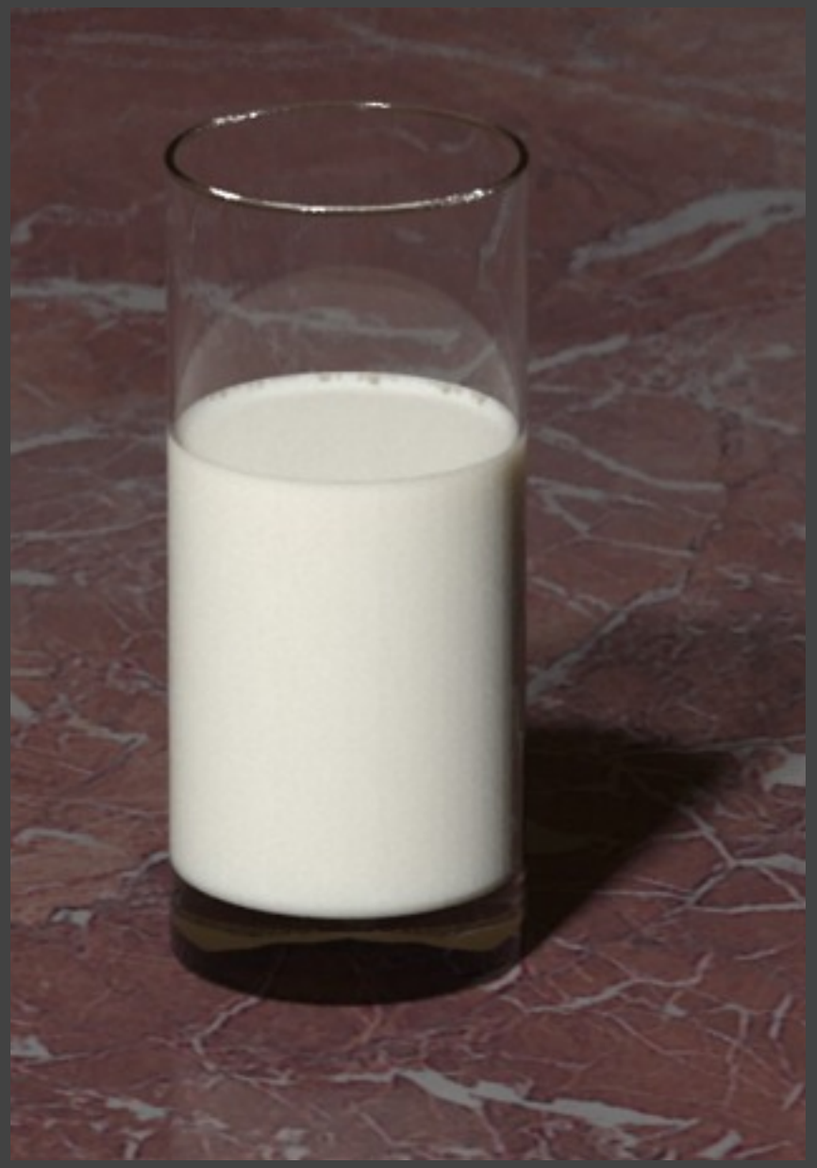
Multiple scattering

Steve Marschner
Cornell CS 6630 Fall 2009

Light diffusion

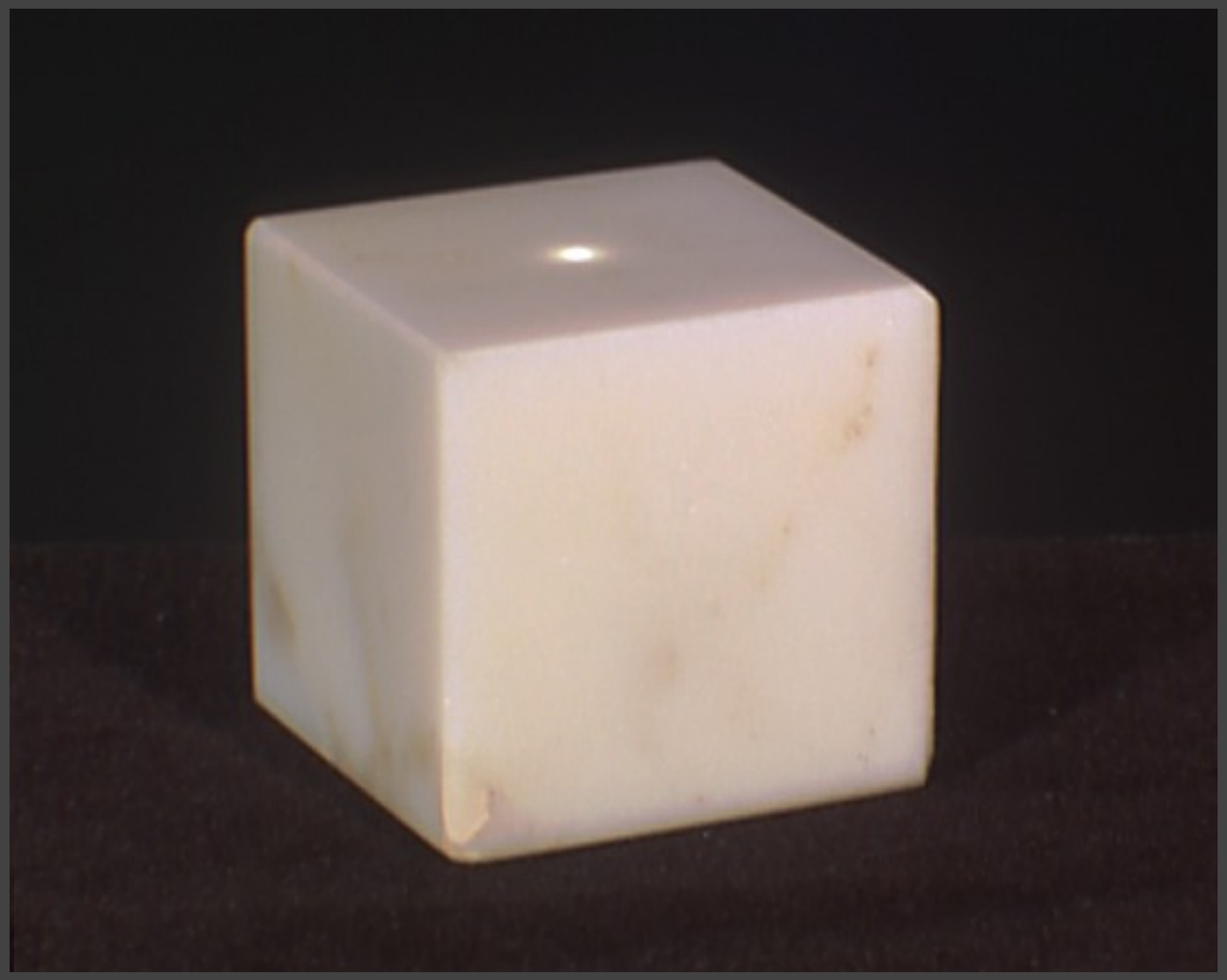
Skim milk

Skim milk



Whole milk

Marble sample

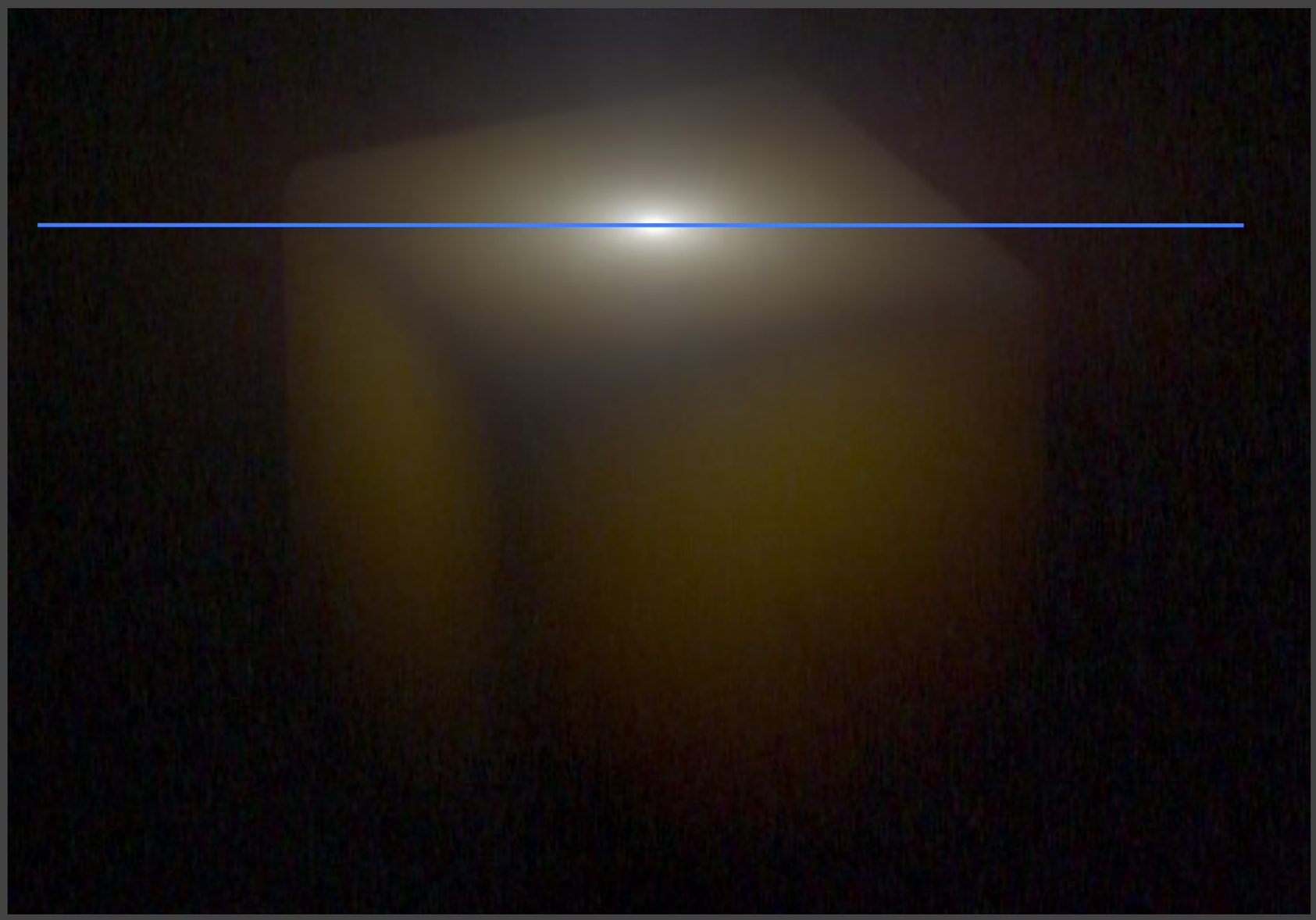


40mm cube of statuary marble

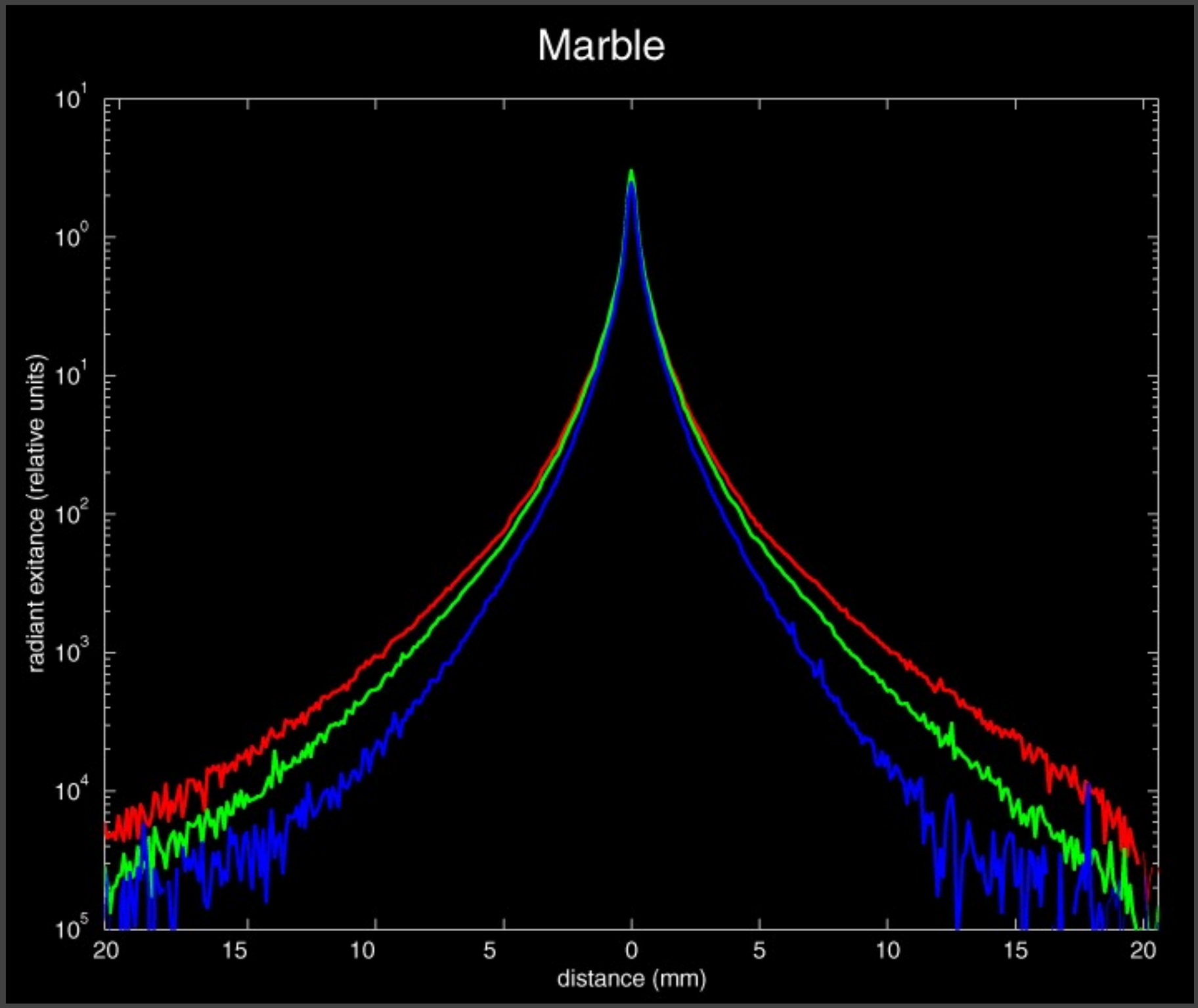
HDR photograph

(log scaled image)

HDR photograph



(log scaled image)


Marble
radiant exitance (relative units)
10^1
10^0
10^1
10^2
10^3
10^4
10^5
20
15
10
5
0
5
10
15
20
distance (mm)

Granular materials

Hair

Direct/indirect separation

Single scattering

All scattering

[Nayar et. al, SIGGRAPH'06]

Kajiya-style path tracing, surfaces, version 0:

rayRadianceEst(x, ω):
    y = traceRay(x, ω)
    return emittedRadiance(y, –ω) + reflectedRadianceEst(y, –ω)

reflectedRadianceEst(x, ω):
    ω′ = uniformRandomPSA(n(x))
    return π * brdf(x, ω, ω′) * rayRadianceEst(x, ω′)

Kajiya-style path tracing, homogeneous volume, version 0:

rayRadianceEst(x, ω):
    y = traceRay(x, ω)
    s = selectDistance(x, ω)
    if s < |y – x| then
        x′ = x – sω
        return scatteredRadianceEst(x′, ω′)
    else
        return emittedRadiance(y, –ω) + reflectedRadianceEst(y, –ω)

selectDistance(x, ω):
    ξ = random()
    return –log(ξ) / σ_t

scatteredRadianceEst(x, ω):
    ω′ = uniformRandomSA()
    return 4π * σ_s / σ_t * phaseFn(x, ω, ω′) * rayRadianceEst(x, ω′)

reflectedRadianceEst(x, ω):
    ω′ = uniformRandomPSA(n(x))
    return π * brdf(x, ω, ω′) * rayRadianceEst(x, ω′)

Kajiya-style path tracing, surfaces, version 0.5:

rayRadianceEst(x, ω):
    y = traceRay(x, ω)
    return emittedRadiance(y, –ω) + reflectedRadianceEst(y, –ω)

reflectedRadianceEst(x, ω):
    if random() < survivalProbability:
        ω′ = uniformRandomPSA(n(x))
        return π * brdf(x, ω, ω′) * rayRadianceEst(x, ω′) / survivalProbability
    else
        return 0

Kajiya-style path tracing, homogeneous volume, version 0.5:

rayRadianceEst(x, ω):
    y = traceRay(x, ω)
    s = selectDistance(x, ω)
    if s < |y – x| then
        x′ = x – sω
        return scatteredRadianceEst(x′, ω′)
    else
        return emittedRadiance(y, –ω) + reflectedRadianceEst(y, –ω)

selectDistance(x, ω):
    ξ = random()
    return –log(ξ) / σ_t

scatteredRadianceEst(x, ω):
    if random() < survivalProbability:
        ω′ = uniformRandomSA()
        return 4π * σ_s / σ_t * phaseFn(x, ω, ω′) * rayRadianceEst(x, ω′) / survivalProbability
    else
        return 0;

reflectedRadianceEst(x, ω):
    if random() < survivalProbability:
        ω′ = uniformRandomPSA(n(x))
        return π * brdf(x, ω, ω′) * rayRadianceEst(x, ω′) / survivalProbability
    else
        return 0

Kajiya-style path tracing, surfaces, version 0.75:

rayRadianceEst(x, ω):
    y = traceRay(x, ω)
    return emittedRadiance(y, –ω) + reflectedRadianceEst(y, –ω)

reflectedRadianceEst(x, ω):
    if random() < survivalProbability:
        ω′, pdf = brdfSample(x, n(x))
        return brdf(x, ω, ω′) * rayRadianceEst(x, ω′) / (pdf * survivalProbability)
    else
        return 0

Kajiya-style path tracing, homogeneous volume, version 0.75:

rayRadianceEst(x, ω):
    y = traceRay(x, ω)
    s = selectDistance(x, ω)
    if s < |y – x| then
        x′ = x – sω
        return scatteredRadianceEst(x′, ω′)
    else
        return emittedRadiance(y, –ω) + reflectedRadianceEst(y, –ω)

selectDistance(x, ω):
    ξ = random()
    return –log(ξ) / σ_t

scatteredRadianceEst(x, ω):
    if random() < survivalProbability:
        ω′, pdf = phaseFunctionSample()
        return σ_s / σ_t * phaseFn(x, ω, ω′) * rayRadianceEst(x, ω′) / (pdf * survivalProbability)
    else
        return 0;

reflectedRadianceEst(x, ω):
    if random() < survivalProbability:
        ω′, pdf = brdfSample(x, n(x))
        return brdf(x, ω, ω′) * rayRadianceEst(x, ω′) / (pdf * survivalProbability)
    else
        return 0

Kajiya-style path tracing, surfaces, version 1.0:

rayRadianceEst(x, ω):
    y = traceRay(x, ω)
    return emittedRadiance(y, –ω)
        + reflectedRadianceEst(y, –ω)

directRadianceEst(x, ω):
    ω′, pdf = luminaireSamplePSA(x, n(x))
    y = traceRay(x, ω′)
    return brdf(x, ω, ω′)
        * emittedRadiance(y, –ω′) / pdf

reflectedRadianceEst(x, ω):
    return directRadianceEst(x, ω)
        + indirectRadianceEst(x, ω)

indirectRadianceEst(x, ω):
    if random() < survivalProbability:
        ω′, pdf = brdfSample(x, n(x))
        y = traceRay(x, ω′)
        return brdf(x, ω, ω′)
            * reflectedRadianceEst(y, –ω′)
            / (pdf * survivalProbability)
    else:
        return 0

Kajiya-style path tracing, homogeneous volume, version 1.0:

```
rayRadianceEst(x, ω, e = 1):
    y = traceRay(x, ω)
    s = selectDistance(x, ω)
    if s < |y – x| then
        x′ = x – sω
        return scatteredRadianceEst(x′, ω′)
    else
        return e * emittedRadiance(y, –ω)
            + reflectedRadianceEst(y, –ω)

selectDistance(x, ω):
    ξ = random()
    return –log(ξ) / σt

directReflectedEst(x, ω):
    ω′, pdf = luminaireSamplePSA(x, n(x))
    y = traceRay(x, ω′)
    return brdf(x, ω, ω′)
        * attenuation(x, y)
        * emittedRadiance(y, –ω′) / pdf

directScatteredEst(x, ω):
    ω′, pdf = luminaireSampleSA(x)
    y = traceRay(x, ω′)
    return phaseFn(x, ω, ω′)
        * attenuation(x, y)
        * emittedRadiance(y, –ω′) / pdf

scatteredRadianceEst(x, ω):
    return directScatteredEst(x, ω)
        + indirectScatteredEst(x, ω)

reflectedRadianceEst(x, ω):
    return directReflectedEst(x, ω)
        + indirectReflectedEst(x, ω)

indirectReflectedEst(x, ω):
    if random() < survivalProbability:
        ω′, pdf = brdfSample(x, n(x))
        return brdf(x, ω, ω′)
            * rayRadianceEst(x, ω′, 0)
            / (pdf * survivalProbability)
    else:
        return 0

indirectScatteredEst(x, ω):
    if random() < survivalProbability:
        ω′, pdf = phaseFunctionSample()
        return σs / σt * phaseFn(x, ω, ω′)
            * rayRadianceEst(x, ω′, 0)
            / (pdf * survivalProbability)
    else
        return 0;
```