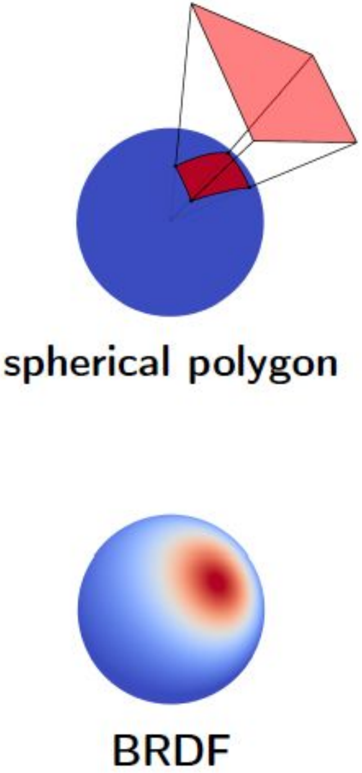
Real-Time Polygonal-Light Shading with Linearly Transformed Cosines

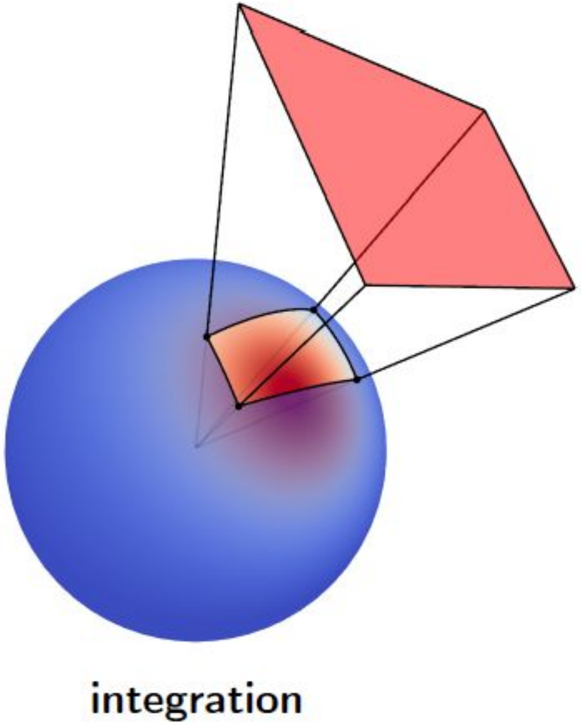
Eric Heitz at al. 2016 SIGGRAPH

Motivations

Realistic & Real-time CG

Hemi-spherical Integration over Polygons





Recall: Rendering Equation

$$L(x \rightarrow \Phi) = L_e(x \rightarrow \Phi) + \int_\Omega L(x \leftarrow \Psi) f_r(x, \Psi \rightarrow \Phi) \cos\theta_x d\omega_\Psi$$

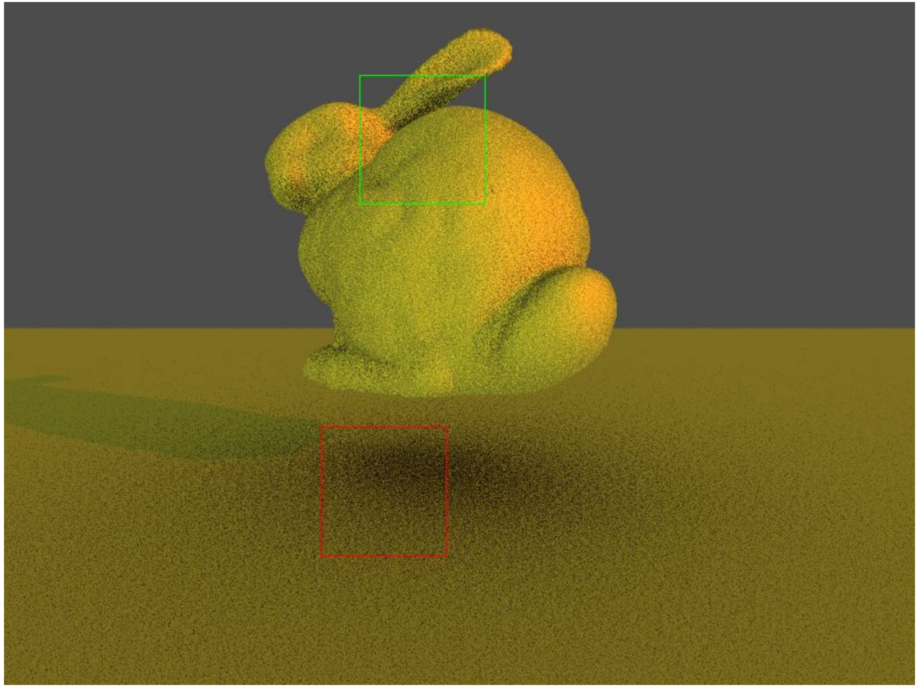
hemispherical integration!

$$L(\omega_o) = L_e(\omega_o) + \int_\Omega L(\omega_i) f_r(\omega_i, \omega_o) \cos\theta_i d\omega_i$$

What We've Learned

Path tracing(Monte Carlo)

$$\sum_{i=1}^{N} \frac{L(\omega_i) f_r(\omega_i, \omega_o) cos(\theta_i)}{p(\omega_i)}$$



Downside

- large amount of rays
- noise, artifacts

Analytic Solutions

Is it possible to solve the original spherical integrations analytically?

$$\int_{\Omega} L(\omega_i) f_r(\omega_i, \omega_o) \cos\theta_i d\omega_i = ????$$

Analytic Solutions

Is it possible to solve the original spherical integrations analytically?

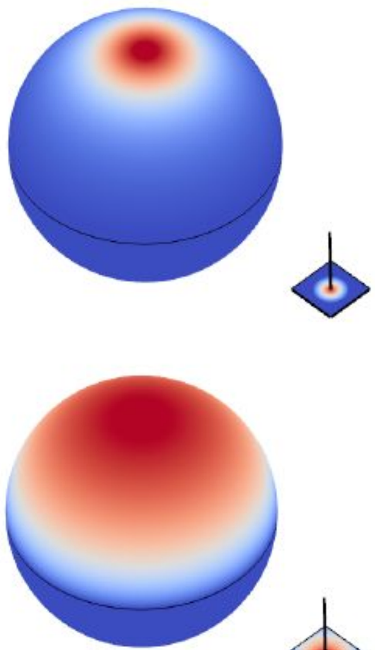
$$\int_\Omega L(\omega_i) f_r(\omega_i, \omega_o) \cos\theta_i d\omega_i = ????$$

Answer: Nope

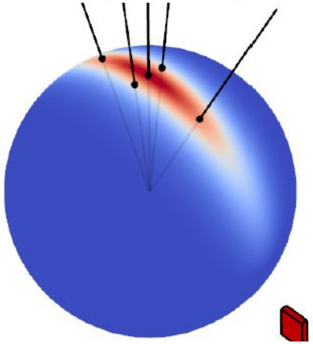
State-of-the-art physically based models → sophisticated to integrate

Sophisticated Shapes of BRDFs

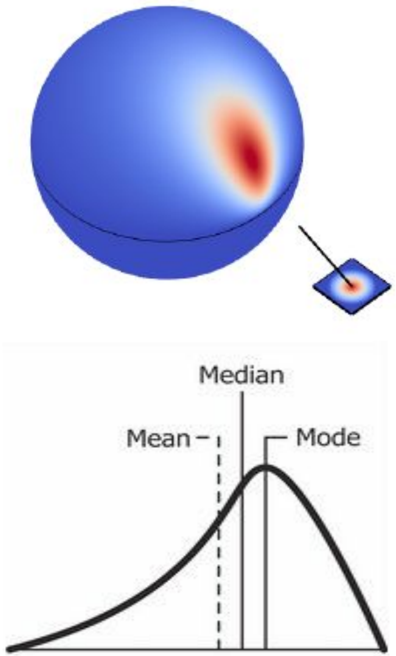
varying roughness



anisotropy



skewness



Approximate Method Is Required

To approximate integrations

The authors suggested

Linearly Transformed Cosines(LTCs) to approximate:

$$\int_{\Omega} L(\omega_i) \boxed{f_r(\omega_i, \omega_o) \cos \theta_i} d\omega_i$$

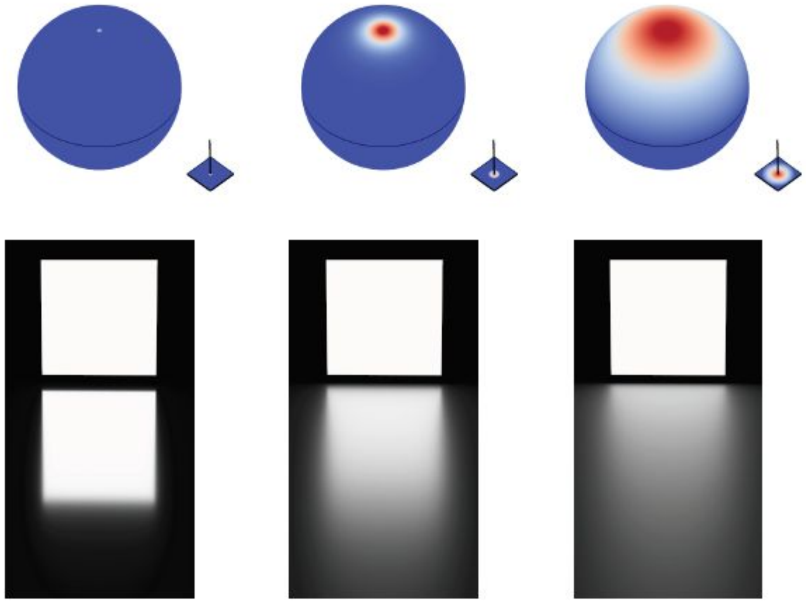
$$\approx D(\omega_i)$$

Summary

1. Find out a new distribution
2. approximate BRDFs
3. cover a wide variety of materials

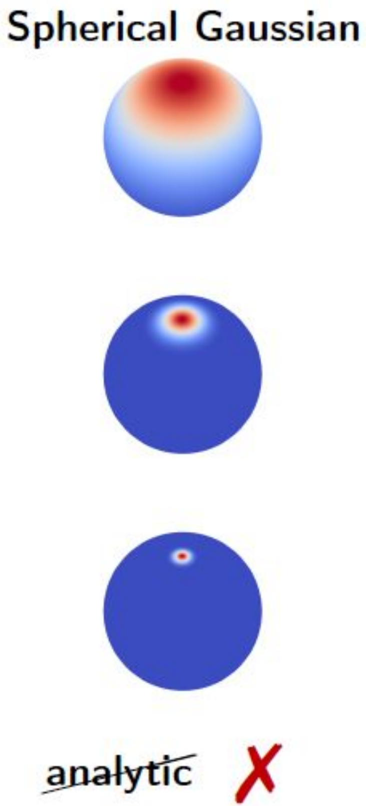
   : almost specular to very glossy

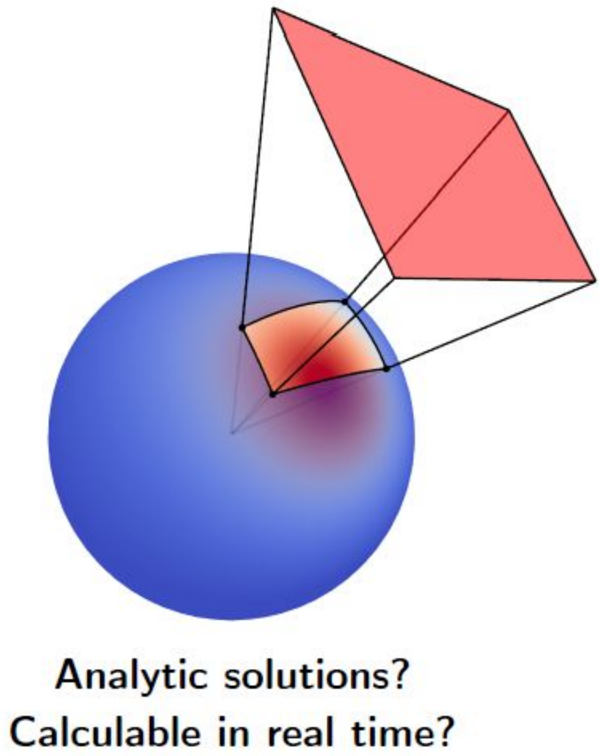
4. integration has to be done in real-time



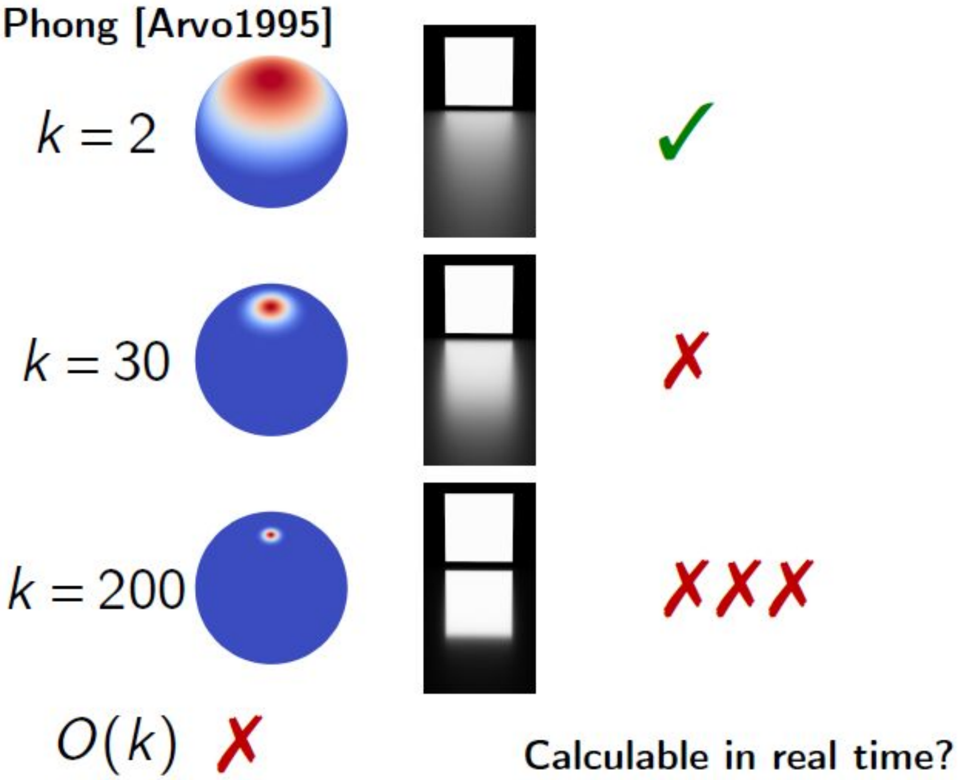
Finding Proper Distributions

Let’s Try with Some Distributions

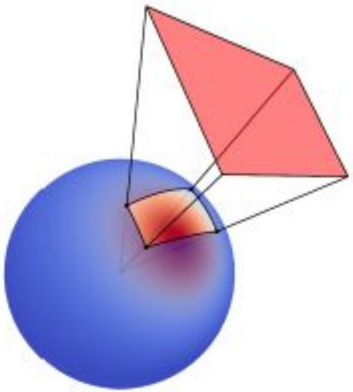




Let's Try with Some Distributions

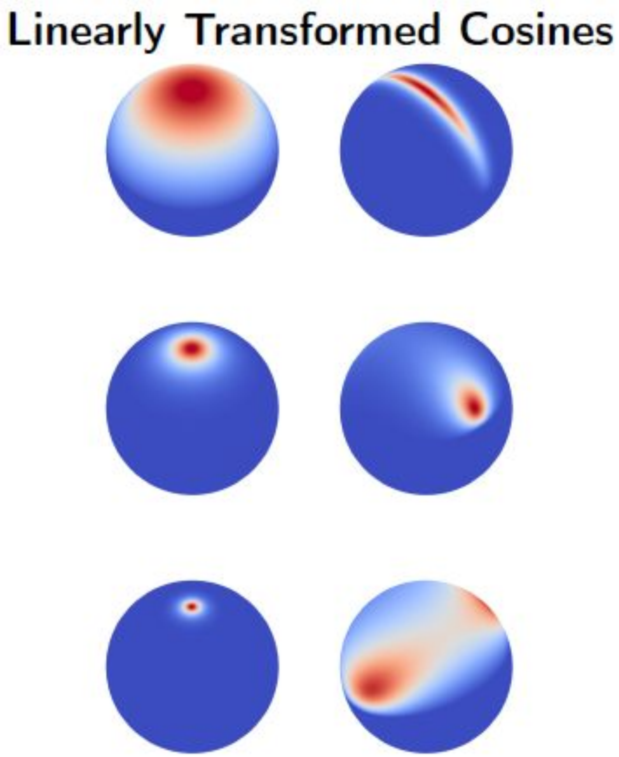


Let’s Try with Some Distributions



Is it possible
to extend the set of
mathematically exact
solutions?

Solution: Linearly Transformed Cosines(LTCs)



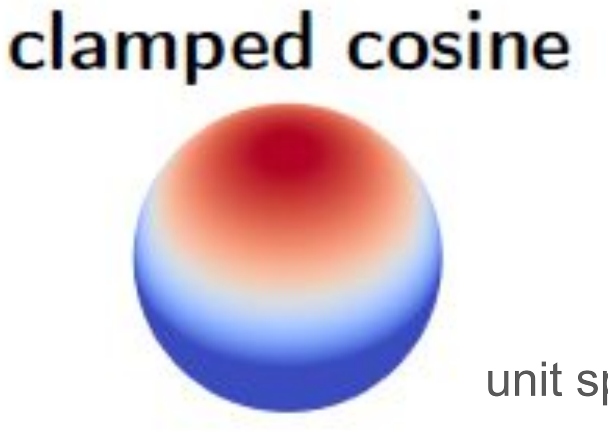
Linear transformations

- rotate, expand, squeeze, shear

Thus, can make almost arbitrary shapes!!!

Linearly Transformed Cosines

(Clamped) Cosine Distributions



unit sphere

$$D_0(\omega_0 = (x, y, z)) = \frac{1}{\pi}\max(\cos\theta_z, 0) = \frac{1}{\pi}\max(z, 0)$$

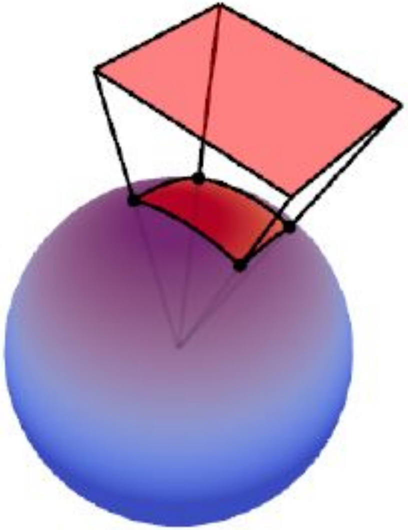
Main Characteristic of Cosine

Integration is so easy!!

$$\int_{P_0} D_0(\omega_0) d\omega_0 = \int_{P_0} \cos\theta \sin\theta d\theta d\phi$$

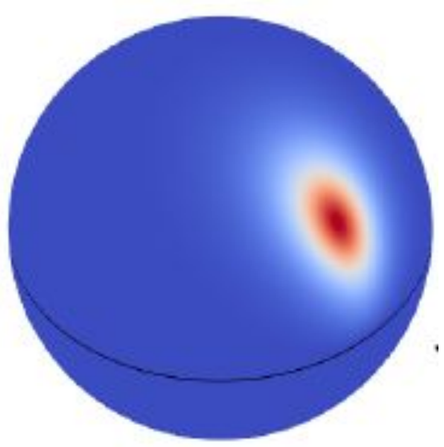
The complexity: O(# of vertices) = almost nothing

[Baum et al. 1989]



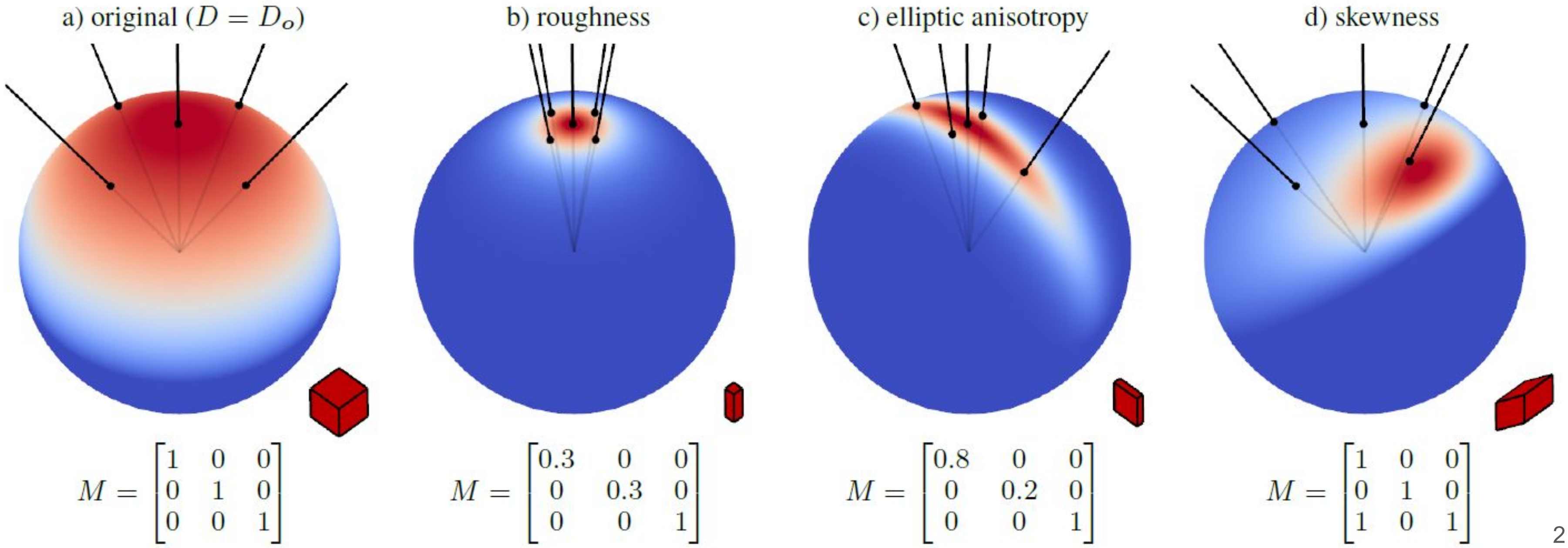
BRDFs vs Clamped Cosine Distributions

Unfortunately, BRDFs are very different from the original cosine distribution…



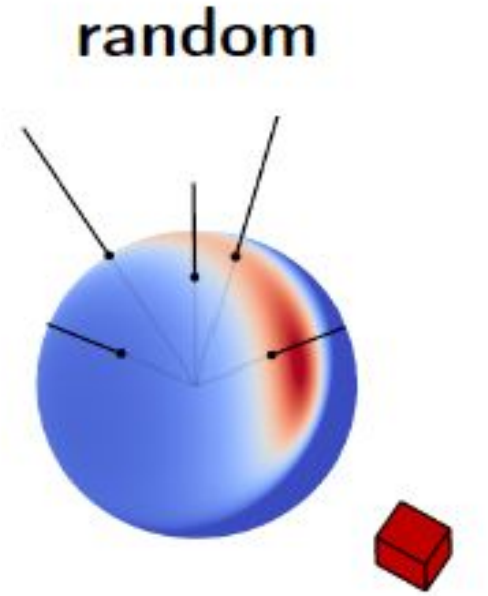
Linearly Transformed Cosines

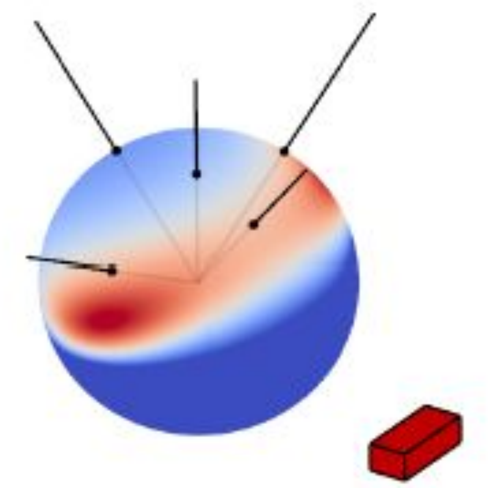
Somehow, we can cover various shapes of BRDF by using LTCs



Linearly Transformed Cosines

We can make even funny shapes





So far….

1. Polygon lights, Spherical integrations!!!
2. Try to find analytic solutions (real-time rendering)
3. Cosine integration is very simple.
4. Linearly transformed cosine distributions can approximate various shapes of BRDFs

So far….

1. Polygon lights, Spherical integrations!!!
2. Try to find analytic solutions (real-time rendering)
3. **Cosine integration is very simple.**
4. **Linearly transformed cosine distributions can approximate various shapes of BRDFs**

How can we connect 3 and 4??

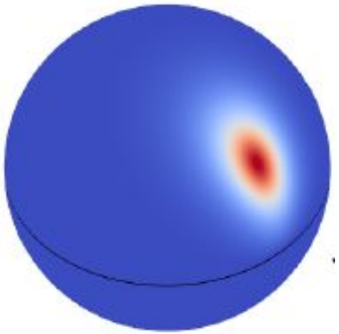
Mathematics

Reminiscence of Calculus I

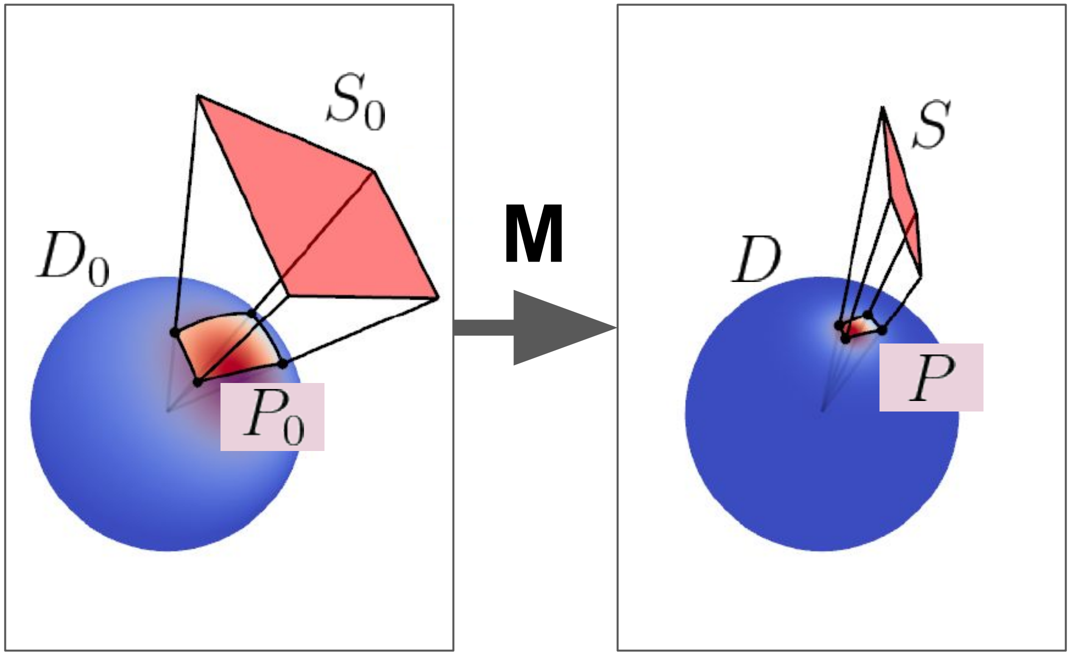
Change of variables (variable transformations)

$$\int_a^b \boxed{f(g(x))g'(x)}\,dx = \int_{g^{-1}(a)}^{g^{-1}(b)} \boxed{f(y)}\,dy$$

Complicated　　　　Simple



Connection Between Cosine and LTC



$$MS_0 = S$$

$$\frac{MP_0}{\|MP_0\|} = P$$

$$for\ \omega_0 \in P_0,\ \omega \in P$$

$$\frac{M\omega_0}{\|M\omega_0\|} = \omega$$

Cosine to LTC

$$\frac{M\omega_0}{\|M\omega_0\|} = \omega$$

$$\omega_0 = \frac{M^{-1}\omega}{\|M^{-1}\omega\|}$$

$$\int_{P_0} D_0(\omega_0) d\omega_0$$

$$= \int_P D_0\left(\frac{M^{-1}\omega}{\|M^{-1}\omega\|}\right) \frac{d\omega_0}{d\omega} d\omega$$

$$= \int_P D_0\left(\frac{M^{-1}\omega}{\|M^{-1}\omega\|}\right) \frac{|M^{-1}|}{\|M^{-1}\omega\|^3} d\omega = \int_P D(\omega) d\omega$$

LTC to Cosine

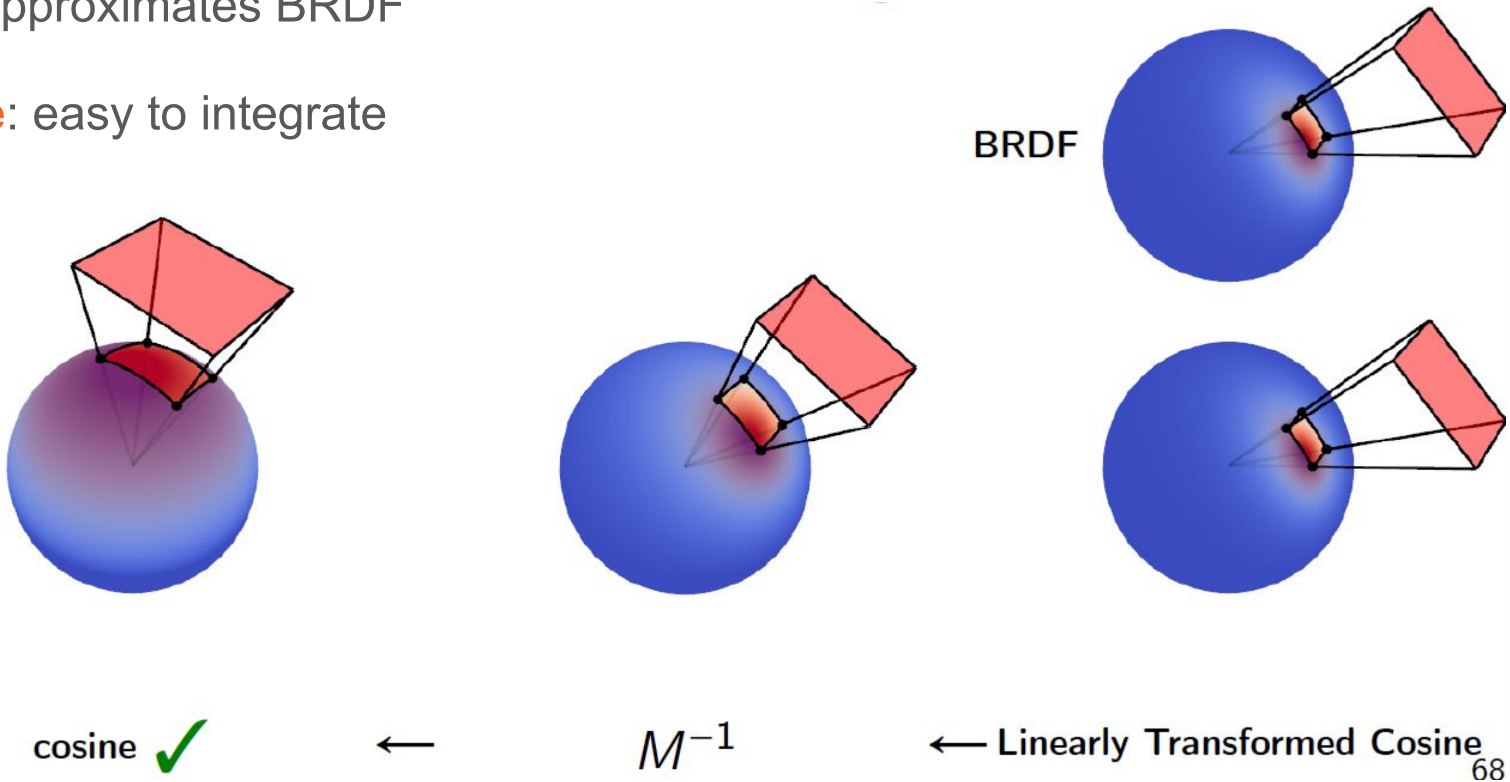
reciprocally….

$$\int_P D(\omega)d\omega = \int_{P_0} D\left(\frac{M\omega_0}{||M\omega_0||}\right)\frac{|M|}{||M\omega_0||^3}d\omega_0 = \int_{P_0} D_0(\omega_0)d\omega_0$$

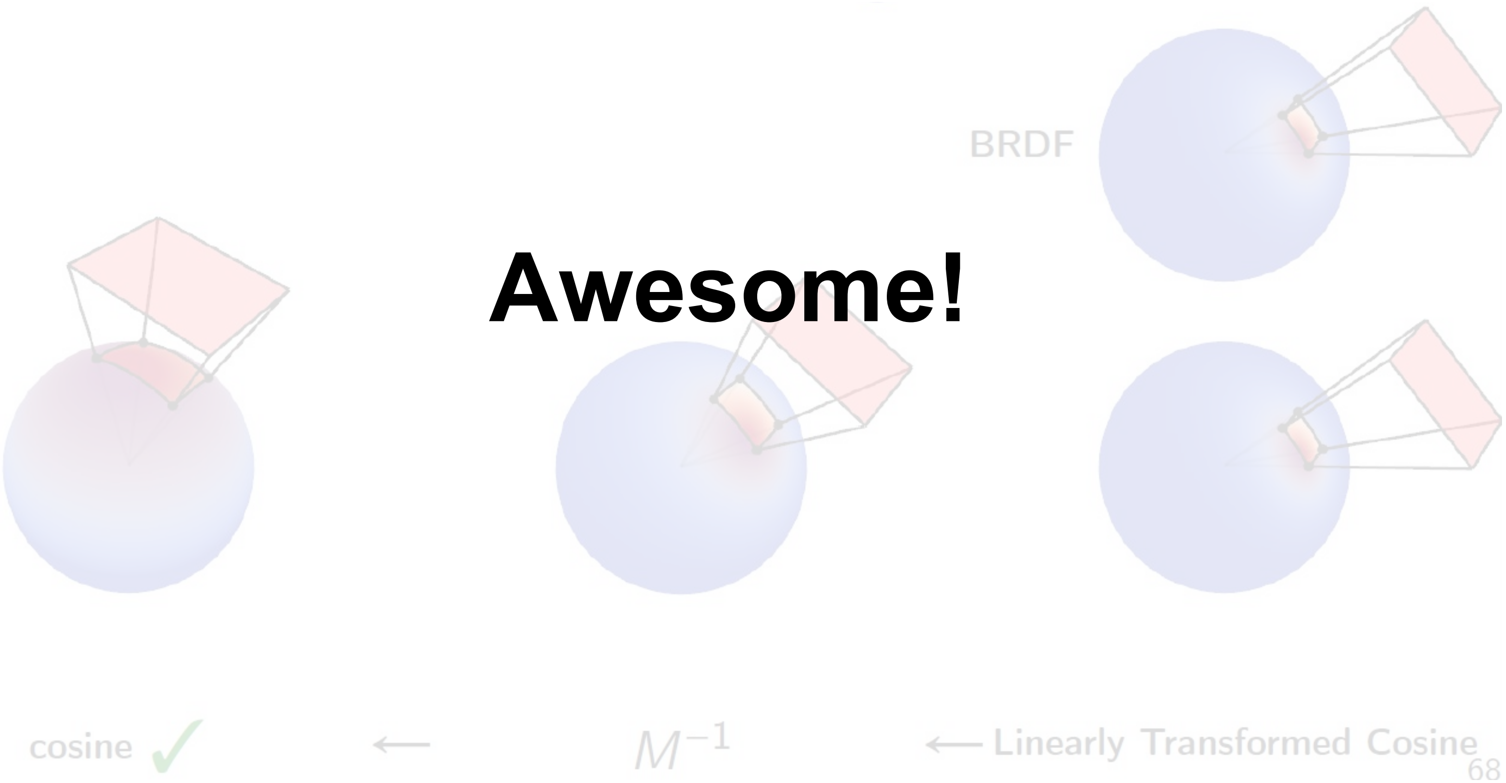
Sum Up Everything

LTC: approximates BRDF

Cosine: easy to integrate



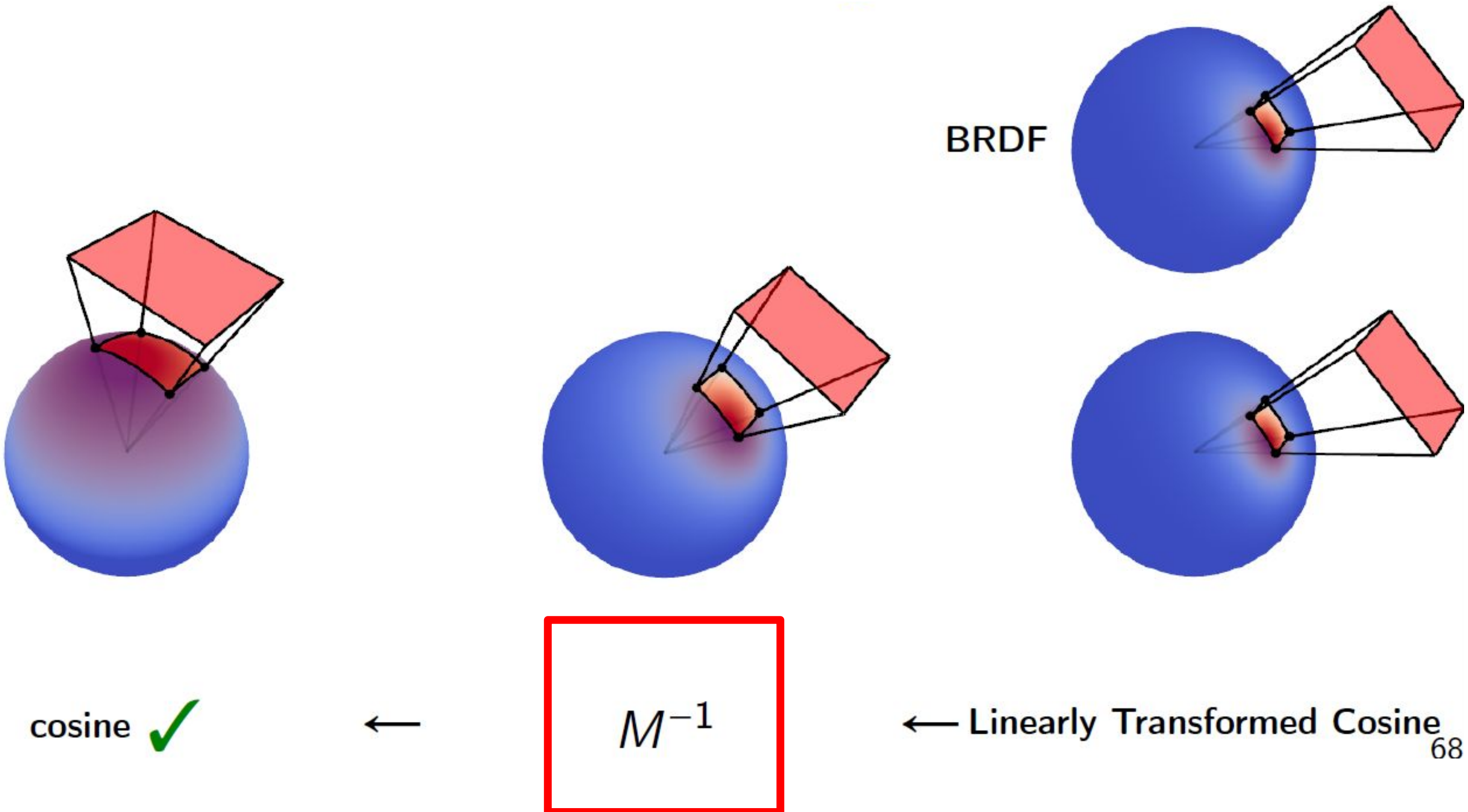
Awesome!



Summary: Main Benefits of Using LTCs

1. Cover important characteristics of BRDFs: anisotropy and skewness
2. Integration can be done in real time

One Dirty Thing Is...



BRDF Fitting: Somewhat Hand-wavy….

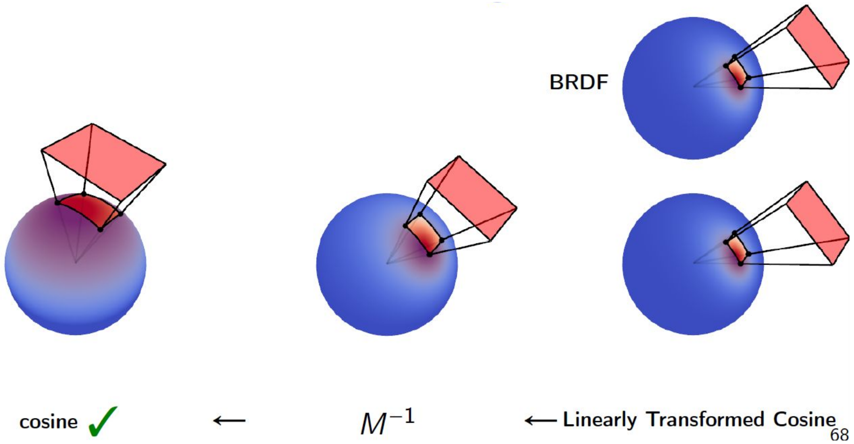
Best-fit LTC: minimize L3 norm → the best visual results

$$D(\omega_i) \approx f_r(\omega_i, \omega_o) \cos\theta_i$$

Manually precompute a set of LTCs with varying roughness and ω_o

Overall Procedure

1. Before runtime,
   a. Find M and LTC provides the best fit
   b. Store Ms
2. Runtime,
   a. For each material(BRDF) and polygonal light,
   b. Inverse transform the polygon
   c. Compute the integral of Clamped Cosine over inverse-transformed polygon



LTCs to the rendering equation

Shading with Constant Polygonal Lights

$$\int_P L(\omega_i) f_r(\omega_i, \omega_o) \cos\theta_i d\omega_i$$

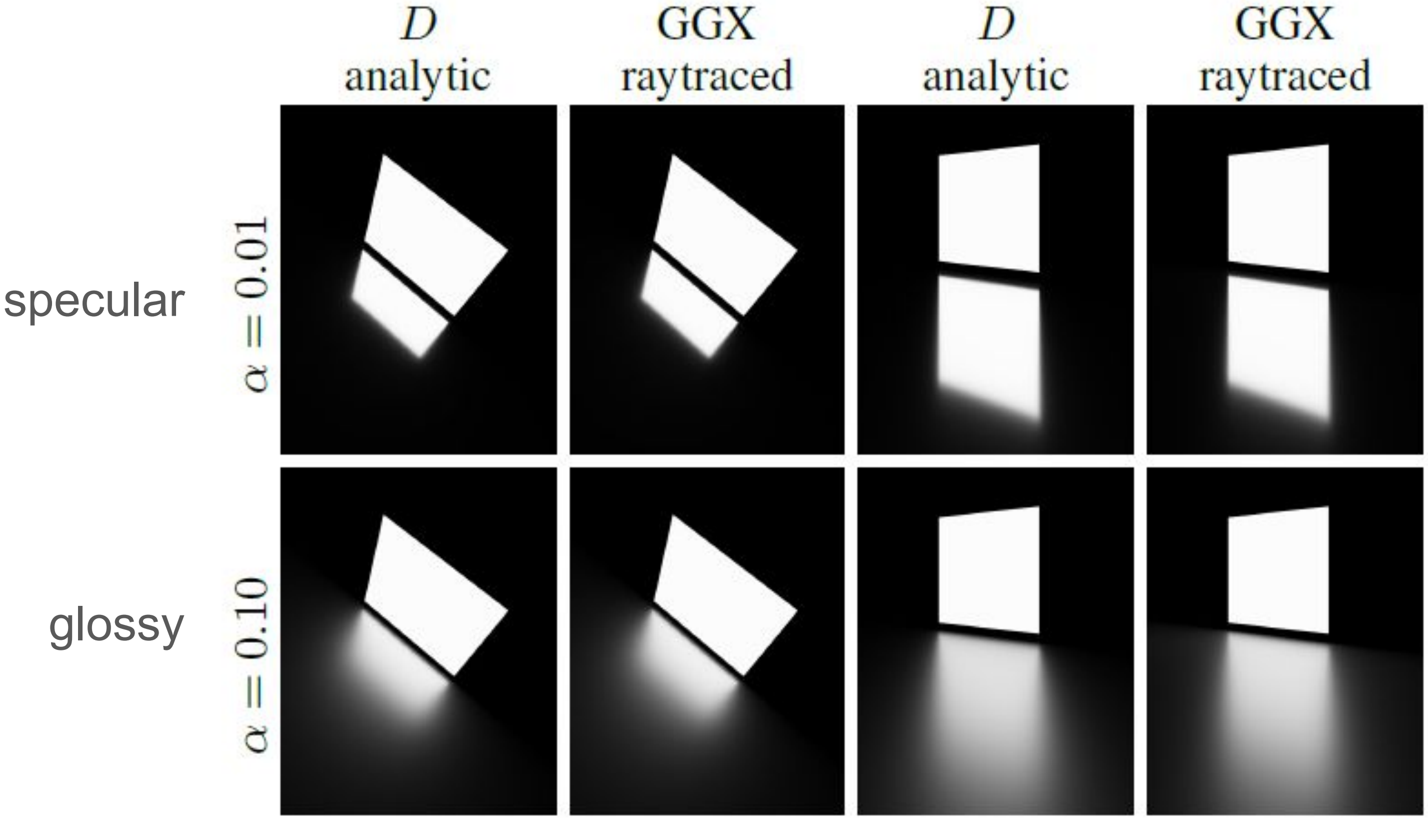
$$L(\omega) = L$$

$$= L \int_P f_r(\omega_i, \omega_o) \cos\theta_i d\omega_i$$

$$\approx L \int_P D(\omega_i) d\omega_i$$

$$= L \int_{P_0} D_0(\omega_0) d\omega_0$$

Shading with Constant Polygonal Lights



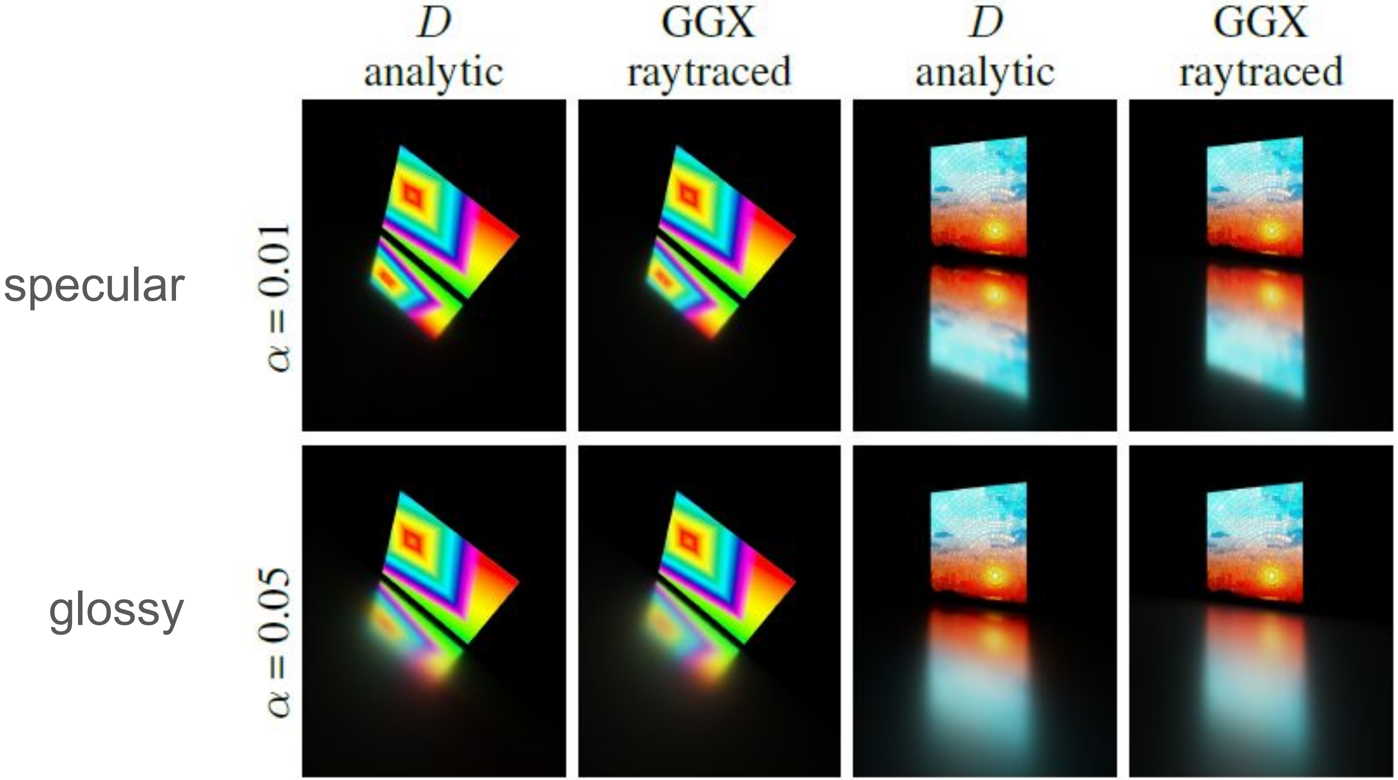
Shading with Textured Polygonal Lights

$$\int_P L(\omega_i) f_r(\omega_i, \omega_o) \cos\theta_i d\omega_i$$

Texture prefiltering

Texture fetching

Shading with Textured Polygonal Lights



Game Engine Integration

Conclusion

Conclusion