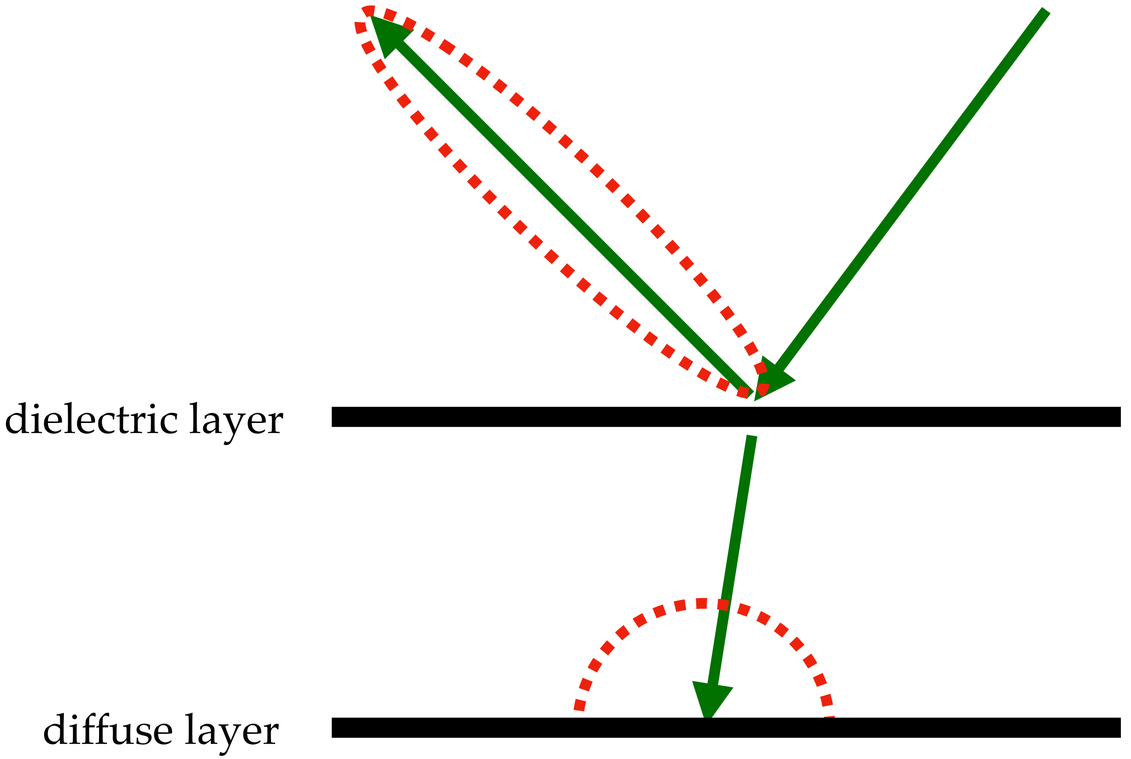
Layered BSDFs

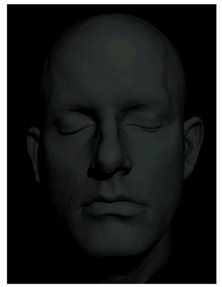
UCSD CSE 272
Advanced Image Synthesis

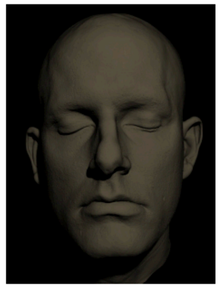
Tzu-Mao Li

Real-world materials are **multi-layered**

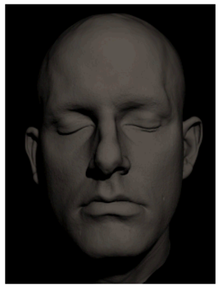


Real-world materials are **multi-layered**


Epidermis Reflectance


Epidermis Transmittance

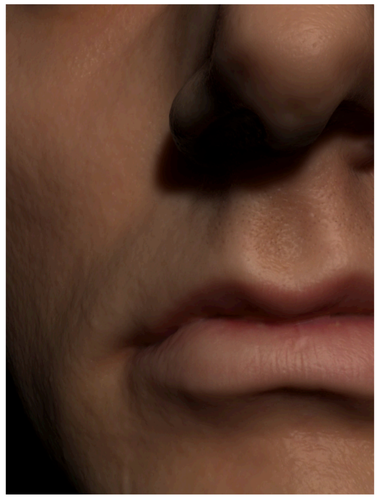

Upper Dermis Reflectance

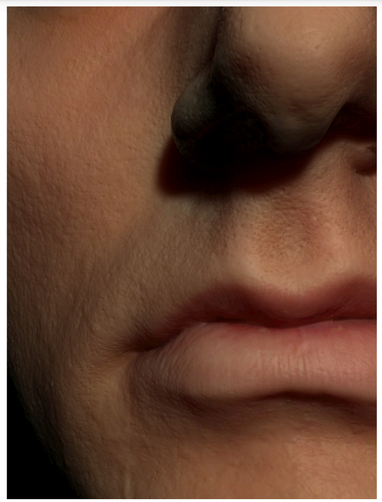

Upper Dermis Transmittance


Bloody Dermis Reflectance

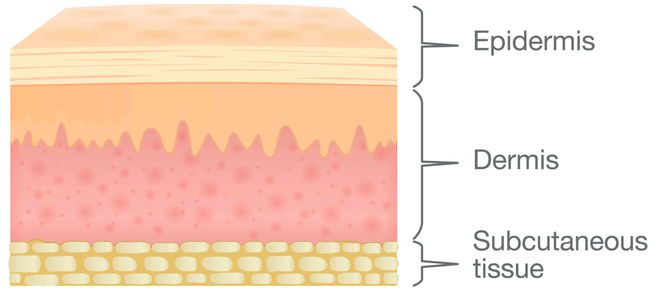

Surface Roughness


All Layers





single-layer multi-layer



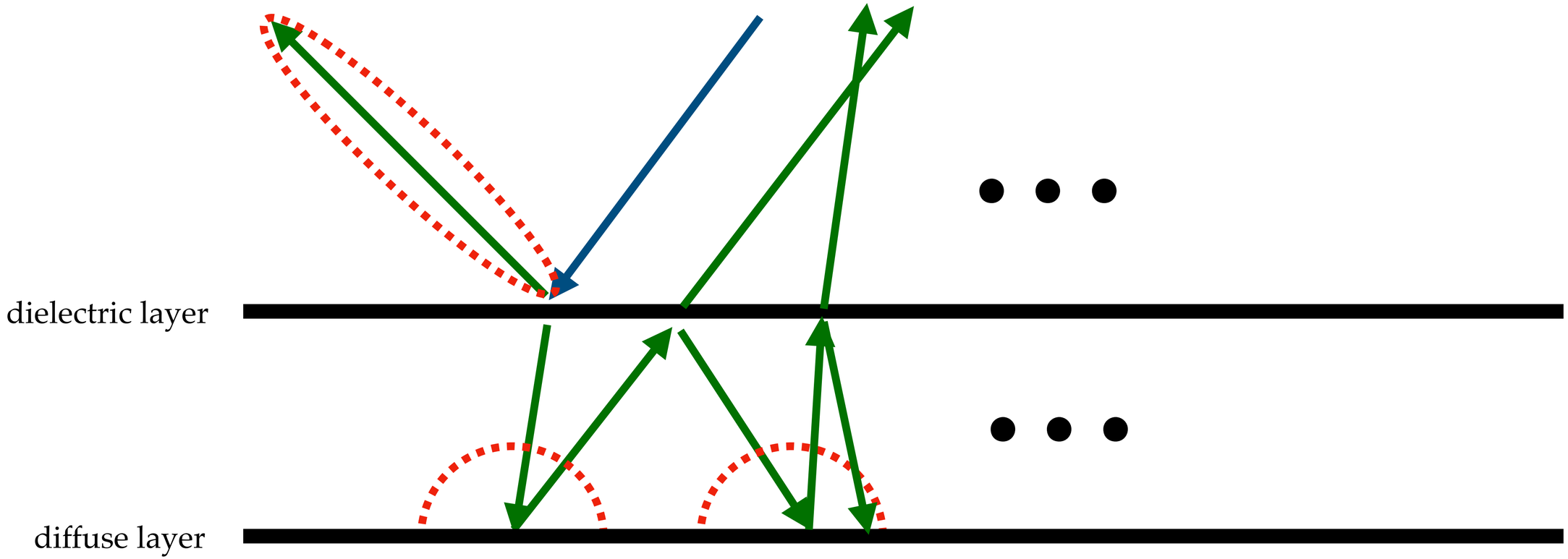
https://fldscc.com/2020/01/30/three-layers-skin/

Light Diffusion in Multi-Layered Translucent Materials

Craig Donner Henrik Wann Jensen

University of California, San Diego

Naive combination of BSDF lobes does not accurately model scattering between interfaces



Why is wet material darker?

(rendering)

(photo)

image credit
https://rnrtires.com/tips-guides/top-tips-for-driving-on-wet-roads/

Rendering of Wet Materials

Henrik Wann Jensen
Justin Legakis
Julie Dorsey

Massachusetts Institute of Technology

Why is wet material darker?

A thin water film on a Lambertian surface can also cause the surface to become darker [10]. The main cause for this darkening is the possibility of total internal reflection at the water-air boundary. Some of the light reflected from the Lambertian surface will be reflected back to the surface by the water-air interface. This light is then subject to another round of absorption by the surface (see Figure 1) before it is reflected again. This can lead to a sequence of multiple absorptions, resulting in a darkening of the surface.

Rendering of Wet Materials

Henrik Wann Jensen
Justin Legakis
Julie Dorsey

Massachusetts Institute of Technology

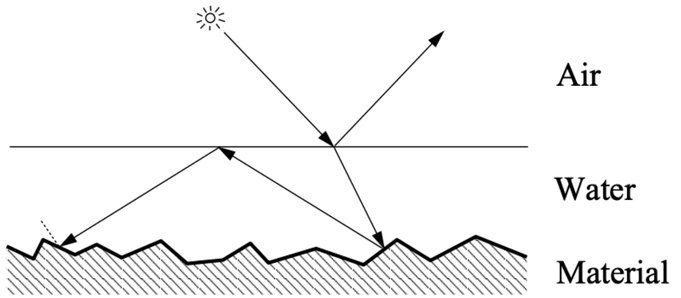
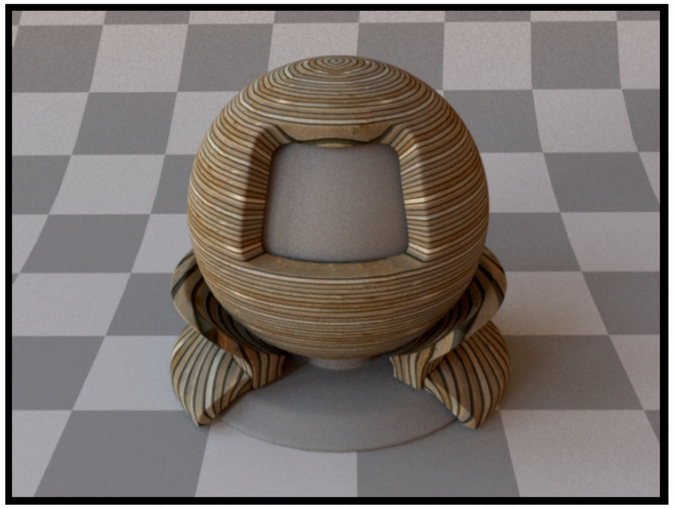
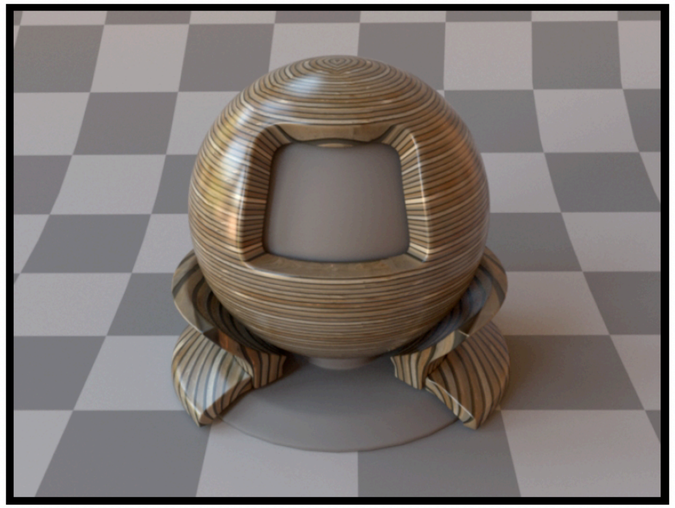


Fig. 1. A layer of water above the surface reflects less light due to the internal reflection at the water-air interface.

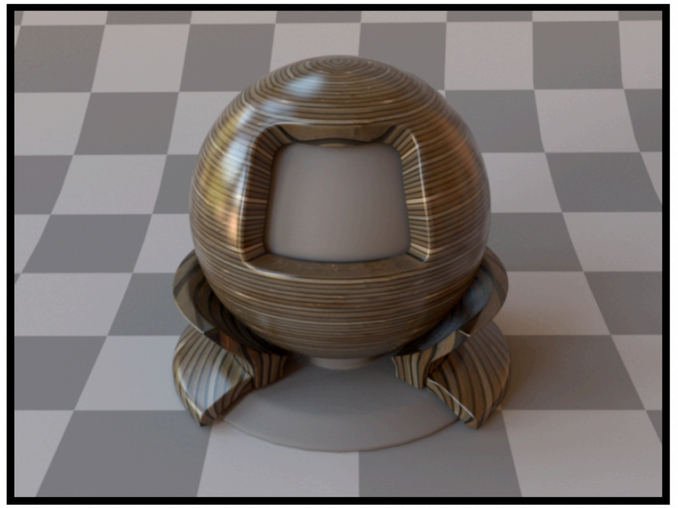
Multiple-scattering causes non-linear shift on reflectance



diffuse only

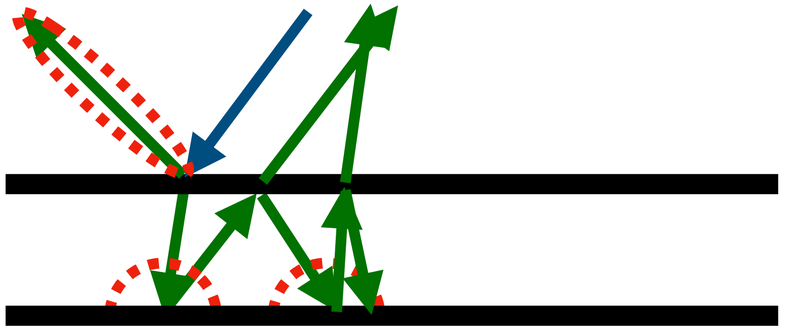


diffuse + specular lobes



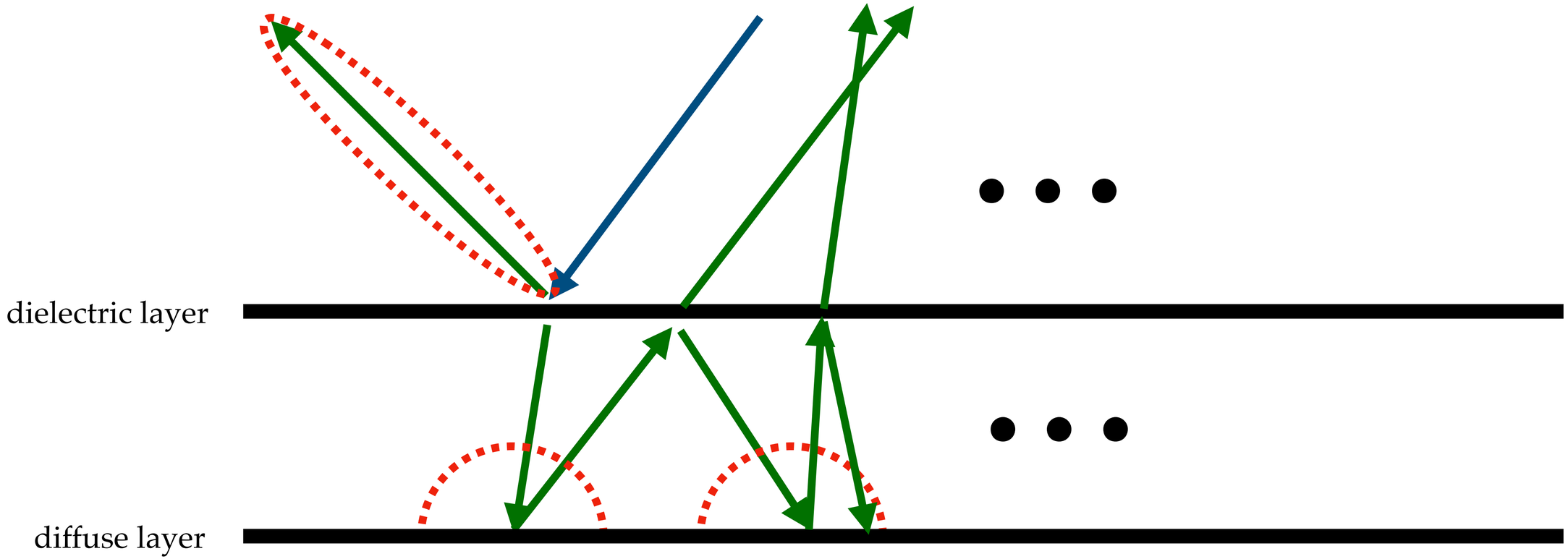
multiple scattering
between diffuse & specular

quiz: what is the consequence of this to the end users of the renderer?



from Mitsuba documentation
https://www.mitsuba-renderer.org/releases/current/documentation.pdf

How would you model the internal scattering?



Today: 3 approaches of modeling layered BSDFs

A Comprehensive Framework for Rendering Layered Materials

Wenzel Jakob Eugene D'Eon Otto Jakob Steve Marschner

In *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2014)*

Efficient Rendering of Layered Materials using an Atomic Decomposition with Statistical Operators

Laurent Belcour (Unity Technologies)

Published in ACM Transactions on Graphics (proc. of SIGGRAPH 2018)

paper bib paper code video slides

Position-Free Monte Carlo Simulation for Arbitrary Layered BSDFs

Yu Guo[1], Miloš Hašan[2], Shuang Zhao[1]
[1]University of California, Irvine [2]Autodesk

ACM Transactions on Graphics (SIGGRAPH Asia 2018), 37(6), 2018

Today: 3 approaches of modeling layered BSDFs

A Comprehensive Framework for Rendering Layered Materials

Wenzel Jakob Eugene D'Eon Otto Jakob Steve Marschner

In *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2014)*

Efficient Rendering of Layered Materials using an Atomic Decomposition with Statistical Operators

Laurent Belcour (Unity Technologies)

Published in ACM Transactions on Graphics (proc. of SIGGRAPH 2018)

paper bib paper code video slides

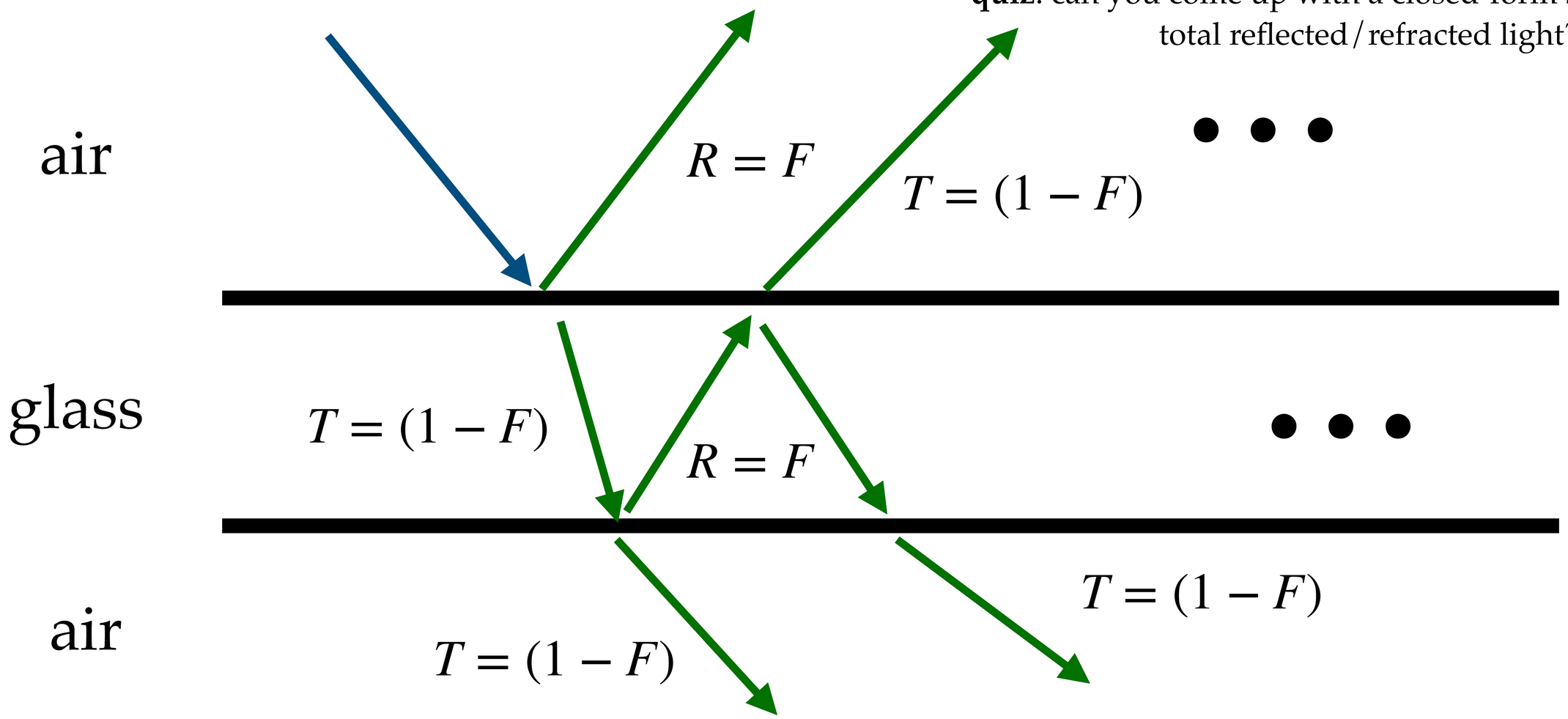
Position-Free Monte Carlo Simulation for Arbitrary Layered BSDFs

Yu Guo[1], Miloš Hašan[2], Shuang Zhao[1]
[1]University of California, Irvine [2]Autodesk

ACM Transactions on Graphics (SIGGRAPH Asia 2018), 37(6), 2018

Special case: pile of glass plates [Stokes 1860]

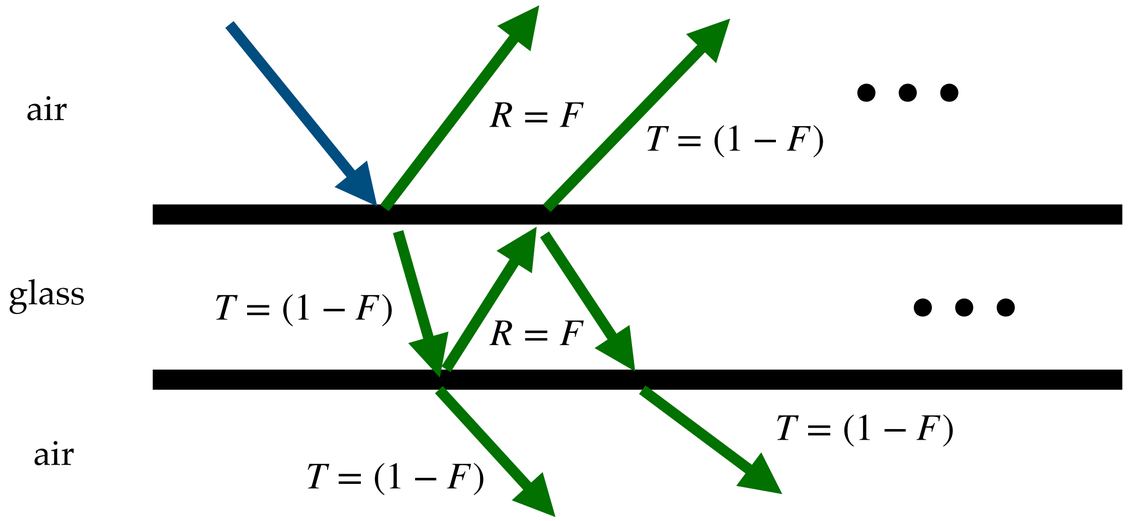
quiz: can you come up with a closed-form solution of the total reflected / refracted light?



Special case: pile of glass plates [Stokes 1860]

$$R_{\text{total}} = R + TRT + TR^3T + \cdots = R + T^2 \sum_{i=0}^{\infty} R^{2i+1} = R + \frac{RT^2}{1 - R^2}$$

$$T_{\text{total}} = TT + TR^2T + TR^4T + \cdots = T^2 \sum_{i=0}^{\infty} R^{2i} = \frac{T^2}{1 - R^2}$$

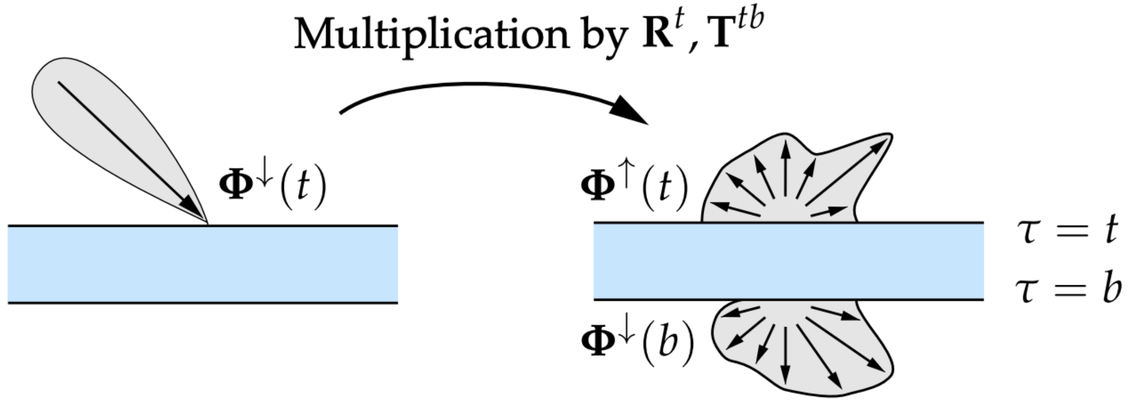


General case: adding equation [Grant and Hunt 1969]

- replace Fresnel equations with matrix vector multiplications

$$\mathbf{\Phi}^{\uparrow}(t) = \mathbf{R}^{t}\mathbf{\Phi}^{\downarrow}(t) + \mathbf{T}^{bt}\mathbf{\Phi}^{\uparrow}(b)$$

$$\mathbf{\Phi}^{\downarrow}(b) = \mathbf{R}^{b}\mathbf{\Phi}^{\uparrow}(b) + \mathbf{T}^{tb}\mathbf{\Phi}^{\downarrow}(b)$$



(aka discrete ordinates method https://en.wikipedia.org/wiki/Discrete_ordinates_method)

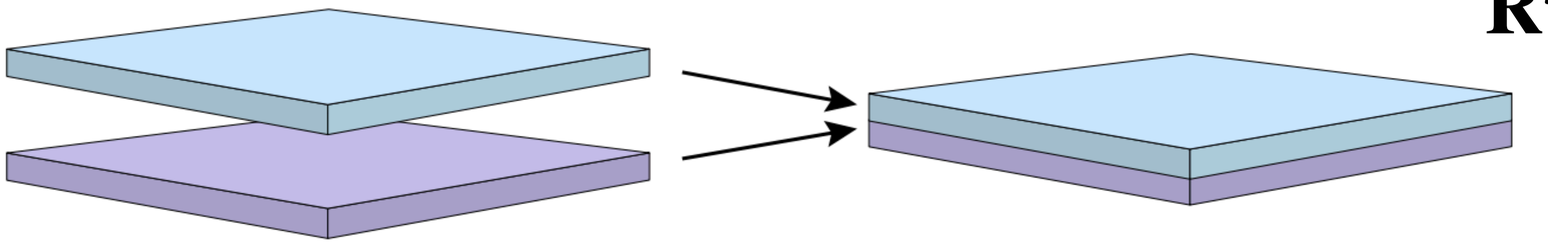
Figure from Wenzel Jakob et al.
https://rgl.s3.eu-central-1.amazonaws.com/media/papers/Jakob2014Comprehensive_1.pdf

General case: adding equation [Grant and Hunt 1969]

goal: figure out the aggregate reflectance/transmittance when we stack layers

$\mathbf{R}_1^t, \mathbf{T}_1^{bt}, \mathbf{R}_1^b, \mathbf{T}_1^{tb}$

$\mathbf{R}_2^t, \mathbf{T}_2^{bt}, \mathbf{R}_2^b, \mathbf{T}_2^{tb}$



$\tilde{\mathbf{R}}^t, \tilde{\mathbf{T}}^{bt}, \tilde{\mathbf{R}}^b, \tilde{\mathbf{T}}^{tb}$

the air/glass layers are a special case!

Figure from Wenzel Jakob et al.
https://rgl.s3.eu-central-1.amazonaws.com/media/papers/Jakob2014Comprehensive_1.pdf

General case: adding equation [Grant and Hunt 1969]

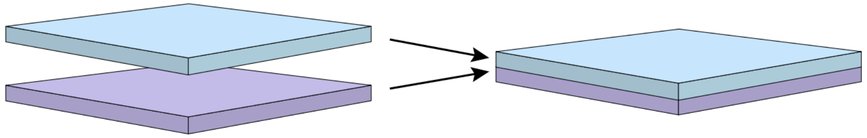
- stacking layer is analogous to summing geometric series for matrices (be careful with non-commutativity of matrix multiplication)

$$\tilde{\mathbf{R}}^t = \mathbf{R}_1^t + \mathbf{T}_1^{bt}(I - \mathbf{R}_2^t\mathbf{R}_1^b)^{-1}\mathbf{R}_2^t\mathbf{T}_1^{tb}$$

$$\tilde{\mathbf{R}}^b = \mathbf{R}_1^b + \mathbf{T}_1^{tb}(I - \mathbf{R}_2^b\mathbf{R}_1^t)^{-1}\mathbf{R}_2^b\mathbf{T}_1^{bt}$$

$$\tilde{\mathbf{T}}^{tb} = \mathbf{T}_2^{tb}\left(I - \mathbf{R}_1^b\mathbf{R}_2^t\right)^{-1}\mathbf{T}_1^{tb}$$

$$\tilde{\mathbf{T}}^{bt} = \mathbf{T}_2^{bt}\left(I - \mathbf{R}_2^t\mathbf{R}_1^b\right)^{-1}\mathbf{T}_1^{bt}$$



$$R_{\text{total}} = R + TRT + TR^3T + \cdots = R + T^2\sum_{i=0}^{\infty} R^{2i+1} = R + \frac{RT^2}{1-R^2}$$

$$T_{\text{total}} = TT + TR^2T + TR^4T + \cdots = T^2\sum_{i=0}^{\infty} R^{2i} = \frac{T^2}{1-R^2}$$

Figure from Wenzel Jakob et al.
https://rgl.s3.eu-central-1.amazonaws.com/media/papers/Jakob2014Comprehensive_1.pdf

Adding equation for thick interfaces: adding-doubling method [van de Hulst 1980]

- start from a thin layer, stack the same layer repeatedly

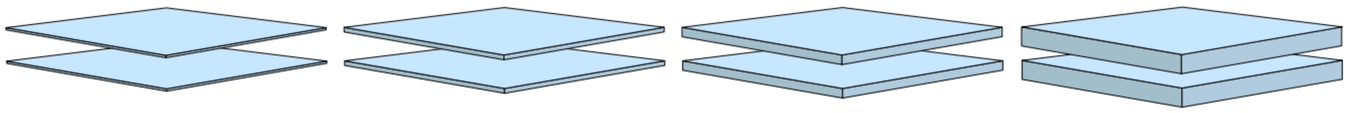
$$\tilde{\mathbf{R}}^t = \mathbf{R}_1^t + \mathbf{T}_1^{bt}(I - \mathbf{R}_2^t\mathbf{R}_1^b)^{-1}\mathbf{R}_2^t\mathbf{T}_1^{tb}$$

$$\tilde{\mathbf{R}}^b = \mathbf{R}_1^b + \mathbf{T}_1^{tb}(I - \mathbf{R}_2^b\mathbf{R}_1^t)^{-1}\mathbf{R}_2^b\mathbf{T}_1^{bt}$$

$$\tilde{\mathbf{T}}^{tb} = \mathbf{T}_2^{tb}\left(I - \mathbf{R}_1^b\mathbf{R}_2^t\right)^{-1}\mathbf{T}_1^{tb}$$

$$\tilde{\mathbf{T}}^{bt} = \mathbf{T}_2^{bt}\left(I - \mathbf{R}_2^t\mathbf{R}_1^b\right)^{-1}\mathbf{T}_1^{bt}$$

the thickness grows exponentially

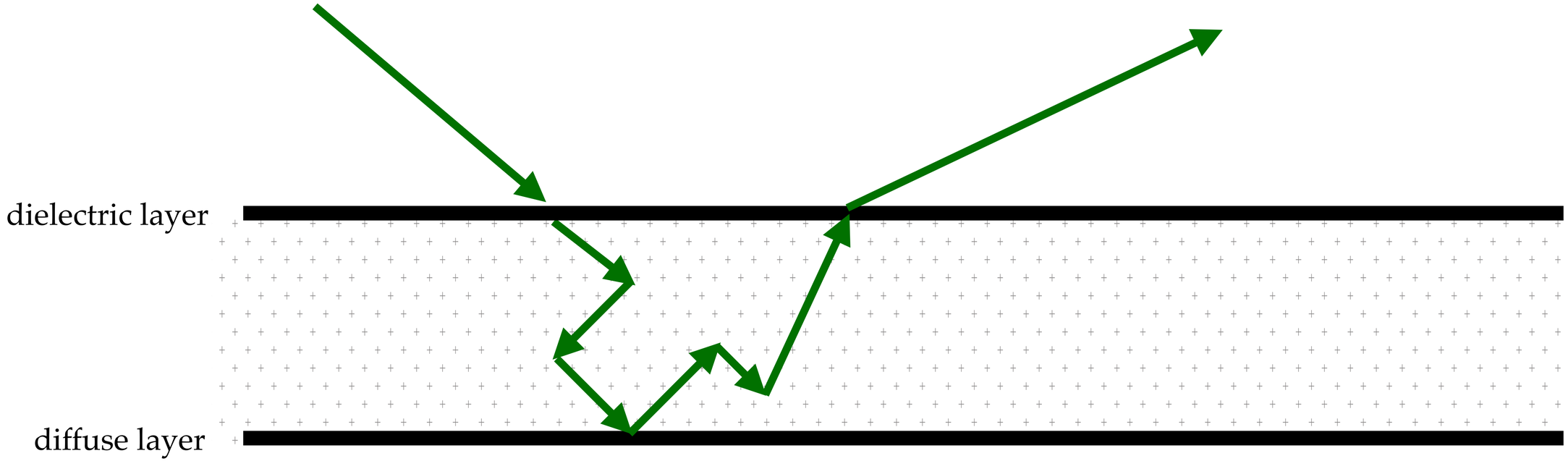


adding-doubling is commonly used for solving light transport in biomedical optics and atmospheric science!

Figure from Wenzel Jakob et al.
https://rgl.s3.eu-central-1.amazonaws.com/media/papers/Jakob2014Comprehensive_1.pdf

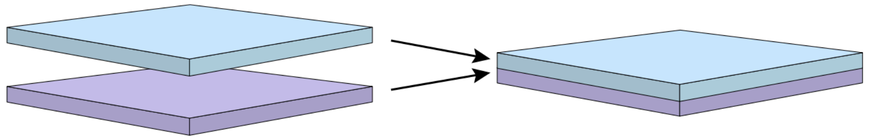
Detail: need to also account for volumetric effects between layers

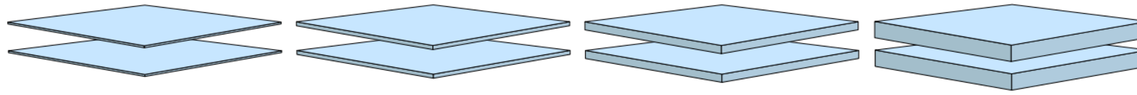
- all methods we talked about today can be easily extended to do this



General layered BSDF construction [Jakob 2014]

- measure R & T of a single (thin) layer (either an analytical BSDF or measured one)
  - store a big matrix for all different angles
- combine different layers using equation below





$$\tilde{\mathbf{R}}^t = \mathbf{R}_1^t + \mathbf{T}_1^{bt}(I - \mathbf{R}_2^t\mathbf{R}_1^b)^{-1}\mathbf{R}_2^t\mathbf{T}_1^{tb}$$

$$\tilde{\mathbf{R}}^b = \mathbf{R}_1^b + \mathbf{T}_1^{tb}(I - \mathbf{R}_2^b\mathbf{R}_1^t)^{-1}\mathbf{R}_2^b\mathbf{T}_1^{bt}$$

$$\tilde{\mathbf{T}}^{tb} = \mathbf{T}_2^{tb}\left(I - \mathbf{R}_1^b\mathbf{R}_2^t\right)^{-1}\mathbf{T}_1^{tb}$$

$$\tilde{\mathbf{T}}^{bt} = \mathbf{T}_2^{bt}\left(I - \mathbf{R}_2^t\mathbf{R}_1^b\right)^{-1}\mathbf{T}_1^{bt}$$

Figure from Wenzel Jakob et al.
https://rgl.s3.eu-central-1.amazonaws.com/media/papers/Jakob2014Comprehensive_1.pdf

Detail: use a Fourier basis to compress the matrices

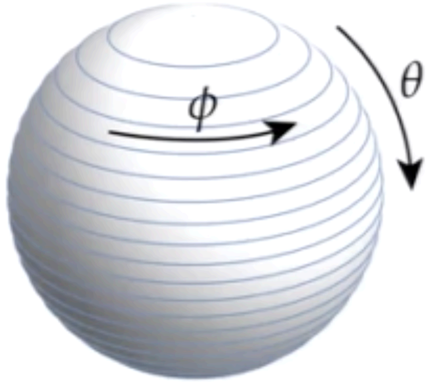
- observation: changes in azimuthal direction are usually low frequency

Chandrasekhar basis

$$L_o(\omega_o) = \int_{S^2} f(\omega_i, \omega_o)\, L_i(\omega_i)\, |\cos\theta_i|\, \mathrm{d}\omega_i$$

Discretize in θ

Fourier transform in ϕ



$$\underset{n\text{-vector}}{\mathbf{L}_o^{(k)}} = \underset{n\times n\text{-matrix}}{\mathbf{F}^{(k)}}\ \underset{n\text{-vector}}{\mathbf{L}_i^{(k)}}$$

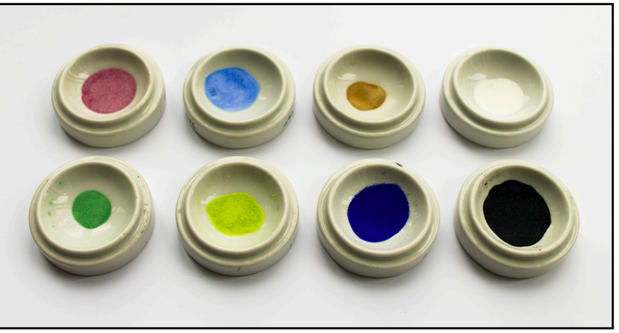
the Fourier basis BSDF is implemented in pbrt
https://www.pbr-book.org/3ed-2018/Reflection_Models/Fourier_Basis_BSDFs

Figure from Wenzel Jakob et al.
https://dl.acm.org/doi/10.1145/2601097.2601139

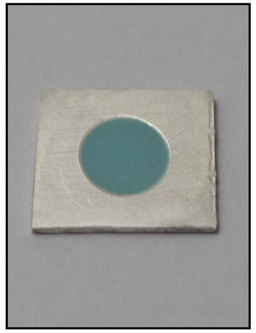
Layered-BSDFs match well with real-world materials!

photo

rendering



(a) A sample of jewelry enamels in powdered form



(b) Enamel test swatch

Figure from Wenzel Jakob et al.
https://rgl.s3.eu-central-1.amazonaws.com/media/papers/Jakob2014Comprehensive_1.pdf

Downside of this method?

$$\tilde{\mathbf{R}}^t = \mathbf{R}_1^t + \mathbf{T}_1^{bt}(I - \mathbf{R}_2^t\mathbf{R}_1^b)^{-1}\mathbf{R}_2^t\mathbf{T}_1^{tb}$$
$$\tilde{\mathbf{R}}^b = \mathbf{R}_1^b + \mathbf{T}_1^{tb}(I - \mathbf{R}_2^b\mathbf{R}_1^t)^{-1}\mathbf{R}_2^b\mathbf{T}_1^{bt}$$
$$\tilde{\mathbf{T}}^{tb} = \mathbf{T}_2^{tb}\left(I - \mathbf{R}_1^b\mathbf{R}_2^t\right)^{-1}\mathbf{T}_1^{tb}$$
$$\tilde{\mathbf{T}}^{bt} = \mathbf{T}_2^{bt}\left(I - \mathbf{R}_2^t\mathbf{R}_1^b\right)^{-1}\mathbf{T}_1^{bt}$$

Downside of the direct implementation of adding equation

- need to store a huge sparse matrix (in Fourier domain)
  - a few megabytes per BSDF
- texturing is hard

$$\tilde{\mathbf{R}}^t = \mathbf{R}_1^t + \mathbf{T}_1^{bt}(I - \mathbf{R}_2^t\mathbf{R}_1^b)^{-1}\mathbf{R}_2^t\mathbf{T}_1^{tb}$$
$$\tilde{\mathbf{R}}^b = \mathbf{R}_1^b + \mathbf{T}_1^{tb}(I - \mathbf{R}_2^b\mathbf{R}_1^t)^{-1}\mathbf{R}_2^b\mathbf{T}_1^{bt}$$
$$\tilde{\mathbf{T}}^{tb} = \mathbf{T}_2^{tb}\left(I - \mathbf{R}_1^b\mathbf{R}_2^t\right)^{-1}\mathbf{T}_1^{tb}$$
$$\tilde{\mathbf{T}}^{bt} = \mathbf{T}_2^{bt}\left(I - \mathbf{R}_2^t\mathbf{R}_1^b\right)^{-1}\mathbf{T}_1^{bt}$$

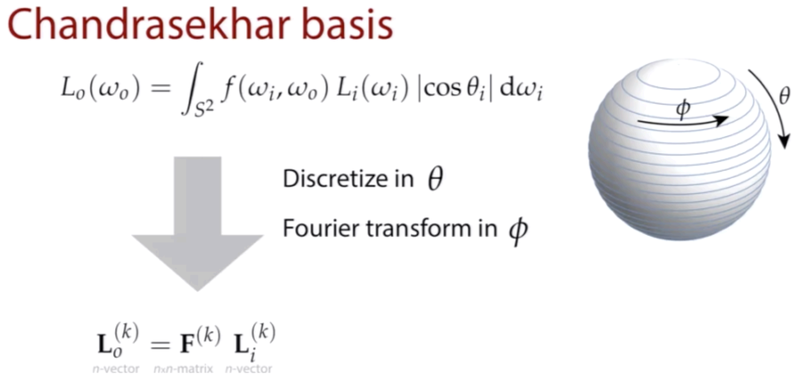


Figure from Wenzel Jakob et al.
https://dl.acm.org/doi/10.1145/2601097.2601139

Next: a more compact representation

A Comprehensive Framework for Rendering Layered Materials

Wenzel Jakob Eugene D'Eon Otto Jakob Steve Marschner

In *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2014)*

Efficient Rendering of Layered Materials using an Atomic Decomposition with Statistical Operators

Laurent Belcour (Unity Technologies)

Published in ACM Transactions on Graphics (proc. of SIGGRAPH 2018)

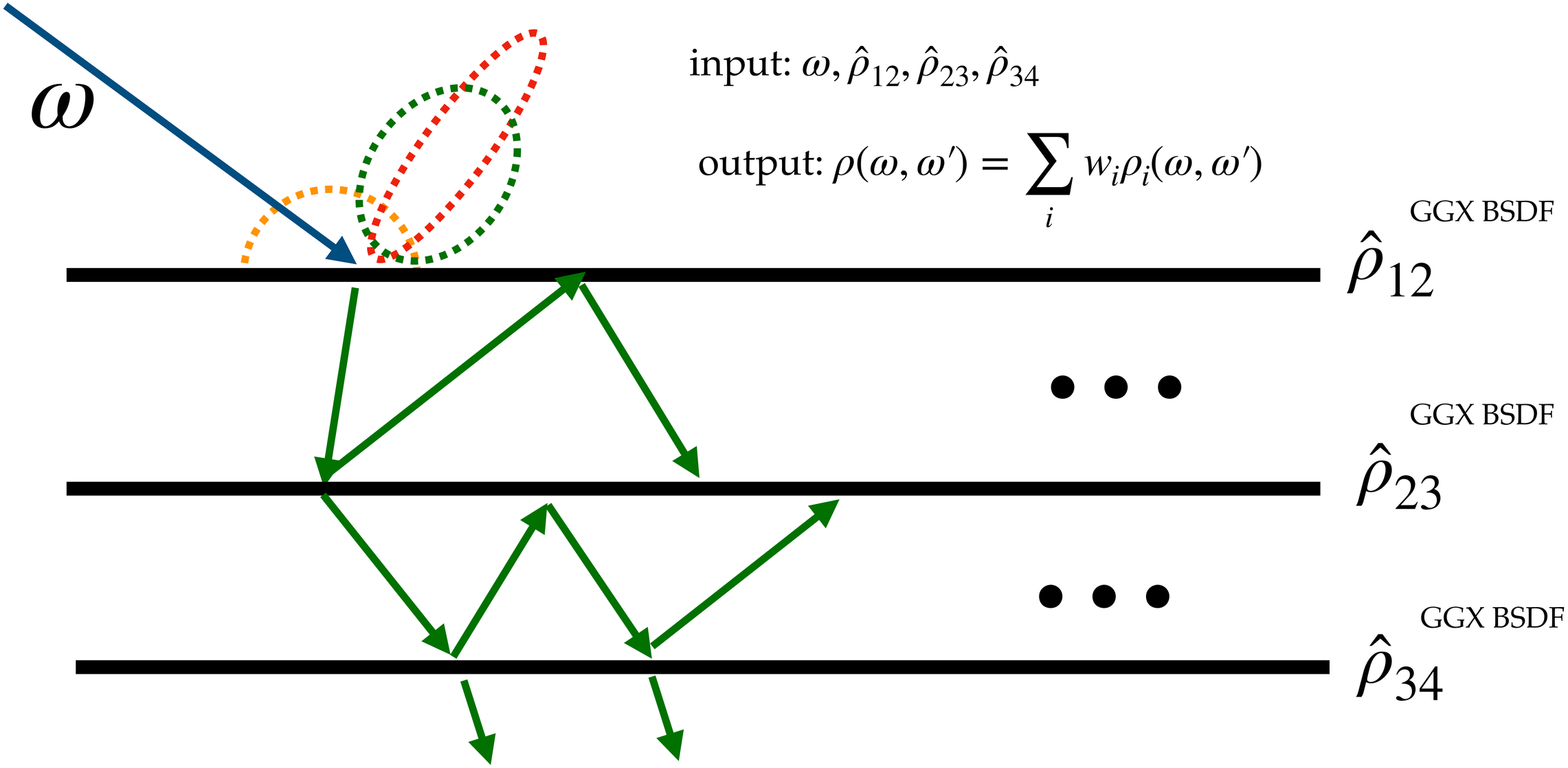
paper bib paper code video slides

Position-Free Monte Carlo Simulation for Arbitrary Layered BSDFs

Yu Guo[1], Miloš Hašan[2], Shuang Zhao[1]
[1]University of California, Irvine [2]Autodesk

ACM Transactions on Graphics (SIGGRAPH Asia 2018), 37(6), 2018

Goal: use sum of GGX lobes to represent multiple scattering between layers

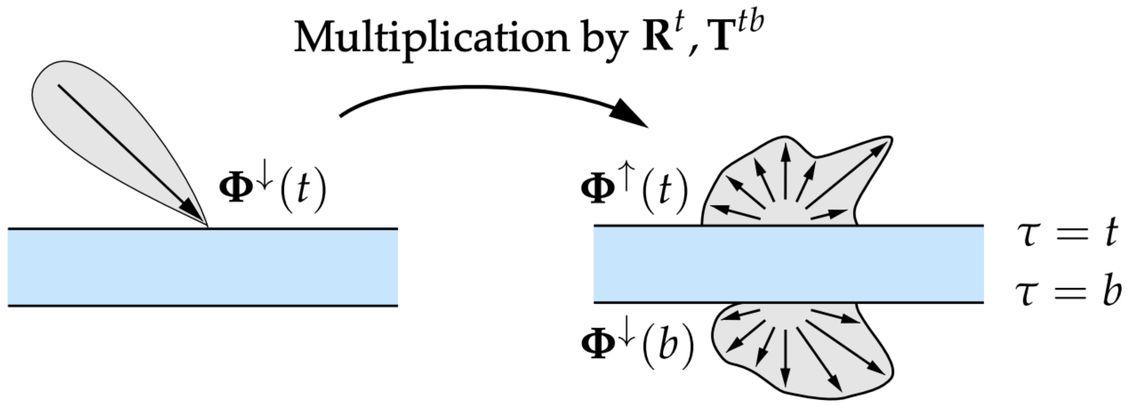


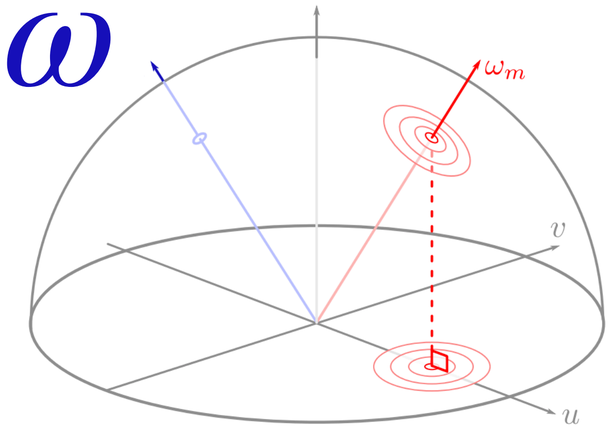
Idea: approximate the R&T matrices using Gaussians for a given direction ω

for a given direction, the matrix vector multiplication falls back to the scalar case

$$\mathbf{\Phi}^{\uparrow}(t) = \mathbf{R}^{t}\mathbf{\Phi}^{\downarrow}(t) + \mathbf{T}^{bt}\mathbf{\Phi}^{\uparrow}(b)$$

$$\mathbf{\Phi}^{\downarrow}(b) = \mathbf{R}^{b}\mathbf{\Phi}^{\uparrow}(b) + \mathbf{T}^{tb}\mathbf{\Phi}^{\downarrow}(b)$$



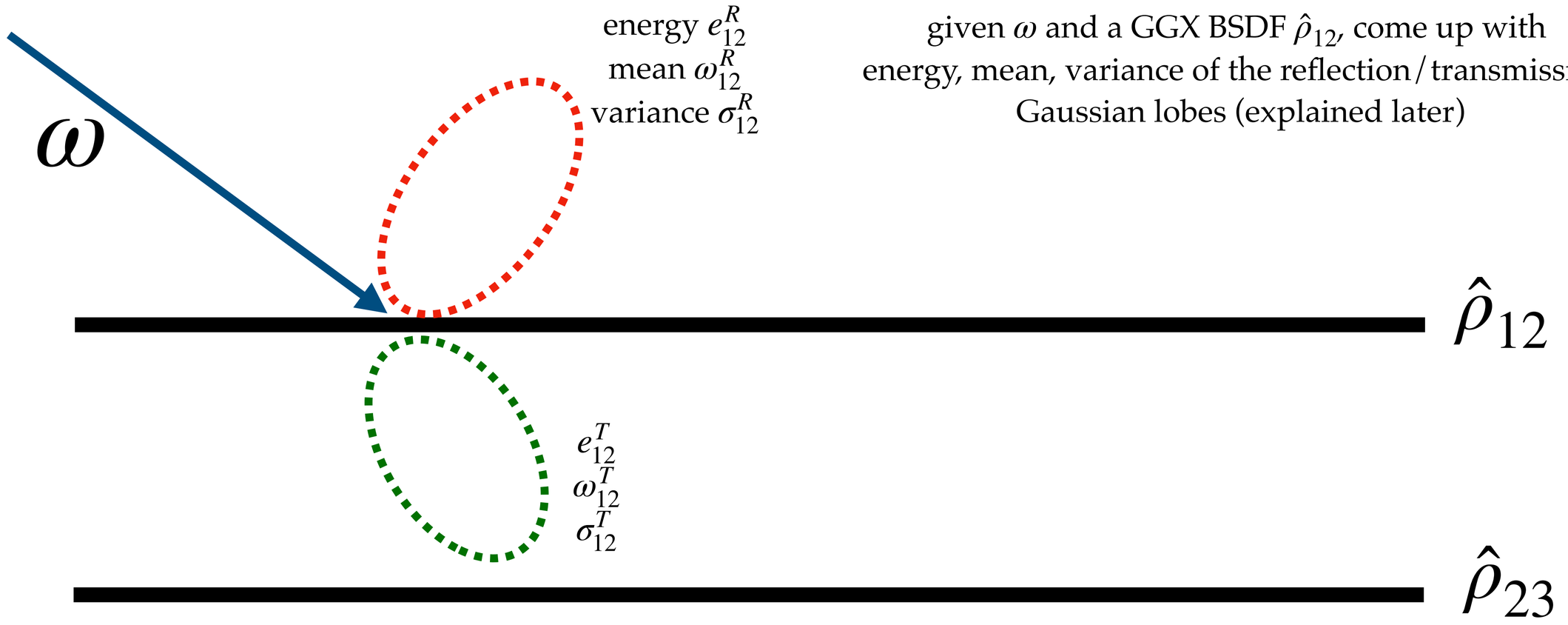


in practice, put the Gaussian in the projected disk (slope space)

input: $\omega, \hat{\rho}_{12}, \hat{\rho}_{23}, \hat{\rho}_{34}$

output: $\rho(\omega, \omega') = \sum_i w_i \rho_i(\omega, \omega')$

Belcour's algorithm



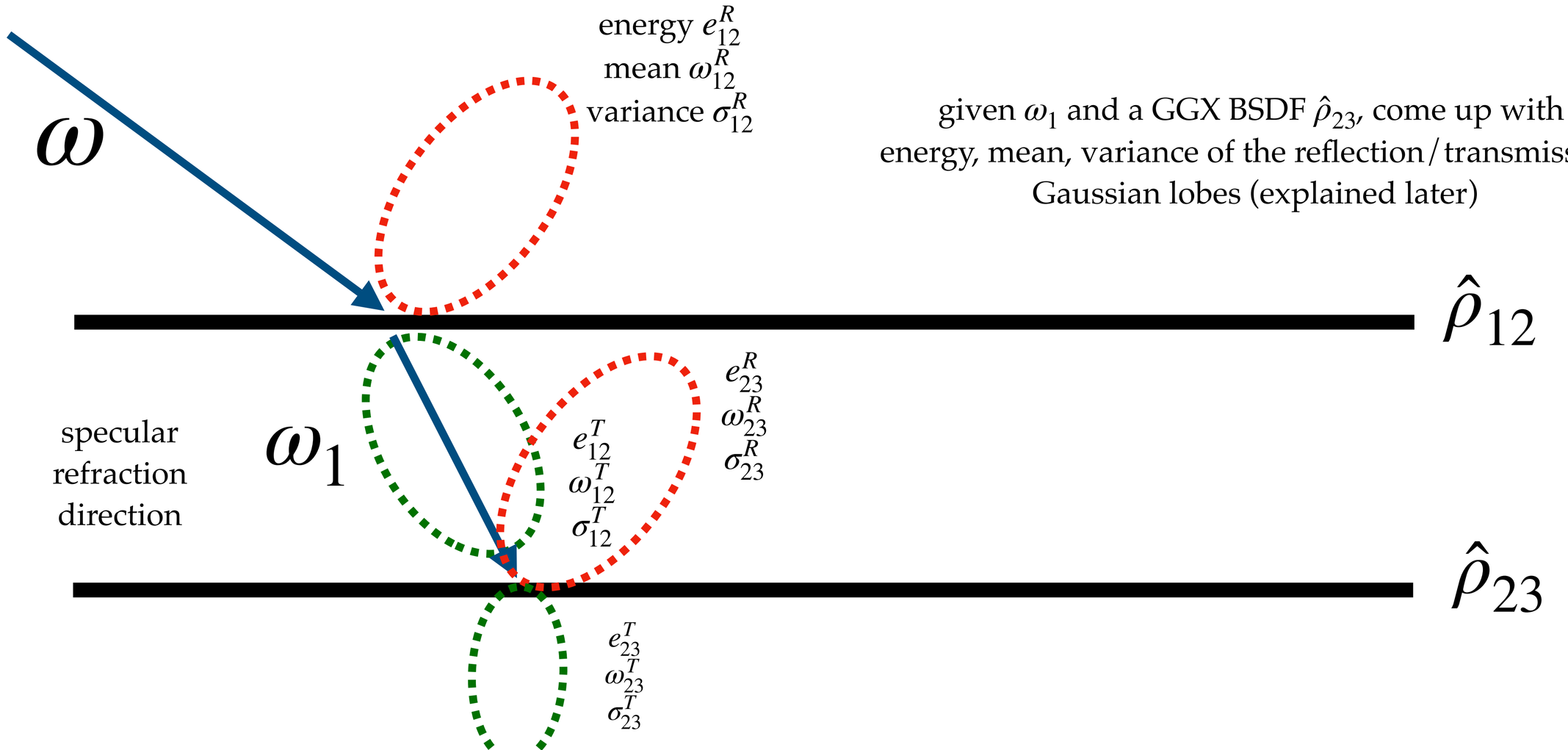
given ω and a GGX BSDF $\hat{\rho}_{12}$, come up with energy, mean, variance of the reflection/transmission Gaussian lobes (explained later)

input: $\omega, \hat{\rho}_{12}, \hat{\rho}_{23}, \hat{\rho}_{34}$

output: $\rho(\omega, \omega') = \sum_i w_i \rho_i(\omega, \omega')$

Belcour's algorithm

$e_{12}^R \rho(\omega, \omega')$ with roughness $r = f_R(\sigma_{12}^R)$



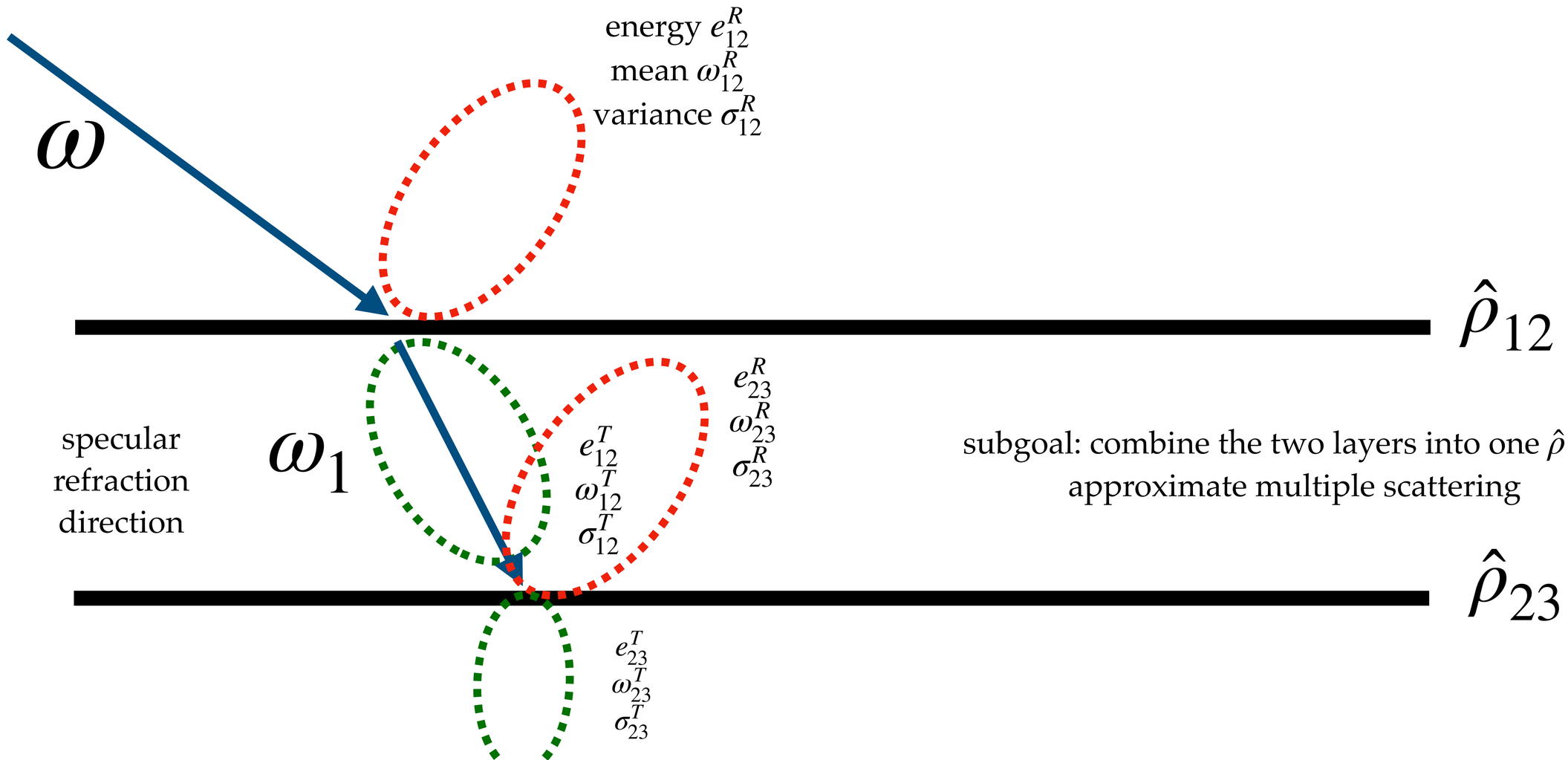
given ω_1 and a GGX BSDF $\hat{\rho}_{23}$, come up with energy, mean, variance of the reflection/transmission Gaussian lobes (explained later)

input: $\omega, \hat{\rho}_{12}, \hat{\rho}_{23}, \hat{\rho}_{34}$

output: $\rho(\omega, \omega') = \sum_i w_i \rho_i(\omega, \omega')$

Belcour's algorithm

$e_{12}^R \rho(\omega, \omega')$ with roughness $r = f_R(\sigma_{12}^R)$

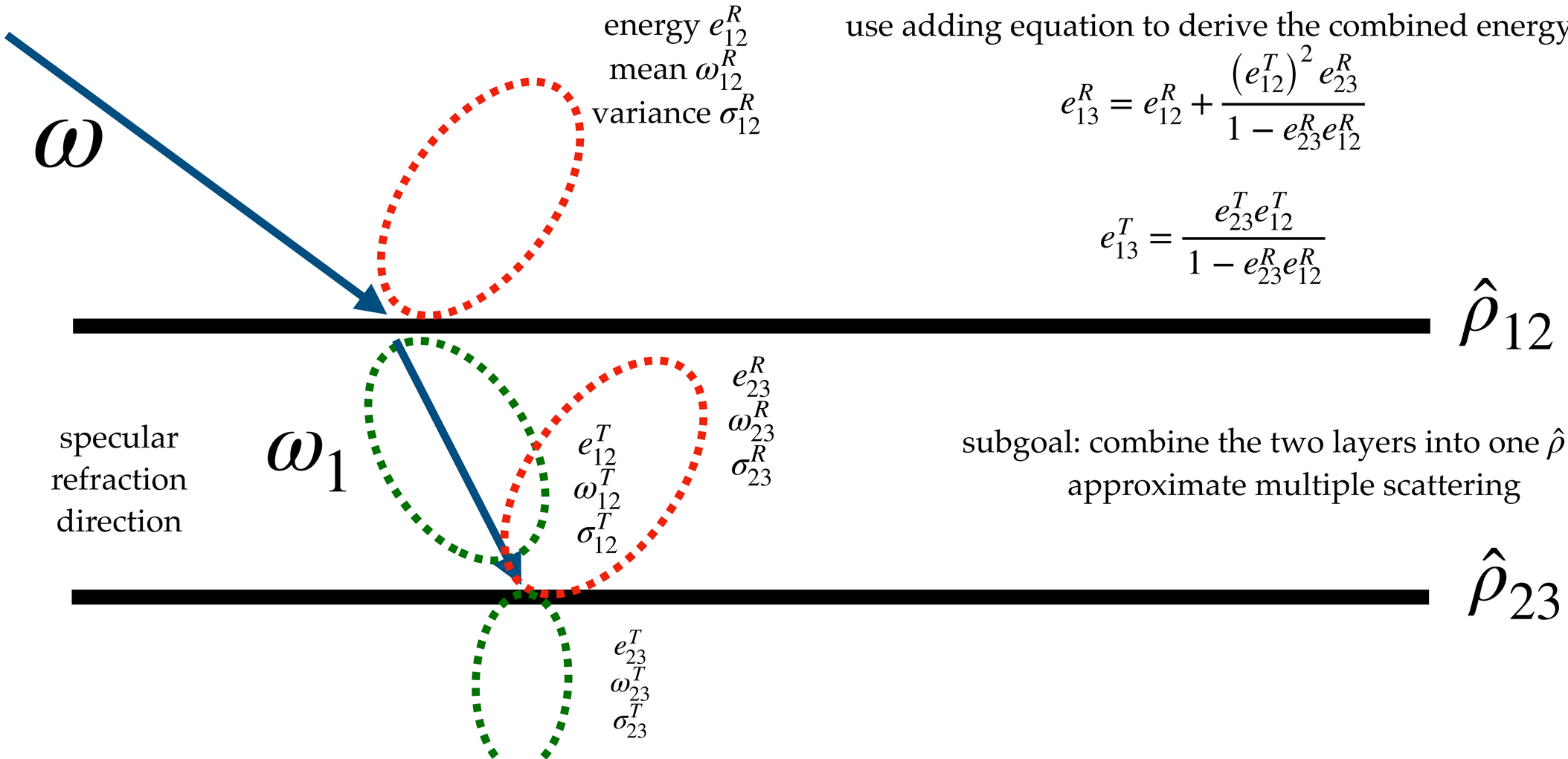


input: $\omega, \hat{\rho}_{12}, \hat{\rho}_{23}, \hat{\rho}_{34}$

output: $\rho(\omega, \omega') = \sum_i w_i \rho_i(\omega, \omega')$

Belcour's algorithm

$e_{12}^R \rho(\omega, \omega')$ with roughness $r = f_R(\sigma_{12}^R)$



use adding equation to derive the combined energy:

$$e_{13}^R = e_{12}^R + \frac{\left(e_{12}^T\right)^2 e_{23}^R}{1 - e_{23}^R e_{12}^R}$$

$$e_{13}^T = \frac{e_{23}^T e_{12}^T}{1 - e_{23}^R e_{12}^R}$$

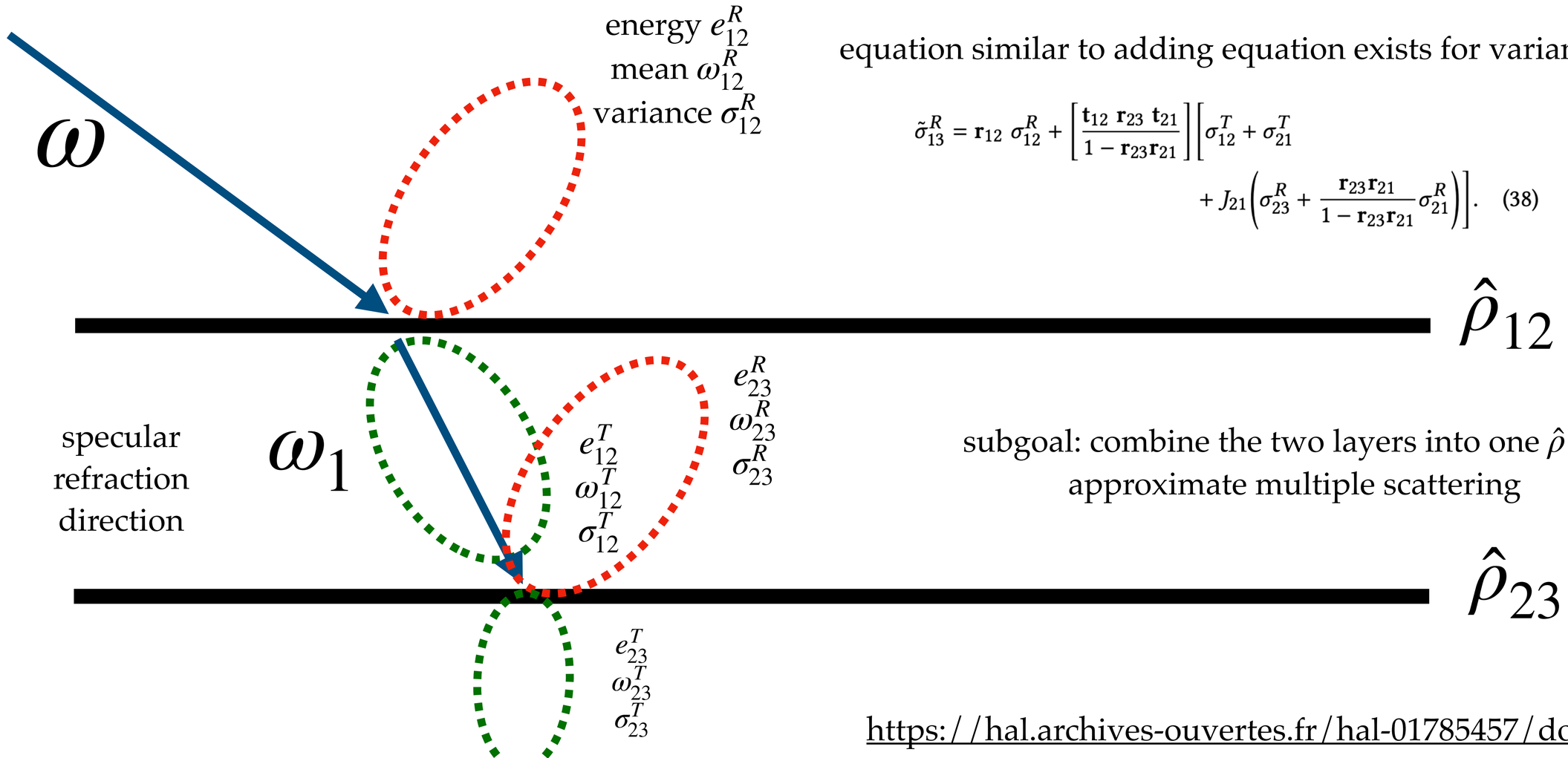
subgoal: combine the two layers into one $\hat{\rho}_{13}$ to approximate multiple scattering

input: $\omega, \hat{\rho}_{12}, \hat{\rho}_{23}, \hat{\rho}_{34}$

output: $\rho(\omega, \omega') = \sum_i w_i \rho_i(\omega, \omega')$

Belcour's algorithm

$e_{12}^R \rho(\omega, \omega')$ with roughness $r = f_R(\sigma_{12}^R)$



equation similar to adding equation exists for variance

$$\tilde{\sigma}_{13}^R = \mathbf{r}_{12}\,\sigma_{12}^R + \left[\frac{\mathbf{t}_{12}\,\mathbf{r}_{23}\,\mathbf{t}_{21}}{1-\mathbf{r}_{23}\mathbf{r}_{21}}\right]\left[\sigma_{12}^T + \sigma_{21}^T + J_{21}\left(\sigma_{23}^R + \frac{\mathbf{r}_{23}\mathbf{r}_{21}}{1-\mathbf{r}_{23}\mathbf{r}_{21}}\sigma_{21}^R\right)\right]. \quad (38)$$

subgoal: combine the two layers into one $\hat{\rho}_{13}$ to approximate multiple scattering

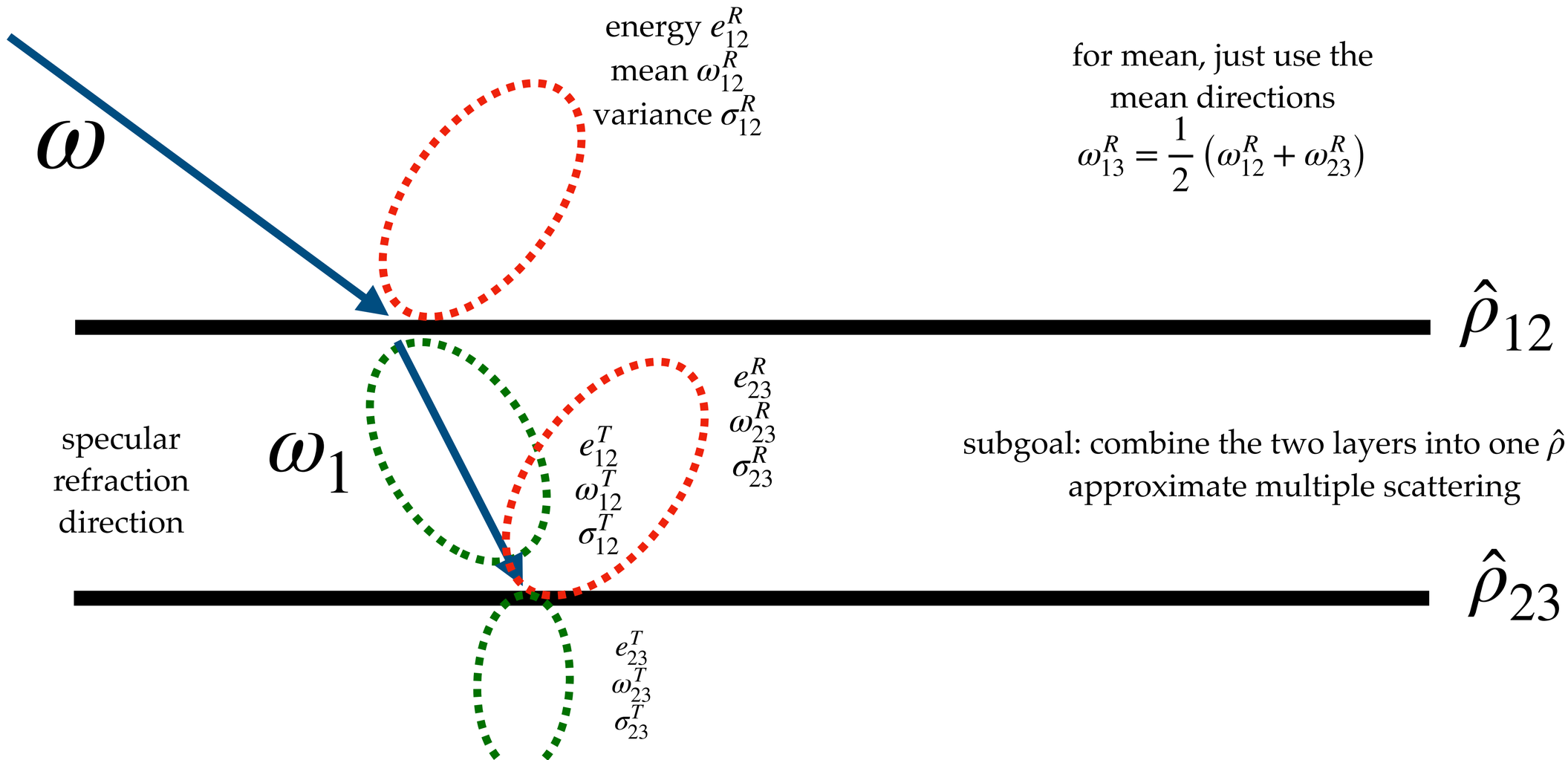
https://hal.archives-ouvertes.fr/hal-01785457/document

input: $\omega, \hat{\rho}_{12}, \hat{\rho}_{23}, \hat{\rho}_{34}$

output: $\rho(\omega, \omega') = \sum_i w_i \rho_i(\omega, \omega')$

Belcour's algorithm

$e_{12}^R \rho(\omega, \omega')$ with roughness $r = f_R(\sigma_{12}^R)$



for mean, just use the mean directions

$$\omega_{13}^R = \frac{1}{2}\left(\omega_{12}^R + \omega_{23}^R\right)$$

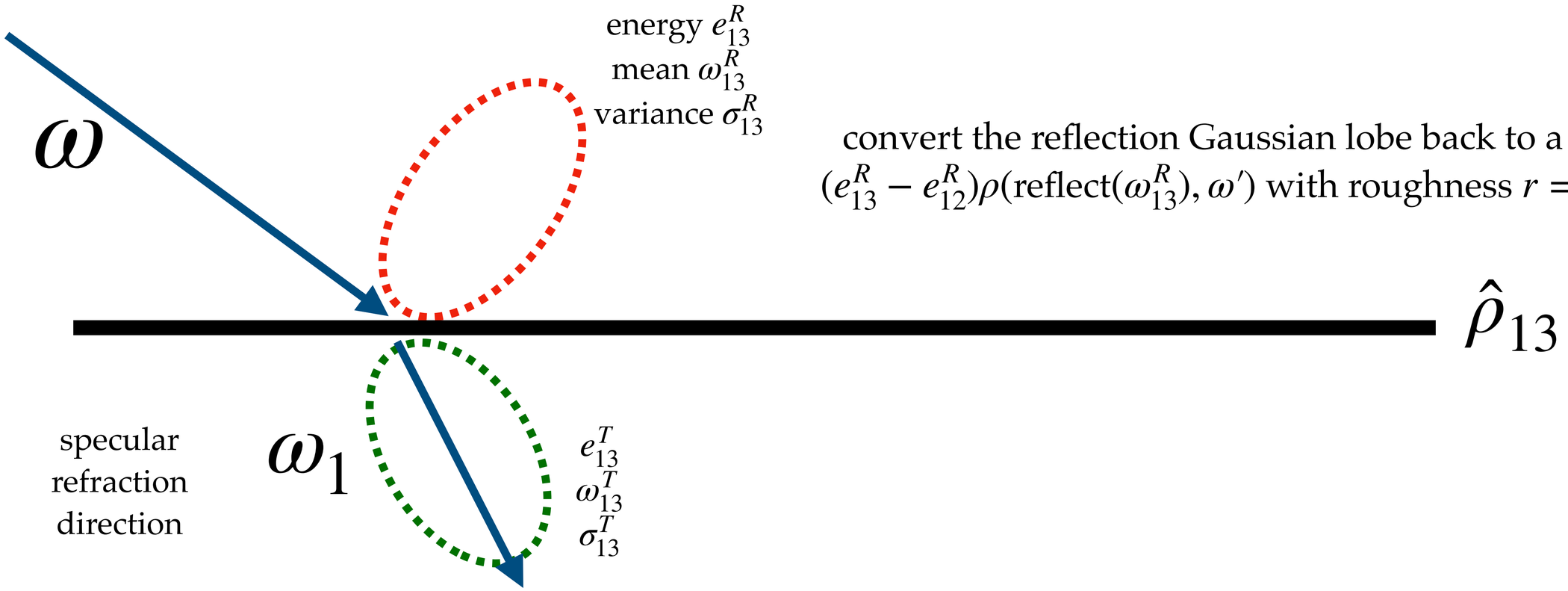
subgoal: combine the two layers into one $\hat{\rho}_{13}$ to approximate multiple scattering

input: $\omega, \hat{\rho}_{12}, \hat{\rho}_{23}, \hat{\rho}_{34}$

output: $\rho(\omega, \omega') = \sum_i w_i \rho_i(\omega, \omega')$

Belcour's algorithm

$e_{12}^R \rho(\omega, \omega')$ with roughness $r = f_R(\sigma_{12}^R)$



convert the reflection Gaussian lobe back to a GGX

$(e_{13}^R - e_{12}^R)\rho(\text{reflect}(\omega_{13}^R), \omega')$ with roughness $r = f(\sigma_{13}^R)$

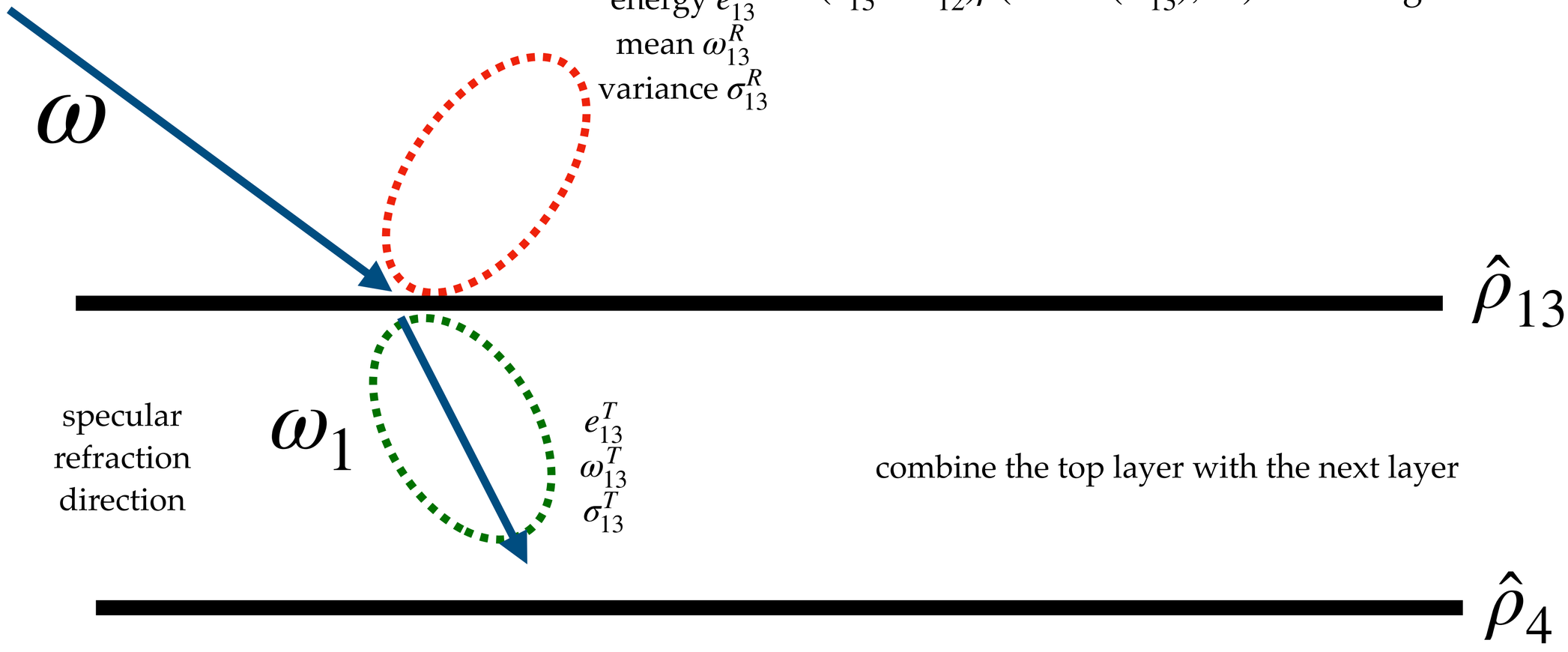
input: $\omega, \hat{\rho}_{12}, \hat{\rho}_{23}, \hat{\rho}_{34}$

output: $\rho(\omega, \omega') = \sum_i w_i \rho_i(\omega, \omega')$

Belcour's algorithm

$e_{12}^R \rho(\omega, \omega')$ with roughness $r = f_R(\sigma_{12}^R)$

$(e_{13}^R - e_{12}^R)\rho(\text{reflect}(\omega_{13}^R), \omega')$ with roughness $r = f_R(\sigma_{13}^R)$



input: $\omega, \hat{\rho}_{12}, \hat{\rho}_{23}, \hat{\rho}_{34}$

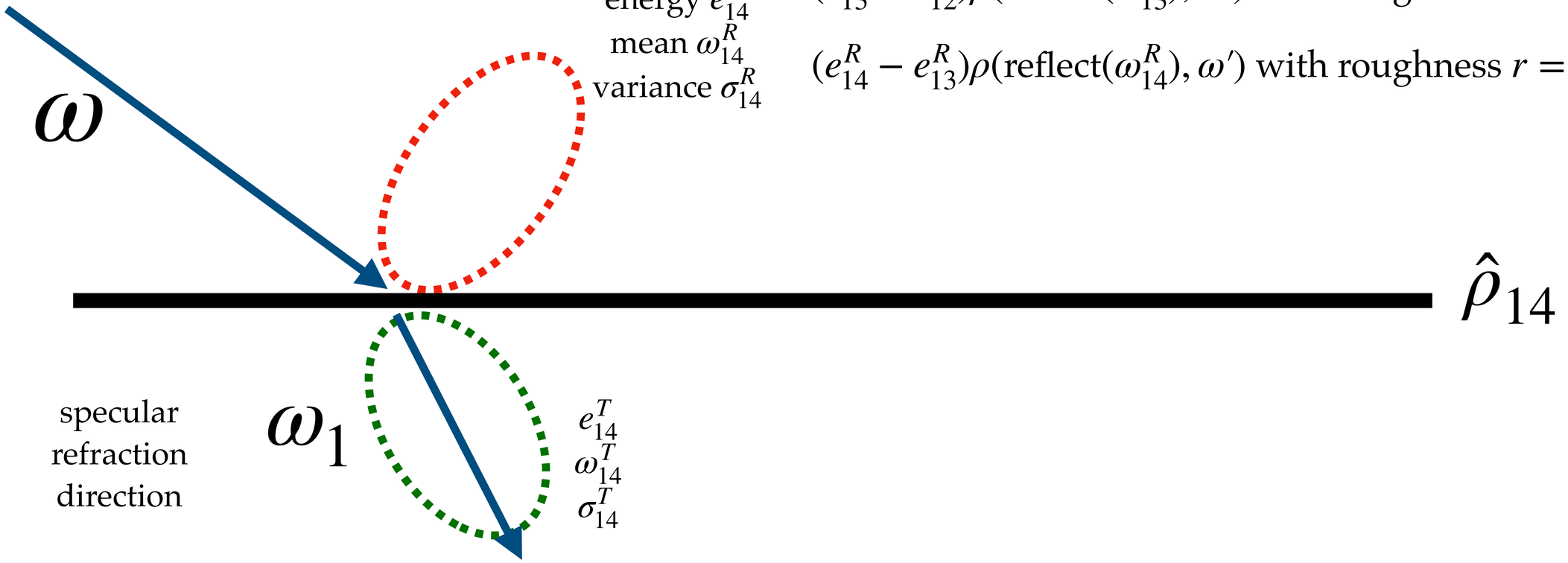
output: $\rho(\omega, \omega') = \sum_i w_i \rho_i(\omega, \omega')$

Belcour's algorithm

$e_{12}^R \rho(\omega, \omega')$ with roughness $r = f_R(\sigma_{12}^R)$

$(e_{13}^R - e_{12}^R)\rho(\text{reflect}(\omega_{13}^R), \omega')$ with roughness $r = f_R(\sigma_{13}^R)$

$(e_{14}^R - e_{13}^R)\rho(\text{reflect}(\omega_{14}^R), \omega')$ with roughness $r = f_R(\sigma_{14}^R)$



Belcour's algorithm

input: $\omega, \hat{\rho}_{12}, \hat{\rho}_{23}, \hat{\rho}_{34}$

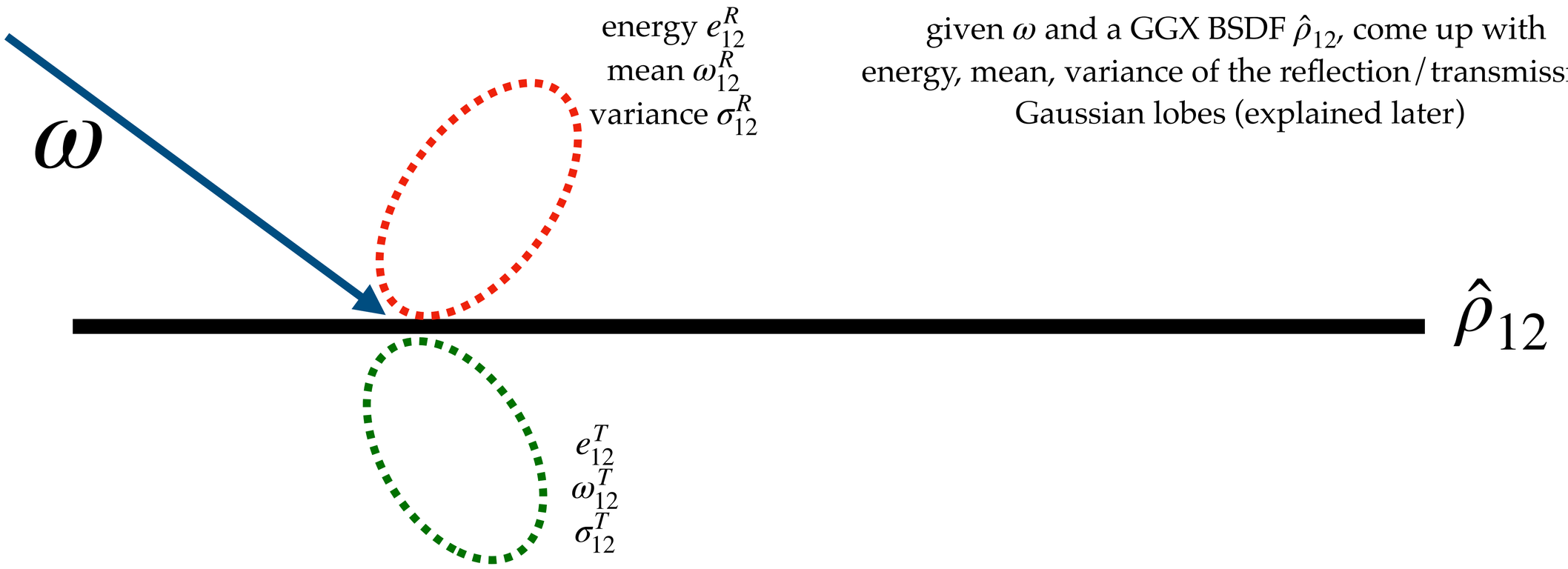
output: $\rho(\omega, \omega') = \sum_i w_i \rho_i(\omega, \omega')$ ⟵

$e^R_{12}\rho(\omega, \omega')$ with roughness $r = f_R(\sigma^R_{12})$

$(e^R_{13} - e^R_{12})\rho(\text{reflect}(\omega^R_{13}), \omega')$ with roughness $r = f_R(\sigma^R_{13})$

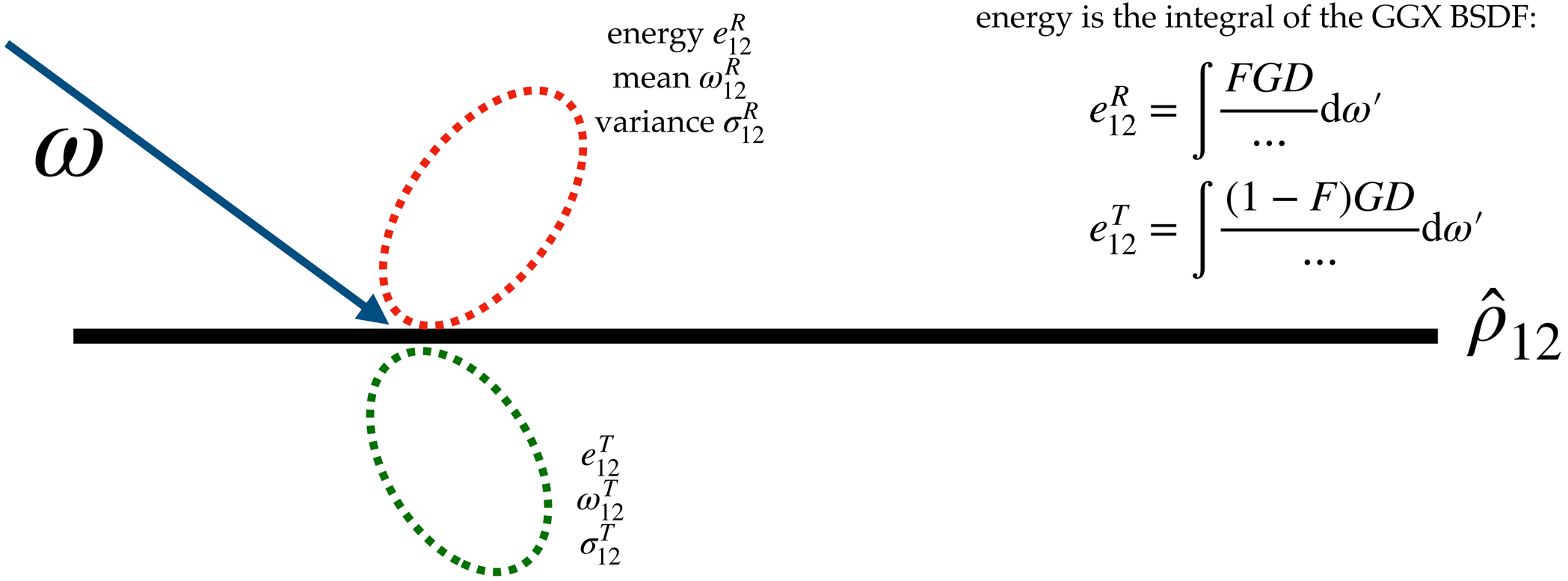
$(e^R_{14} - e^R_{13})\rho(\text{reflect}(\omega^R_{14}), \omega')$ with roughness $r = f_R(\sigma^R_{14})$

How do obtain the Gaussian lobes given GGX?



given ω and a GGX BSDF $\hat{\rho}_{12}$, come up with energy, mean, variance of the reflection/transmission Gaussian lobes (explained later)

How do obtain the Gaussian lobes given GGX?

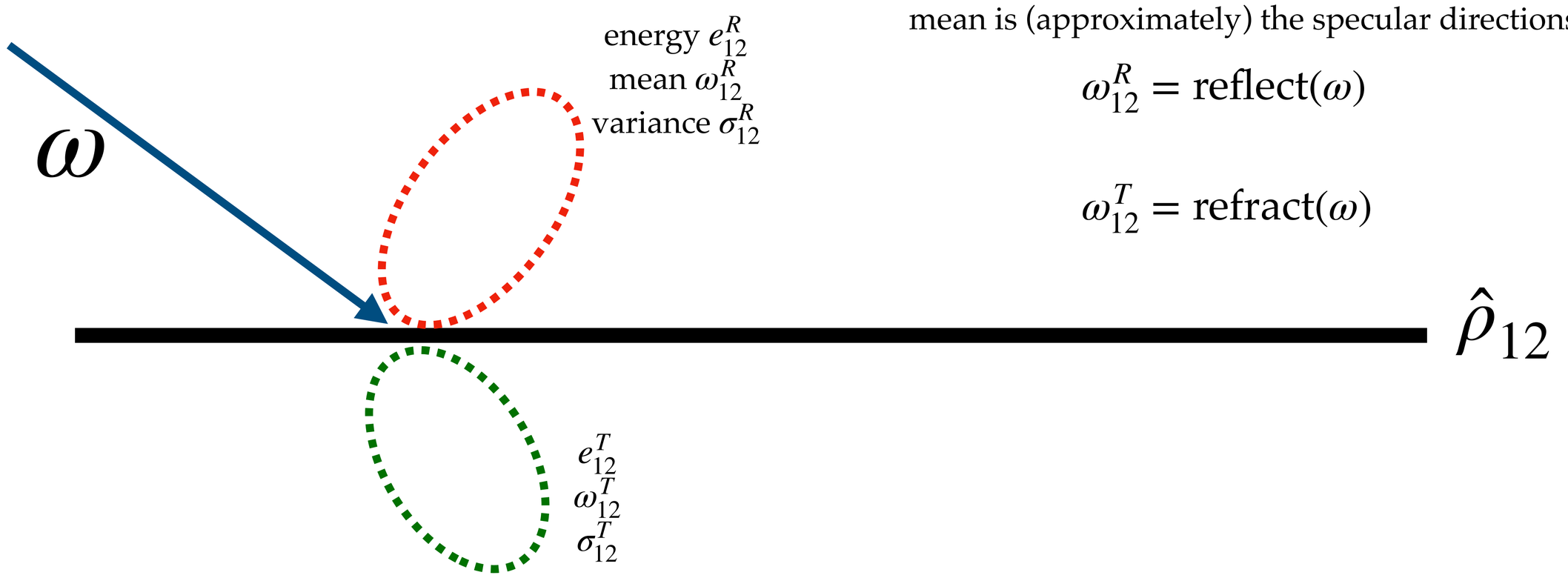


energy is the integral of the GGX BSDF:

$$e_{12}^R = \int \frac{FGD}{\ldots} \mathrm{d}\omega'$$

$$e_{12}^T = \int \frac{(1-F)GD}{\ldots} \mathrm{d}\omega'$$

How do obtain the Gaussian lobes given GGX?

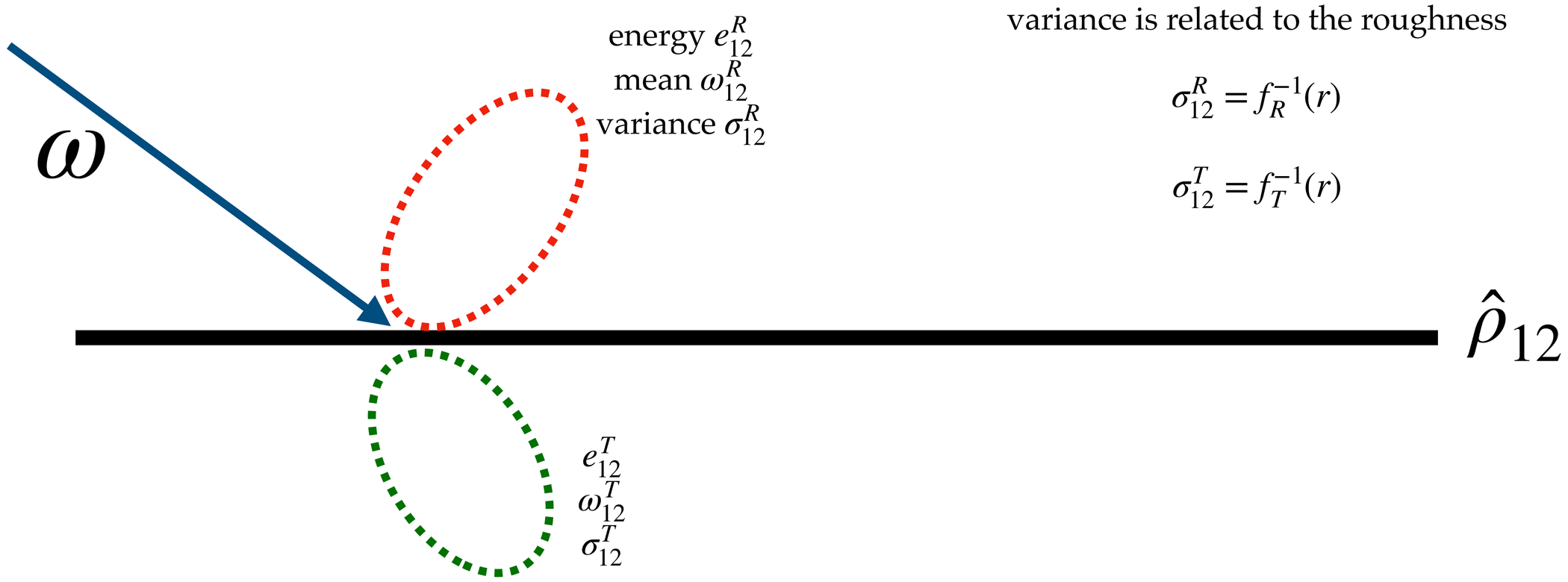


mean is (approximately) the specular directions:

$$\omega_{12}^R = \text{reflect}(\omega)$$

$$\omega_{12}^T = \text{refract}(\omega)$$

How do obtain the Gaussian lobes given GGX?



variance is related to the roughness

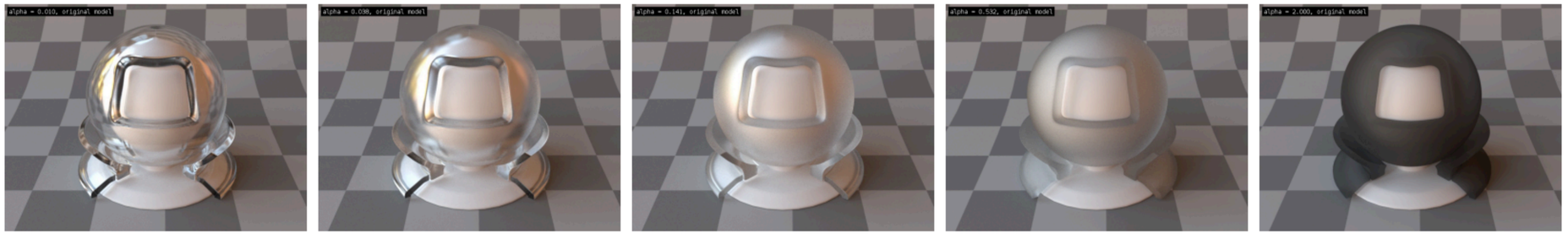
$$\sigma_{12}^R = f_R^{-1}(r)$$

$$\sigma_{12}^T = f_T^{-1}(r)$$

Detail: important to consider multiple-scattering inside microfacet models as well

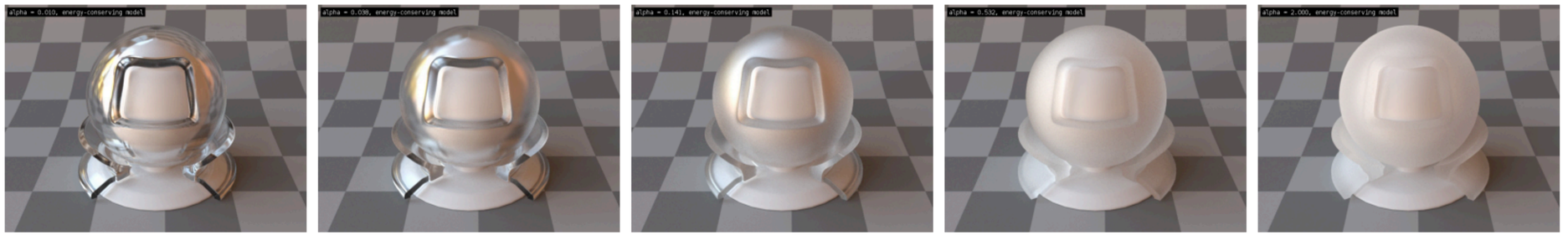
increased roughness

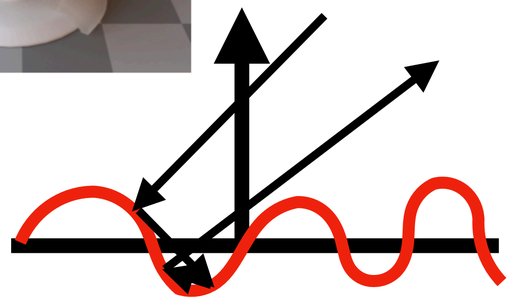
single-scattering microfacet



(a) $\alpha = 0.01$ **(b)** $\alpha = 0.04$ **(c)** $\alpha = 0.14$ **(d)** $\alpha = 0.53$ **(e)** $\alpha = 2$

multiple-scattering microfacet

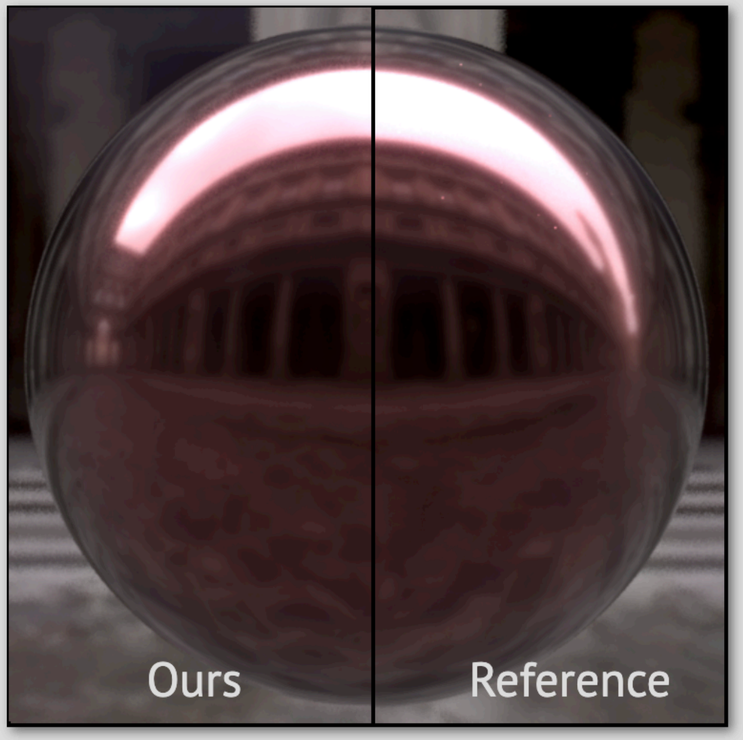




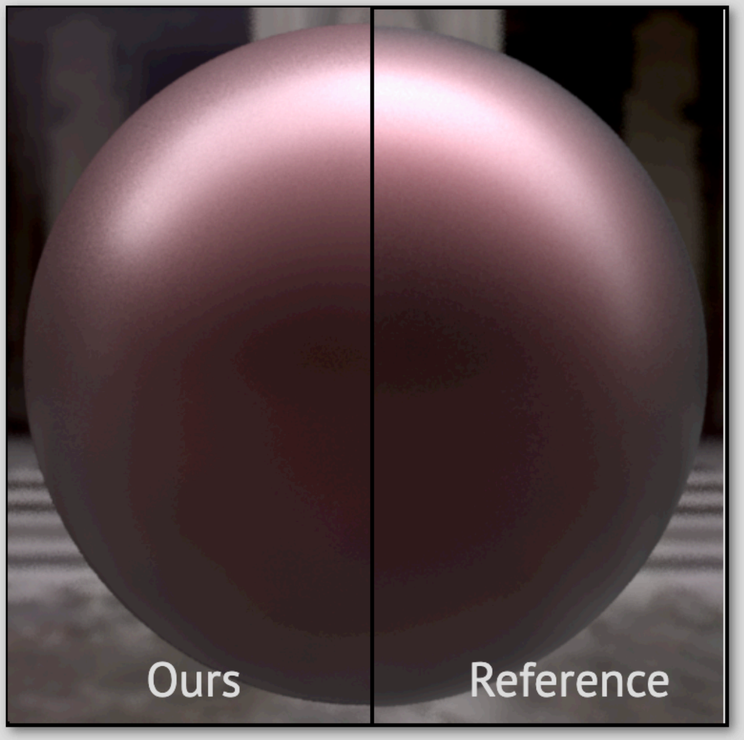
images from Wenzel Jakob et al.
https://rgl.s3.eu-central-1.amazonaws.com/media/papers/Jakob2014Comprehensive_1.pdf

See Heitz et al. “Multiple-Scattering Microfacet BSDFs with the Smith Model”

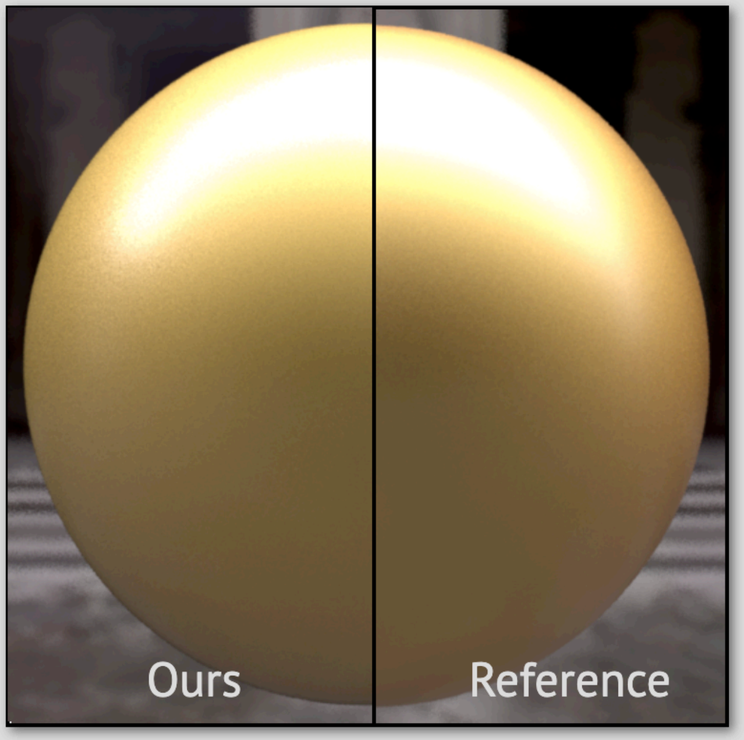
Belcour's algorithm produces very tight approximation



Metal foil



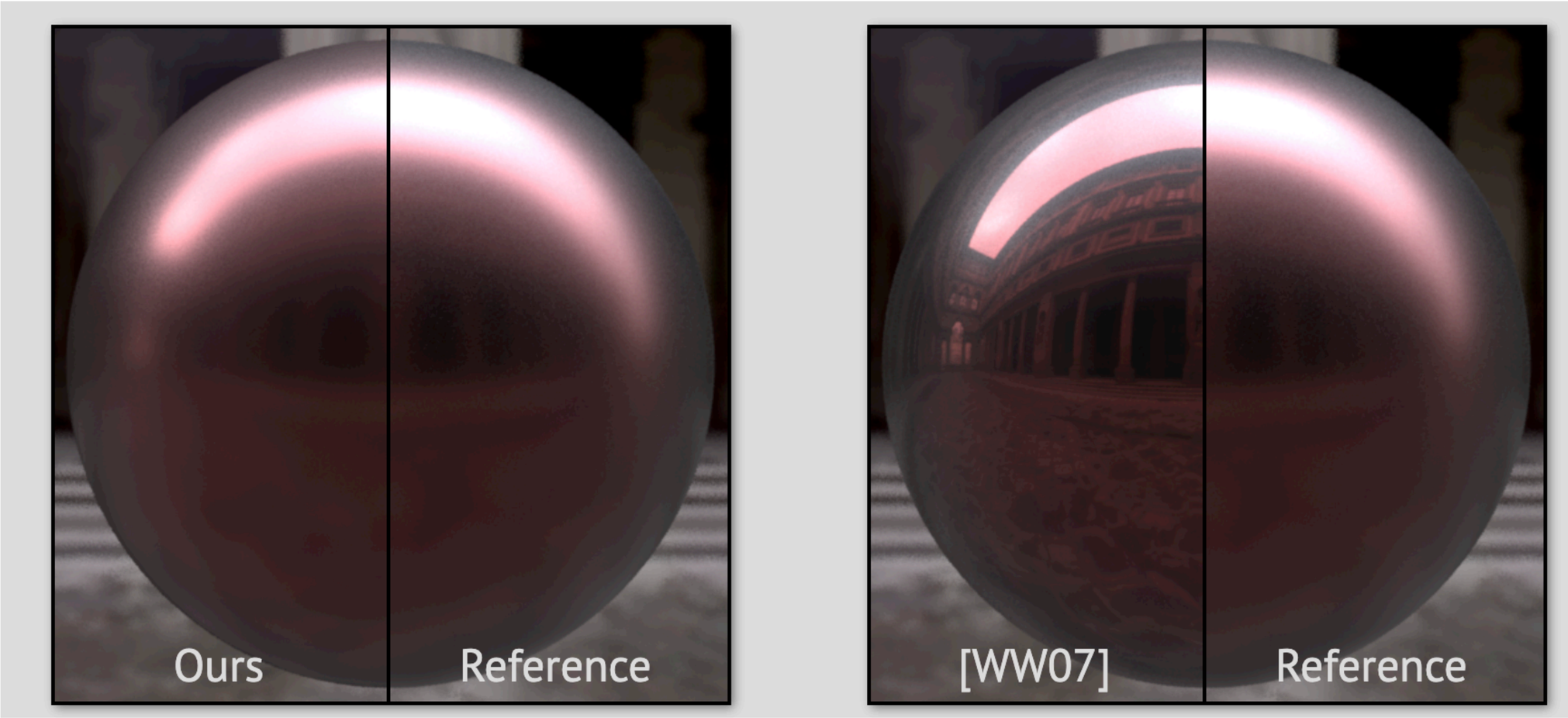
Rough metal



Gold Coated

https://belcour.github.io/blog/slides/2018-brdf-realtime-layered/slides.html#/17

More accurate than a more direct blending of BSDFs



https://belcour.github.io/blog/slides/2018-brdf-realtime-layered/slides.html#/17

Runs in realtime!

BEETLE scene

Material editing session

Nvidia GeForce GTX 980

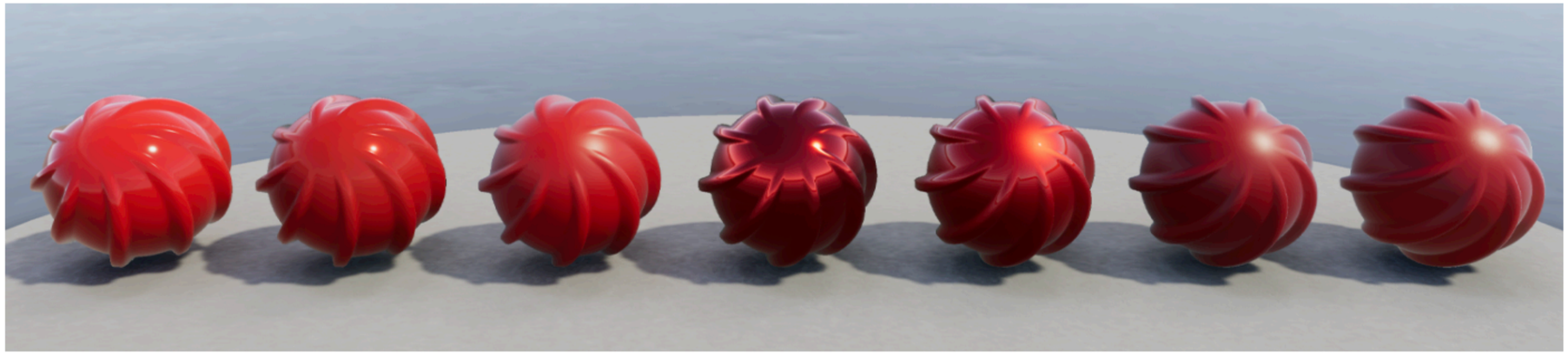
movie from Laurent Belcour

https://belcour.github.io/blog/slides/2018-brdf-realtime-layered/slides.html#/24

Implemented in Unity

StackLit

The StackLit Master Stack can render materials that are more complex than the Lit Master Stack. It includes all the features available in the Lit shader and, in some cases, provides more advanced or higher quality versions. For example, it uses a more advanced form of specular occlusion and also calculates anisotropic reflections for area lights in the same way the Lit shader does for other light types. It also takes into account light interactions between two vertically stacked physical layers, along with a more complex looking general base layer.



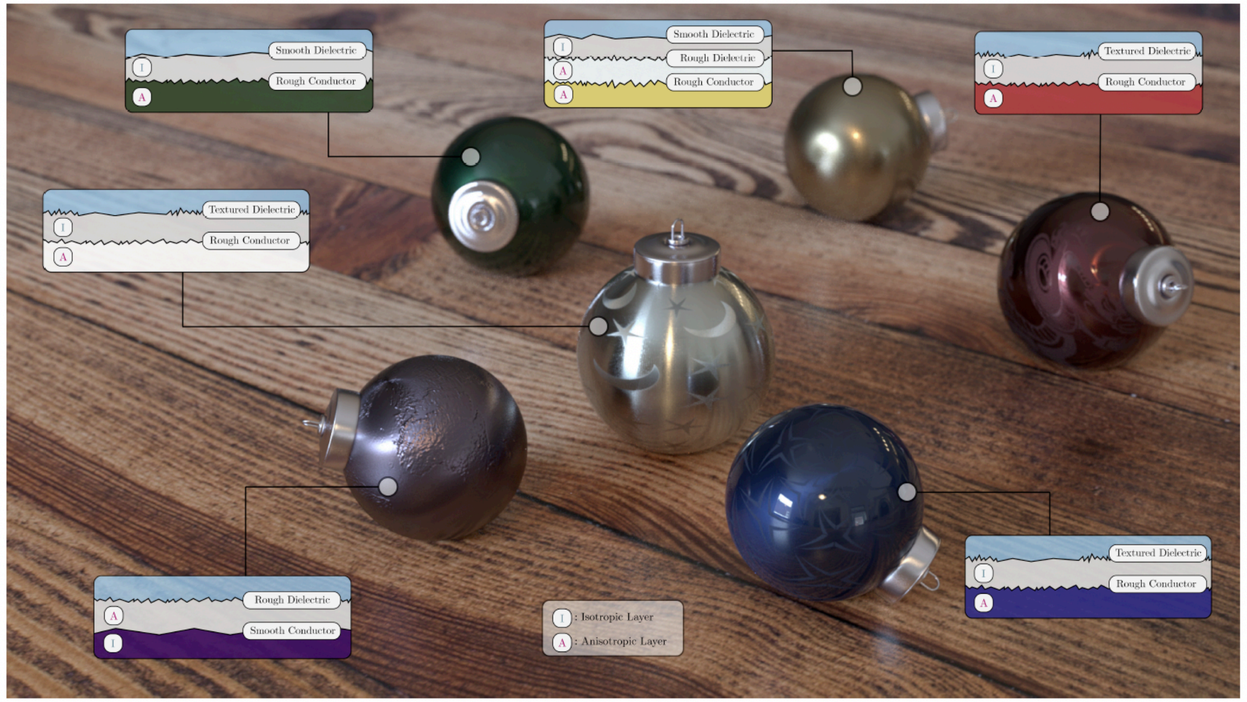
https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@12.0/manual/master-stack-stacklit.html

Easily generalizable to anisotropic GGX

Rendering Layered Materials with Anisotropic Interfaces

Philippe Weier
Unity Technologies, EPFL

Laurent Belcour
Unity Technologies



Somewhat generalizable to diffuse BSDFs

Rendering Layered Materials with Diffuse Interfaces

Heloise de Dinechin
Unity Technologies, EPFL

Laurent Belcour
Unity Technologies

Next: Monte Carlo simulation

A Comprehensive Framework for Rendering Layered Materials

Wenzel Jakob Eugene D'Eon Otto Jakob Steve Marschner

In *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2014)*

Efficient Rendering of Layered Materials using an Atomic Decomposition with Statistical Operators

Laurent Belcour (Unity Technologies)

Published in ACM Transactions on Graphics (proc. of SIGGRAPH 2018)

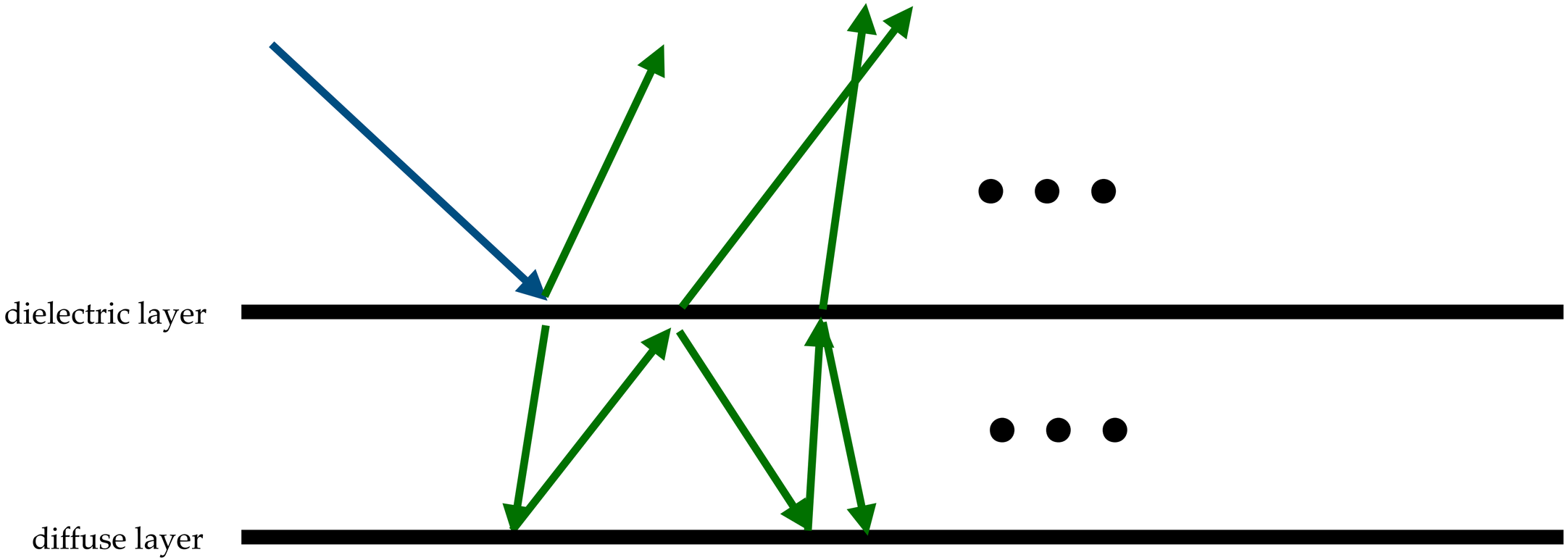
paper bib paper code video slides

Position-Free Monte Carlo Simulation for Arbitrary Layered BSDFs

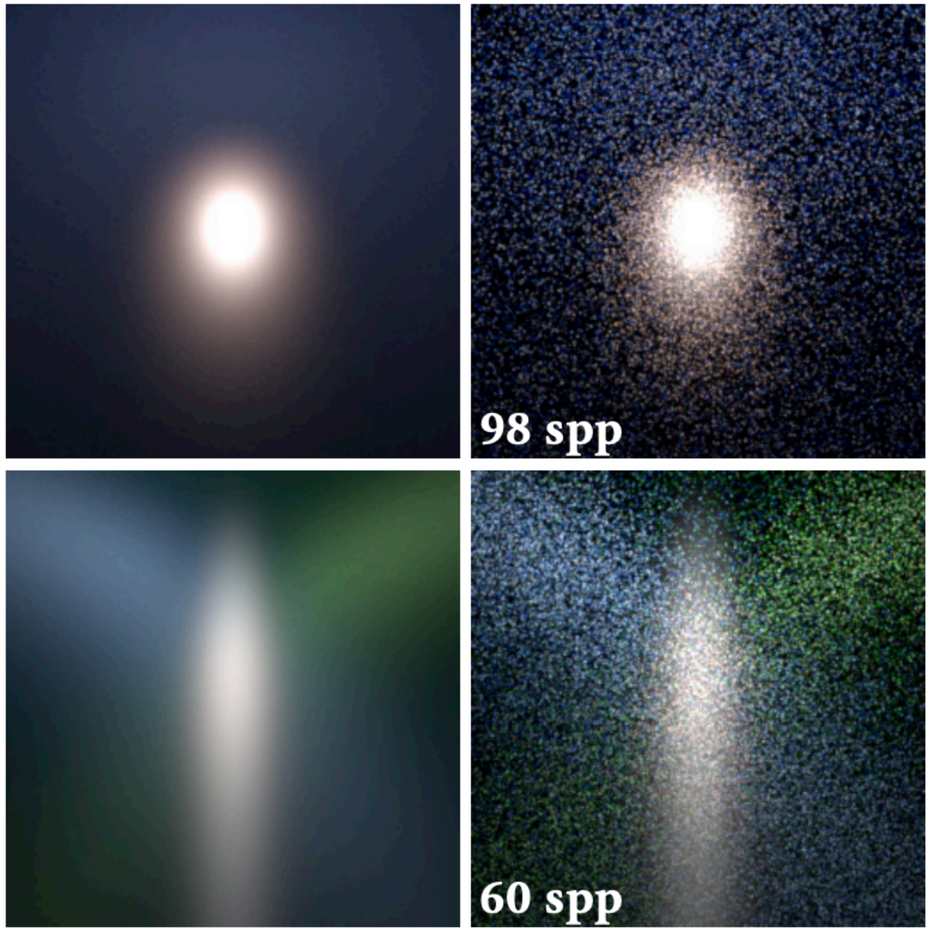
Yu Guo[1], Miloš Hašan[2], Shuang Zhao[1]
[1]University of California, Irvine [2]Autodesk

ACM Transactions on Graphics (SIGGRAPH Asia 2018), 37(6), 2018

Idea: just compute the multiple scattering using Monte Carlo integration!



Brute-force simulation is still too slow



Reference Standard PT

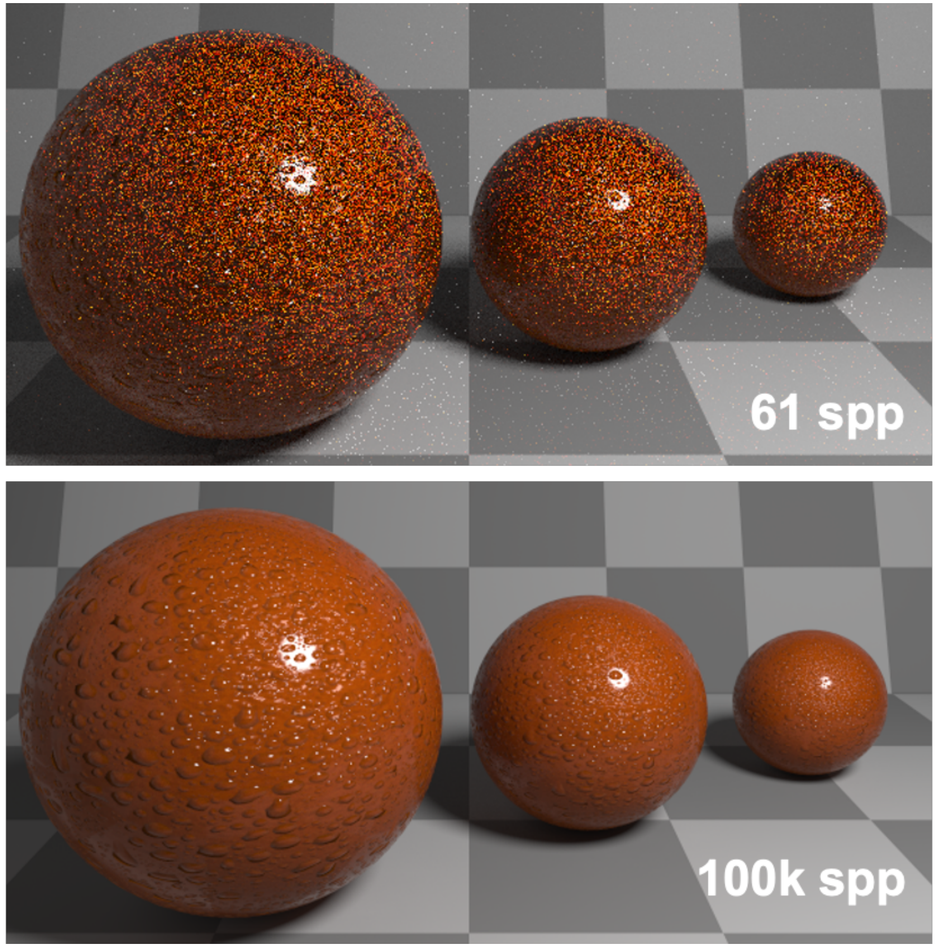
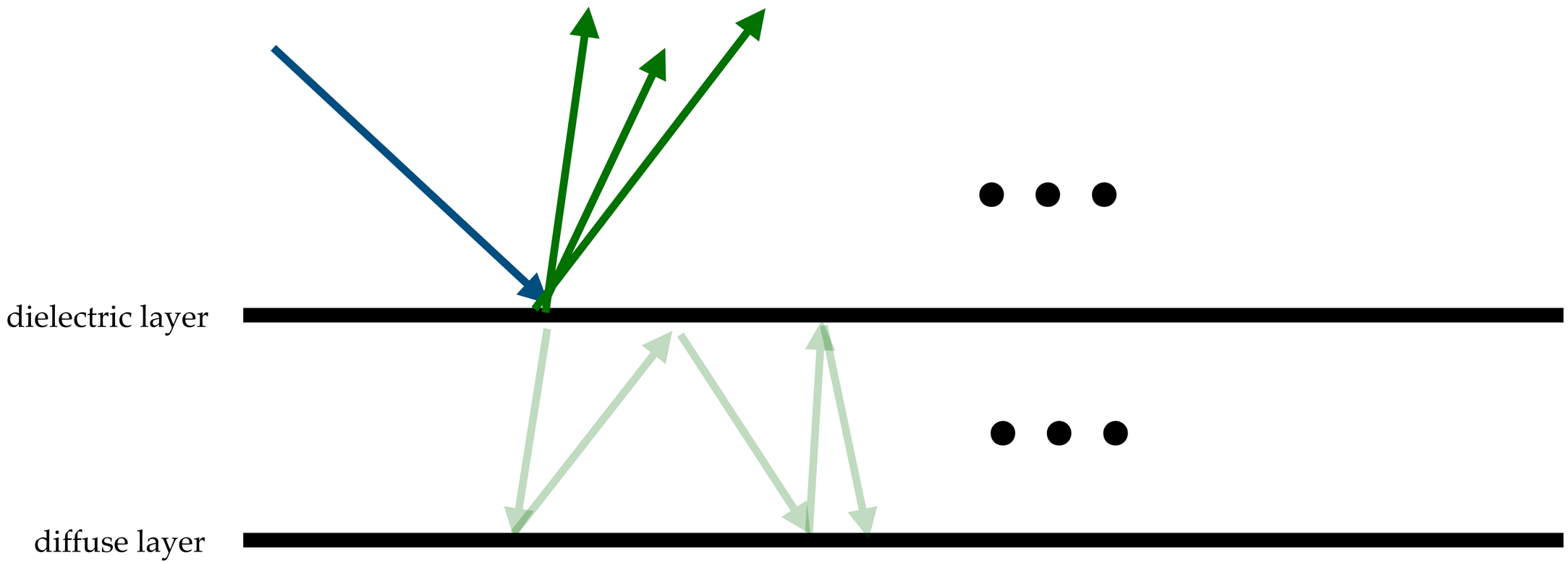


image from Yu Guo et al. & Gamboa et al.
https://www.ics.uci.edu/~yug10/webpage/pdf/2018TOG.pdf
https://beltegeuse.github.io/research/publication/2020_layered/

Observation:
the horizontal distance doesn't matter in a BSDF



Idea: solve a small “1D” rendering equation inside the material

mathematically: it’s a change of variable!

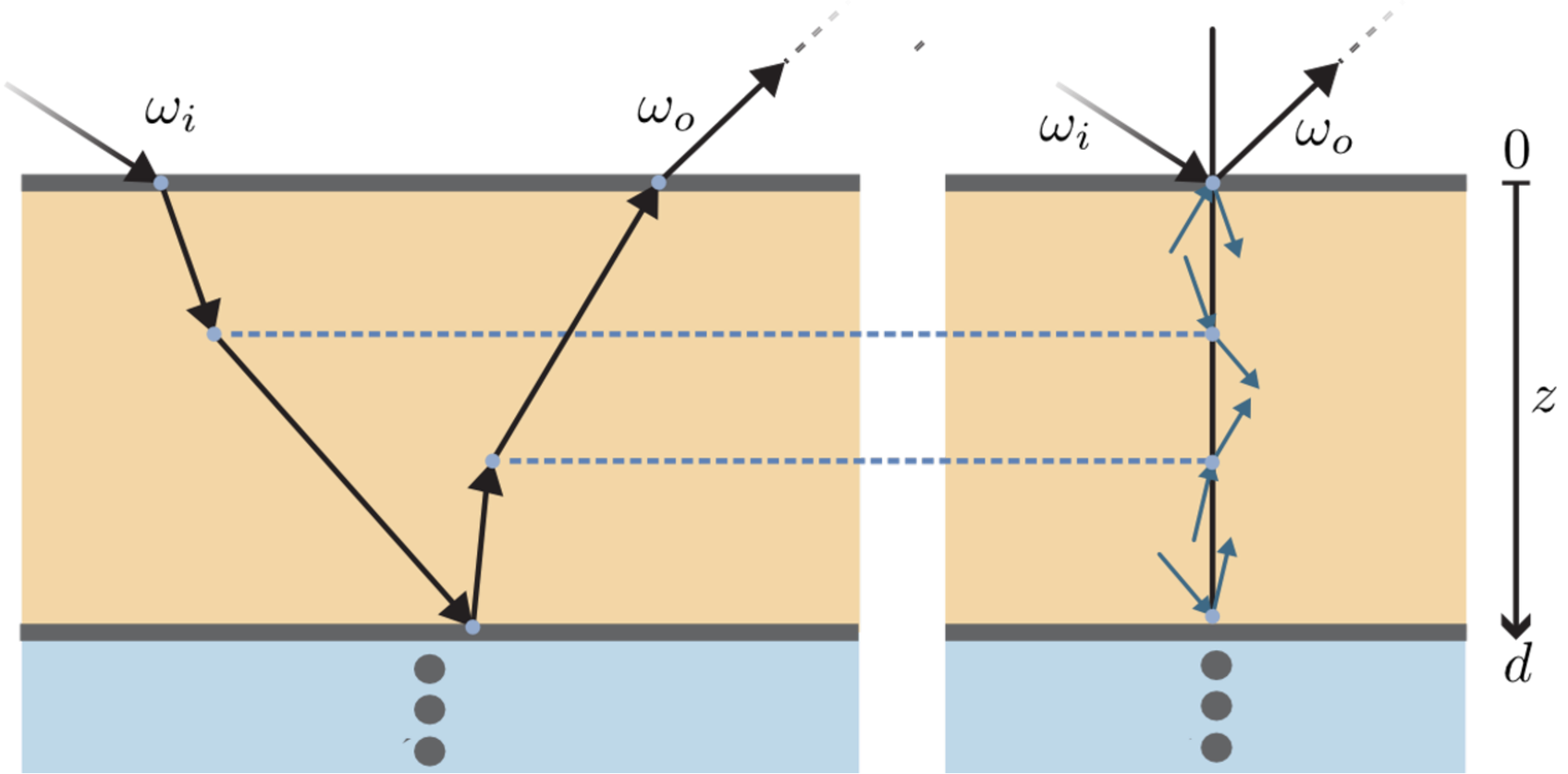
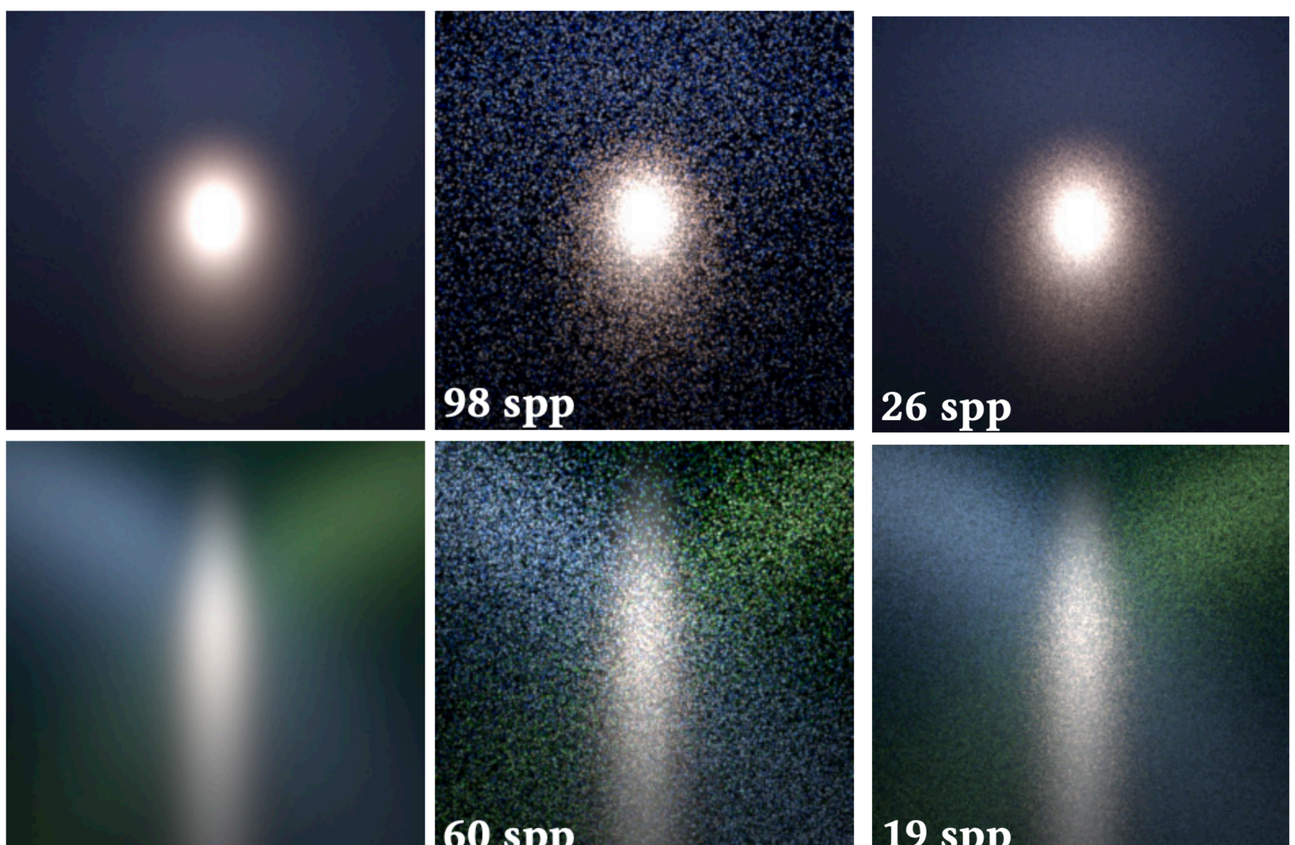
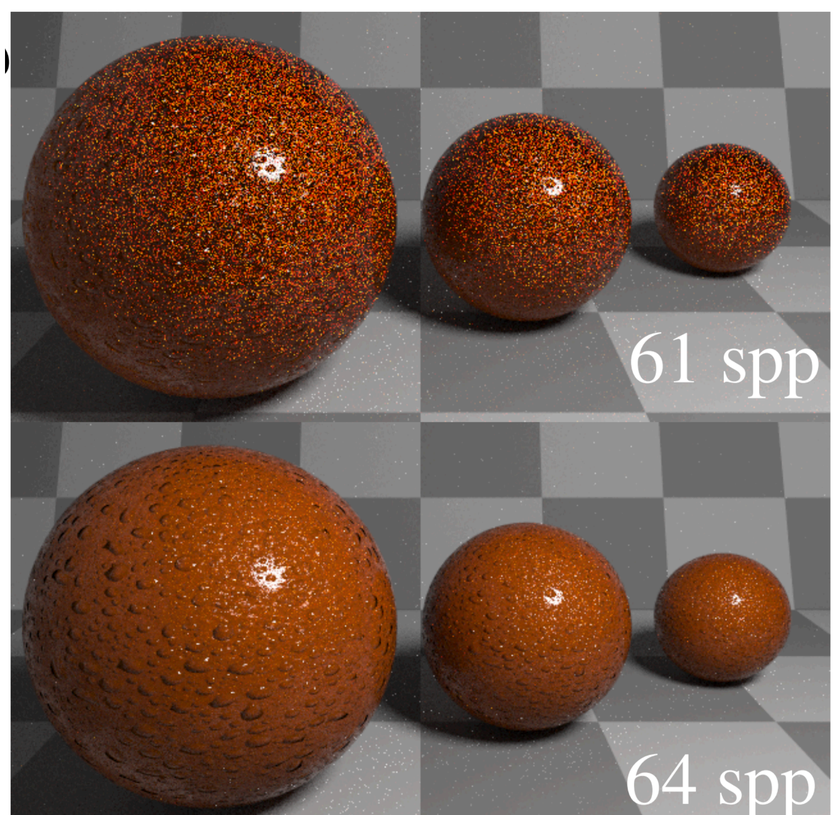


Figure from Gamboa et al.
https://beltegeuse.github.io/research/publication/2020_layered/

The simplified rendering equation has much smaller variance



Reference | Standard PT | Guo et al.



Gamboa et al.

image from Yu Guo et al. & Gamboa et al.
https://www.ics.uci.edu/~yug10/webpage/pdf/2018TOG.pdf
https://beltegeuse.github.io/research/publication/2020_layered/

More accurate than Belcour?

- Belcour's code doesn't directly support multi-layer high-res normal maps
  - mipmapping for layered BSDF could be an interesting research topic

https://beltegeuse.github.io/research/publication/2020_layered/ Gamboa et al.

Today: 3 approaches of modeling layered BSDFs

A Comprehensive Framework for Rendering Layered Materials

Wenzel Jakob Eugene D'Eon Otto Jakob Steve Marschner

In *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2014)*

Efficient Rendering of Layered Materials using an Atomic Decomposition with Statistical Operators

Laurent Belcour (Unity Technologies)

Published in ACM Transactions on Graphics (proc. of SIGGRAPH 2018)

paper bib paper code video slides

Position-Free Monte Carlo Simulation for Arbitrary Layered BSDFs

Yu Guo[1], Miloš Hašan[2], Shuang Zhao[1]
[1]University of California, Irvine [2]Autodesk

ACM Transactions on Graphics (SIGGRAPH Asia 2018), 37(6), 2018

discussion: what are the pros and cons of the three methods?
if you want to implement one of these in your renderer,
which one will you choose? why?

History (in graphics)

1997

Plane Parallel Radiance Transport for

Global Illumination in Vegetation

Nelson Max

Curtis Mobley

Brett Keating

En-Hua Wu

continuous adding equation!

$$\frac{dI_u(z)}{dz} = I_u(z)\tau_{uu}(z) + I_d(z)\rho_{du}(z)$$

$$-\frac{dI_d(z)}{dz} = I_u(z)\rho_{ud}(z) + I_d(z)\tau_{dd}(z)\,.$$

2001

An Illumination Model for a Skin Layer Bounded by Rough Surfaces

Jos Stam

Alias | wavefront
1218 Third Ave, 8th Floor,
Seattle, WA 98101

2000

Monte Carlo Evaluation Of Non-Linear Scattering Equations For Subsurface Reflection

Matt Pharr Pat Hanrahan

Stanford University

Monte Carlo solution to the adding equation

$$\begin{aligned}
\mathbf{S}_{a\to a} &= \mathbf{S}_a + \mathbf{S}_a\mathbf{S}_b\mathbf{S}_a + \mathbf{S}_a\mathbf{S}_b\mathbf{S}_a\mathbf{S}_b\mathbf{S}_a + \cdots && \text{(3.6a)}\\
&= \sum_{n=0}^{\infty}(\mathbf{S}_a\mathbf{S}_b)^n\mathbf{S}_a && \text{(3.6b)}\\
&= (\mathbf{I} - \mathbf{S}_a\mathbf{S}_b)^{-1}\mathbf{S}_a && \text{(3.6c)}\\
&= \mathbf{S}_a + \mathbf{S}_a\mathbf{S}_b\mathbf{S}_{a\to a}. && \text{(3.6d)}
\end{aligned}$$

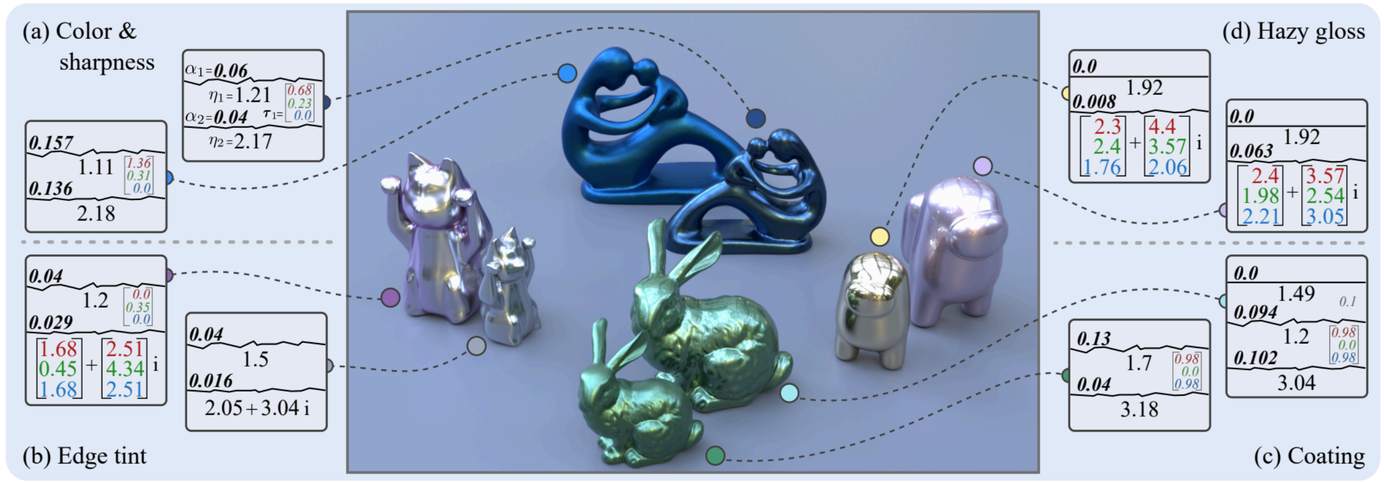
predecessor of Wenzel's approach

ComputeRT:
  For $k = 0, \cdots, N$ do
    Compute the scattering matrix $\mathbf{M}_k$ (Appendix A)
    Compute the reflection and transmission matrices $\mathbf{R}_{ij}$ and $\mathbf{T}_{ij}$ (Appendix D)
    Compute eigenstructure of $\mathbf{M}_k$
    For $i = 1, \cdots, M$
      Solve linear system for incoming direction μ_i (Equation 15)
      Transmit radiances out of the layer (Equation 16)
      Set the i-th columns of $\mathbf{R}_k$ and $\mathbf{T}_k$
    next
  next

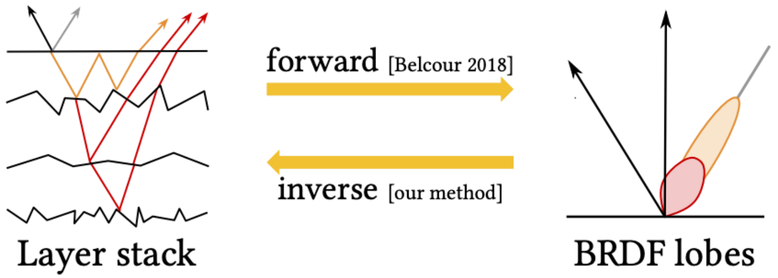
Fun recent research about layered materials

An Inverse Method for the Exploration of Layered Material Appearance

MÉGANE BATI, IOGS, Université de Bordeaux, France
PASCAL BARLA, Inria, France
ROMAIN PACANOWSKI, Inria, France

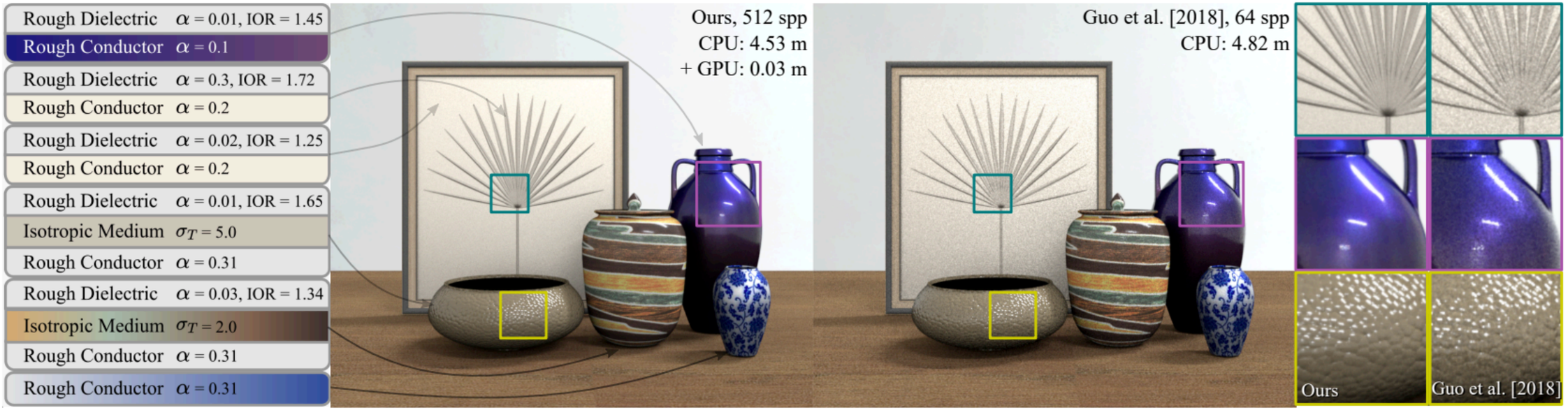


the inverse version of Belcour's algorithm



Fun recent research about layered materials

Neural Layered BRDFs



Jiahui Fan
Nanjing University of Science and Technology
China
fjh@njust.edu.cn

Beibei Wang†
Nankai University,
Nanjing University of Science and Technology
China
beibei.wang@nankai.edu.cn

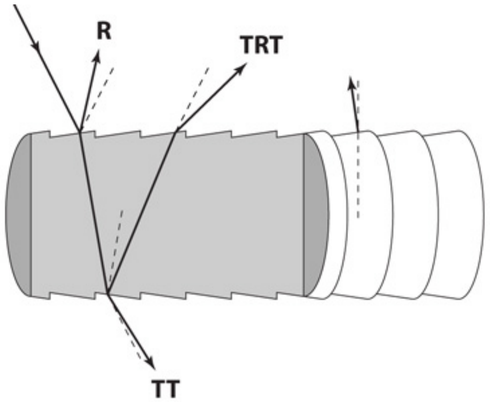
Miloš Hašan
Adobe Research
USA

Jian Yang†
Nanjing University of Science and Technology
China
csjyang@njust.edu.cn

Ling-Qi Yan
University of California, Santa Barbara
USA
lingqi@cs.ucsb.edu

Next: hair & cloth

- rendering fiber structures



https://www.cs.drexel.edu/~david/Classes/Papers/p271-kajiya.pdf
https://www.cs.cornell.edu/~srm/publications/SG03-hair.pdf
https://shuangz.com/projects/practical_cloth-sa20/