REAL-TIME SUBSURFACE SCATTERING

A comparative analysis of approximation techniques

Bennett Poh · 2026

01-
WHY SUBSURFACE SCATTERING MATTERS

Subsurface Scattering - **OFF**

Light reflects only at the skin surface, producing sharp shading transitions and an unnatural, plastic appearance.

Subsurface Scattering - **ON**

Light penetrates the skin and scatters internally, producing soft shading transitions and a natural, lifelike appearance.

Subsurface Scattering - **OFF**

The material appears dense and concrete-like, with the light interacting only at the surface.

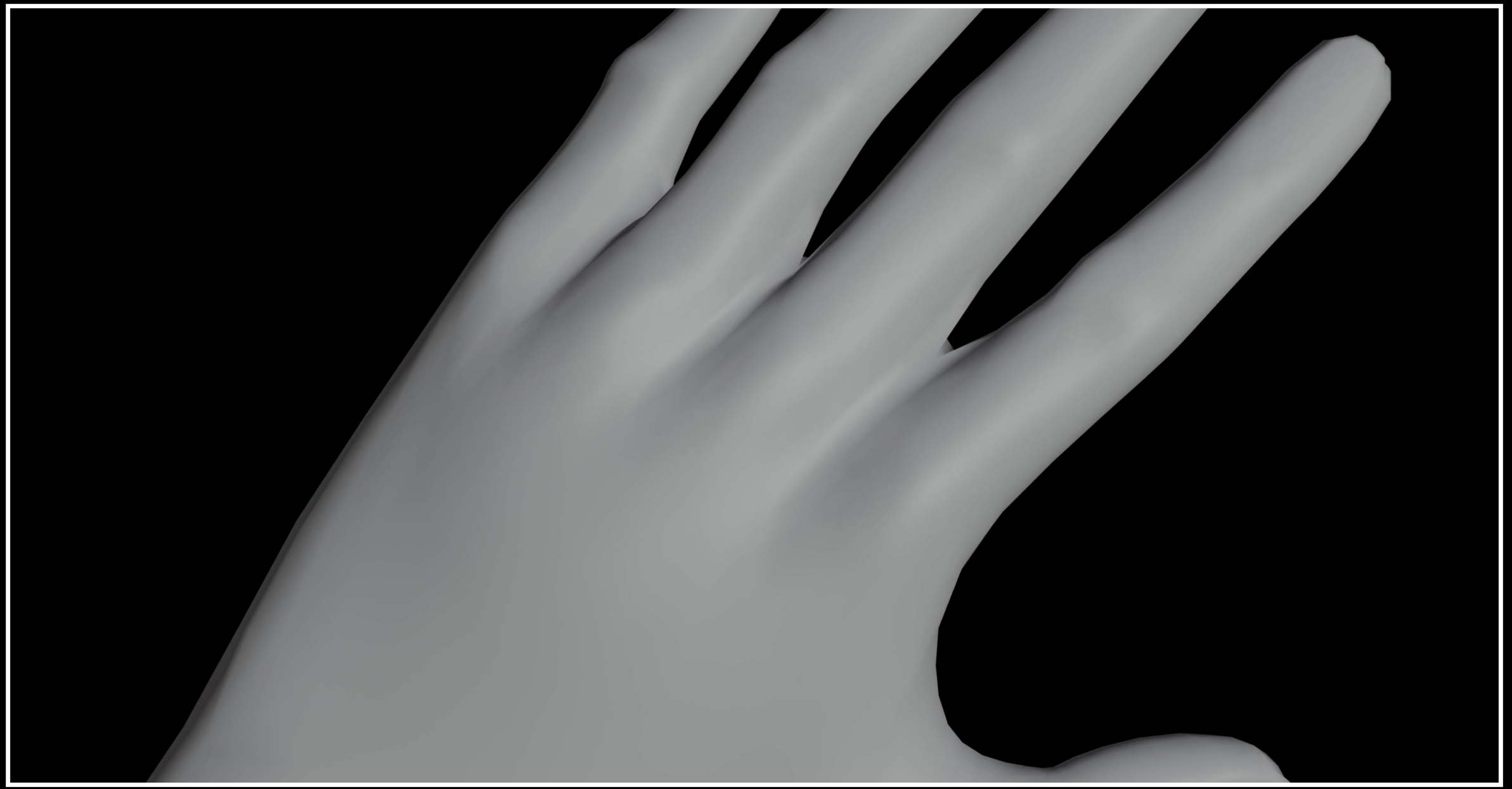
Subsurface Scattering - **ON**

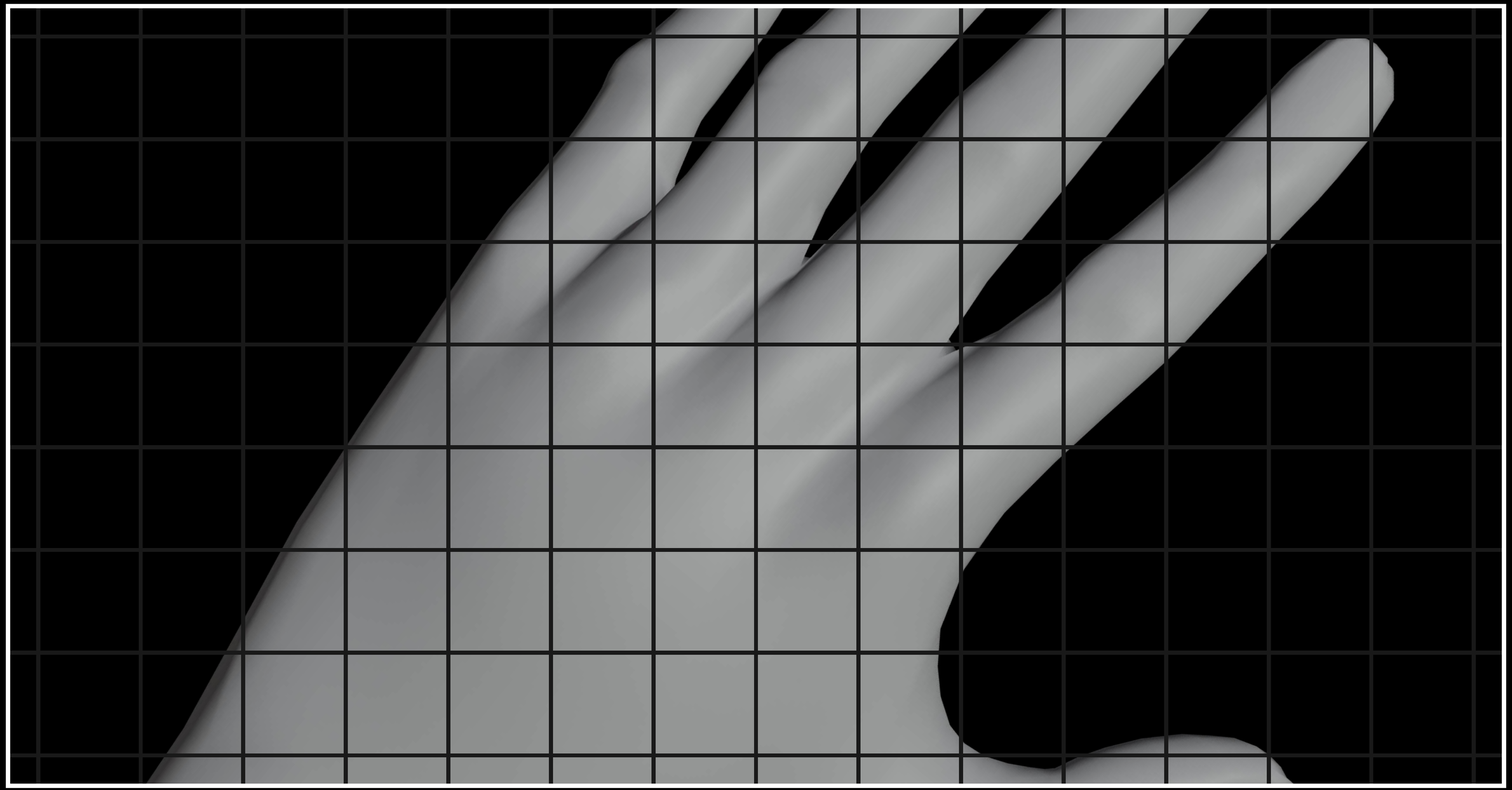
The material appears soft and waxy, as light penetrates and diffuses beneath the surface.

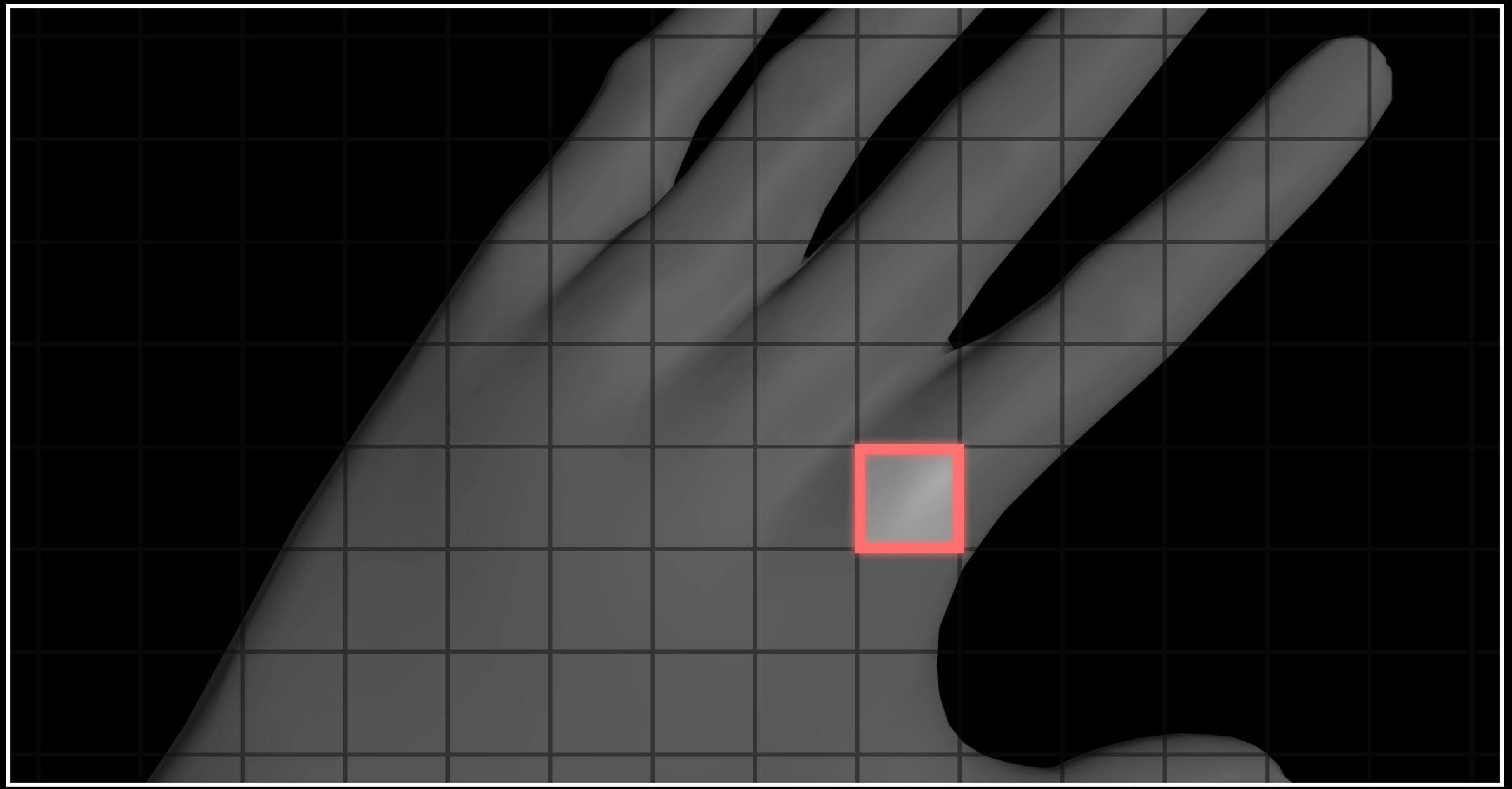
02-

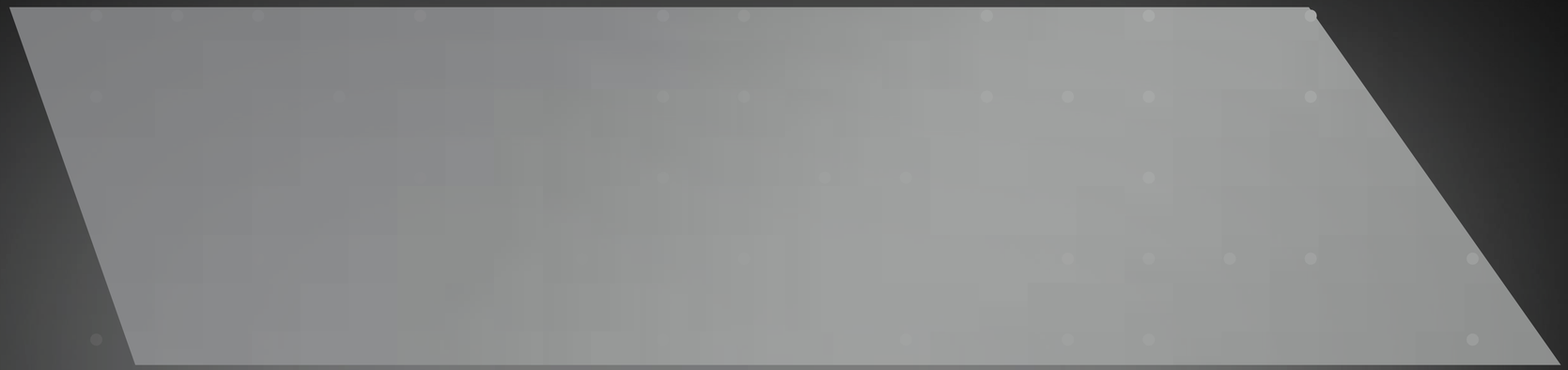
PHYSICAL BASIS OF SUBSURFACE SCATTERING

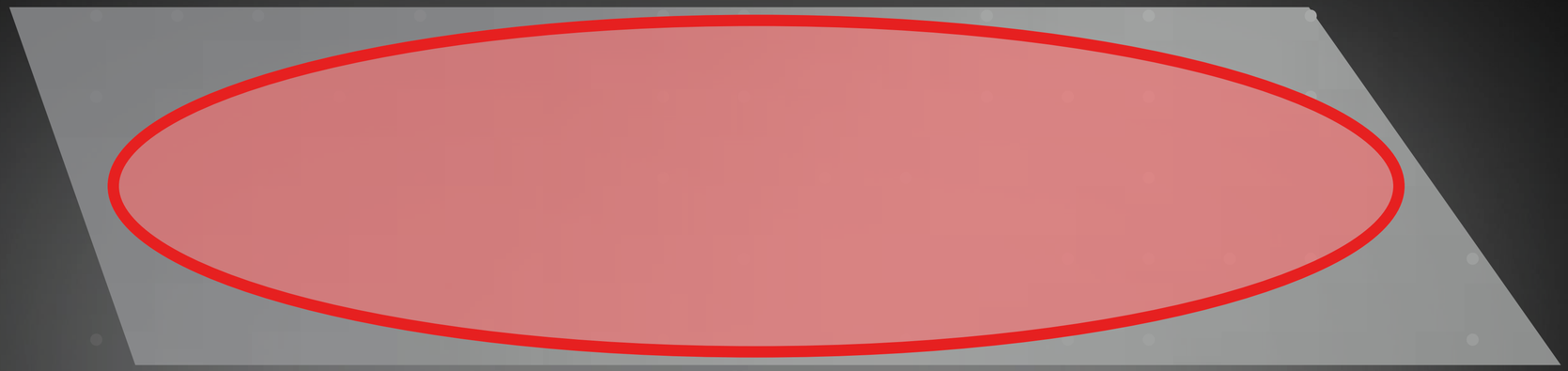
Before discussing the real-time techniques in this study, it is important to understand the physical basis of SSS and why it poses a challenge for real-time rendering.

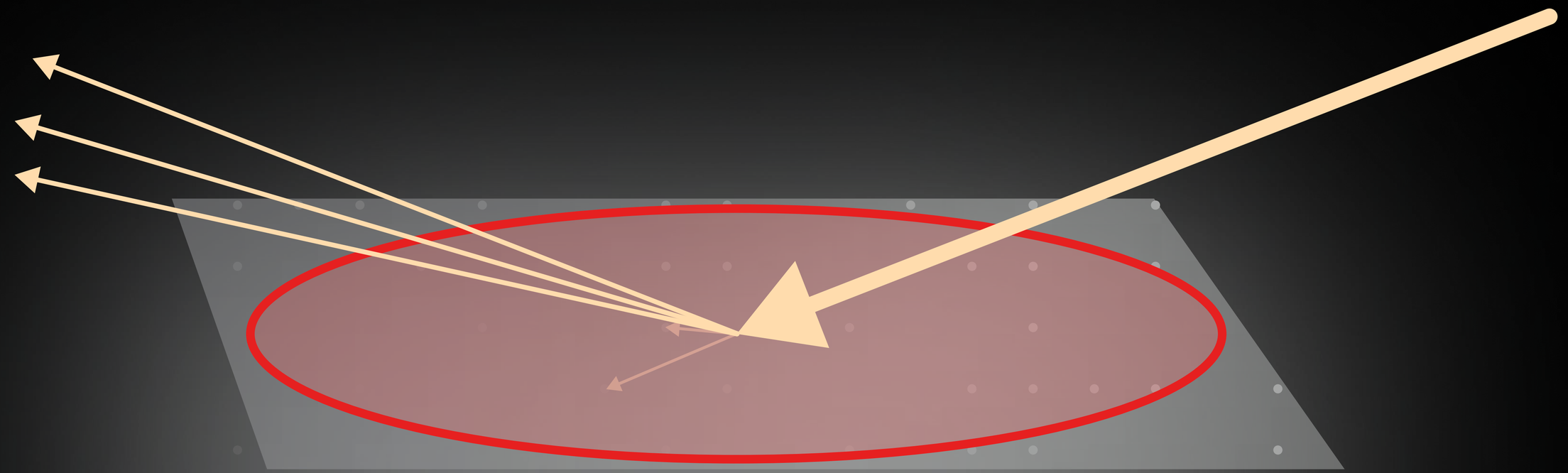


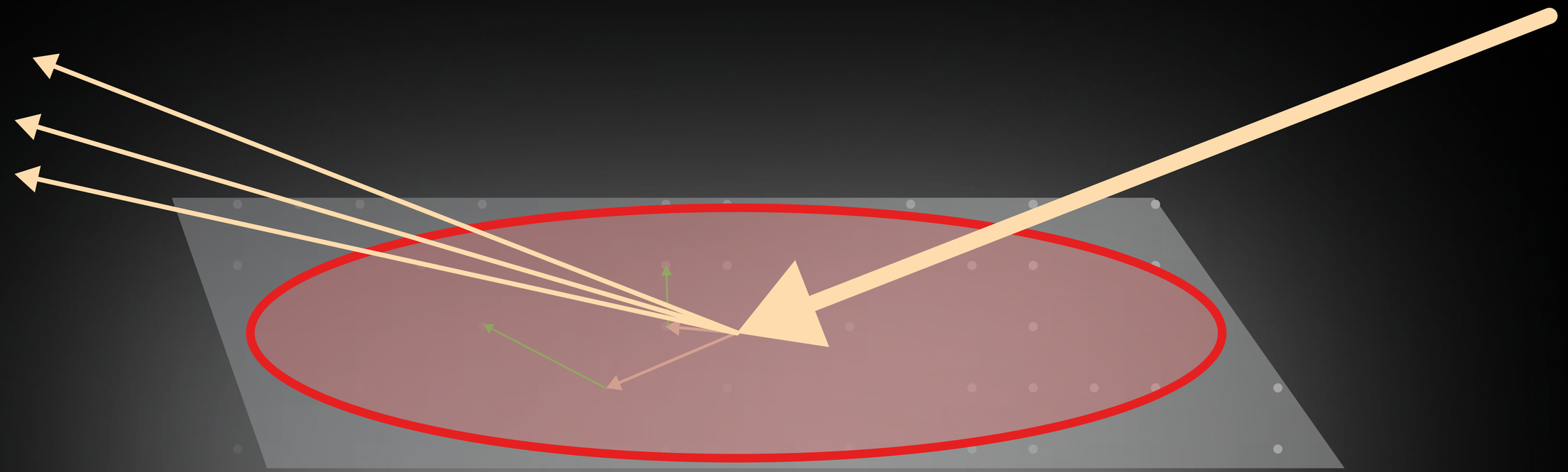


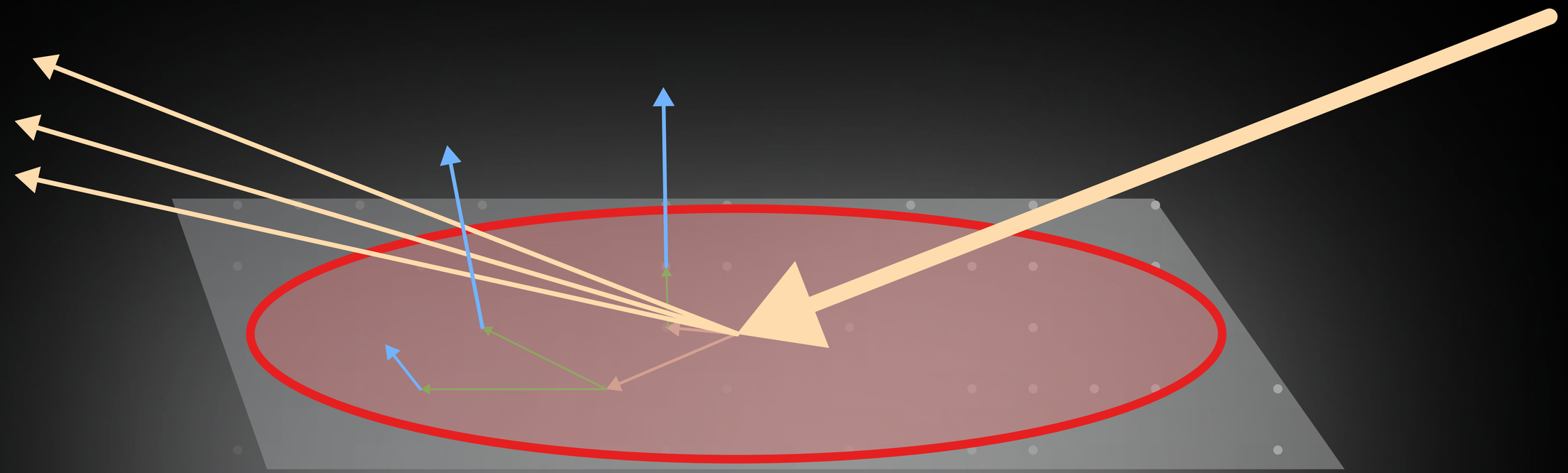


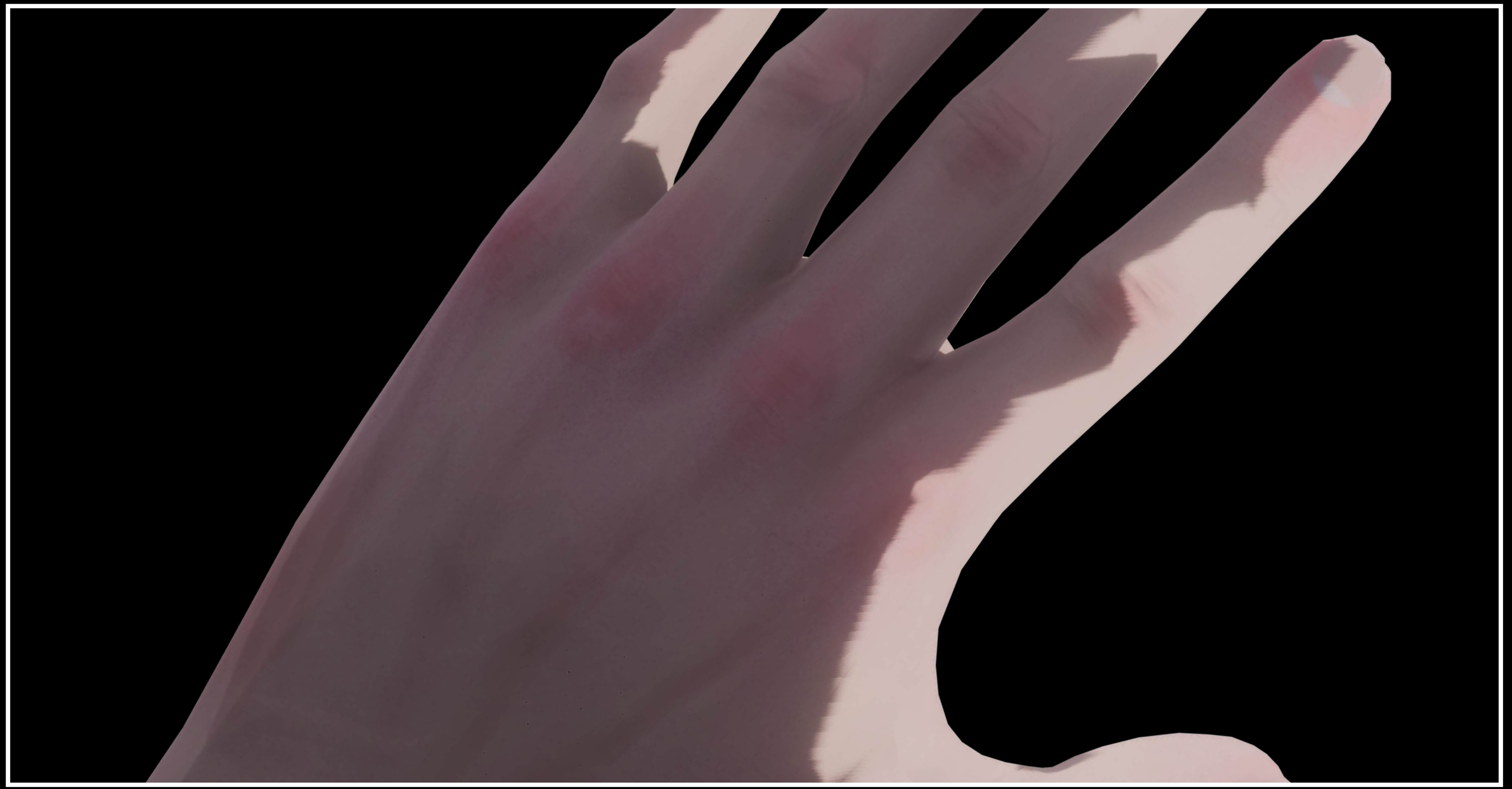


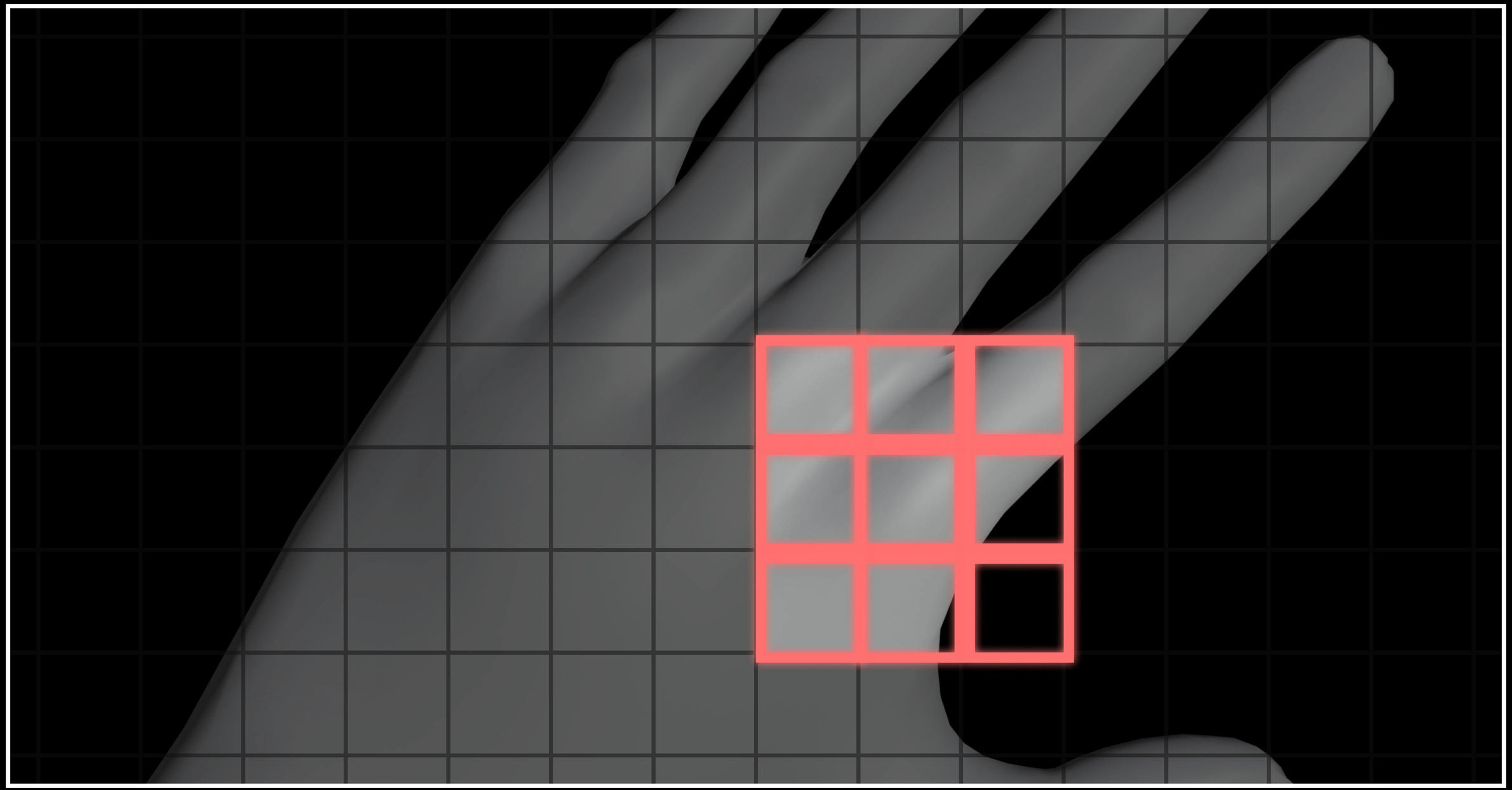


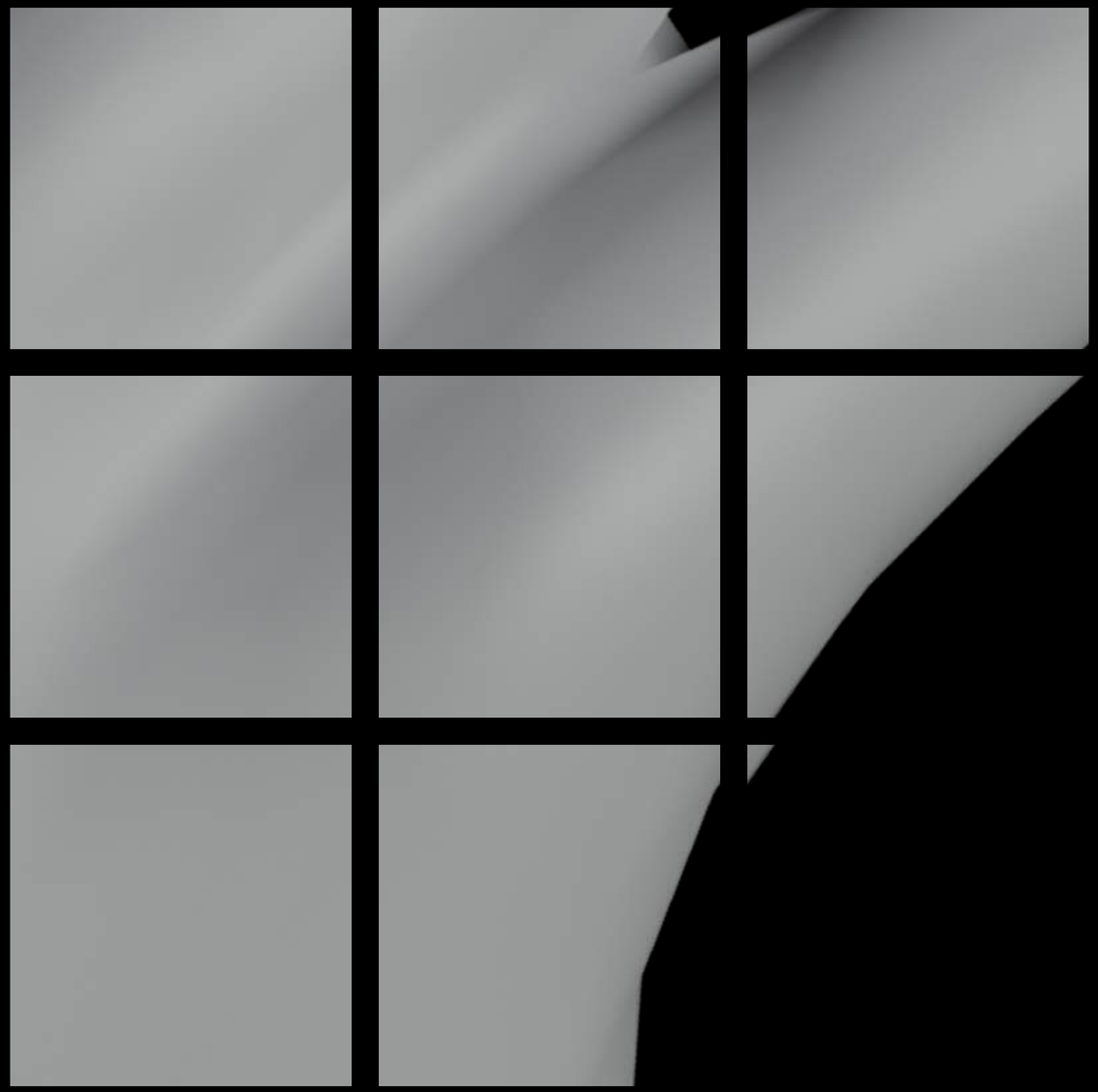


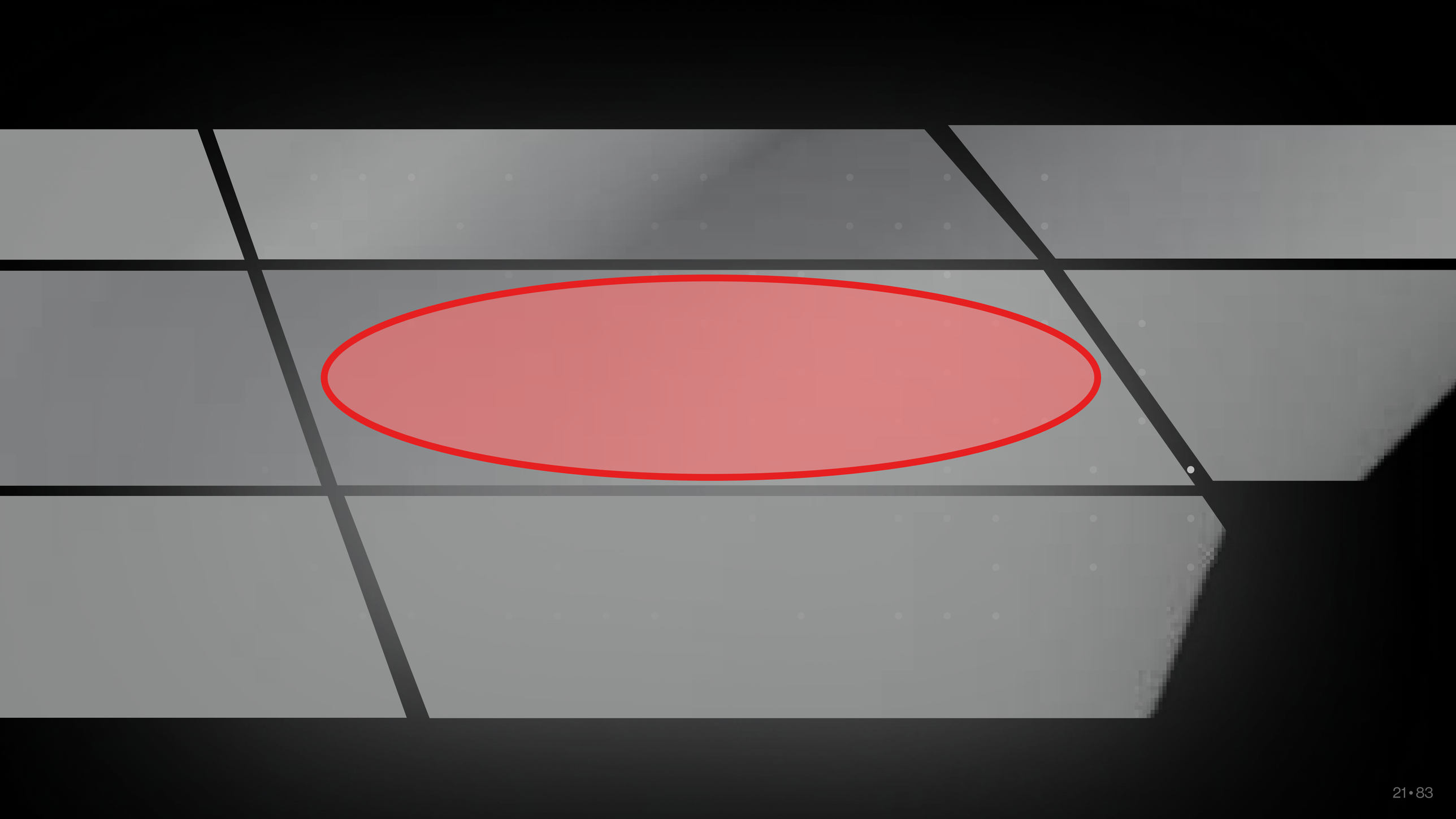


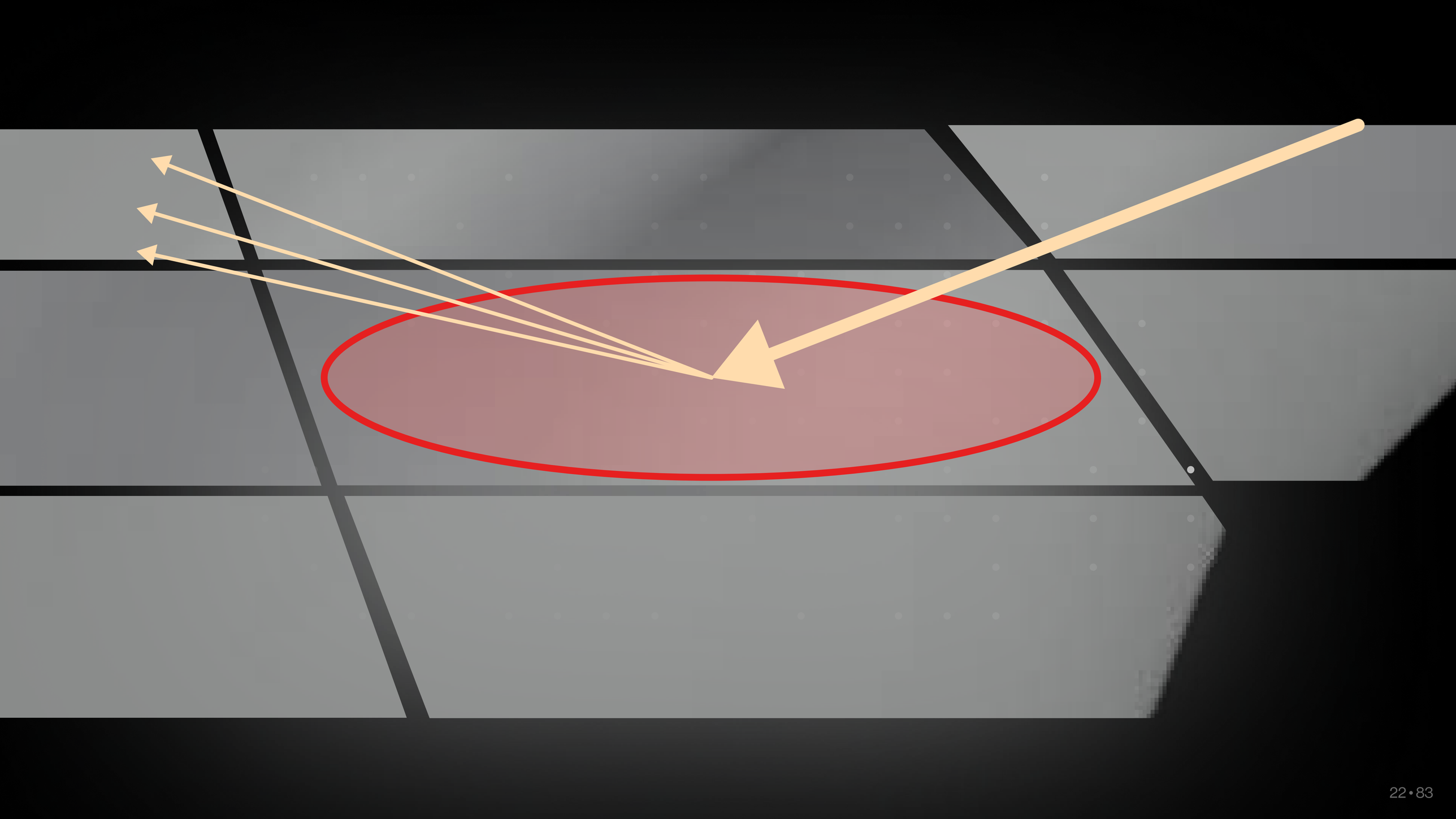


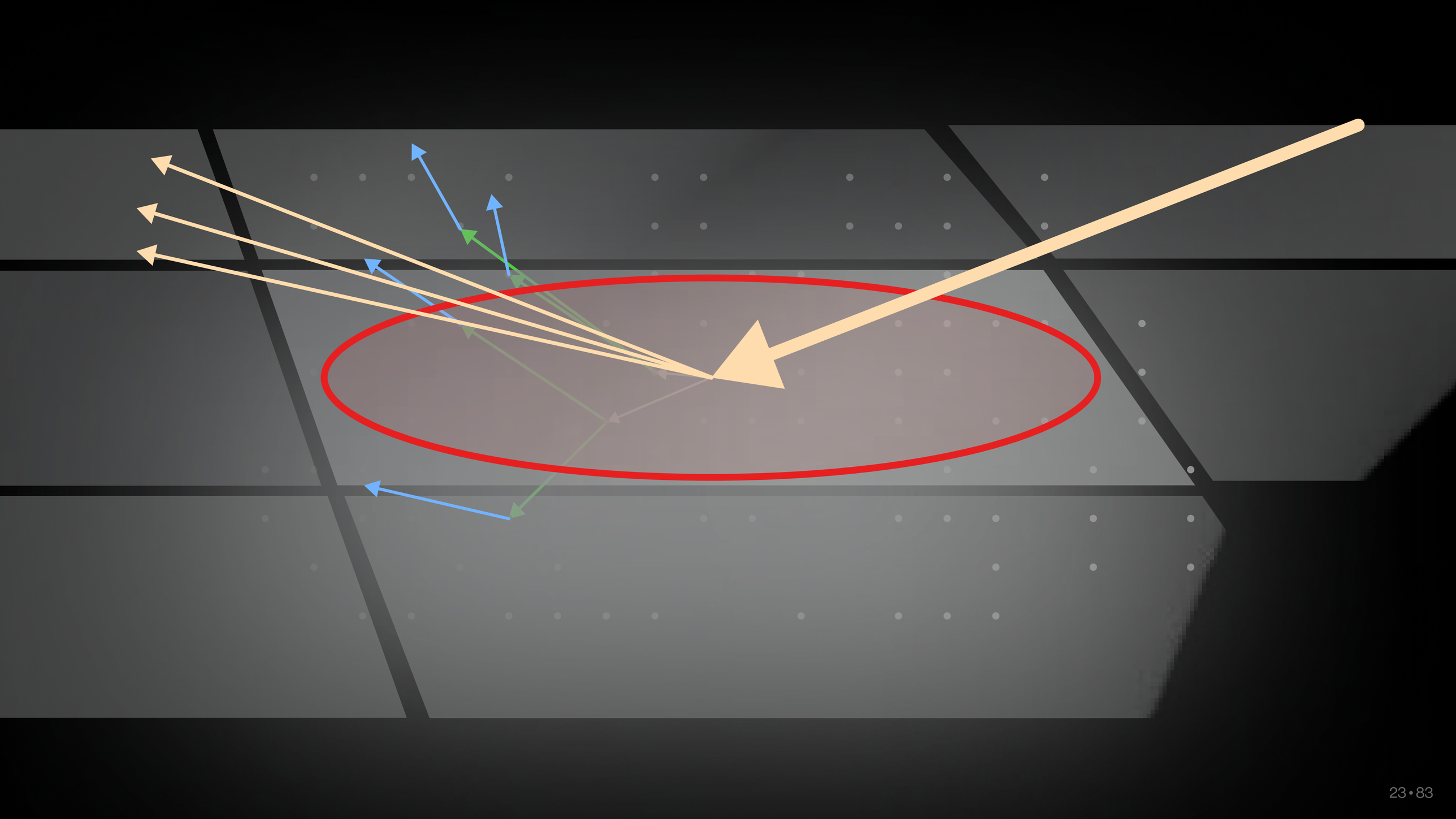


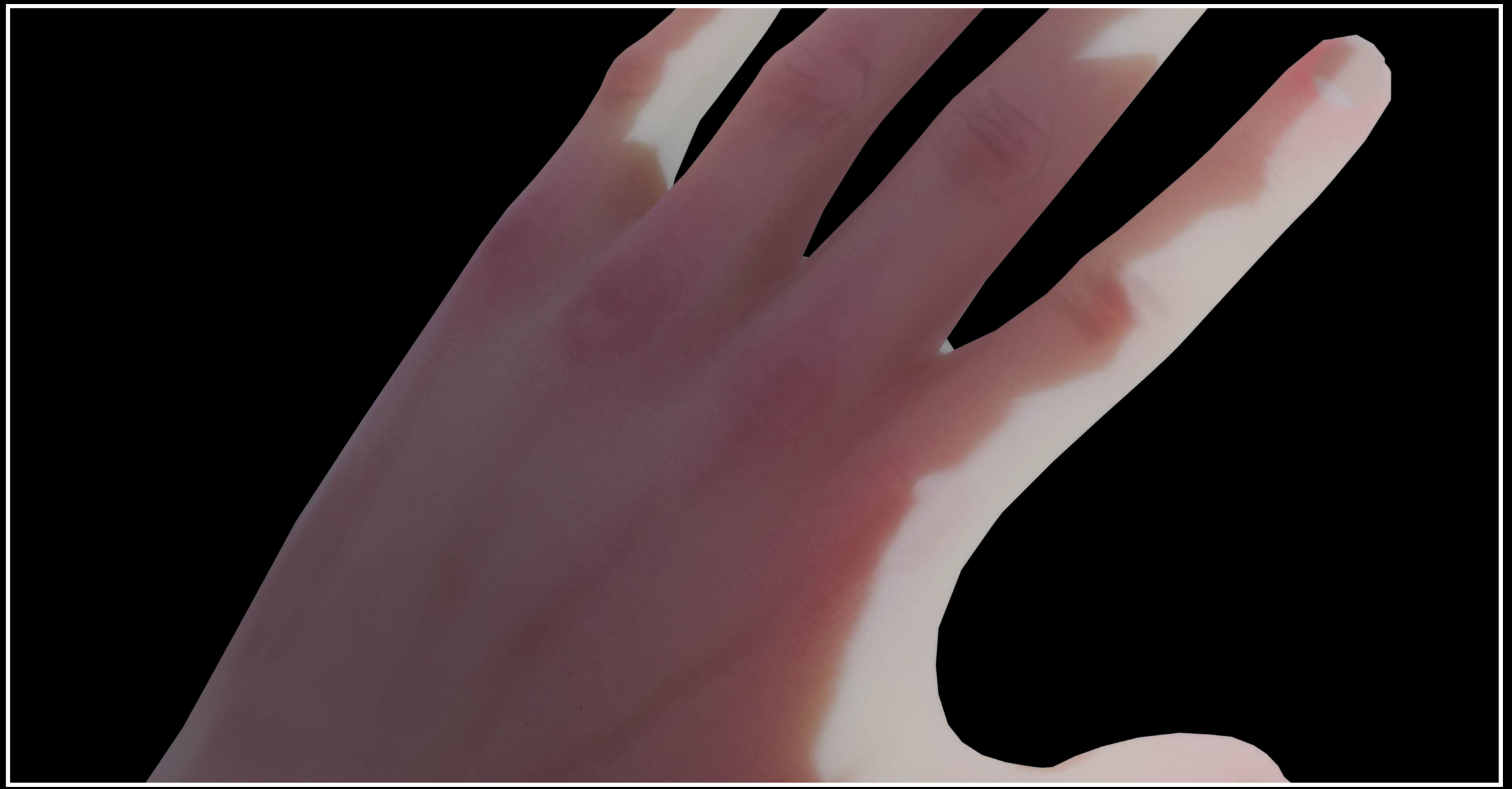












Subsurface Scattering - **OFF**

Subsurface Scattering - **ON**

Subsurface scattering can be mathematically described by a Bidirectional Surface Scattering Reflectance Distribution Function (BSSRDF):

$$S(x_i, \omega_i;\, x_o, \omega_o) = C\, F_t(x_i, \omega_i)\, R(|x_o - x_i|)\, F_t(x_o, \omega_o).$$

Outgoing radiance is obtained by integrating incident radiance over all surface points and directions:

$$L_o(x_o, \omega_o) = \int_A \int_{2\pi} S(x_i, \omega_i;\, x_o, \omega_o)\, L_i(x_i, \omega_i)\, (\mathbf{n}_i \cdot \omega_i)\, \mathrm{d}\omega_i\, \mathrm{d}A(x_i).$$

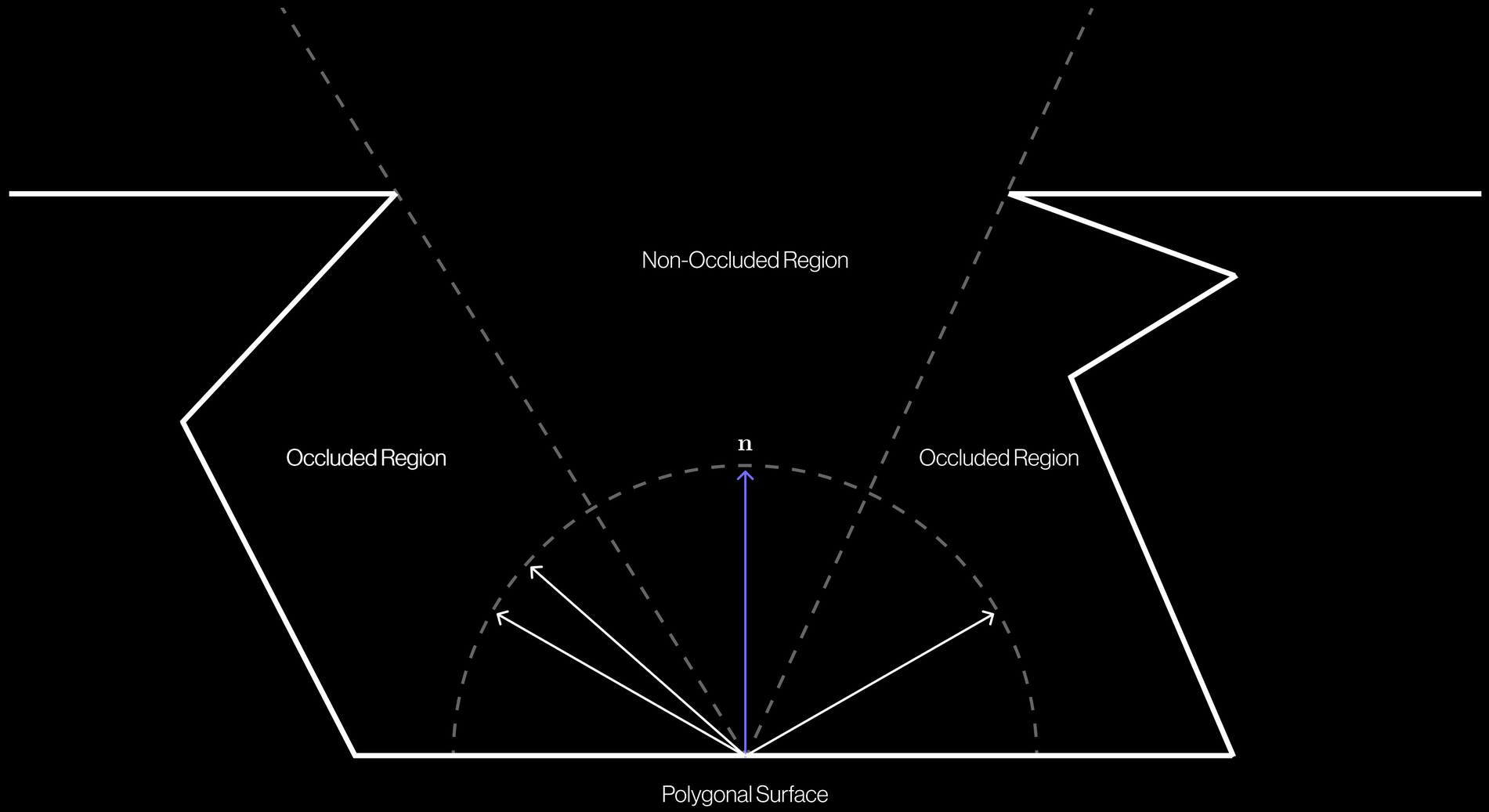
Fully physically based approaches are too expensive!

03-
REAL-TIME APPROXIMATION TECHNIQUES

Simple Translucency Approximation

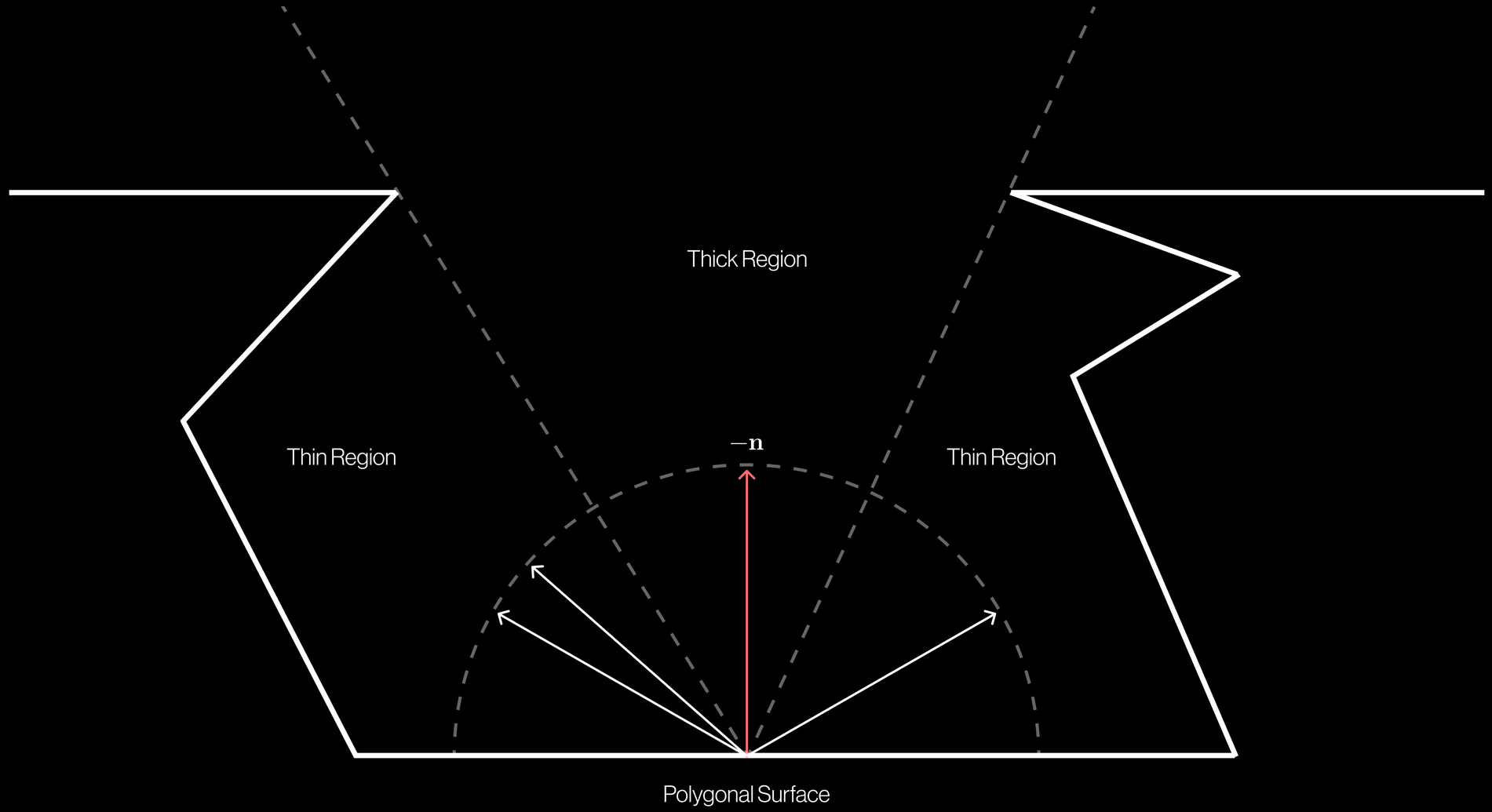
Image source: Rendered in Frostbite 2 Engine by Colin Barré-Brisebois & Marc Bouchard

The goal was to strip down SSS to the core visual cues .



Computing local material thickness

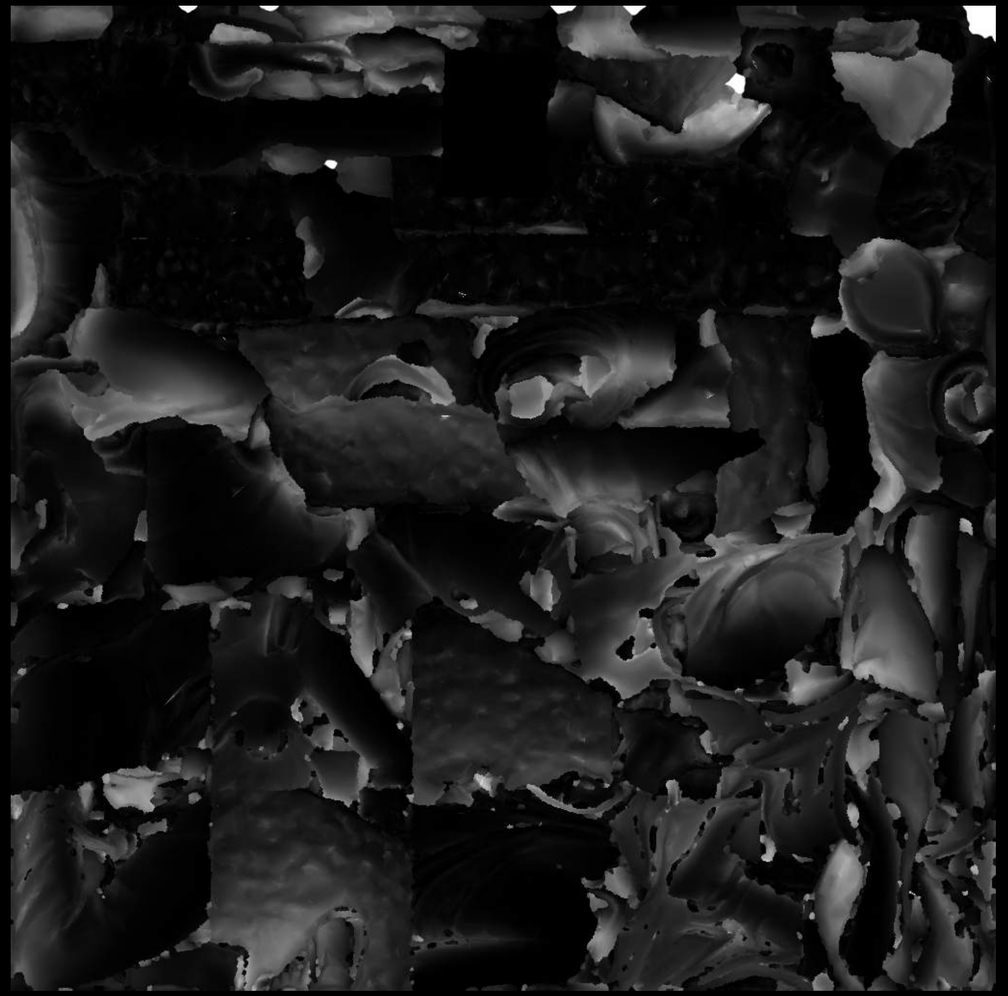
Ambient occlusion measures how enclosed a surface point is by nearby geometry by sampling the hemisphere around its normal.



Computing local material thickness

By flipping the surface normals inwards, AO samples the object's interior, causing thin regions to appear more occluded than thick ones.

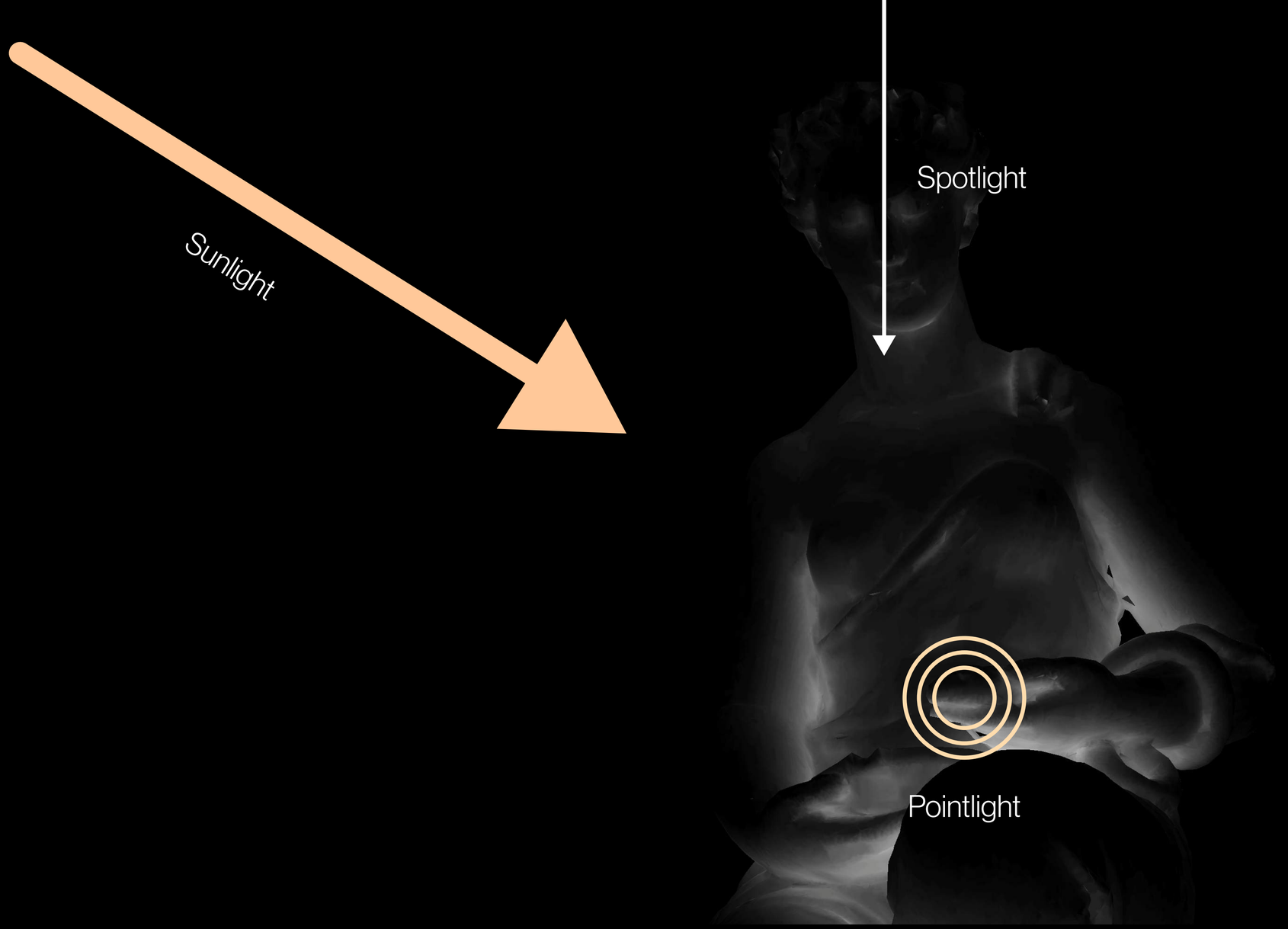
Invert



Computing local material thickness

The result is inverted so thin regions become high values and thick regions low, matching translucency behaviour.

Resulting Thickness Map



Lighting setup for the following part...

Angular Dependent Spreading

Translucency is approximated with a view-dependent back-lighting term, modulated by local thickness.

$$w_{\text{back}}(\mathbf{x}) = \max\Big(0,\ \mathbf{V} \cdot (-\mathbf{L})\Big)$$

View from the sunlight

View from opposite the sunlight

Angular Dependent Spreading

The light direction is perturbed along the surface normal to emphasise silhouettes, then evaluated with a powered back-lighting term.

$$\mathbf{L}_T = \frac{\mathbf{L} + d_T \mathbf{N}}{\|\mathbf{L} + d_T \mathbf{N}\|}$$

$$T_{\text{dir}}(\mathbf{x}) = s_T \left[\max\Big(0,\, \mathbf{V} \cdot (-\mathbf{L}_T)\Big)\right]^{p_T}$$

View from the sunlight

View from opposite the sunlight

Angular Dependent Spreading

The final translucency term modulates the directional response by local thickness and light attenuation, with a small ambient offset.

$$T(\mathbf{x}) = A_L(\mathbf{x})\, t(\mathbf{x})\, [T_{\text{dir}}(\mathbf{x}) + a_T]$$

View from the sunlight

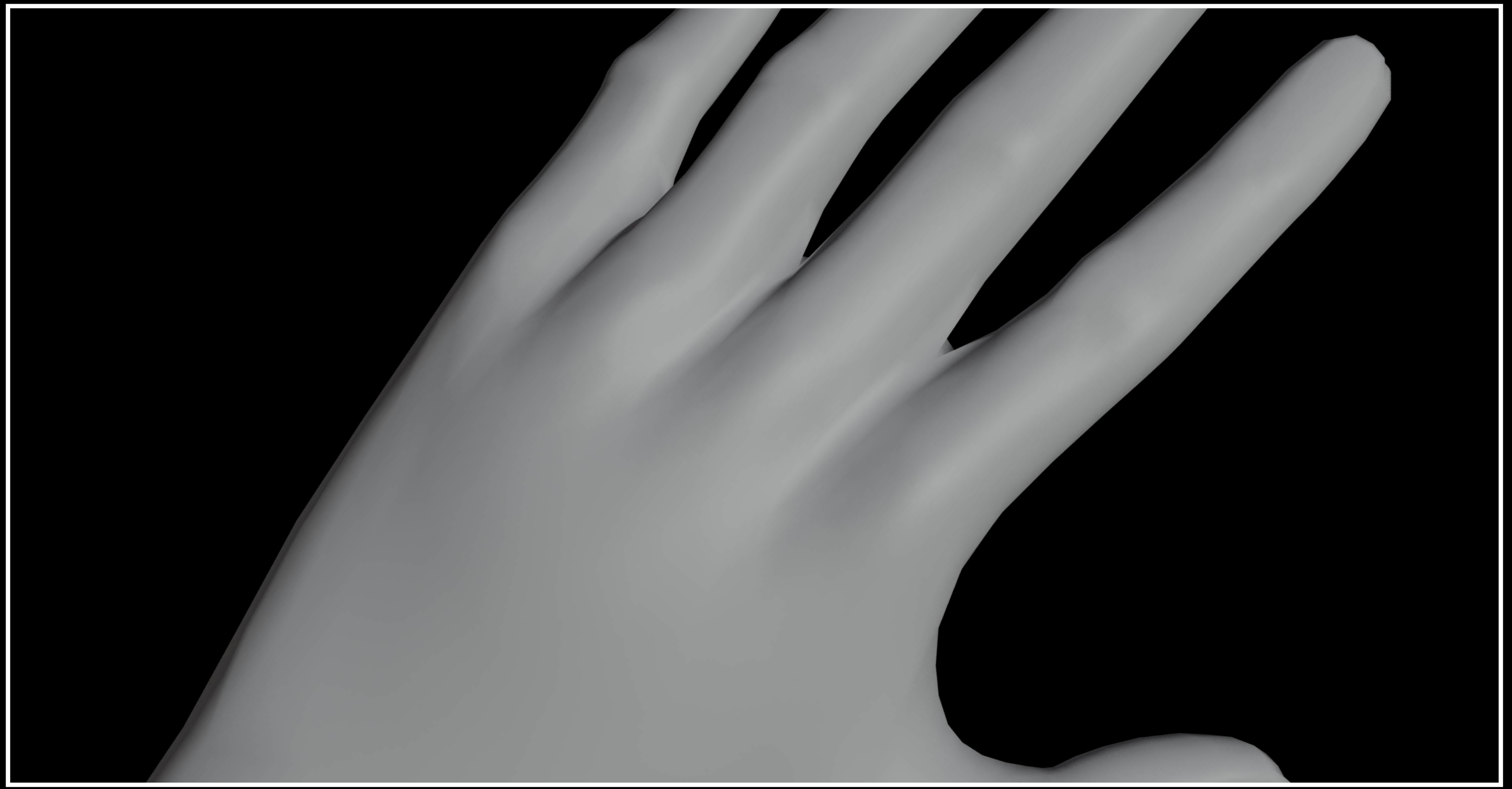
View from opposite the sunlight

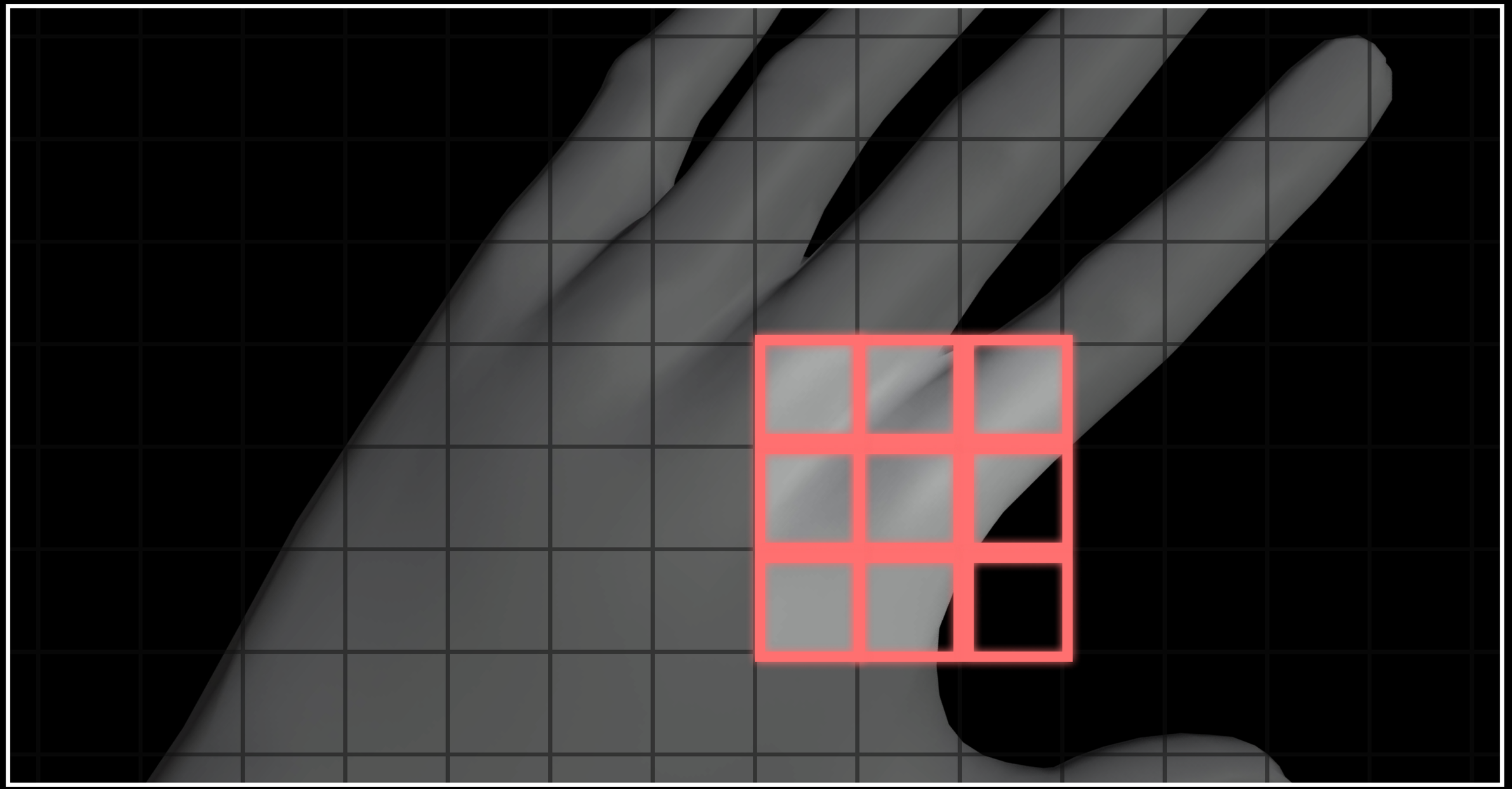
Final Result

Separable Subsurface Scattering

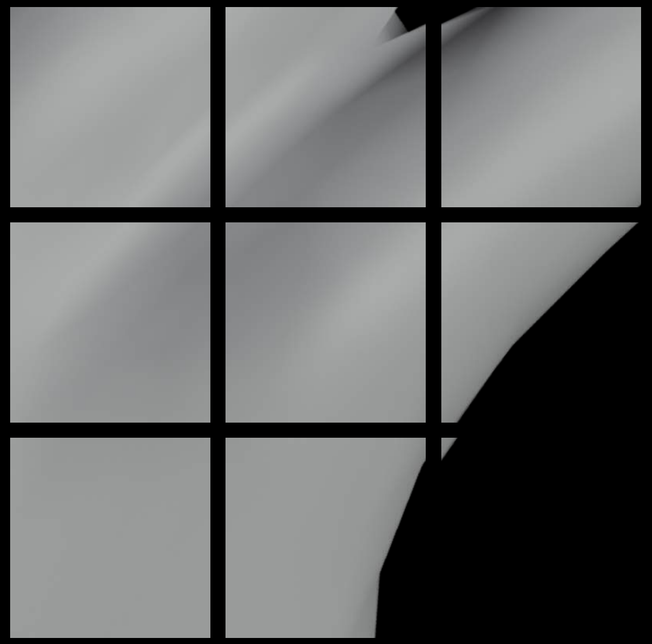
Image source: Black Myth: Wukong by Game Science rendered in Unreal Engine

How to gather lighting data from neighbouring surface points?



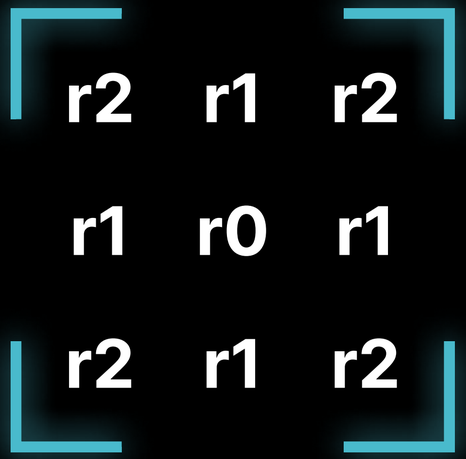


Let’s assign some example values to each “pixel”



12	18	15
20	30	22
14	19	16

A diffusion reflectance profile describes how neighbouring surface points contribute to the final shading.



For illustration, we discretise a diffusion reflectance profile into a small convolution kernel.

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Perform 2D convolution

$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \times \begin{bmatrix} 12 & 18 & 15 \\ 20 & 30 & 22 \\ 14 & 19 & 16 \end{bmatrix} \overset{\Sigma}{=} 20.9375$$

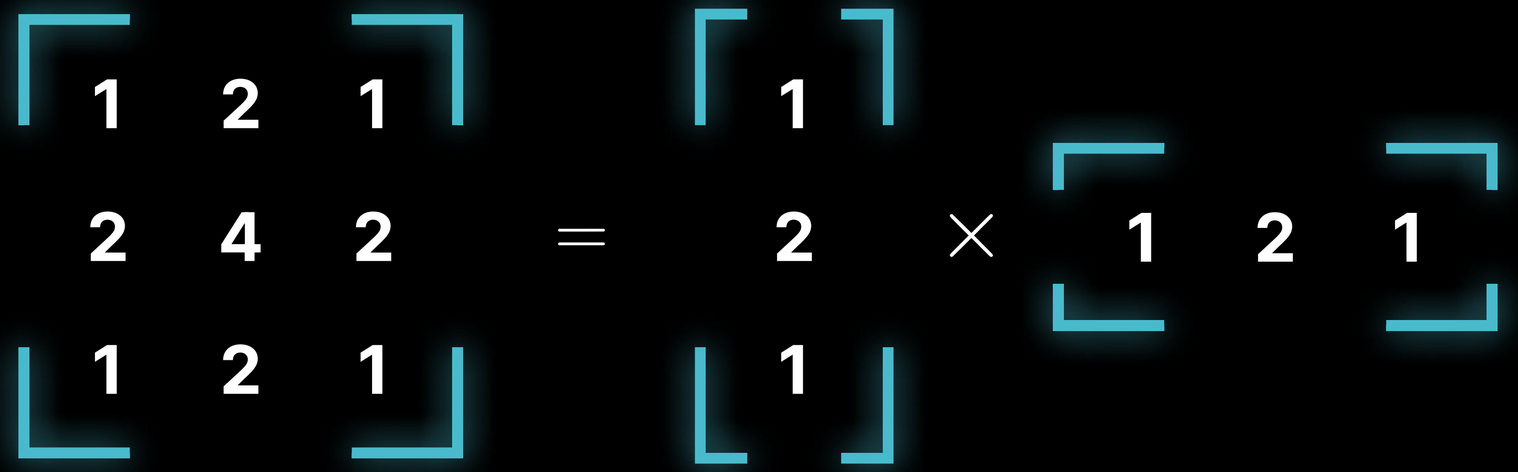
What if we could improve $O(N^2)$ to $O(N)$?

Split the kernel!

By separating the 2D kernel into two 1D kernels, we could achieve linear computational cost.

$$A(x, y) = a(x)\, a(y)$$

By separating the 2D kernel into two 1D kernels, we could achieve linear computational cost.



Here, instead of a full N×N convolution, the kernel is evaluated using two 1D passes with N samples each.

$$\frac{1}{4} \times \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \times \begin{bmatrix} 12 & 18 & 15 \\ 20 & 30 & 22 \\ 14 & 19 & 16 \end{bmatrix} = \begin{bmatrix} 15.75 \\ 25.5 \\ 17 \end{bmatrix}$$

Here, instead of a full N×N convolution, the kernel is evaluated using two 1D passes with N samples each.

$$\frac{1}{4} \times \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \times \begin{bmatrix} 15.75 \\ 25.5 \\ 17 \end{bmatrix} \overset{\Sigma}{=} 20.9375$$

How do we form our kernels?

Input | Artist-friendly kernel | Pre-integrated kernel | Ground Truth

Pre-integrated kernel

By pre-integrating the diffuse reflectance profile offline, we obtain a kernel that can be efficiently applied at runtime using separable filtering.

Artist-friendly kernel

A small combination of Gaussian functions, enabling intuitive control over short- and long-range scattering that the pre-integrated kernel lacks.

Image source: Jorge Jimenez's Separable Subsurface Scattering

Christensen-Burley's Normalised Diffusion Model

Image source: The Heretic by Unity rendered in Unity HDRP

To motivate the model, let's examine the diffuse reflectance profiles obtained by brute-force Monte Carlo simulation.

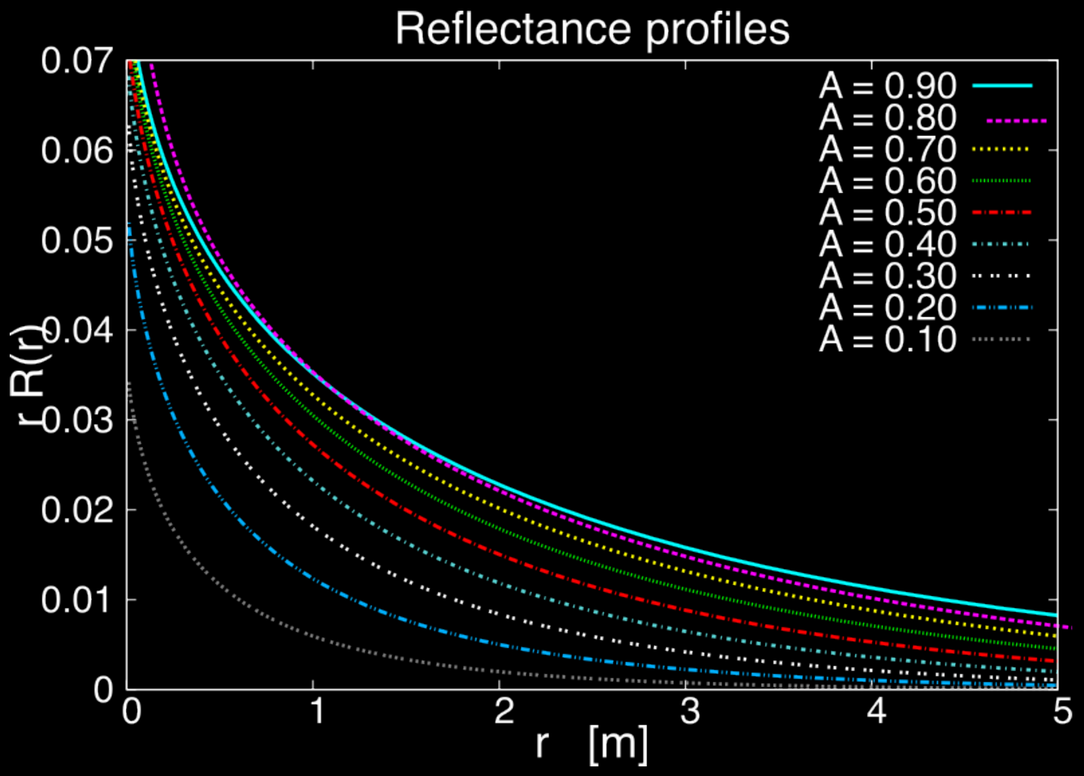
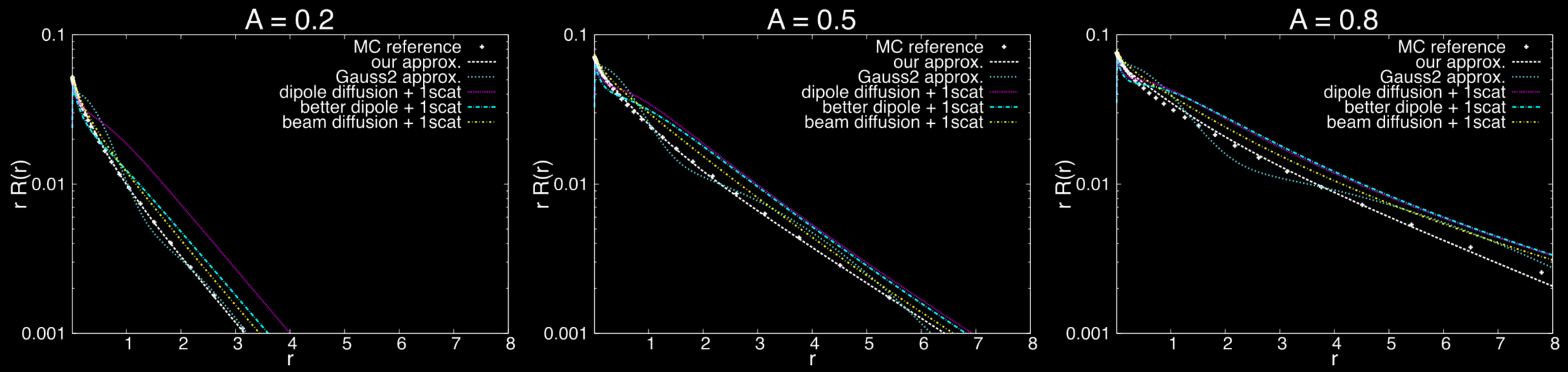


Image source: Per H. Christensen and Brent Burley's Approximate Reflectance Profiles for Efficient Subsurface Scattering

Rather than modelling the full physical light transport, this approach matches the shape of the diffusion reflectance profile $R(r)$ using a simple analytic curve.

$$R(r) = \frac{e^{-r/d} + e^{-r/(3d)}}{8\pi d\, r}$$

And it works!



Fit of various reflectance profiles for surface albedos A = 0.2, 0.5 and 0.8

Image source: Per H. Christensen and Brent Burley's Approximate Reflectance Profiles for Efficient Subsurface Scattering

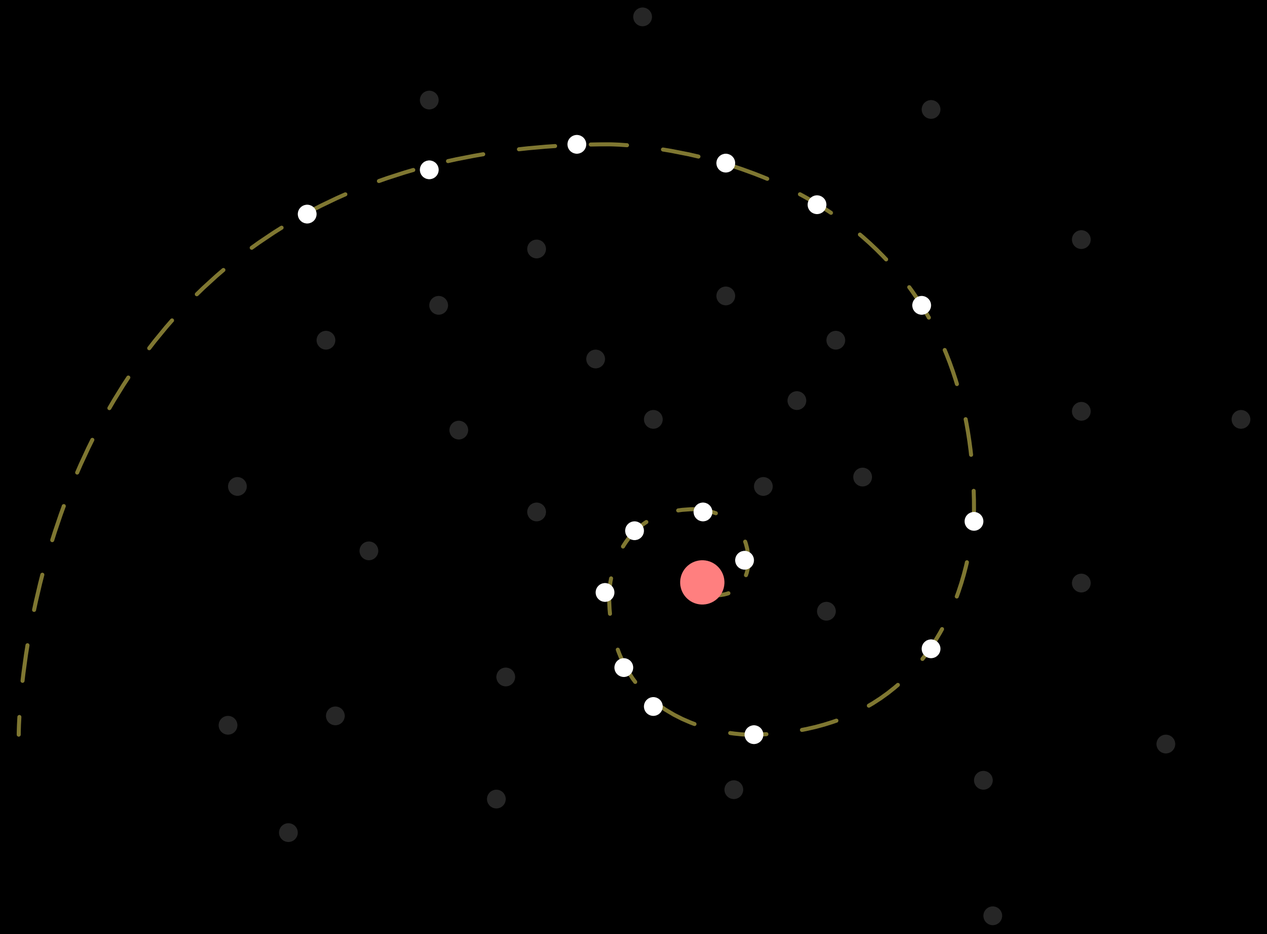
The remainder of this work focusses on finding a physically meaningful representation of the diffusion length d, which controls how far light spreads beneath the surface.

How does convolution work here?

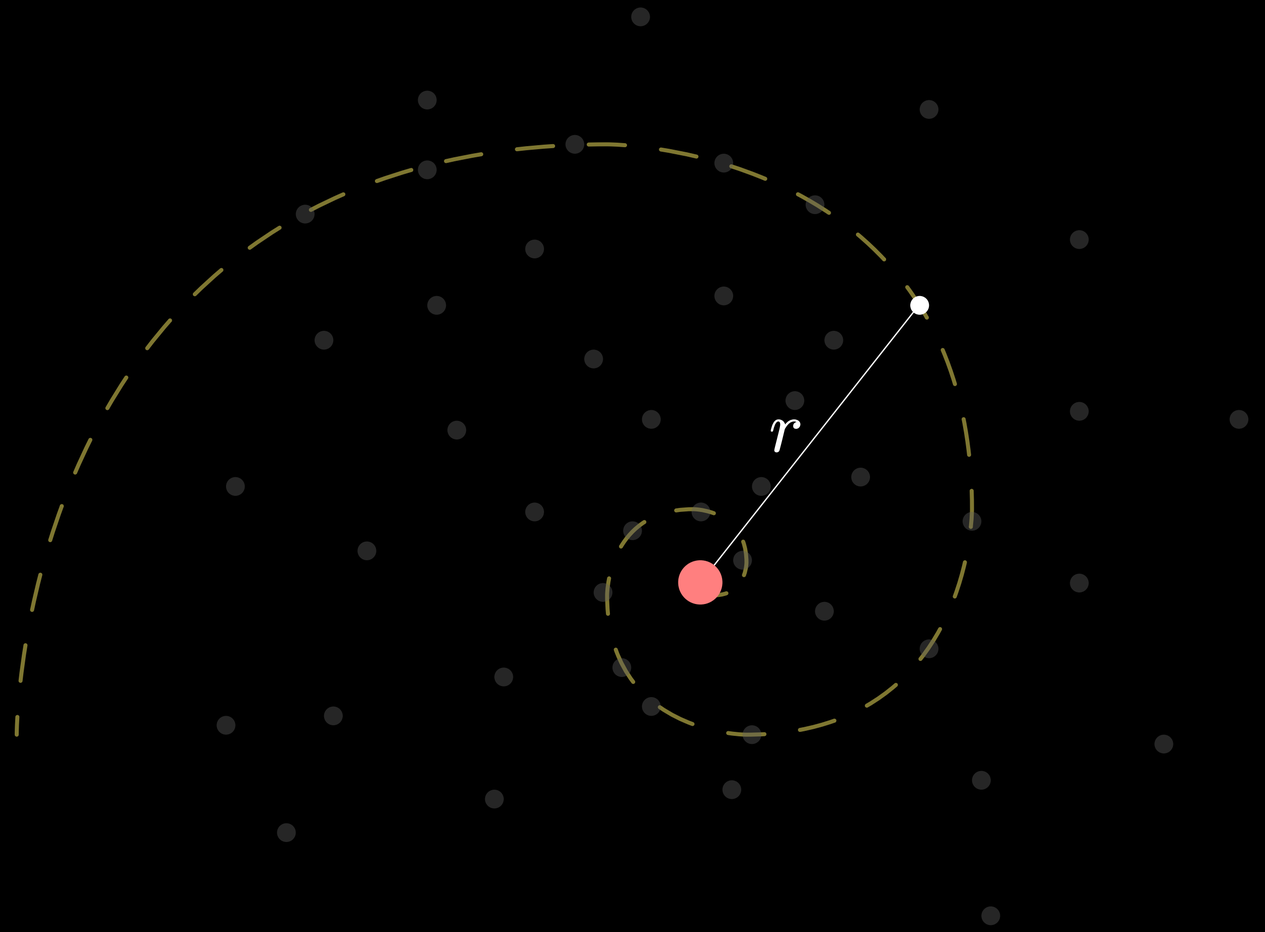
To achieve better performance, we use a fixed, precomputed spiral sampling pattern defined by the golden angle.

For a pixel p at runtime,

Sample neighbouring surface points using the defined sampling pattern:



For each neighbour, calculate its distance from p



The final colour of the pixel p is the convolution of the neighbours' irradiance with the diffusion reflectance function at their distances r_i

$$L_{\text{sss}}(p) = \sum_i R(r_i)\, L_i$$

04-

VISUAL FIDELITY VS. PERFORMANCE

Visual fidelity evaluates the models' ability to capture the most important aspects of SSS.

Visual Fidelity	Non-local subsurface light diffusion	Distance dependent colour bleeding	Energy conserving
STA			
SSSS			
CBNDM			

STA is a local shading model that uses backlighting to mimic SSS effects.

Visual Fidelity	Non-local subsurface light diffusion	Distance dependent colour bleeding	Energy conserving
STA	No		
SSSS			
CBNDM			

STA does not evaluate the results per colour channel.

Visual Fidelity	Non-local subsurface light diffusion	Distance dependent colour bleeding	Energy conserving
STA	No	No	
SSSS			
CBNDM			

STA simply adds transmitted light on top of existing shading without considering energy conservation.

Visual Fidelity	Non-local subsurface light diffusion	Distance dependent colour bleeding	Energy conserving
STA	No	No	No
SSSS			
CBNDM			

SSSS uses 1D kernels to sample its neighbours.

Visual Fidelity	Non-local subsurface light diffusion	Distance dependent colour bleeding	Energy conserving
STA	No	No	No
SSSS	Yes		
CBNDM			

The convolutions in SSSS could theoretically be done per colour channel if the diffusion profile of each colour (R,G,B) is provided.

Visual Fidelity	Non-local subsurface light diffusion	Distance dependent colour bleeding	Energy conserving
STA	No	No	No
SSSS	Yes	Yes (theoretically)	
CBNDM			

Energy conservation is only guaranteed in the convolution step.

Visual Fidelity	Non-local subsurface light diffusion	Distance dependent colour bleeding	Energy conserving
STA	No	No	No
SSSS	Yes	Yes (theoretically)	No
CBNDM			

CBNDM samples its neighbours with a spiral pattern defined by the golden angle.

Visual Fidelity	Non-local subsurface light diffusion	Distance dependent colour bleeding	Energy conserving
STA	No	No	No
SSSS	Yes	Yes (theoretically)	No
CBNDM	Yes		

CBNDM is normally evaluated per colour channel in practice.

Visual Fidelity	Non-local subsurface light diffusion	Distance dependent colour bleeding	Energy conserving
STA	No	No	No
SSSS	Yes	Yes (theoretically)	No
CBNDM	Yes	Yes	

CBNDM redistributes a fixed amount of energy rather than introducing additional illumination.

Visual Fidelity	Non-local subsurface light diffusion	Distance dependent colour bleeding	Energy conserving
STA	No	No	No
SSSS	Yes	Yes (theoretically)	No
CBNDM	Yes	Yes	Yes

STA has linear cost in screen pixels or vertices, depending on whether it runs in the fragment or vertex shader.

m_{screen} = #pixels on screen
m = #shaded pixels on screen
v = #vertices on the mesh
n = #number of neighbour samples
	Cost

STA	O(m) or O(v)
SSSS	
CBNDM	

SSSS performs two separable 1D convolutions, giving a cost of $O(m_{screen}n)$ before optimisation, and O(mn) if a stencil mask is used.

m_{screen} = #pixels on screen
m = #shaded pixels on screen
v = #vertices on the mesh
n = #number of neighbour samples
	Cost

STA	O(m) or O(v)
SSSS	$O(m_{screen}n)$ or O(mn)
CBNDM	

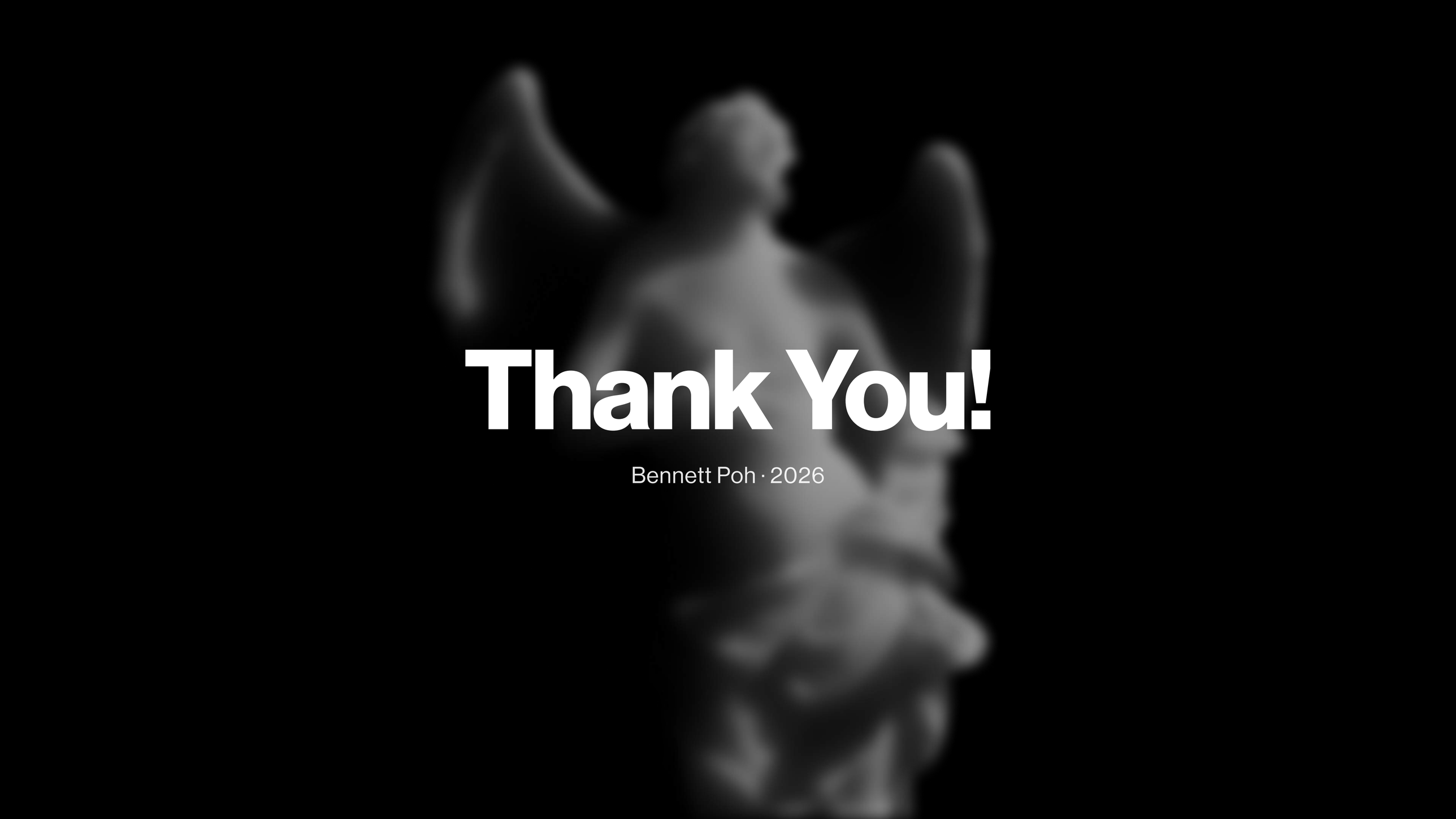
CBNDM evaluates a non-separable diffusion profile in a single pass, incurring the same costs as SSSS.

m_{screen} = #pixels on screen
m = #shaded pixels on screen
v = #vertices on the mesh
n = #number of neighbour samples
	Cost

STA	O(m) or O(v)
SSSS	O(m_{screen}n) or O(mn)
CBNDM	O(m_{screen}n) or O(mn)

05-

CONCLUSIONS AND TRADE-OFFS


Thank You!
Bennett Poh · 2026