tics & (depth < >voca

: = inside ? 1 ( 1 ) ht = nt / nc, ddn bs2t = 1.0f - n⊓t ~ ( 0, N ); ≫)

at a = nt - nc, b = nt at Tr = 1 - (R0 + (1 - R0 Fr) R = (D <sup>=</sup> nnt - N - (dd)

\* diffuse; = true;

. efl + refr)) && (depth < MAXDEPTI

D, N ); refl \* E \* diffuse; = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly ff; radiance = SampleLight( &rand, I ) e.x + radiance.y + radiance.z) > **16 (2**)

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Psurvis at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely following Sec /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:  $\epsilon(x,x')$ 

= g(x, x')

## **INFOMAGR – Advanced Graphics**

Jacco Bikker - November 2021 - February 2022

# Lecture 2 - "Whitted"

(x, x', x'')I(x', x'')dx''

## Welcome!



ics & (depth < Mox000

: = inside ? l | ... ht = nt / nc, ddn - ... os2t = 1.0f - nnt - n O, N ); ð)

at a = nt - nc, b = nt - r at Tr = 1 - (R0 + (1 - R0 Tr) R = (D <sup>#</sup> nnt - N = (ddm -

= \* diffuse = true;

-:fl + refr)) && (depth < MODEPT

), N ); refl \* E \* diffus = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, closed if; radiance = SampleLight( &rand, I, &L, &ilent 2.x + radiance.y + radiance.z) > 0) &

v = true; at brdfPdf = EvaluateDiffuse( L, N ) Pourvive at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) (nud

andom walk - done properly, closely following Sec. /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Today's Agenda:

- Introduction: Appel
- Whitted
- Cook



## Appel

at a = nt - nc,

= true:

), N ); = true;

#### AXDEPTH)

survive = SurvivalP lf: radiance = SampleLi .x + radiance.y +

v = true; at brdfPdf = Evalua at3 factor = diffus at weight = Mis2( at cosThetaOut = E \* ((weight \* cos

andom walk - done /ive)

at3 brdf = SampleDi urvive; pdf;

1 = E \* brdf \* (dot( N, R ) / pdf); sion = true:

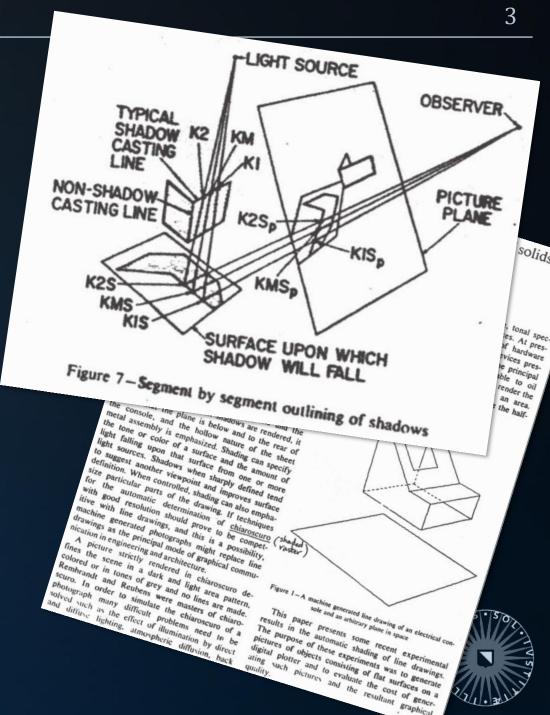
di

Some Techniques for Shading Machine Renderings of Solids, Arthur Appel, 1964.

Idea: use rays to find geometry and shadows.

"Graphics" for games in 1964: https://en.wikipedia.org/wiki/The\_Sumerian\_Game





## Appel

ics & (depth < Not00 ⊡

at a = nt - nc, b = nt - nc at Tr = 1 - (R0 + (1 - R0 Ir) R = (D = nnt - N = (ddn

= \* diffuse; = true;

-•fl + refr)) && (denth < MA)

D, N ); refl \* E \* diffu: = true;

AXDEPTH)

survive = SurvivalProl estimation - doing if if; radiance = SampleLight 2.x + radiance.y + rad

w = true; at brdfPdf = EvaluateDiffuse( L, N ) Ps at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

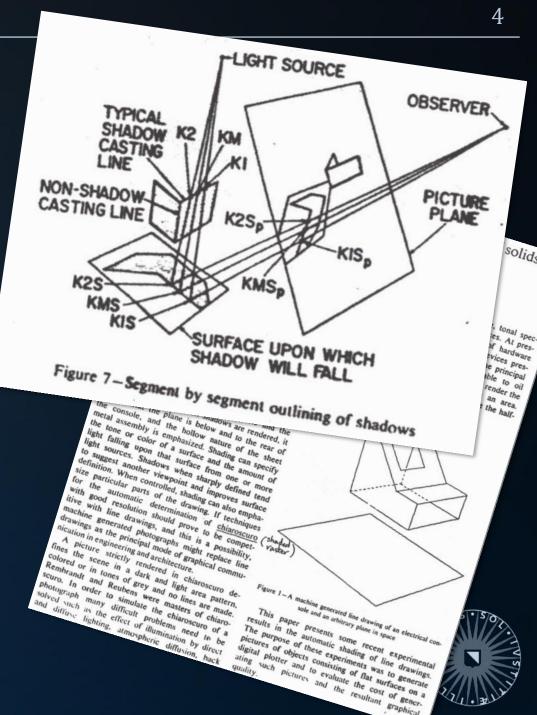
andom walk - done properly, closely following Soul /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, apdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

Some Techniques for Shading Machine Renderings of Solids, Arthur Appel, 1964.

Idea: use rays to find geometry and shadows.

This method is very time consuming, usually requiring for useful results several thousand times as much calculation time as a wire frame drawing. About one half of this time is devoted to determining the



## Appel

tics & (depth < Poxee

c = inside 7 1 ht = nt / nc, ddn bs2t = 1.0f - nnt D, N ); D)

at a = nt - nc, b = nt - n at Tr = 1 - (R0 + (1 - R0 Fr) R = (D = nnt - N - (dd

= \* diffuse = true;

. :fl + refr)) && (depth

), N ); ∙efl \* E \* diff = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, doing if; radiance = SampleLight( &rand, I, &L, &II) 2.x + radiance.y + radiance.z) > 0) &&

E

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* PsurvL at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely following Sec /ive)

; t3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf ; urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

Recap

 $P_1$ 

U

 $P_0$ 

Plane:  $P \cdot \vec{N} + d = 0$ Ray:  $P(t) = O + t\vec{D}$ Substituting for P(t), we get  $(O + t\vec{D}) \cdot \vec{N} + d = 0$   $t = -(O \cdot \vec{N} + d)/(\vec{D} \cdot \vec{N})$  $P = O + t\vec{D}$ 

Sphere:  $(P - C) \cdot (P - C) - r^2 = 0$ 

Substituting for P(t), we get

 $(0+t\vec{D}-\vec{C})\cdot(0+t\vec{D}-\vec{C})-r^2 = 0$  $\vec{D}\cdot\vec{D}\ t^2 + 2\vec{D}\cdot(0-\vec{C})\ t + (0-\vec{C})^2 - r^2 = 0$ 

 $at^2 + bt + c = 0 \rightarrow t = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$ 

 $a = \vec{D} \cdot \vec{D}$   $b = 2\vec{D} \cdot (0 - C)$  $c = (0 - C) \cdot (0 - C) - r^2$ 



## Appel

Ray:  $P(t) = 0 + t\vec{D}$ 

: = inside ? 1 1 1 ht = nt / nc, ddn os2t = 1.0f - nmt ? 0, N ); 3)

at a = nt - nc, b = nt - nc at Tr = 1 - (R0 + (1 - R0 Ir) R = (D = nnt - N - (00)

= \* diffuse; = true;

efl + refr)) && (depth < MODEPTH

), N ); refl \* E \* diffus = true;

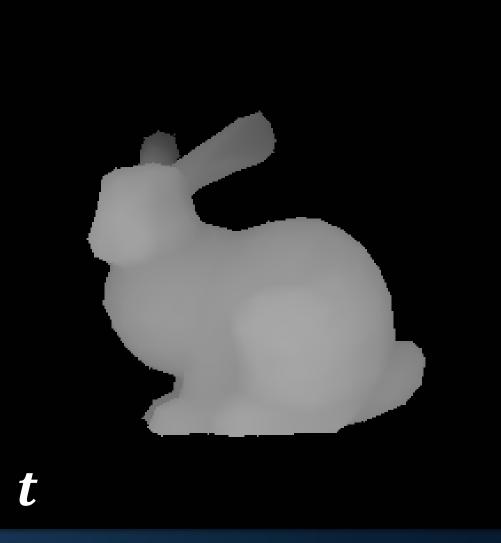
AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, closed Hf; radiance = SampleLight( &rand, I, &L, &light) 2.x + radiance.y + radiance.z) > 0) &&

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pourvive at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* (rad

andom walk - done properly, closely following Soul /ive)

; t3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); Sion = true:





## Appel

tics & (depth < ≯voca

c = inside ? 1 ( ) ) ht = nt / nc, ddn ps2t = 1.0f - nnt \* 2, N ); 3)

at a = nt - nc, b = nt - nc at Tr = 1 - (R0 + (1 - R0 Fr) R = (D <sup>=</sup> nnt - N <sup>-</sup> (dd)

= \* diffuse = true;

. efl + refr)) && (depth < MOXDEPTH)

D, N ); refl \* E \* diffus = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly if; radiance = SampleLight( &rand, I, &L, &II ext + radiance.y + radiance.z) > 0) && (doctor)

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Psurvive at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* (radial);

andom walk - done properly, closely following Sec. /ive)

; t3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); Sion = true:





ics & (depth < Mox000

: = inside ? l | ... ht = nt / nc, ddn - ... os2t = 1.0f - nnt - n O, N ); ð)

at a = nt - nc, b = nt - r at Tr = 1 - (R0 + (1 - R0 Tr) R = (D <sup>#</sup> nnt - N = (ddm -

= \* diffuse = true;

-:fl + refr)) && (depth < MODEPT

), N ); refl \* E \* diffus = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, closed if; radiance = SampleLight( &rand, I, &L, &ilent 2.x + radiance.y + radiance.z) > 0) &

v = true; at brdfPdf = EvaluateDiffuse( L, N ) Pourvive at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) (nud

andom walk - done properly, closely following Sec. /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Today's Agenda:

- Introduction: Appel
- Whitted
- Cook



## Whitted

An Improved Illumination Model for Shaded Display

In 1980, "State of the Art" consisted of:

- Rasterization
- Shading: either diffuse  $(N \cdot L)$  or specular  $((N \cdot H)^n)$ , both not taking into account fall-off (Phong)
- Reflection, using environment maps (Blinn & Newell \*)
- Stencil shadows (Williams \*\*)



andom walk - done properl /ive)

= true:

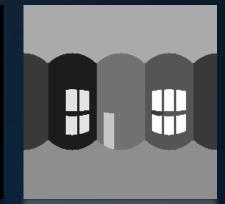
lf;

efl + refr)) && (dep

at3 brdf = SampleDiffuse( diffuse, N, r1 urvive; pdf; 1 = E \* brdf \* (dot( N, R ) / pdf); sion = true







\* : Blinn, J. and Newell, M. 1976. Texture and Reflection in Computer Generated Images.

\*\* : Williams, L. 1978. Casting curved shadows on curved surfaces..



## Whitted

at a = n

), N );

AXDEPTH)

lf:

efl + refr)) && (dept

refl \* E \* diffuse;

survive = SurvivalProbability(

radiance = SampleLight( &rand,

An Improved Illumination Model for Shaded Display

In 1980, "State of the Art" consisted of:

- Rasterization
- Shading: either diffuse (N · L) or specular ((N · H)<sup>n</sup>), both not taking into account fall-off (Phong)
- Reflection, using environment maps (Blinn & Newell)
- Stencil shadows (Williams)

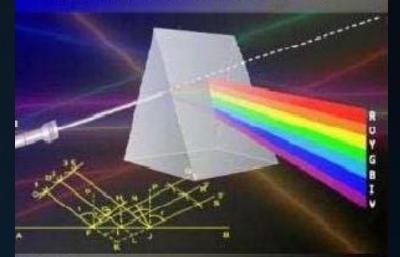
### Goal:

Solve reflection and refraction

#### Improved model:

Based on classical ray optics

## RAY OPTICS (PHYSICS : For Classes IX - XII)





National Council of Educational Research and Training



e.x + radiance.y + radiance.z) > 0) 88 (3 w = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Ps at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at weight = 0 = 0 = 0 = 0 = 0

at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) = (

andom walk - done properly, closely following Same /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Whitted

ics & (depth < Model)

: = inside ? 1 ht = nt / nc, ddn bs2t = 1.0f - nmt 0, N ); ∂)

at a = nt - nc, b = nt + nc at Tr = 1 - (R0 + (1 - R0 Fr) R = (D = nnt - N - (ddn

= \* diffuse; = true;

efl + refr)) && (depth < MAXDEPTH

), N ); refl \* E \* diffuse; = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly if; radiance = SampleLight( &rand, I, &L, &Light e.x + radiance.y + radiance.z) > 0) && doing

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Psurvise at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* C

andom walk - done properly, closely followi /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

An Improved Illumination Model for Shaded Display\*

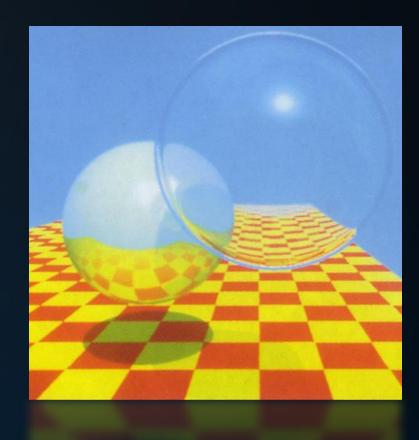
Physical basis of Whitted-style ray tracing:

Light paths are generated (backwards) from the camera to the light sources, using rays to simulate optics.

Color Trace( ray r )
 I, N, mat = NearestIntersection( scene, r )

return mat.color \* DirectIllumination( I, N )

\* : T. Whitted. An Improved Illumination Model for Shaded Display. Commun. ACM, 23(6):343–349, 1980.





## Whitted

tics & (depth < Moosen

: = inside ? 1 1 1 2 ht = nt / nc, ddn - 1 ps2t = 1.0f - nmt - 1 2, N ); 3)

at a = nt - nc, b = nt + n at Tr = 1 - (R0 + (1 - R0 Fr) R = (D <sup>\*</sup> nnt - N \* (33)

= \* diffuse; = true;

efl + refr)) && (depth < MAXDEPTI

D, N ); refl \* E \* diffuse = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, closed if; radiance = SampleLight( &rand, I, &L, &l e.x + radiance.y + radiance.z) > 0) &&

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* P at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf

andom walk - done properly, closely following 30 /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

An Improved Illumination Model for Shaded Display

Color Trace( ray r )

I,  $\vec{N}$ , mat = NearestIntersection( scene, r ) return mat.color \* DirectIllumination( I,  $\vec{N}$  )

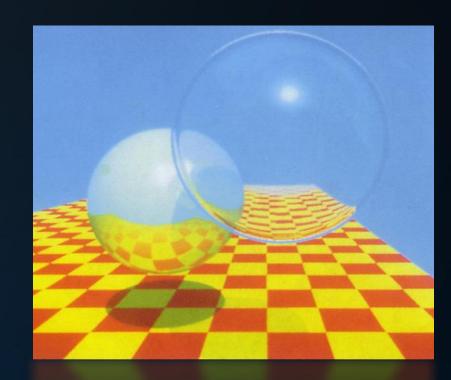
Direct illumination:

*Summed* contribution of *unoccluded* point light sources, taking into account:

Distance to I

Angle between  $\vec{L}$  and  $\vec{N}$ Intensity of light source

Note that this requires a ray per light source.





## Whitted

tics Adepth: < MacDarth

= inside ? 1
ht = nt / nc, ddn
os2t = 1.0f - nnt n
0, N );
3)

at a = nt - nc, b = nt at Tr = 1 - (R0 + (1 - R0 Tr) R = (D = nnt - N - (3 - 1)

\* diffuse; = true;

. :fl + refr)) && (depth < MOXDED)

D, N ); refl \* E \* diffuse; = true;

#### AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly if; radiance = SampleLight( &rand, I, &L, &L 2.x + radiance.y + radiance.z) > 0) &&

v = true;

at brdfPdf = EvaluateDiffuse( L, N )
at3 factor = diffuse \* INVPI;
at weight = Mis2( directPdf, brdfPdf
at cosThetaOut = dot( N, L );

E \* ((weight \* cosThetaOut) / directPd

andom walk - done properly, closely following /ive)

```
;
t3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, lpdf );
rvive;
pdf;
n = E * brdf * (dot( N, R ) / pdf);
sion = true:
```

## An Improved Illumination Model for Shaded Display

## Color Trace( ray r )

I,  $\vec{N}$ , mat = NearestIntersection( scene, r )

if (mat == DIFFUSE)

```
return mat.color * DirectIllumination( I, \vec{N} )
```

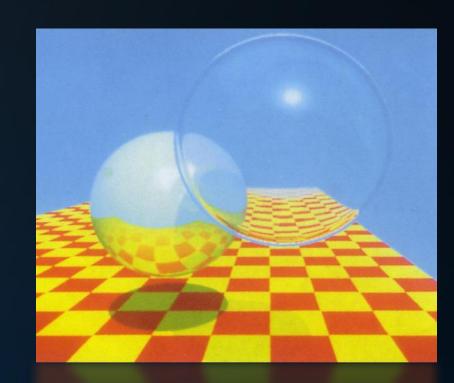
if (mat == MIRROR)

```
return mat.color * Trace( I, reflect( r.\vec{D}, \vec{N} )
```

#### Indirect illumination:

For perfect specular object (mirrors) we extend the primary ray with an extension ray:

- We still modulate transport with the material color
- We do not apply  $\vec{N} \cdot \vec{L}$ 
  - We do not calculate direct illumination





## Whitted

nics & (depth < ≫xx00

: = inside ? 1 1 1 3 ht = nt / nc, ddn os2t = 1.0f - nnt \* 1 2, N ); 3)

at a = nt - nc, b = nt - n at Tr = 1 - (R0 + (1 - R0 Fr) R = (D = nnt - N - (3)

= \* diffuse; = true;

efl + refr)) && (depth < MAXD

), N ); ∙efl \* E \* diffu = true;

AXDEPTH)

survive = SurvivalProbability( diffuse
estimation - doing it properly
if;
radiance = SampleLight( &rand, I, &L, &LI);
ext + radiance.y + radiance.z) > 0) &&

v = true;

at brdfPdf = EvaluateDiffuse( L, N ) Psurvi at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely following Soci /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf ) urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Reflection

Given a ray direction  $\vec{D}$  and a normalized surface normal  $\vec{N}$ , the reflected vector  $\vec{R} = \vec{D} - 2(\vec{D} \cdot \vec{N})\vec{N}$ .

#### Derivation:

$$\vec{V} = \vec{N}(\vec{D} \cdot \vec{N})$$
  

$$\vec{U} = \vec{D} - \vec{V}$$
  

$$\vec{R} = \vec{U} + (-\vec{V})$$
  

$$\vec{R} = \vec{D} - \vec{N}(\vec{D} \cdot \vec{N}) - \vec{N}(\vec{D} \cdot \vec{N})$$
  

$$\vec{R} = \vec{D} - 2(\vec{D} \cdot \vec{N})\vec{N}$$





## Whitted

at a = nt - nc,

), N );

AXDEPTH)

v = true;

lf;

efl + refr)) && (depth

survive = SurvivalProbability( di

radiance = SampleLight( &rand, I, e.x + radiance.y + radiance.z) >

at brdfPdf = EvaluateDiffuse( L,

at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf at cosThetaOut = dot( N, L )<u>:</u>\_\_\_\_\_\_

refl \* E \* diffuse;

<u>Question 1</u>: For direct illumination, we take into account:

- Material color
- Distance to light source
- $\vec{N} \cdot \vec{L}$

Why?

<u>Question 2</u>: We use the summed contribution of all light sources.

Is this correct?

<u>Question 3</u>: Why do we not sample the light sources for a pure specular surface? *(can you cast a shadow on a bathroom mirror?)* 

<u>Question 4</u>: Show geometrically that, for normalized vectors  $\vec{D}$  and  $\vec{N}$ ,  $\vec{R} = \vec{D} - 2(\vec{D} \cdot \vec{N})\vec{N}$  yields a normalized vector.



andom walk - done properly, closely followi /ive) :

E \* ((weight \* cosThetaOut) / directPdf

, t33 brdf = SampleDiffuse( diffuse, N, r1, r2, 8R, 8pdf urvive; .pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true: 15

## Whitted

tics & (depth < ™xxxxx

: = inside ? 1 1 1 nt = nt / nc, ddm os2t = 1.0f - nmt ~ 0, N ); 0)

at a = nt - nc, b = nt at Tr = 1 - (R0 + (1 - R0) Fr) R = (D <sup>+</sup> nnt - N - (dd)

= \* diffuse; = true;

. :fl + refr)) && (depth < MAXDEPTH

D, N ); refl \* E \* diffuse; = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly Hf; radiance = SampleLight( &rand, I, &L, &Le e.x + radiance.y + radiance.z) > 0) &

v = true;

at brdfPdf = EvaluateDiffuse( L, N ) \* Psurviv at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) (();

andom walk - done properly, closely following Sou. /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, 8R, 8pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

An Improved Illumination Model for Shaded Display

Handling partially reflective materials:

Color Trace( ray r )
 I, N, mat = NearestIntersection( scene, r )
 s = mat.specularity
 d = 1 - mat.specularity
 return mat.color \* (
 s \* Trace( ray( I, reflect( r.D, N) ) ) +
 d \* DirectIllumination( I, N) )

Note: this is not efficient. (why not?)





## Whitted

ics & (depth < ≫0000

: = inside ? 1 1 0 ht = nt / nc, ddn os2t = 1.0f - nnt o 0, N ); 0)

at a = nt - nc, b = nt at Tr = 1 - (R0 + (1 - R0 Ir) R = (D <sup>=</sup> nnt - N

= \* diffuse; = true;

efl + refr)) && (depth < MAXDED

D, N ); refl \* E \* diffuse; = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, closed if; radiance = SampleLight( &rand, I, &L, &light) e.x + radiance.y + radiance.z() > 0) && (doing)

v = true;

at brdfPdf = EvaluateDiffuse( L, N ) \* Psurvis at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) (Ps

andom walk - done properly, closely following Sec. /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, apd urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## An Improved Illumination Model for Shaded Display

## Dielectrics

```
Color Trace( ray r )
    I, N, mat = NearestIntersection( scene, r )
    if (mat == DIFFUSE)
        return mat.color * DirectIllumination( I, N)
    if (mat == MIRROR)
        return mat.color * Trace( I, reflect( r.D, N)
    if (mat == GLASS)
        return mat.color * ?
```





## Whitted

tics & (depth < MADOT

: = inside ? 1 ; . ht = nt / nc, ddn os2t = 1.0f - nnt ° 2, N ); ≥)

= \* diffuse; = true;

efl + refr)) && (depth < MODEPTH

), N ); refl \* E \* diffu = true;

AXDEPTH)

survive = SurvivalProbability( diffus
estimation - doing it properly
if;
addiance = SampleLight( &rand, I, &L
e.x + radiance.y + radiance.z) > 0) >

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Psu at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely following Sol. /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Dielectrics

The direction of the transmitted vector  $\vec{T}$  depends on the refraction indices  $n_1, n_2$  of the media separated by the surface. According to Snell's Law:

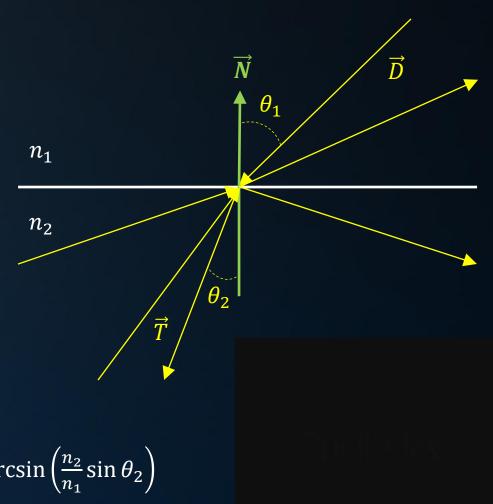
 $n_1 sin \theta_1 = n_2 sin \theta_2$ 

or

$$\frac{n_1}{n_2}sin heta_1 = sin heta_2$$

Note: left term may exceed 1, in which case  $\theta_2$  cannot be computed. Therefore:

$$\frac{n_1}{n_2}\sin\theta_1 = \sin\theta_2 \iff \sin\theta_1 \le \frac{n_2}{n_1} \Rightarrow \theta_{critical} = \arcsin\left(\frac{n_2}{n_1}\sin\theta_2\right)$$





## Whitted

tics ≹ (depth < Moder

: = inside ? 1 : 1 : 1 ht = nt / nc, ddn bs2t = 1.0f - nnt 2, N ); 3)

at a = nt - nc, b = nt = n at Tr = 1 - (R0 + (1 - R0)  $\Gamma$ r) R = (D = nnt - N - (ddn)

= \* diffuse; = true;

. efl + refr)) && (depth < MAXDEDII

), N ); refl \* E \* diffuse = true;

#### AXDEPTH)

survive = SurvivalProbability( diffuse )
estimation - doing it properly, closed
f;
radiance = SampleLight( &rand, I, &L, &ll)
e.x + radiance.y + radiance.z) > 0) && (d)

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Ps at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely following Sec /ive)

; t3 Brdf = SampleDiffuse( diffuse, N, r1, r2, &R, apd urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:





## Whitted

ics & (depth < Model

: = inside ? 1 | 1 | 1 ht = nt / nc, ddn | 1 ps2t = 1.0f - nmt | 1 2, N ); 3)

at a = nt - nc, b = Mt = n at Tr = 1 - (R0 + (1 - R0 Fr) R = (D = nnt - N = (Jdn

= \* diffuse = true:

. efl + refr)) && (depth < MAXDEPTI

D, N ); refl \* E \* diffu: = true;

#### AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, closed if; radiance = SampleLight( &rand, I, &L, &ll e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) + radiance.z) > 0) && doing e.x + radiance.y + radiance.z) + ra

w = true; at brdfPdf = EvaluateDiffuse( L, N ) Psu at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely following Swa /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf ); pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:



https://en.wikipedia.org/wiki/Snell%27s\_window



## Whitted

ics & (depth < ™XXXX

= inside 7 1 1 1 0 ht = nt / nc, ddn - 0 os2t = 1.0f - nπt - α 0, N ); 3)

at a = nt - nc, b = nt - nc at Tr = 1 - (R0 + (1 - R0 Fr) R = (D = nnt - N

= \* diffuse; = true;

efl + refr)) && (depth < MODE

D, N ); refl \* E \* diff = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, closed if; radiance = SampleLight( &rand, I, &L, &ll e.x + radiance.y + radiance.z) > 0) && (d)

w = true; at brdfPdf = EvaluateDiffuse( L, N ) at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L );

E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely following : /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, apdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Dielectrics

$$\frac{n_1}{n_2}\sin\theta_1 = \sin\theta_2 \Leftrightarrow \sin\theta_1 \le \frac{n_2}{n_1}$$

$$k = 1 - \left(\frac{n_1}{n_2}\right)^2 \left(1 - \cos\theta_1^2\right)$$

$$\vec{T} = \begin{cases} TIR, & \text{for } k < 0\\ \frac{n_1}{n_2}\vec{D} + \vec{N}\left(\frac{n_1}{n_2}\cos\theta_1 - \sqrt{k}\right), & \text{for } k \ge 0 \end{cases}$$

Note: 
$$cos\theta_1 = \vec{N} \cdot -\vec{D}$$
, and  $\frac{n_1}{n_2}$  should be calculated only once.

\* For a full derivation, see http://www.flipcode.com/archives/reflection\_transmission.pdf

## $\overrightarrow{N}$ $\overrightarrow{D}$ 4 $\theta_1$ $n_1$ $n_2$ $\theta_2$ $\vec{T}$



## Whitted

ics & (depth < PVXCC

: = inside ? 1 : 1 : 3 ht = nt / nc, ddn = bs2t = 1.0f - nnt 2, N ); 3)

at a = nt - nc, b = nt - n at Tr = 1 - (R0 + (1 - R0 Fr) R = (D = nnt - N = (ddn

= \* diffuse = true;

efl + refr)) && (depth < NACCO

D, N ); refl \* E \* diffus = true;

AXDEPTH)

survive = SurvivalProbability( diff estimation - doing it properly, if; radiance = SampleLight( &rand, I, e.x + radiance.y + radiance.z) > @

w = true; at brdfPdf = EvaluateDiffuse( L, N at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPc at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / dire

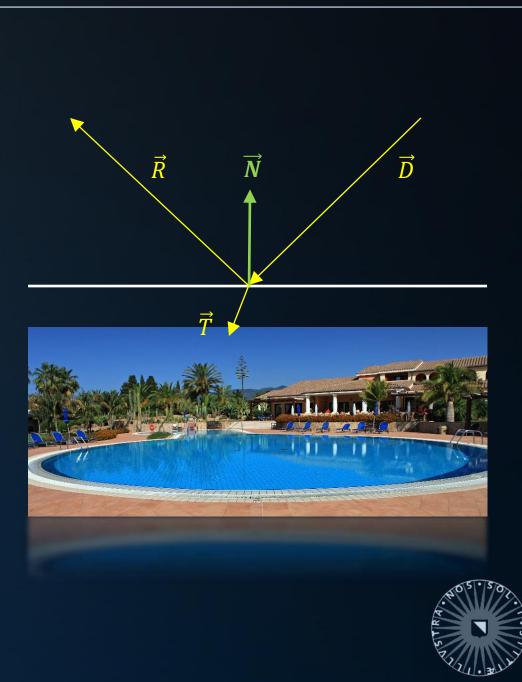
andom walk - done properly, closely /ive)

; at3 brdf = SampleDiffuse( diffuse, urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Dielectrics

#### A typical dielectric transmits *and* reflects light.





## Whitted

nics & (depth < Modes

: = inside ? 1 | 1 | 7 ht = nt / nc, ddn bs2t = 1.0f - nnt 2, N ); 3)

at a = nt - nc, b = nt + nc at Tr = 1 - (R0 + (1 - R0  $\Gamma$ ) R = (D = nnt - N + (d0)

= \* diffuse; = true;

. efl + refr)) && (depth < MAXD

D, N ); refl \* E \* diff = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly if; radiance = SampleLight( &rand, I, &L e.x + radiance.y + radiance.z) > 0) &&

w = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Psurviv at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* (r);

andom walk - done properly, closely following 300 /ive)

st3 brdf = SampleDiffuse( diffuse, N, r1, r2 Where: cos hetaurvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Dielectrics

 $F_r =$ 

A typical dielectric transmits *and* reflects light.

Based on the Fresnel equations, the reflectivity of the surface for non-polarized light is formulated as:

$$=\frac{1}{2}\left(\left(\frac{n_1\cos\theta_i-n_2\cos\theta_t}{n_1\cos\theta_i+n_2\cos\theta_t}\right)^2+\left(\frac{n_1\cos\theta_t-n_2\cos\theta_i}{n_1\cos\theta_t+n_2\cos\theta_i}\right)^2\right)$$

Reflectance for s-polarized light

Reflectance for p-polarized light

Reflectance for unpolarized light

$$t_t = \sqrt{1 - \left(\frac{n_1}{n_2}\sin\theta_i\right)^2}$$

## Whitted

sics & (depth < Monson

: = inside ? 1 1 1 0 ht = nt / nc, ddn 0 ps2t = 1.0f - nnt 0 D, N ); 3)

at a = nt - nc, b = nt at Tr = 1 - (R0 + (1 - R0 Fr) R = (D <sup>=</sup> nnt - N

= \* diffuse; = true;

efl + refr)) && (depth < MAXDEPTH

D, N ); refl \* E \* diffu = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, closed if; radiance = SampleLight( &rand, I, &L, &light) 2.x + radiance.y + radiance.z) > @\_\_\_\_\_\_

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Psurvive at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* (ref

andom walk - done properly, closely following Same. /ive)

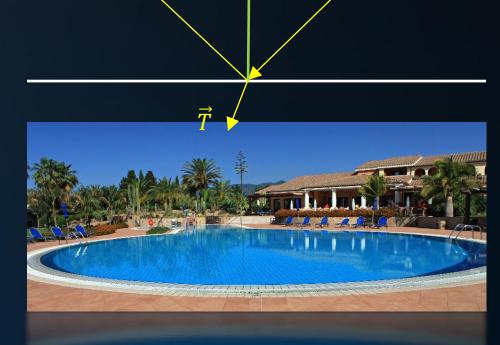
; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Dielectrics

 $F_r = \cdots$ 

Based on the law of conservation of energy:

 $F_t = 1 - F_r$ 



 $\vec{N}$ 

 $\vec{R}$ 



 $\overrightarrow{D}$ 

## Whitted

#### sics & (depth < MARDERT

- : = inside ? 1 = 1.0 ht = nt / nc, ddn bs2t = 1.0f - nnt = onn D, N ); 3)
- at a = nt nc, b = nt = n at Tr = 1 - (R0 + (1 - R0 Fr) R = (D \* nnt - N \* (330
- = \* diffuse; = true;
- efl + refr)) && (depth < M
- D, N ); ∵efl \* E \* diffu = true;
- AXDEPTH)
- survive = SurvivalProbability( diffuse estimation - doing it properly.close If; radiance = SampleLight( &rand, I, &L, 2.x + radiance.y + radiance.z) > 0) &
- v = true; at brdfPdf = EvaluateDiffuse( L, N at factor = diffuse \* INNET;
- at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf
- andom walk done properly, closely following /ive)
- ; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Ray Tracing

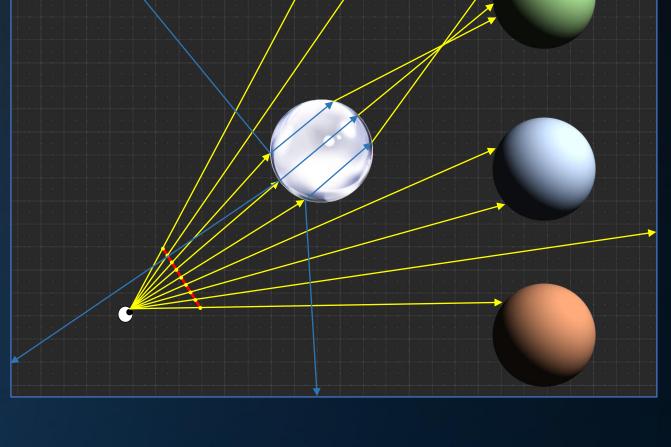
## World space

- Geometry
- Eye

- Screen plane
- Screen pixels
- Primary rays
- Intersections
- Point light
  - Shadow rays

## Light transport

- Extension rays
- Light transport



 $\mathbf{X}$ 



## Whitted

nics & (depth < ⊅000001

at a = nt - nc, b = nt at Tr = 1 - (R0 + (1 - R0 Ir) R = (D <sup>=</sup> nnt - N

= \* diffuse; = true;

efl + refr)) && (depth < MAXDEPTH

), N ); refl \* E \* diffu: = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, If; radiance = SampleLight( &rand, I, &t, \$100 e.x + radiance.y + radiance.z) > <u>0) && con</u>

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Psurviv at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* (P8

andom walk - done properly, closely following Sour /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Ray Tree

Using Whitted-style ray tracing, hitting a surface point may spawn:

- a shadow ray for each light source;
- a reflection ray;
- a ray transmitted into the material.

The reflected and transmitted rays may hit another object with the same material.

 $\rightarrow$  A single primary ray may lead to a very large number of ray queries.



## Whitted

rics & (depth < ≫ocoo

at a = nt - nc, b = nt - nc at Tr = 1 - (R0 + (1 - R0 Fr) R = (D = nnt - N - (ddn

= \* diffuse = true;

efl + refr)) && (depth < MAXDEPTI

D, N ); refl \* E \* diffus = true;

#### AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly closed if; radiance = SampleLight( &rand, I, &L, &light) 2.x + radiance.y + radiance.z) > 0) && (details)

w = true; at brdfPdf = EvaluateDiffuse( L, N ) Pourvive at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) (and

andom walk - done properly, closely following Sec. /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf ) urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

<u>Question 5</u>: imagine a scene with several point lights and dielectric materials. Considering the law of conservation of energy, what can you say about the energy transported by each individual ray?



## Whitted

Beer's Law

hics & (depth < NAXDEET

at a = nt - nc, b = nt + r at Tr = 1 - (R0 + (1 - R0 Fr) R = (D = nnt - N = (ddn

= \* diffuse = true;

. efl + refr)) && (depth < MAXDEPTI

), N ); ∵efl \* E \* diffus = true;

#### AXDEPTH)

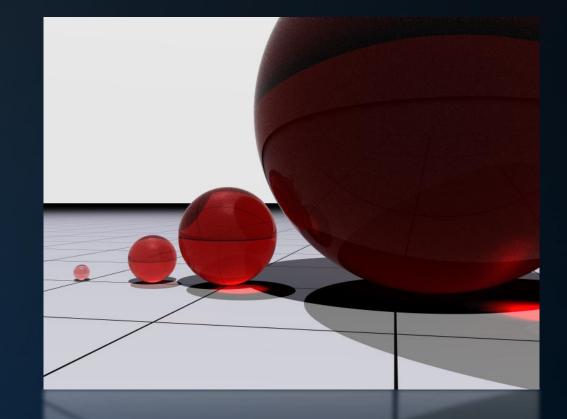
survive = SurvivalProbability( diffuse estimation - doing it properly, close if; radiance = SampleLight( &rand, I, &L, & e.x + radiance.y + radiance.z) > 0) &&

w = true; at brdfPdf = EvaluateDiffuse( L, N ) \* P at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf

andom walk - done properly, closely following 3 /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, Bpdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

# $I_0$ $C, \alpha$ $I_1$





## Whitted

nics & (depth < NoCOS

: = inside ? 1 + 1 0 ht = nt / nc, ddn bs2t = 1.0f - nnt 2, N ); 3)

at a = nt - nc, b = nt at Tr = 1 - (R0 + (1 - R0 Tr) R = (D = nnt - N - (d)

= \* diffuse = true;

. efl + refr)) && (depth < MADEPIO

), N ); •efl \* E \* diffu = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, close if; radiance = SampleLight( &rand, I, &t, 2.x + radiance.y + radiance.z) > 0) &&

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Psurv at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely following Sour /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Beer's Law

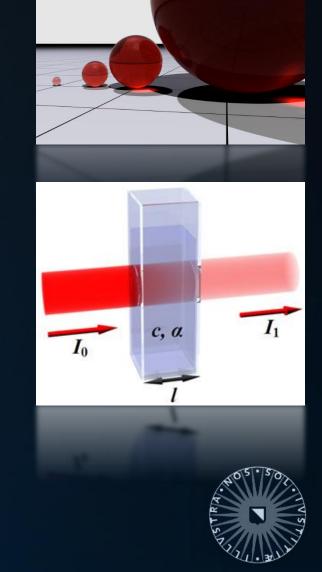
Light travelling through a medium loses intensity due to absorption.

The intensity I(d) that remains after travelling d units through a substance with absorption a is:

 $I(d) = I(0)e^{-\ln(a)d}$ 

#### In pseudocode:

I.r	*=	exp(	-a.r	*	d	);
I.g	*=	exp(	-a.g	*	d	);
I.b	*=	exp(	-a.b	*	d	);



## Whitted

at a = nt

efl + refr)) && (depth

), N ); refl \* E \* diffuse;

AXDEPTH)

survive = SurvivalProbability( dif lf; radiance = SampleLight( &rand, I, e.x + radiance.y + radiance.z) > 0

v = true; at brdfPdf = EvaluateDiffuse( L, N at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely foll: /ive)

at3 brdf = SampleDiffuse( diffuse, N, r1, r2, 8R, A urvive; pdf; 1 = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Whitted - Summary

A Whitted-style ray tracer implements the following optical phenomena:

- Direct illumination of multiple light sources, taking into account
  - Visibility **Distance** attenuation A shading model: N dot L for diffuse
- Pure specular reflections, with recursion
- Dielectrics, with Fresnel, with recursion
  - Beer's Law

The ray tracer supports any primitive for which a ray/primitive intersection can be determined.



ics & (depth < Mox000

: = inside ? l | ... ht = nt / nc, ddn - ... os2t = 1.0f - nnt - n O, N ); ð)

at a = nt - nc, b = nt - n at Tr = 1 - (R0 + (1 - R0 Tr) R = (D <sup>#</sup> nnt - N = (ddm -

= \* diffuse = true;

-:fl + refr)) && (depth < MODEPT

), N ); refl \* E \* diffus = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, closed if; radiance = SampleLight( &rand, I, &L, &ilent 2.x + radiance.y + radiance.z) > 0) &

v = true; at brdfPdf = EvaluateDiffuse( L, N ) Pourvive at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) (nud

andom walk - done properly, closely following Sec. /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Today's Agenda:

- Introduction: Appel
- Whitted
- Cook



## Cook

## 'Distributed Ray Tracing'

vics k (depth < Poxosini)

= inside ? 1 t = nt / nc, ddn s2t = 1.0f - nnt - c , N );

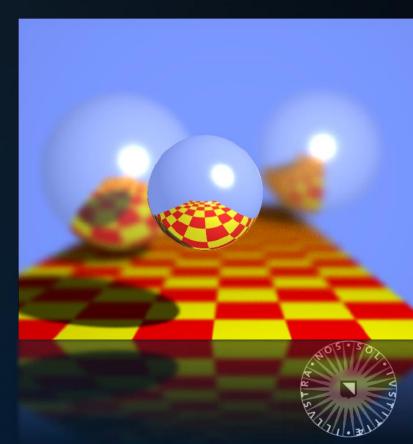


andom walk - done properly, closely following rive)

at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, Bpdf ) urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:



Whitted-style ray tracing does not handle glossy reflections, depth of field, motion blur.



## Cook

ics & (depth < MaxDept

: = inside ? 1 1 1 2 ht = nt / nc, ddn - 1 ps2t = 1.0f - nmt - 1 2, N ); 3)

at a = nt - nc, b = nt - nc at Tr = 1 - (R0 + (1 - R0) Fr) R = (D <sup>+</sup> nnt - N - (dd)

= \* diffuse; = true;

. efl + refr)) && (depth < MAXDEPTI

D, N ); refl \* E \* diffu: = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, closed if; radiance = SampleLight( &rand, I, &L, &ll e.x + radiance.y + radiance.z) > 0) &&

w = true; at brdfPdf = EvaluateDiffuse( L, N ) Psurviv at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) (();

andom walk - done properly, closely following /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

'Distributed Ray Tracing'

Whitted-style ray tracing does not handle glossy reflections, depth of field, motion blur:

Ray tracing is a point sampling process.

#### Cook et al.\*:

Replace point sampling by integrals:

- Perform anti-aliasing by integrating over the pixel
- Add motion blur by integrating over time
  - Calculate depth of field by integrating over the aperture.

\* : Cook et al., 1984. Distributed Ray Tracing.



ics & (depth < Mox000

: = inside ? l | ... ht = nt / nc, ddn - ... os2t = 1.0f - nnt - n O, N ); ð)

at a = nt - nc, b = nt - n at Tr = 1 - (R0 + (1 - R0 Tr) R = (D <sup>#</sup> nnt - N = (ddm -

= \* diffuse = true;

-:fl + refr)) && (depth < MODEPT

), N ); refl \* E \* diffus = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly, closed if; radiance = SampleLight( &rand, I, &L, &ilent 2.x + radiance.y + radiance.z) > 0) &

v = true; at brdfPdf = EvaluateDiffuse( L, N ) Pourvive at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) (nud

andom walk - done properly, closely following Sec. /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Today's Agenda:

- Introduction: Appel
- Whitted
- Cook



hics & (depth < Not000

: = inside ? 1 ht = nt / nc, ddn bs2t = 1.0f - nnt 0, N ); a)

at a = nt - nc, b = nt - nc at Tr = 1 - (R0 + (1 - R0 Ir) R = (D = nnt - N = (ddn)

= \* diffuse; = true:

efl + refr)) && (depth < MADEPT)

D, N ); refl \* E \* diffuse; = true;

AXDEPTH)

survive = SurvivalProbability( diffuse estimation - doing it properly ff; radiance = SampleLight( &rand, I, &L, &lla e.x + radiance.y + radiance.z) >\_\_0)

w = true; at brdfPdf = EvaluateDiffuse(L, N) \* Psi at3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot(N, L); <u>E \* ((weig</u>ht \* cosThetaOut) / directPdf)

andom walk - done properly, closely following Son. /ive)

; at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, apd urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

# **INFOMAGR – Advanced Graphics**

Jacco Bikker - November 2021 – February 2022

# END of "Whitted"

next lecture: "Acceleration Structures"



