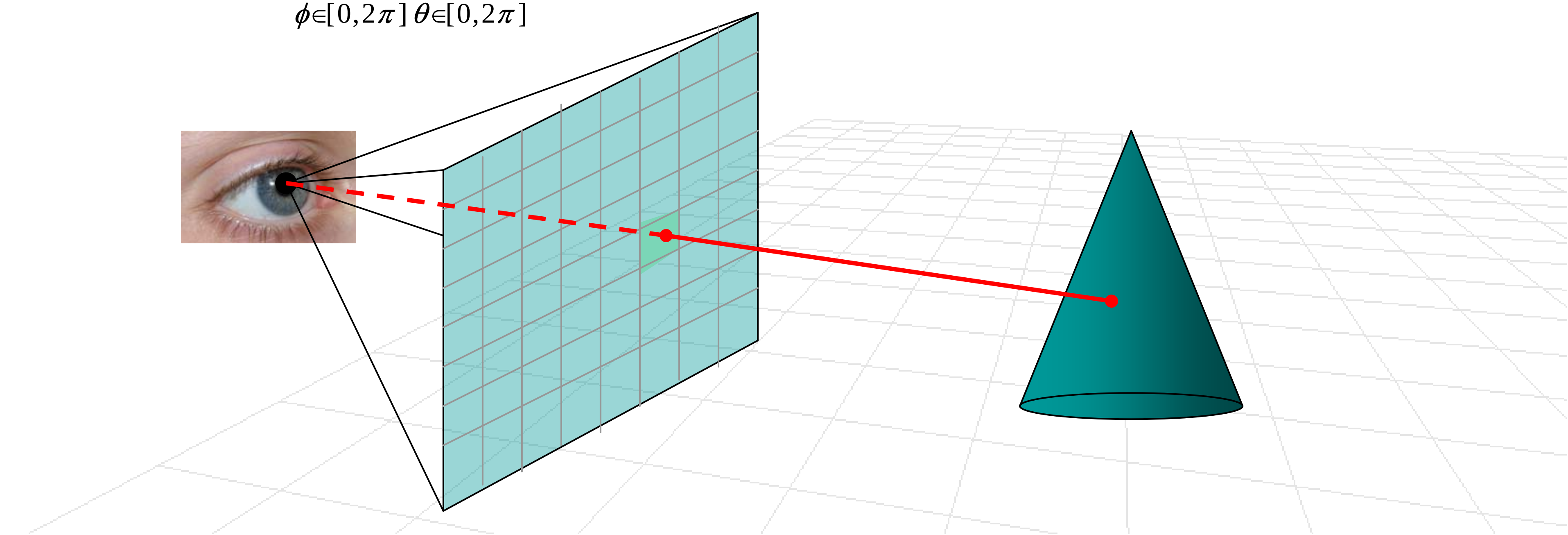
Radiance at a Point

- Recall that radiance (shading) at a surface point is given by

$$L(\bar{\boldsymbol{p}},\vec{\boldsymbol{d}}_e) = \int_\Omega \rho(\vec{\boldsymbol{d}}_e,\vec{\boldsymbol{d}}_i) L(\bar{\boldsymbol{p}},-\vec{\boldsymbol{d}}_i)(\vec{\boldsymbol{n}}\cdot\vec{\boldsymbol{d}}_i)\,\boldsymbol{d}\omega$$

- If we parameterize directions in spherical coordinates and assume small differential solid angle, we get

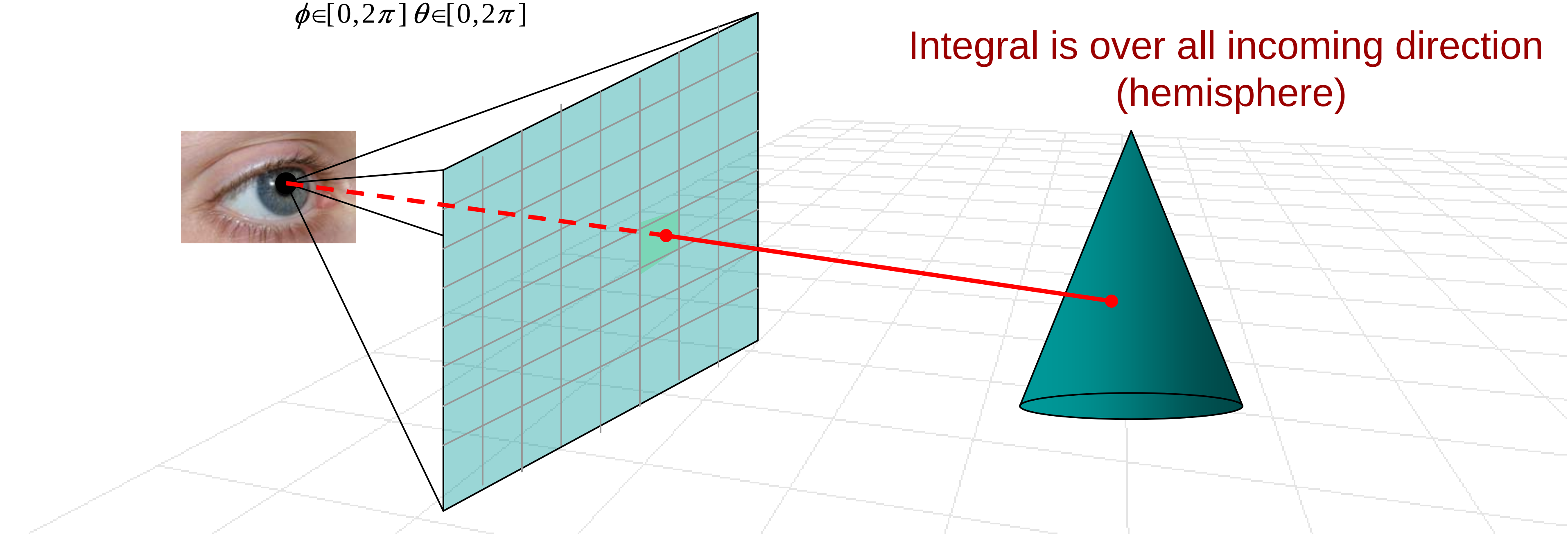
$$L(\bar{\boldsymbol{p}},\vec{\boldsymbol{d}}_e) = \int_{\phi\in[0,2\pi]} \int_{\theta\in[0,2\pi]} \rho\left(\vec{\boldsymbol{d}}_e,\vec{\boldsymbol{d}}_i(\phi,\theta)\right) L\left(\bar{\boldsymbol{p}},-\vec{\boldsymbol{d}}_i(\phi,\theta)\right)\left(\vec{\boldsymbol{n}}\cdot\vec{\boldsymbol{d}}_i(\phi,\theta)\right)\sin\theta\,\boldsymbol{d}\theta\,\boldsymbol{d}\phi$$



Radiance at a Point

- Recall that radiance (shading) at a surface point is given by
$$L(\bar{\boldsymbol{p}},\vec{\boldsymbol{d}}_e) = \int_{\Omega} \rho(\vec{\boldsymbol{d}}_e,\vec{\boldsymbol{d}}_i) L(\bar{\boldsymbol{p}},-\vec{\boldsymbol{d}}_i)(\vec{\boldsymbol{n}}\cdot\vec{\boldsymbol{d}}_i)\, \boldsymbol{d}\omega$$
- If we parameterize directions in spherical coordinates and assume small differential solid angle, we get

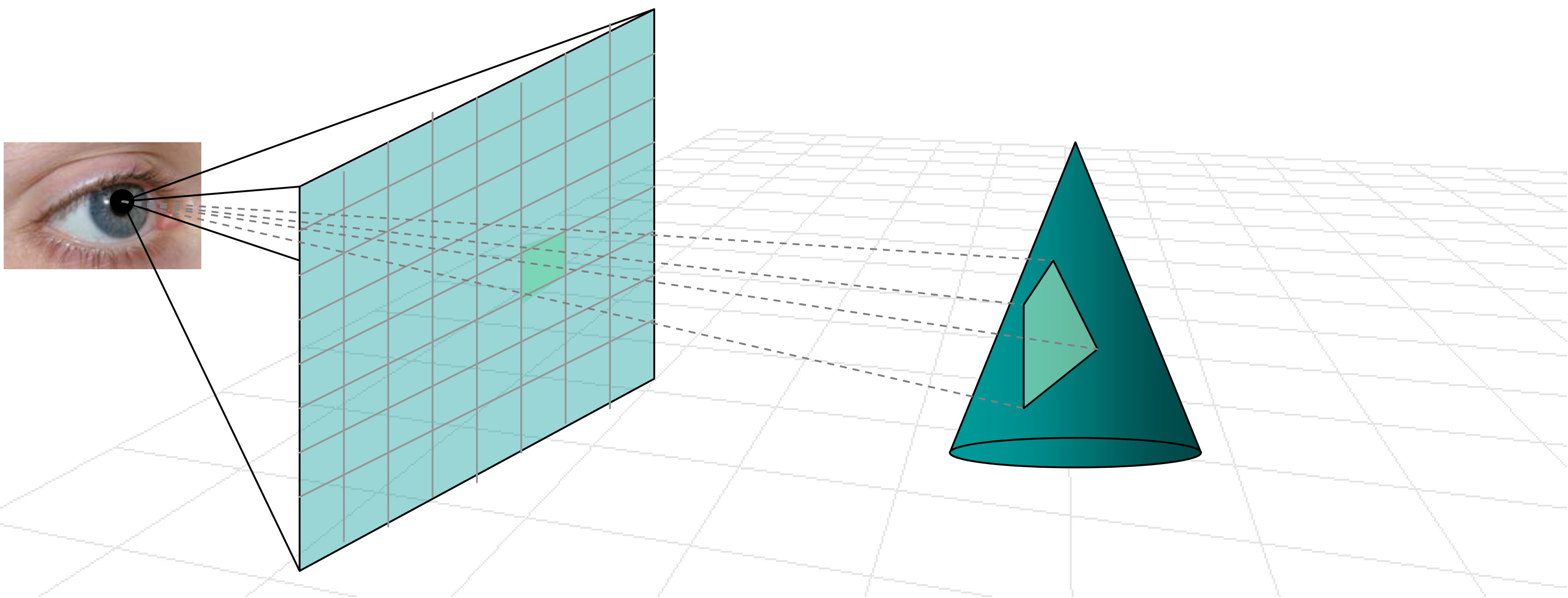
$$L(\bar{\boldsymbol{p}},\vec{\boldsymbol{d}}_e) = \int_{\phi\in[0,2\pi]} \int_{\theta\in[0,2\pi]} \rho\left(\vec{\boldsymbol{d}}_e,\vec{\boldsymbol{d}}_i(\phi,\theta)\right) L\left(\bar{\boldsymbol{p}},-\vec{\boldsymbol{d}}_i(\phi,\theta)\right)\left(\vec{\boldsymbol{n}}\cdot\vec{\boldsymbol{d}}_i(\phi,\theta)\right)\sin\theta\, \boldsymbol{d}\theta\, \boldsymbol{d}\phi$$



Irradiance at a Pixel

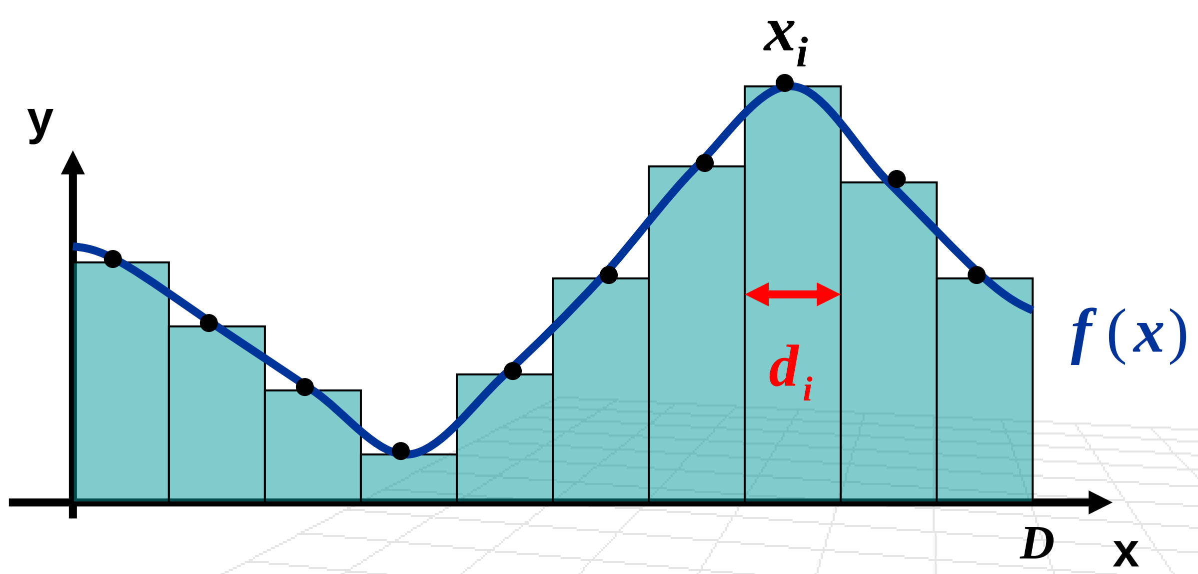
- To compute the color of the pixel, we need to compute total light energy (flux) passing through the pixel (rectangle) (i.e. we need to compute the total irradiance at a pixel)

$$\Phi_{i,j} = \int_{\alpha_{\min} \le \alpha \le \alpha_{\max}} \int_{\beta_{\min} \le \beta \le \beta_{\max}} H(\alpha,\beta)\, d\alpha\, d\beta$$



Numerical Integration (1D Case)

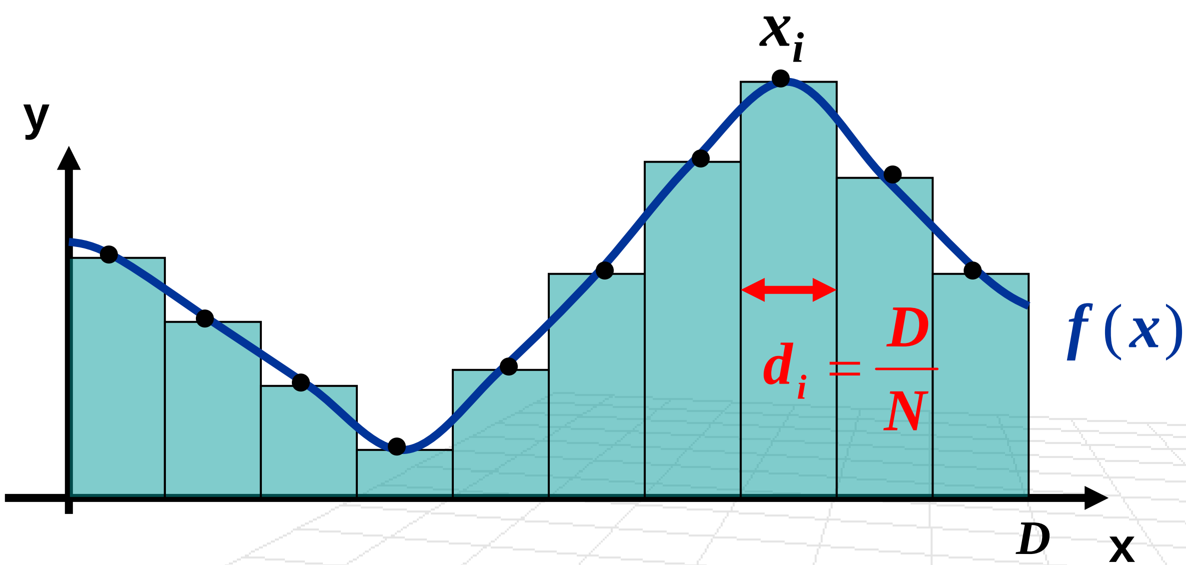
- **Remember:** integral is an area under the curve
- We can approximate any integral numerically as follows



$$\sum_{i=1}^{N} d_i f(x_i) \xrightarrow[N \to \infty]{} \int_0^D f(x)\,dx$$

Numerical Integration (1D Case)

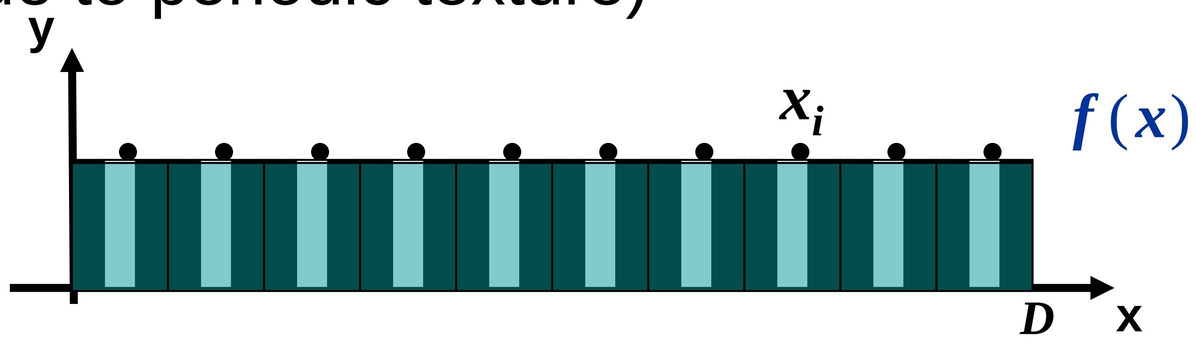
- **Remember:** integral is an area under the curve
- We can approximate any integral numerically as follows



$$\int_0^D f(x)\,dx \approx \sum_{i=1}^{N} \frac{D}{N} f(x_i)$$

Monte Carlo Integration

- **Idea:** randomize points x_i to avoid structured noise (e.g. due to periodic texture)

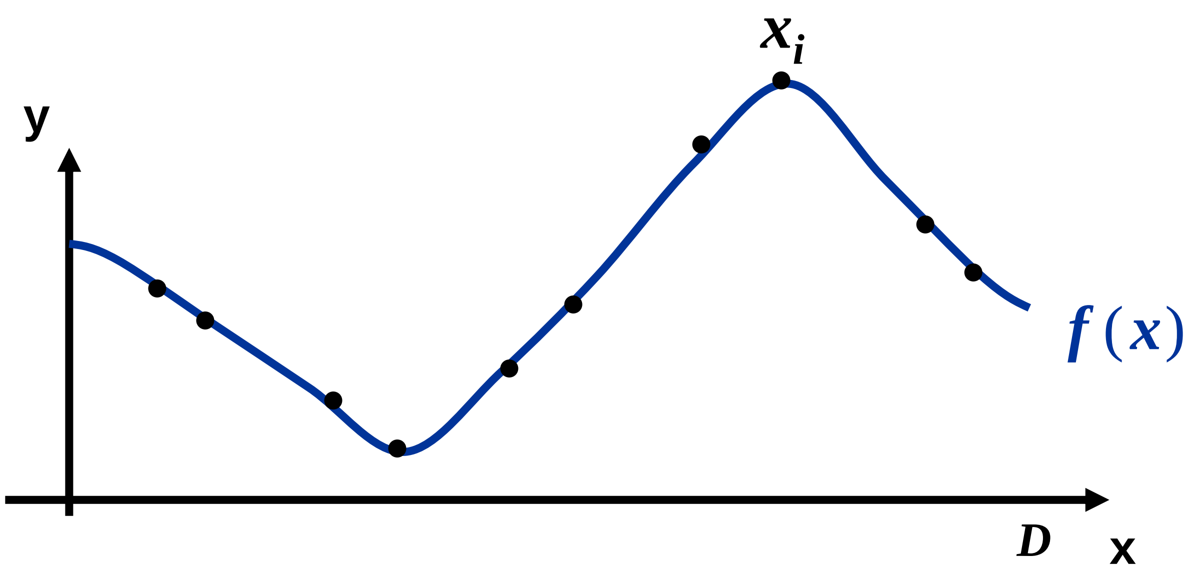


- Draw N random samples x_i independently from uniform distribution $Q(x)=U[0,D]$ (i.e. $Q(x) = 1/D$ is the uniform probability density function)
- Then approximation to the integral becomes

$$\frac{1}{N}\sum w_i f(x_i) \approx \int f(x)\,dx \quad \text{, for } \quad w_i = \frac{1}{Q(x_i)}$$

- **We can also use other Q's for efficiency !!!** (a.k.a. importance sampling)

Monte Carlo Integration



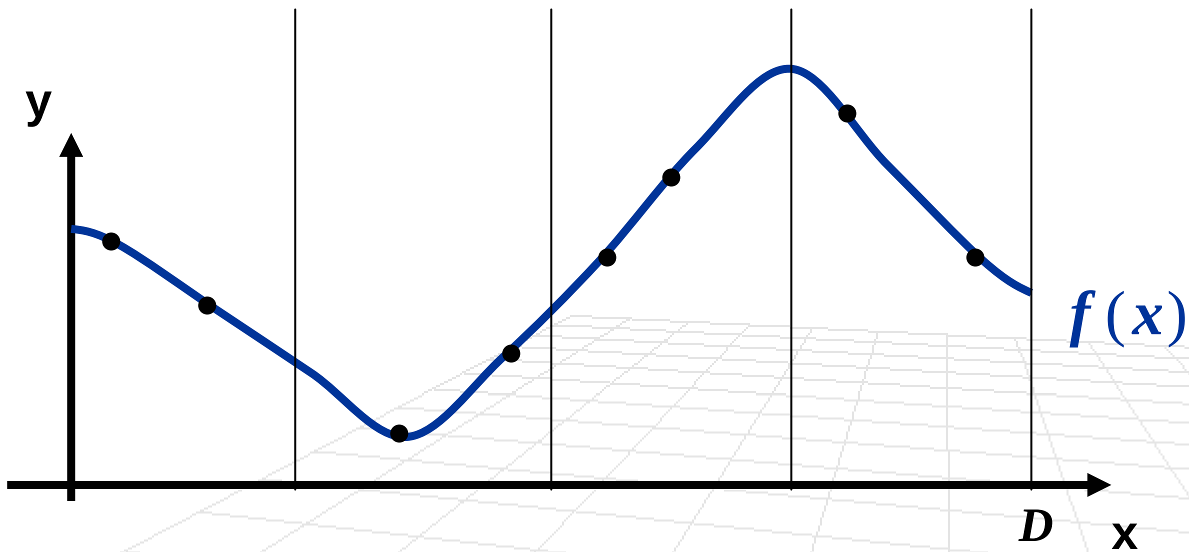
- Then approximation to the integral becomes

$$\frac{1}{N}\sum w_i f(x_i) \approx \int f(x)\,dx \quad \text{, for } \quad w_i = \frac{1}{Q(x_i)}$$

- **We can also use other *Q*'s for efficiency !!!** (a.k.a. importance sampling)

Stratified Sampling

- **Idea:** combination of uniform sampling plus random jitter
- Break domain into T intervals of widths d_t and N_t samples in interval t

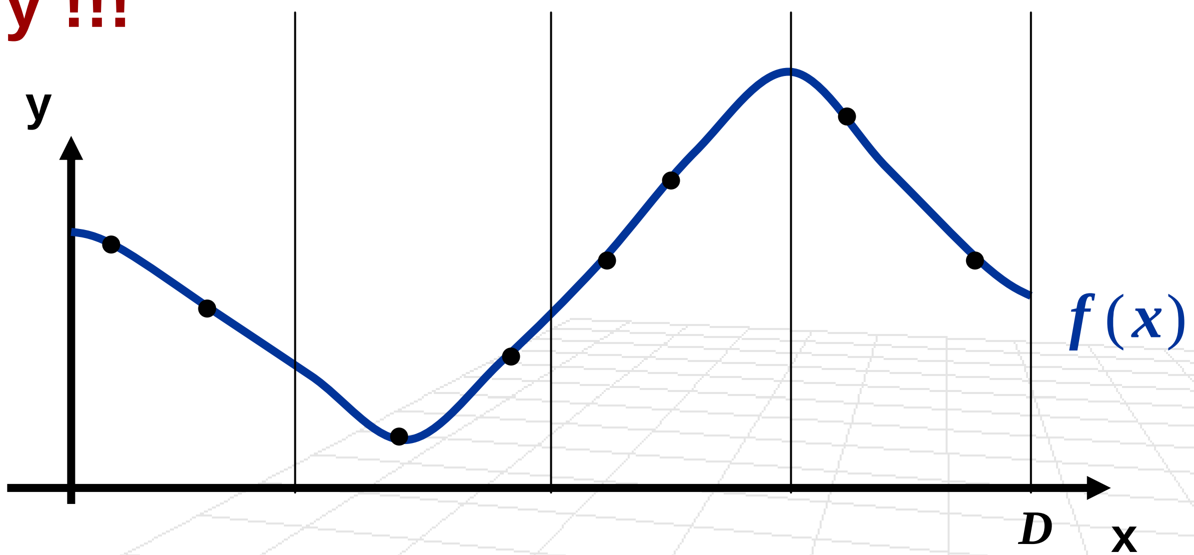


- Integral approximated using the following:

$$\sum_{t=1}^{T} \frac{1}{N_t} \sum_{j=1}^{N_t} d_t f(x_{t,j})$$

Stratified Sampling

- If intervals are uniform $d_t = D/T$ and there are same number of samples in each interval $N_t = N/T$ then this approximation reduces to: $\sum_{t=1}^{T}\sum_{j=1}^{N_t}\frac{D}{N}f(x_{t,j})$
- **In general, the interval size and the # of samples can vary !!!**



- Integral approximated using the following:

$$\sum_{t=1}^{T}\frac{1}{N_t}\sum_{j=1}^{N_t} d_t f(x_{t,j})$$

Back to Distribution Ray Tracing

- Based on one of the approximate integration approaches we need to compute
  - Let's try uniform sampling

$$L(\bar{\boldsymbol{p}},\vec{\boldsymbol{d}}_e) = \int_{\phi\in[0,2\pi]}\int_{\theta\in[0,2\pi]} \rho\left(\vec{\boldsymbol{d}}_e,\vec{\boldsymbol{d}}_i(\phi,\theta)\right)\boldsymbol{L}\left(\bar{\boldsymbol{p}},-\vec{\boldsymbol{d}}_i(\phi,\theta)\right)\left(\vec{\boldsymbol{n}}\cdot\vec{\boldsymbol{d}}_i(\phi,\theta)\right)\sin\theta\,\boldsymbol{d}\theta\,\boldsymbol{d}\phi$$

$$\approx \sum_{\boldsymbol{m}=1}^{\boldsymbol{M}}\sum_{\boldsymbol{n}=1}^{\boldsymbol{N}} \rho\left(\vec{\boldsymbol{d}}_e,\vec{\boldsymbol{d}}_i(\phi_m,\theta_n)\right)\boldsymbol{L}\left(\bar{\boldsymbol{p}},-\vec{\boldsymbol{d}}_i(\phi_m,\theta_n)\right)\left(\vec{\boldsymbol{n}}\cdot\vec{\boldsymbol{d}}_i(\phi_m,\theta_n)\right)\sin\theta\,\Delta\theta\,\Delta\phi$$

where

$$\theta_n = \left(\boldsymbol{n}-\frac{1}{2}\right)\Delta\theta \qquad \Delta\theta = \frac{\pi/2}{\boldsymbol{M}}$$

$$\phi_m = \left(\boldsymbol{m}-\frac{1}{2}\right)\Delta\phi \qquad \Delta\phi = \frac{2\pi}{\boldsymbol{N}}$$

midpoint of the interval (sample point) **Interval width**

Importance Sampling in Distribution Ray Tracing

- **Problem:** Uniform sampling is too expensive (e.g. **100** samples/hemisphere with depth of ray recursion of **4** => $\mathbf{100^4 = 10^8}$ samples per pixel … with $\mathbf{10^5}$ pixels => $\mathbf{10^{15}}$ samples)

- **Solution:** Sample more densely (using importance sampling) where we know that effects will be most significant (e.g. visible surfaces, light sources, etc.)
  - Direction toward point or extended light source are significant
  - Specular and off-axis specular are significant
  - Texture/lightness gradients are significant
  - Sample less with greater depth of recursion

Importance Sampling (review)

- **Idea:** Approximates any integral by samples drawn independently and identically from some desired importance distribution $\boldsymbol{Q(x)}$

$$\frac{1}{N}\sum f(x_i) \approx \int Q(x) f(x) dx, \quad x_i \sim Q(x)$$

- This is not quite what we want, but if we (scale) or divide by $\boldsymbol{Q(x_i)}$

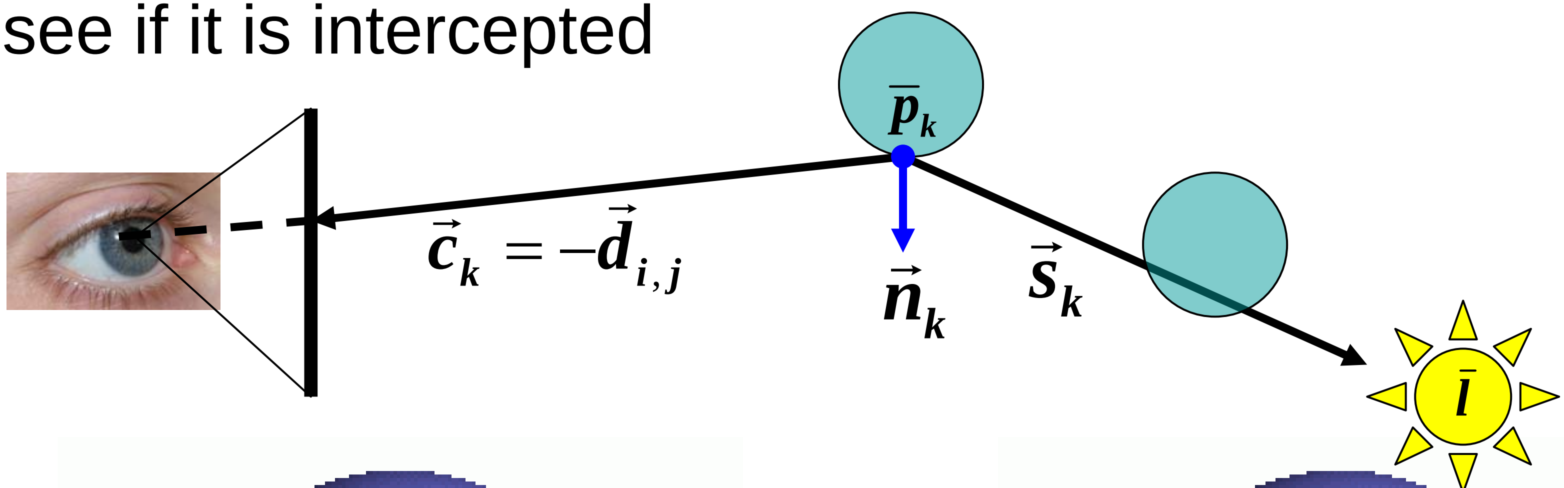
$$\frac{1}{N}\sum w_i f(x_i) \approx \int f(x) dx \text{ , for } w_i = \frac{1}{Q(x_i)}$$

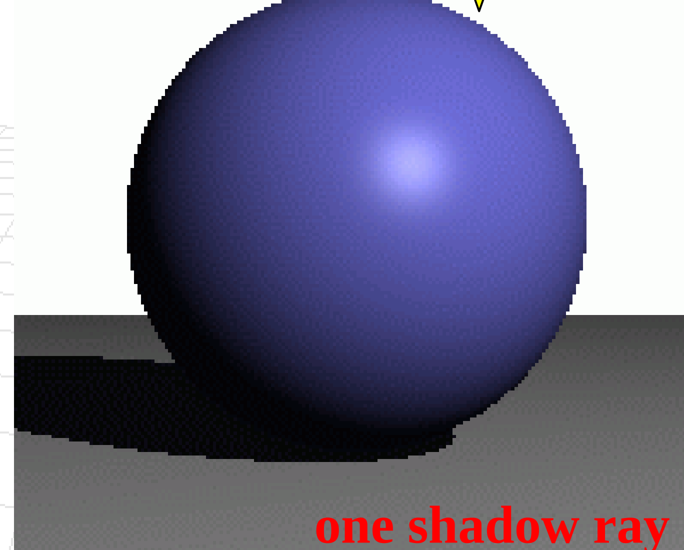
Benefits of Distribution Ray Tracing

- Better global diffuse lighting
  - Color bleeding
  - Bouncing highlights
- Extended light sources
- Anti-aliasing
- Motion blur
- Depth of field
- Subsurface scattering

Shadows in Ray Tracing

- Recall, we shoot a ray towards a light source and see if it is intercepted

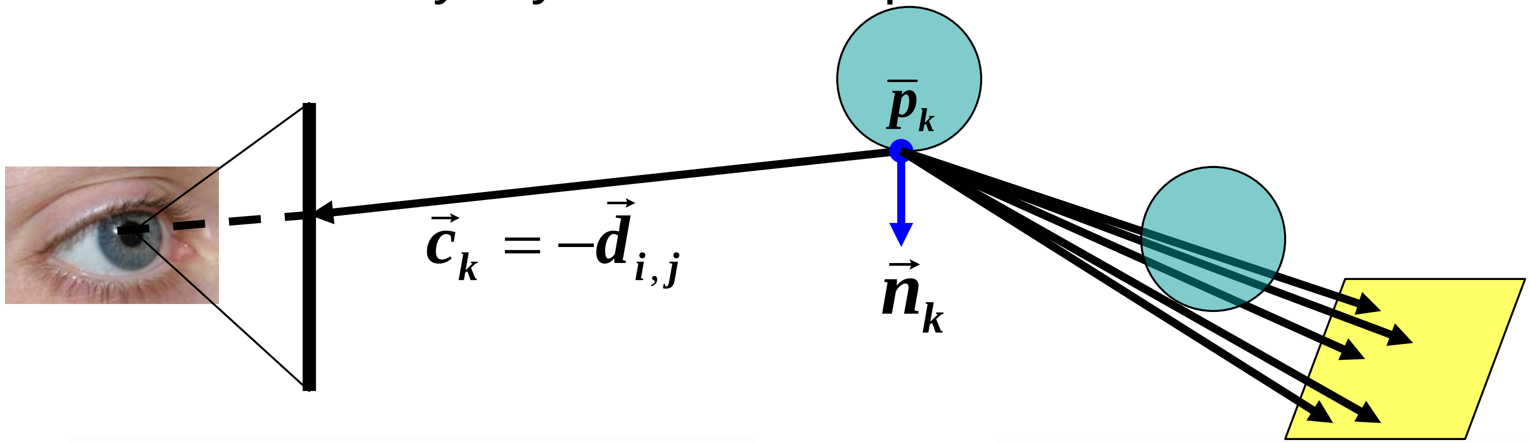


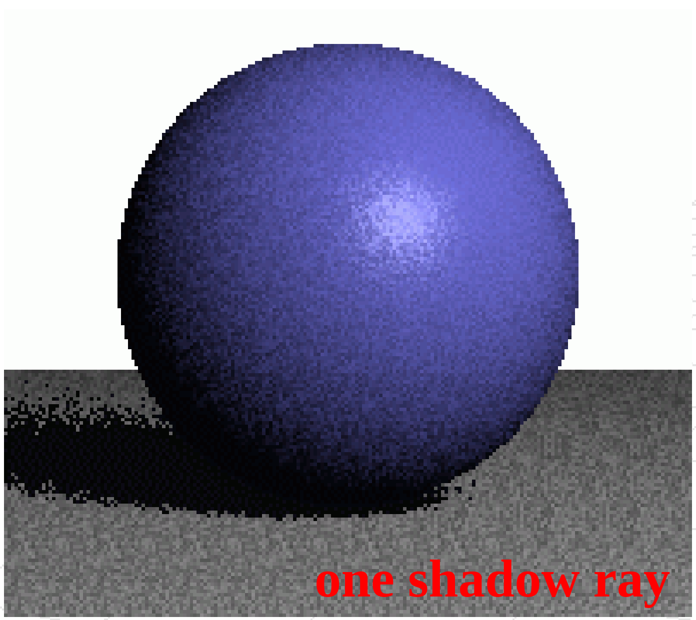


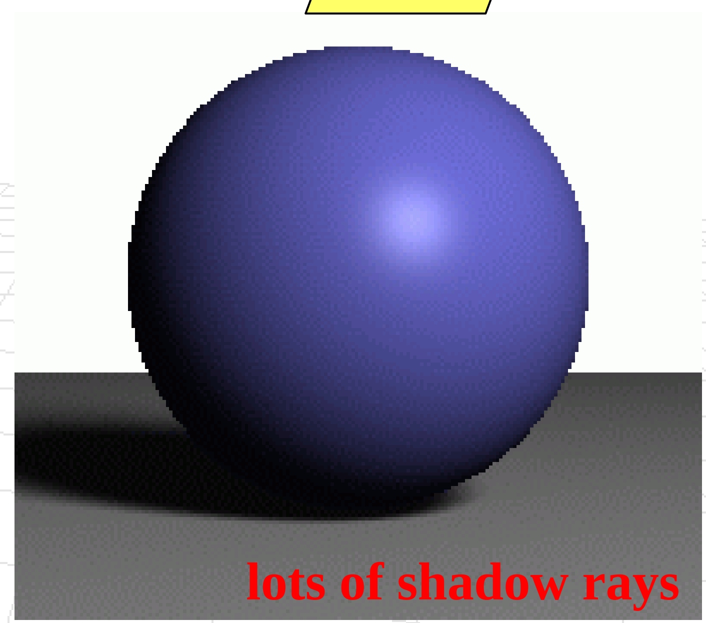
Images from the slides by Durand and Cutler

Soft Shadows with Distribution Ray Tracing

- Lets shoot multiple rays from the same point and attenuate the color based on how many rays are intercepted

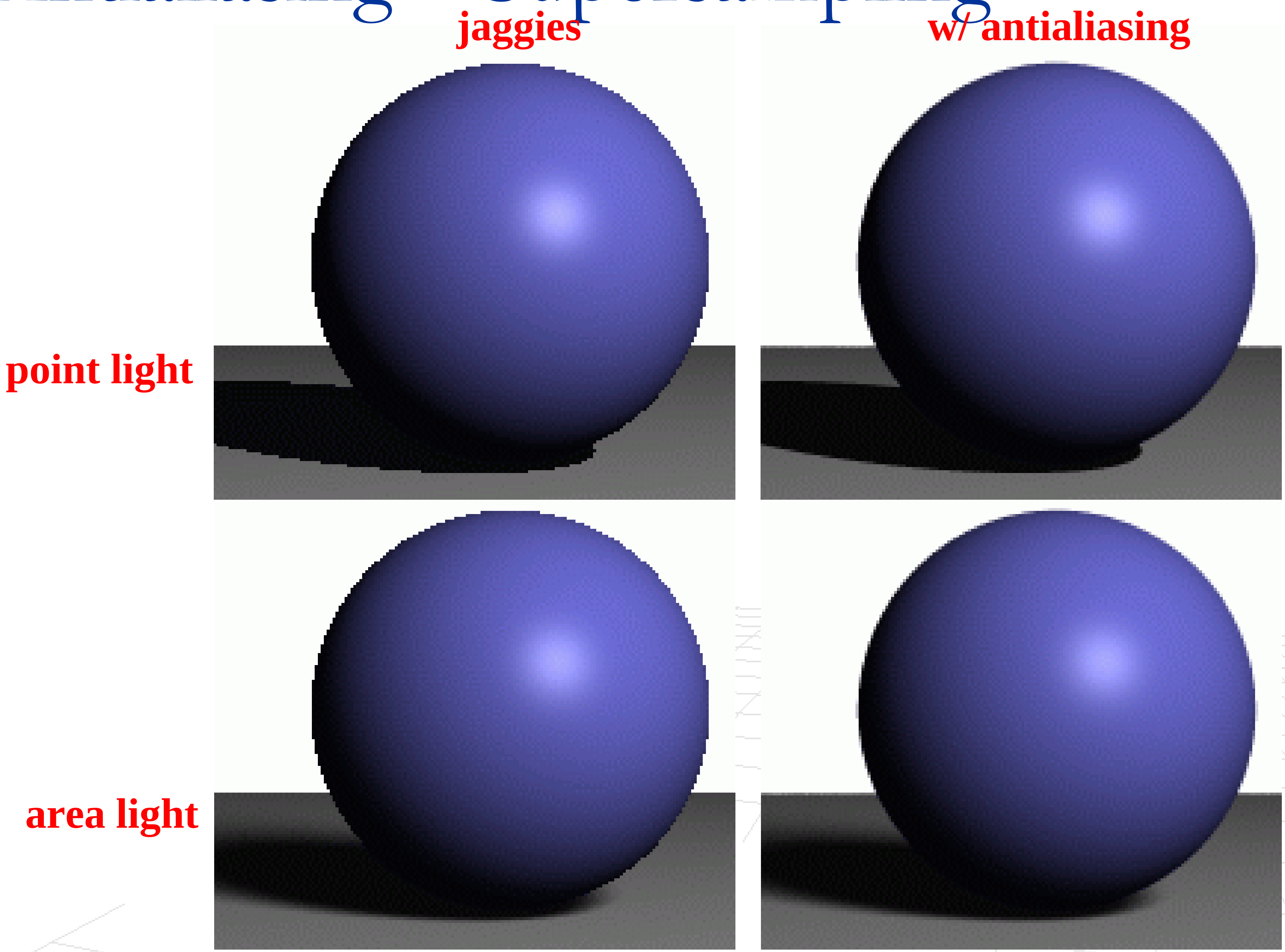






Images from the slides by Durand and Cutler

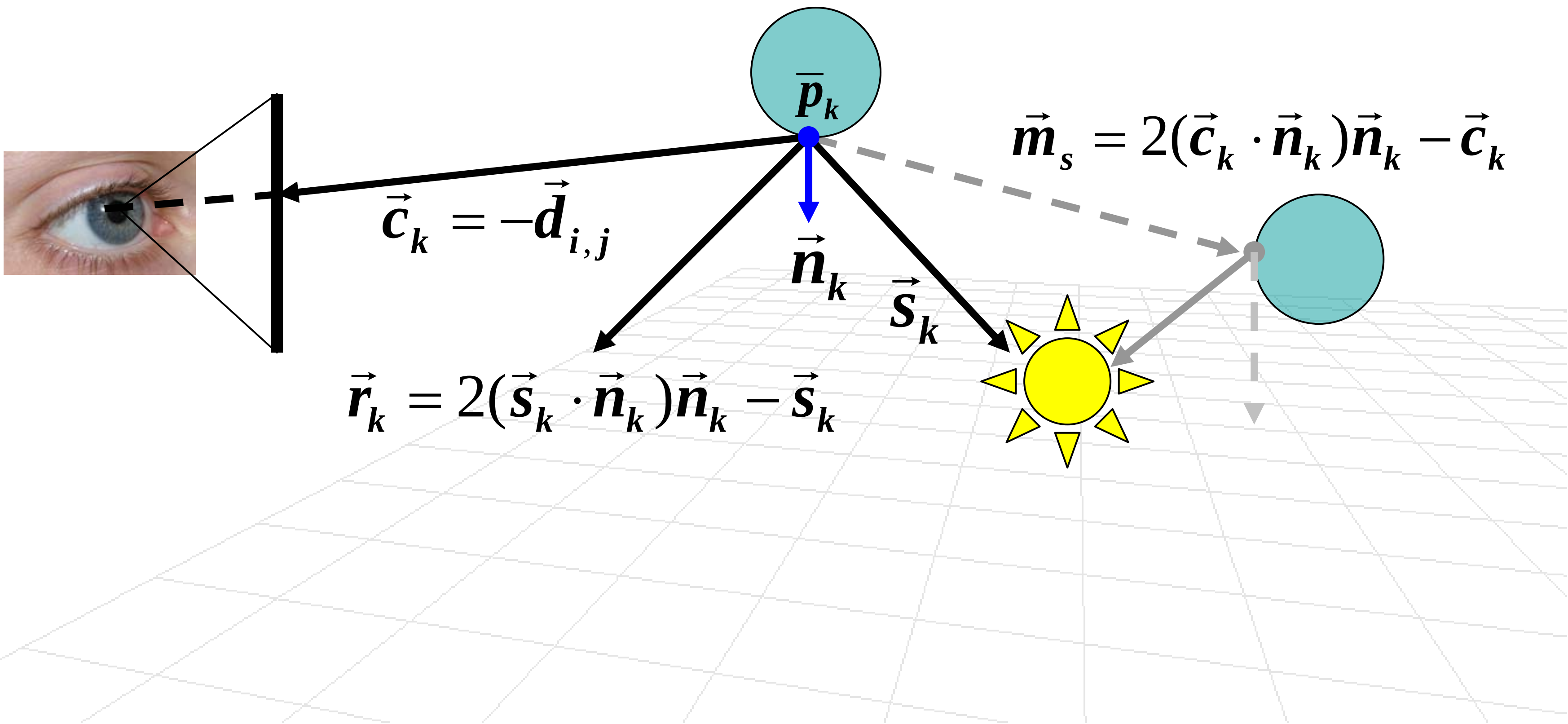
Antialiasing – Supersampling



Images from the slides by Durand and Cutler

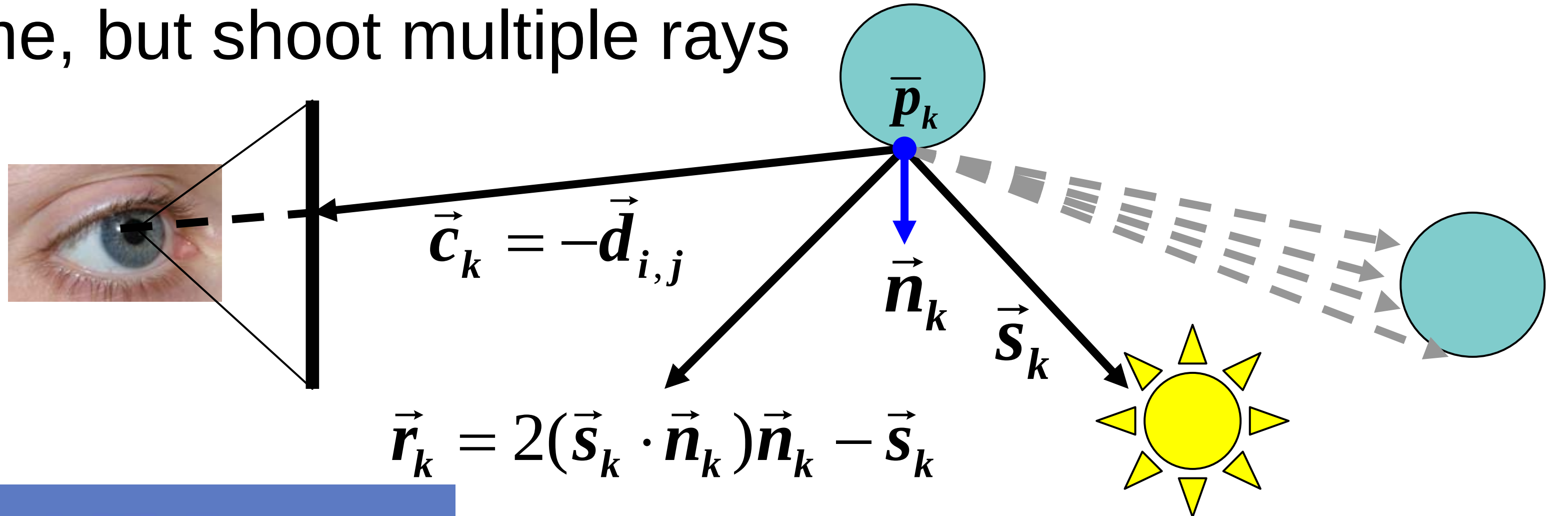
Specular Reflections

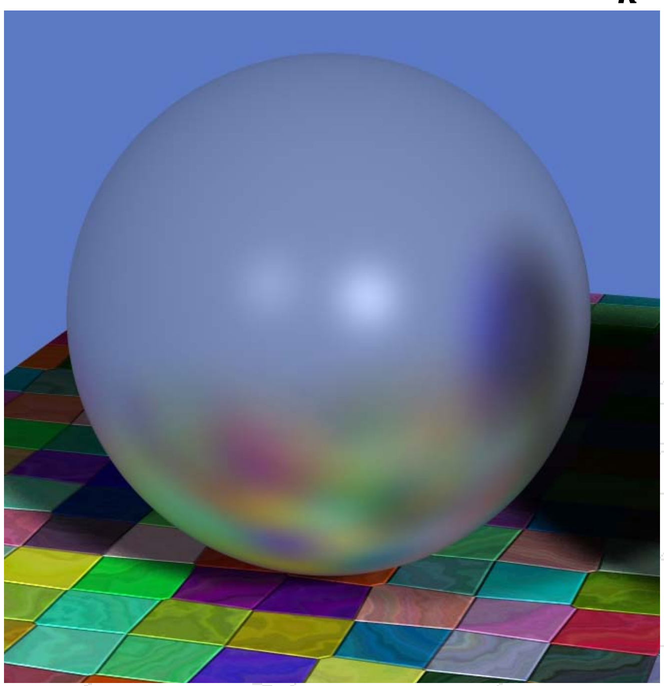
- Recall, we had to shoot a ray in a perfect specular reflection direction (with respect to the camera) and get the radiance at the resulting hit point



Specular Reflections with DRT

- Same, but shoot multiple rays





Justin Legakis