tic: ≰ (depth < NA

: = inside / l it = nt / nc, dd os2t = 1.0f - nn ), N ); D)

st  $a = nt - nc_{1}b - nt$ st Tr = 1 - (R0 + (1))Tr ) R = (D \* nnt - R \* )

= diffuse = true;

-: :fl + refr)) && (depth k HANDII

D, N ); ~efl \* E \* diffus = true;

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property if; radiance = SampleLight( %rand, I e.x + radiance.y + radiance.z) > 0) %

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pourch st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* 0000

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

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

# INFOGR – Computer Graphics

Jacco Bikker - April-July 2016 - Lecture 4: "Ray Tracing (2)"

# Welcome!



tica € (depth < 14.55

: = inside / 1 it = nt / nc, dde os2t = 1.0f - nn: 0, N ); 0)

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

= diffuse; = true;

-: efl + refr)) && (depth < HADIIII

D, N ); ref1 \* E \* diffu = true;

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property if; adiance = SampleLight( &rand, I. .x + radiance.y + radiance.z)

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pauro st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* read

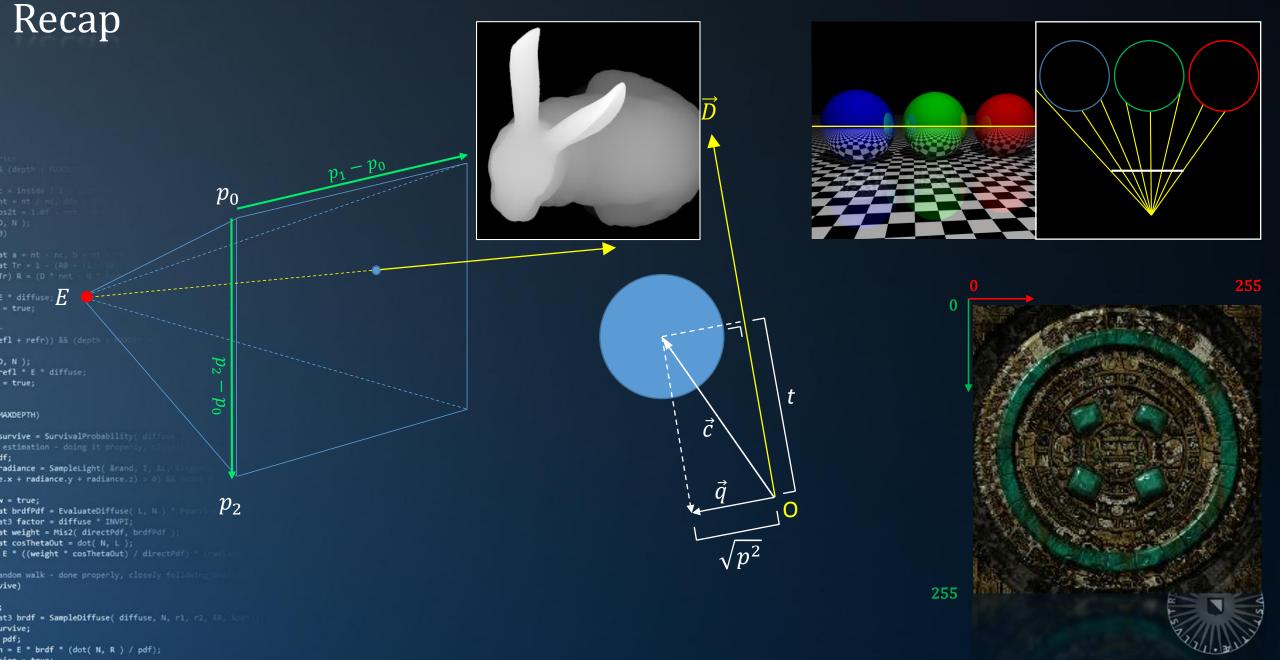
andom walk - done properly, closely following -/ive)

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

## Today's Agenda:

- Recap
- End of the Primary Ray
- Normals
- The Camera
- Assignment P2





3

sion = true:

tica € (depth < 14.55

: = inside / 1 it = nt / nc, dde os2t = 1.0f - nn: 0, N ); 0)

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

= diffuse; = true;

-: efl + refr)) && (depth < HADIIII

D, N ); ref1 \* E \* diffu = true;

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property if; adiance = SampleLight( &rand, I. .x + radiance.y + radiance.z)

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pauro st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* read

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

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

## Today's Agenda:

- Recap
- End of the Primary Ray
- Normals
- The Camera
- Assignment P2



tic: (depth < NA

= inside / 1 nt = nt / nc, dda ps2t = 1.0f - nn p, N ); 3)

st a = nt - nc, b - nt st Tr = 1 - (R0 + (1 Tr) R = (D \* nnt - N

= diffuse = true;

efl + refr)) && (depth K MAXDIIII

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

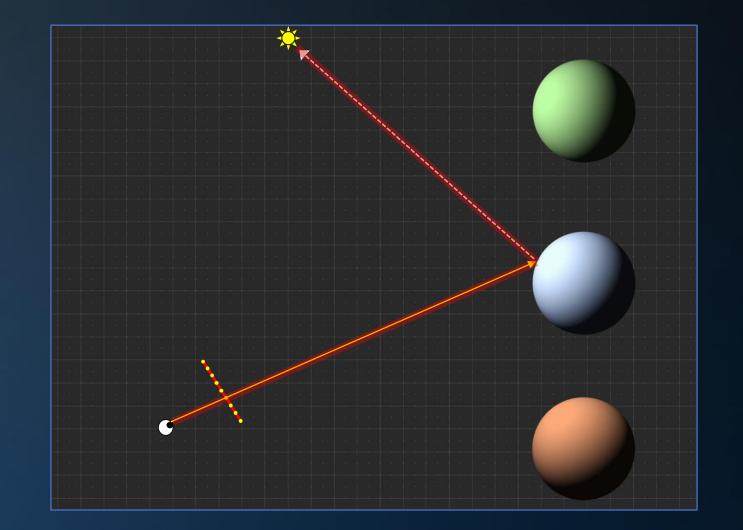
AXDEPTH)

survive = SurvivalProbability difference estimation - doing it property if; adiance = SampleLight( %rand, I, Market e.x + radiance.y + radiance.z) = 0.55

v = true; at brdfPdf = EvaluateDiffuse( L, N.) \* Prove st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* \*\*\*

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

; pt3 brdf = SampleDiffuse( diffuse, N, r1, r2, N, pr pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:





tic: € (depth ⊂ 100⊂

: = inside / l = it = nt / nc, dde os2t = 1.0f = nnt o, N ); 3)

st a = nt - nc, b - nt st Tr = 1 - (R0 + (1 Tr) R = (D \* nnt - R

= diffuse; = true;

efl + refr)) && (depth & MANDERT

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

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it properly ff; radiance = SampleLight( %rand, I, Market e.x + radiance.y + radiance.z) > 0) %

w = true; st brdfPdf = EvaluateDiffuse( L, N ) Pour st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* (0)

andom walk - done properly, closely following /ive)

; st3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, D) urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## The End

We used *primary rays* to find the *primary intersection* point.

Determining light transport:

- Sum illumination from all light sources
- ...If they are *visible*.

We used a primary ray to find the object visible through a pixel: Now we will use a *shadow ray* to determine visibility of a light source.



tice € (depth (1983)

: = inside / 1 it = nt / nc, ddo ps2t = 1.0f - nnt D; N ); B)

st a = nt - nc, b = ntst Tr = 1 - (R0 + (1 - 1))(r) R = (0 - nnt - 1)

E = diffuse; = true;

efl + refr)) && (depth < HADDI

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

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property ff; radiance = SampleLight( &rand, I e.x + radiance.y + radiance.r) = 0

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pours) st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* ()

andom walk - done properly, closely following /ive)

; st3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, sr urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### Shadow Ray

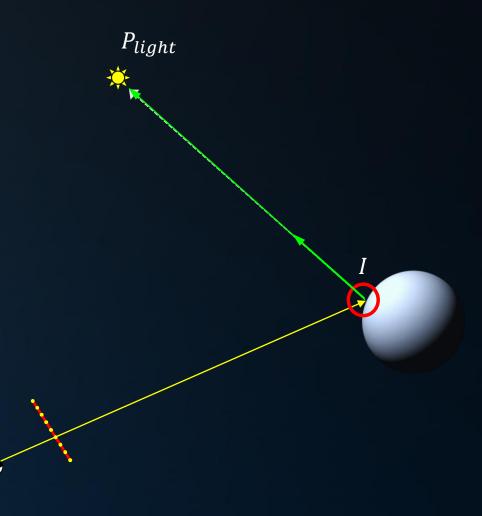
Constructing the shadow ray:

 $\overline{p(t)} = \overline{O} + t\overline{D}$ 

Ray origin: the primary intersection point *I*.

Ray direction:  $P_{light} - I$  (normalized)

Restrictions on *t*:  $0 < t < ||P_{light} - I||$ 





st a = m st Tr = 1 - (RE

efl + refr)) && (de

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

#### AXDEPTH)

survive = SurvivalProbability if: -adiance = SampleLight( &rand x + radiance.v + radiance.z

v = true: st brdfPdf = EvaluateDiffuse( st3 factor = diffuse \* INVPI st weight = Mis2( directPdf, brdfPdf at cosThetaOut = dot( N, L ); E ((weight \* cosThetaOut) / directPdf

andom walk - done properly, closely f vive)

at3 brdf = SampleDiffuse( diffuse, N, r1, r2, UR, rvive; pdf; 1 = E \* brdf \* (dot( N, R ) / pdf); sion = true

## Shadow Ray

Direction of the shadow ray:  $|P_{light}-I||$ 

Equally valid:

Note that we get different intersection points depending on the direction of the shadow ray.

It doesn't matter: the shadow ray is used to determine *if* there is an occluder, not *where*.

### This has two consequences:

- 1. We need a dedicated shadow ray query;
- 2. Shadow ray queries are (on average) twice as fast. (why?)

P<sub>light</sub>-I

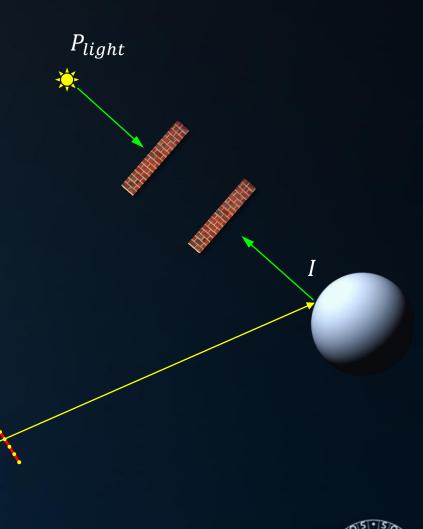
I-P<sub>light</sub>

 $||P_{light}-I||$ 

or

I-P<sub>light</sub>

 $|I-P_{light}||$ 





tic: k (depth < 10.)

:= inside / l it = nt / nc, dde os2t = 1.0f o, N ); 3)

st a = nt - nc, b - nt + + st Tr = 1 - (R0 + (1 Tr) R = (D \* nnt - N \* +

= \* diffuse; = true;

efl + refr)) && (depth k HANDIII

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

AXDEPTH)

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

v = true; t brdfPdf = EvaluateDiffuse( L, N.) \* Pours st3 factor = diffuse \* INVPI; st weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely following /ive)

; st3 brdf = SampleDiffuse( diffuse, N, r1, r2, UR, D) urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Shadow Ray

"In theory, theory and practice are the same. In practice, they are not."

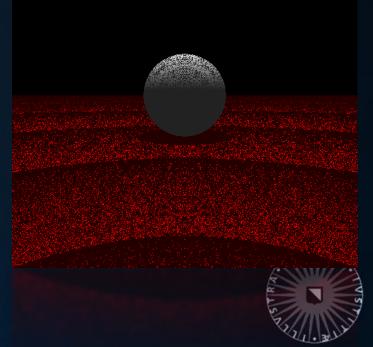
### Problem 1:

Our shadow ray queries report intersections at  $t = \sim 0$ . Why?

Cause: the shadow ray sometimes finds the surface it originated from as an occluder, resulting in *shadow acne*.

Fix: offset the origin by 'epsilon' times the shadow ray direction.

Note: don't forget to reduce  $t_{max}$  by epsilon.



tice € (depth < 10.5

:= inside / L it = nt / nc, dde os2t = 1.8f o, N ); 3)

st a = nt - nc, b + nt + + st Tr = 1 - (R0 + (1 Tr) R = (D \* nnt - N \*

= \* diffuse; = true;

-:fl + refr)) && (depth & MADIIII

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

AXDEPTH)

survive = SurvivalProbability difference estimation - doing it property ff; radiance = SampleLight( &rand, I e.x + radiance.y + radiance.r) = 0

v = true; t brdfPdf = EvaluateDiffuse( L, N) Promote st3 factor = diffuse \* INVPI; st weight = Mis2( directPdf, brdfPdf); st cosThetaOut = dot( N, L); E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely following /ive)

; st3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, sr urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Shadow Ray

"In theory, theory and practice are the same. In practice, they are not."

### Problem 2:

Our shadow ray queries report intersections at  $t = t_{max}$ . Why?

Cause: when firing shadow rays from the light source, they may find the surface that we are trying to shade.

Fix: reduce  $t_{max}$  by 2 \* *epsilon*.



tic: ⊾ (depth'∈ NASS

= inside / : it = nt / nc, ddn 552t = 1.0f = nd 3, N ); 3)

st a = nt - nc, b - nt st Tr = 1 - (R0 + (1 fr) R = (D \* nnt - N \*

= diffuse; = true;

-: :fl + refr)) 88 (depth < HANDIII

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

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it properly if; radiance = SampleLight( &rand, I = 1 .x + radiance.y + radiance.z) = 0 = 0.

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pourse st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* 000

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

; pt3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, source; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### Shadow Ray

"The most expensive shadow rays are those that do not find an intersection."

Why?

(because those rays tested every primitive before concluding that there was no occlusion)



tic: k (depth < NAS⊂

= inside / L it = nt / nc, dde ss2t = 1.8f - nnt -), N ); 3)

at a = nt - nc, b - nt - n at Tr = 1 - (80 + 1 Tr) R = (0 \* nnt - N

E \* diffuse; = true;

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

D, N ); ~efl \* E \* diffuse; = true;

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it properly, if; radiance = SampleLight( &rand, I, L, L) e.x + radiance.y + radiance.z) > 0) = 1

v = true; t brdfPdf = EvaluateDiffuse( L, N.) = Pour st3 factor = diffuse = INVPI; st weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) = 0.000

andom walk - done properly, closely following /ive)

; t3 brdf = SampleDiffuse( diffuse, N, rl, r2, 48, 454 pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true;

## Transport

The amount of energy travelling from the light via the surface point to the eye depends on:

- The brightness of the light source
- The distance of the light source to the surface point
- Absorption at the surface point
- The angle of incidence of the light energy



tice k (depth < 10.5

: = inside / l it = nt / nc, dde os2t = 1.0f = nnt ), N ); 3)

at a = nt - nc, b = nt - ncat Tr = 1 - (R0 + (1 - R0))Tr ) R = (0 \* nnt - R + (1 - R0))

= \* diffuse; = true;

-:fl + refr)) && (depth is HANDIII)

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

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property if; radiance = SampleLight( &rand, I & x + radiance.y + radiance.z) = 0

v = true; at brdfPdf = EvaluateDiffuse( L, N.) \* Provident st3 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, NR, Soft urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### Transport

Brightness of the light source:

Expressed in *watt (W)*, or *joule per second (J/s or Js*<sup>-1</sup>).

Energy is transported by photons.

Photon energy depends on wavelength; energy for a 'yellow' photon is  $\sim 3.31 \cdot 10^{-19}$  J.

A 100W light bulb thus emits  $\sim 3.0 \cdot 10^{20}$  photons per second.



tic: ⊾(depth < 100

: = inside / 1 ht = nt / nc, dde bs2t = 1.0f - ... b, N ); b)

st a = nt - nc, b + nt st Tr = 1 - (80 + (1 Tr) R = (0 \* nnt - n

E \* diffuse; = true;

-:fl + refr)) && (depth & HAADIII

D, N ); ~efl \* E \* diffuse; = true;

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property ff; radiance = SampleLight( &rand, I = b) e.x + radiance.y + radiance.z) = 0

v = true; t brdfPdf = EvaluateDiffuse( L, N ) st3 factor = diffuse \* INVPI; st weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPd

andom walk - done properly, closely following /ive)

; pt3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, Dp3 prvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true;

### Transport

Energy at distance *r*:

For a point light, a brief pulse of light energy spreads out as a growing sphere. The energy is distributed over the surface of this sphere.

It is therefore proportional to the inverse area of the sphere at distance r, i.e.:

$$E/m^2 = E_{light} \frac{1}{4\pi r^2}$$

Light energy thus dissipates at a rate of  $\frac{1}{r^2}$ . This is referred to as *distance attenuation*.



tice ≰ (depth < 10.5

= inside / 1 it = nt / nc, dde os2t = 1.8f - nnt ), N ); 8)

st  $a = nt - nc_{1} b - nt$ st Tr = 1 - (R0 + (1))Tr ) R = (0 \* nnt - N)

= diffuse; = true;

efl + refr)) && (depth < HANDIIII

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

AXDEPTH)

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

v = true; t brdfPdf = EvaluateDiffuse( L, N ) = Pauro st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely following /ive)

; t3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, 1, r pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Transport

Absorption:

Most materials absorb light energy. The wavelengths that are not fully absorbed define the 'color' of a material.

### The reflected light is thus:

 $E_{reflected} = E_{incoming} \cdot C_{material}$ 

Note that  $C_{material}$  cannot exceed 1; the reflected light is never *more* than the incoming light.



tic: k (depth < 100

:= inside / L it = nt / nc, dde os2t = 1.0f - nn: 0, N ); 3)

at a = nt - nc, b - n1 at Tr = 1 - (R0 + (1 - 1 fr) R = (D \* nnt - N \* 1

= diffuse = true;

efl + refr)) && (depth & MANDER

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

AXDEPTH)

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

v = true; at brdfPdf = EvaluateDiffuse( L, N, ) \* Pauro st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* 0

andom walk - done properly, closely following

; ot3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, los) pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### Transport

Energy arriving at an angle:

A small bundle of light arriving at a surface affects a larger area than the cross-sectional area of the bundle.

Per  $m^2$ , the surface thus receives less energy. The remaining energy is proportional to:

 $\cos \alpha$  or:  $\vec{N} \cdot \vec{L}$ .



iles ⊾ (depth (⊂)⊍s:

= inside / L it = nt / nc, dde os2t = 1.8f - nnt D, N ); B)

at a = nt - nc, b - n at Tr = 1 - (R0 + fr) R = (0 \* nnt - N

E = diffuse; = true;

-:fl + refr)) && (depth & HADD

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

AXDEPTH)

survive = SurvivalProbability difference estimation - doing it property ff; radiance = SampleLight( &rand, I e.x + radiance.y + radiance.z) > 0)

v = true; t brdfPdf = EvaluateDiffuse( L, N ) = Pour st3 factor = diffuse = INVPI; st weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

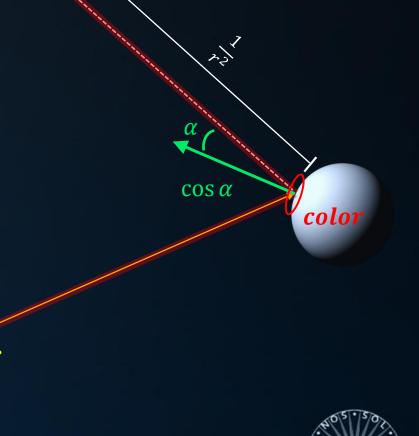
andom walk - done properly, closely following /ive)

; st3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, st urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### Transport

All factors:

- Emitted light : defined as RGB color, floating point
- Distance attenuation:  $\frac{1}{r^2}$
- Absorption, modulate by material color
- N dot L



Elight



tic: k (depth < 10

= inside / 1 it = nt / nc, dde os2t = 1.01 - nnt 5, N ); 3)

at a = nt - nc, b - nt at Tr = 1 - (80 + 1 Tr) R = (0 \* nnt - 8 \*

= diffus = true:

-:**fl + refr)) && (depth k MAND** 

>, N ); ref1 \* E \* dif = true;

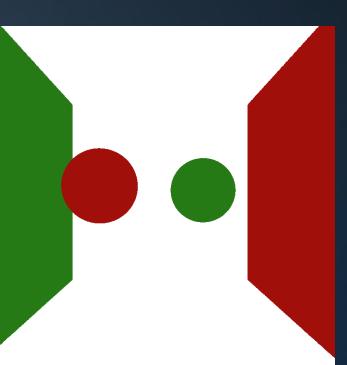
#### AXDEPTH)

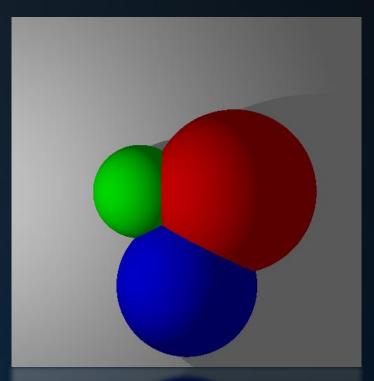
survive = SurvivalProbability difference estimation - doing it property if; radiance = SampleLight( &rand I .x + radiance.y + radiance.r) > 0) &

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pur st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \*

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

; t33 brdf = SampleDiffuse( diffuse, N, F1, F2, BR, 5, 5 urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:







tica € (depth < 14.55

: = inside / 1 it = nt / nc, dde os2t = 1.0f - nn: 0, N ); 0)

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

= diffuse; = true;

-: efl + refr)) && (depth < HADIIII

D, N ); ref1 \* E \* diffu = true;

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property if; adiance = SampleLight( &rand, I. .x + radiance.y + radiance.z)

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pauro st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* read

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

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

## Today's Agenda:

- Recap
- End of the Primary Ray
- Normals
- The Camera
- Assignment P2



tic: • (depth: < 103

: = inside / l nt = nt / nc, dde os2t = 1.0f - nnt 0, N ); 3)

st a = nt - nc, b + nt + + st Tr = 1 - (R0 + (1 - F) fr) R = (D \* nnt - N \*

= diffuse = true;

-•fl + refr)) 88 (death & BAAD

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

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it properly if; radiance = SampleLight( &rand, 1 e.x + radiance.y + radiance.r) > 0) &

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)

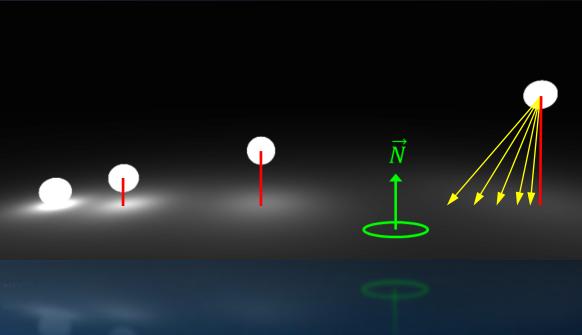
andom walk - done properly, closely following /ive)

; st3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, Doff urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true;

## We Need a Normal

For a plane, we already have the normal.

$$Ax + By + Cz + D = 0$$
 or  $(P \cdot \vec{N}) + D = 0$ 



## Distance attenuation: $1/r^2$

Angle of incidence:  $N \cdot L$ 



"Le: k (depth < 100

= inside / 1 it = nt / nc, ddo os2t = 1.8f - nnt ), N ); 8)

st a = nt - nc, b + nt st Tr = 1 - (80 + (1 - 1 Tr) R = (0 \* nnt - 1

= diffuse; = true;

-:**fl + refr))** && (depth < HANDE

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

AXDEPTH)

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

v = true; t brdfPdf = EvaluateDiffuse( L, N.) = Pours) st3 factor = diffuse = INVPI; at weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) = 000

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

; pt3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, Dp3 pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### We Need a Normal

Question:

How does light intensity relate to scene size? i.e.: if I scale my scene by a factor 2, what should I do to my lights?

 $\rightarrow$  Distance attenuation requires scaling light intensity by  $2^2$ 

→ Scene scale does not affect  $N \cdot L$ .



tic: K (depth < 10.5

: = inside / 1 ht = nt / nc, dde bs2t = 1.0f - nnt b, N ); b)

st a = nt - nc, b + nt + st Tr = 1 - (80 + (1 - 1 Tr) R = (0 \* nnt - 11 -

E \* diffuse; = true;

efl + refr)) && (depth & HANDIIII

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

AXDEPTH)

survive = SurvivalProbability difference estimation - doing it property if; radiance = SampleLight( &rand, I. . .x + radiance.y + radiance.;) = 0

v = true; at brdfPdf = EvaluateDiffuse( L, N.) \* Pour 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)

; pt3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, Loor pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true;

### We Need a Normal

Question:

What happens when a light is near the horizon?

 $\rightarrow$  Angle approaches 90°; cos  $\alpha$  approaches 0

 $\rightarrow$  Light is distributed over an infinitely large surface

Note: below the horizon,  $\cos \alpha$  becomes negative. → Clamp  $\vec{N} \cdot \vec{L}$  to zero.





22

tic: K (depth < 10.

= inside / l it = nt / nc, dde ps2t = 1.0f - nnt >, N ); >)

st a = nt - nc, b - nt + + st Tr = 1 - (R0 + (1 Tr) R = (D \* nnt - N \* +

= diffuse; = true;

efl + refr)) && (depth & MAXDIII

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

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it properly if; adiance = SampleLight( \$rand, I, I, s.x + radiance.y + radiance.z) > 0) \$\$\$

v = true; t brdfPdf = EvaluateDiffuse( L, N ) = Pour st3 factor = diffuse = INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) = 000

andom walk - done properly, closely following /ive)

; st3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, D) urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## We Need a Normal

Normals are also used to *prevent* shadow rays.

### Situation:

A light source is behind the surface we hit with the primary ray:

 $\vec{N} \cdot \vec{L} < 0$ 

In this case, visibility is 0, and we do not cast the shadow ray.



tic: € (depth < 10

:= inside / i ht = nt / nc, dde bs2t = 1.0f - nnt b; N ); b)

st a = nt - nc, b - nt st Tr = 1 - (80 + (1 Tr) R = (0 \* nnt - n -

E <sup>=</sup> diffuse = true;

efl + refr)) && (depth & MANDIII

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

AXDEPTH)

v = true; at brdfPdf = EvaluateDiffuse( L, N.) \* Punct st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* 11

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

; pt3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, statu pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### We Need a Normal

Normals for spheres:

The normal for a sphere at a point *P* on the sphere is parallel to the vector from the center of the sphere to *P*.

$$\vec{N}_P = \frac{P - C}{||P - C||}$$





tic: € (depth < 19

: = inside / l ht = nt / nc, dde 552t = 1.0f - nnt 2, N ); 3)

st a = nt - nc, b - nt st Tr = 1 - (R0 + (1 Tr) R = (D \* nnt - N

E <sup>=</sup> diffuse = true;

-:fl + refr)) && (depth k MAXDIII

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

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property ff; radiance = SampleLight( &rand I &.x + radiance.y + radiance.z) > 0) & f

v = true; at brdfPdf = EvaluateDiffuse( L, N.) \* Praction st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* 1

andom walk - done properly, closely following /ive)

; st3 brdf = SampleDiffuse( diffuse, N, r1, r2, UR, Lord urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### We Need a Normal

Normals for spheres:

When a sphere is hit from the inside, we need to *reverse* the normal.

 $\vec{N}_P = \frac{C - P}{||P - C||}$ 

How to detect this situation when it is not trivial:

1. Calculate the normal in the usual manner (P - C); 2. If  $\vec{N}_P \cdot \vec{D}_{ray} < 0$  then  $\vec{N}_P = -\vec{N}_P$ .



tice k (depth ⊂ 100

= inside / 1 it = nt / nc, dde -552t = 1.8f - nnt 3, N ); 3)

at  $a = nt - nc_{s} b + nt + s$ at Tr = 1 - (R0 + (1 - 1))Tr ) R = (0 + nnt - 1)

= diffuse; = true;

efl + refr)) && (depth is MonDia

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

AXDEPTH)

survive = SurvivalProbability difference estimation - doing it property ff; radiance = SampleLight( &rand I &.x + radiance.y + radiance.r)

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pi st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

andom walk - done properly, closely following /ive)

; pt3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, Doff prvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true;

### Normal Interpolation

Simulating smooth surfaces using normal interpolation:

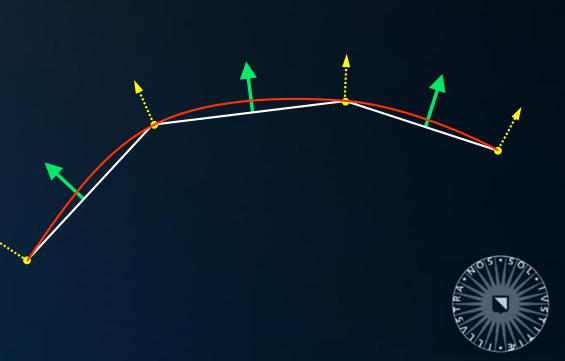
1. Generate *vertex normals*.

A vertex normal is calculated by averaging the normals of the triangles connected to the vertex and normalizing the result.

2. Interpolate the normals over the triangle.

In a ray tracer, use barycentric coordinates to do this. Normalize the interpolated normal.





nice ⊾ (depth (c)100

: = inside / L it = nt / nc, dde os2t = 1.0f - nnt 0, N ); 3)

st a = nt - nc, b - nt st Tr = 1 - (R0 + (1. fr) R = (D \* nnt - 0.

= diffuse; = true;

-:fl + refr)) && (depth is HANDIII

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

#### AXDEPTH)

survive = SurvivalProbabil estimation - doing it pro if; radiance = Sampl e.x + radiance.y

v = true; at brdfPdf = Eva at3 factor = dif at weight = Mis2 at cosThetaOut = E = ((weight

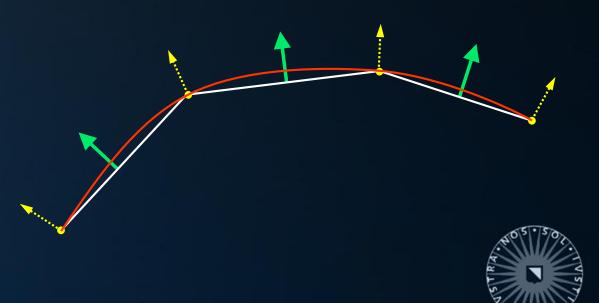
andom walk - don /ive)

; t3 brdf = Sampl urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### Normal Interpolation

Using the interpolated normal:

- Use the interpolated normal in the  $\vec{N} \cdot \vec{L}$  calculation.
- Use the original face normal when checking if a light is visible.



tica € (depth < 14.55

: = inside / 1 it = nt / nc, dde os2t = 1.0f - nn: 0, N ); 0)

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

= diffuse; = true;

-: efl + refr)) && (depth < HADIIII

D, N ); ref1 \* E \* diffu = true;

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property if; adiance = SampleLight( &rand, I. .x + radiance.y + radiance.z)

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pauro st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* read

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

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

## Today's Agenda:

- Recap
- End of the Primary Ray
- Normals
- The Camera
- Assignment P2



at a = nt

), N );

= true;

AXDEPTH)

v = true;

if;

efl + refr)) && (depth

survive = SurvivalProbability( d)

-adiance = SampleLight( &rand, 1

e.x + radiance.y + radiance.z)

st brdfPdf = EvaluateDiffuse( st3 factor = diffuse \* INVPI;

at cosThetaOut = dot( N, L );

st weight = Mis2( directPdf, brdfPdf

E ((weight \* cosThetaOut) / directPdf

efl \* E \* diffuse;

### Preparing a Free Camera

The view frustum is uniquely defined by:

- A camera position
- A target location
- A field of view (angle)
- A rotation around the view vector

### We can limit this further by specifying an 'up' vector, e.g. (0, 1, 0):

- Camera and target position
- Field of view

### Data we need to produce primary rays:

Camera position, three screen corners.

andom walk - done properly, closely following vive)

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

# No. 100 STA

29

nice ⊾ (depth (c)100

= \* inside / : ht = nt / nc, d/e -552t = 1.8f - nn -3, N ); 3)

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

= \* diffuse; = true;

efl + refr)) && (depth is MANDINI

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

#### AXDEPTH)

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 fol /ive)

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

### Most Basic Setup

View direction: V = normalize( target - pos )

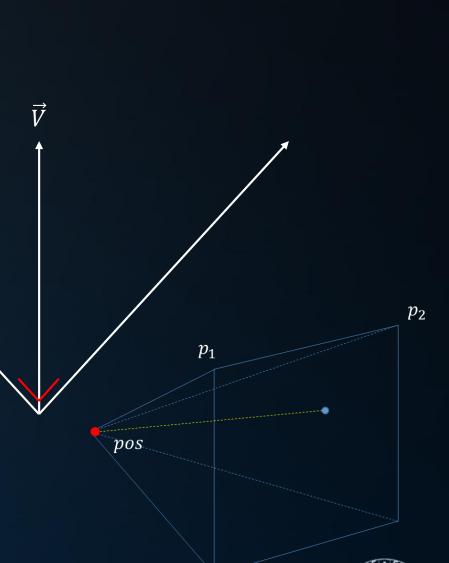
Center of screen: C = pos + V

Vectors to the left and right: (-1, 0, 0) and (1, 0, 0)

Vectors up and down: (0, 1, 0) and (0, -1, 0)

Screen corners: p1 = C + left + up

 $p_{1} = C + ierc + up$   $p_{2} = C + right + up$   $p_{3} = C + left + down$ 





), N ); = true;

#### AXDEPTH)

survive = SurvivalProbability( diff) adiance = SampleLight( &rand, I, LL, e.x + radiance.y + radiance.z) > 0) #

v = true;

at brdfPdf = EvaluateDiffuse( L, N ) st3 factor = diffuse \* INVPI; st weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* (Pur

/ive)

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

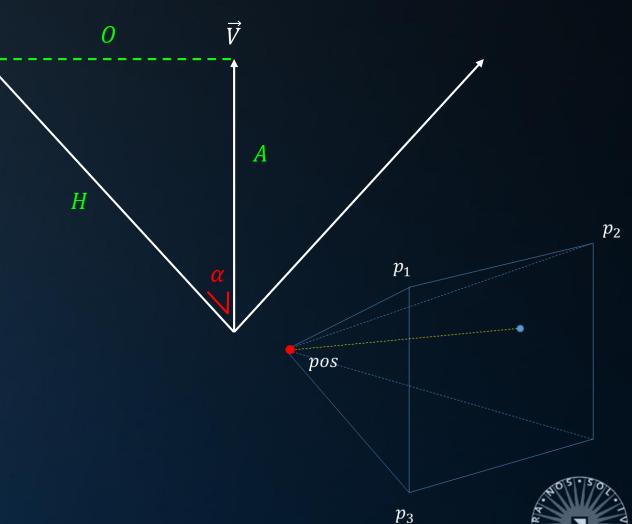
## Adding FOV

### We know that:

0 = 1 $\alpha = \frac{1}{2}FOV$ 

### Using SOHCAHTOA:

$$an \alpha = \frac{O}{A} \rightarrow A = \frac{O}{\tan \alpha}$$





### **Arbitrary Direction**

efl + refr)) && (depth < H

), N ); = true;

AXDEPTH)

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

v = true; at brdfPdf = EvaluateDiffuse( L, N st3 factor = diffuse = INVPI; at weight = Mis2( directPdf, brdfPdf at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf

andom walk - done properly, closely fol vive)

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

View direction: V = normalize( target - pos )

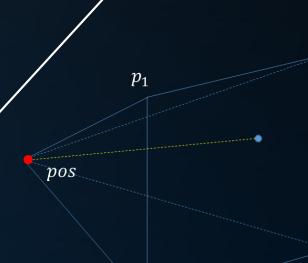
We know that up = (0, 1, 0)

### Therefore

left = normalize( cross( V, up ) ) (note that this is true even when  $\vec{V}$  is not level)

up = cross( V, left );

### Screen corners: p1 = C + left + upp2 = C + right + upp3 = C + left + down



 $\vec{V}$ 



 $p_2$ 

tic: ⊾(depth ∈ 100

= inside / 1 it = nt / nc, ddo ps2t = 1.0f - nnt D, N ); D)

st a = nt - nc, b - nt st Tr = 1 - (80 + (1 Tr) R = (D \* nnt - N

E \* diffuse; = true;

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

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

AXDEPTH)

v = true; at brdfPdf = EvaluateDiffuse( L, N.) \* Process st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \*

andom walk - done properly, closely following /ive)

; pt3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, brd pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

## Aspect Ratio

Basic idea:

If your window width is 1.3x your window height, ||p2-p1|| should be 1.3x ||p3-p1||.

So:

aspectRatio = height / width;

### Screen corners:

p1 = C + left + up \* aspectRatio
p2 = C + right + up \* aspectRatio
p3 = C + left + down \* aspectRatio

 $p_1$ 

pos

 $\vec{V}$ 

 $p_2$ 

### Fisheye Lens

efl + refr)) && (depth

), N ); = true;

AXDEPTH)

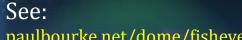
survive = SurvivalProbability adiance = SampleLight( &rand, e.x + radiance.y + radiance.z)

v = true; at brdfPdf = EvaluateDiffuse( L st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* (Public

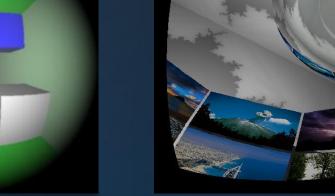
andom walk - done properly, closely follow vive)

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





### paulbourke.net/dome/fisheye



See: rifty-business.blogspot.nl/2013/08/understanding-oculus-rift-distortion.html





tica k (depth ⊂ NAS

= inside / L ht = nt / nc, ddo hs2t = 1.0f - nnt D, N ); D)

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

= diffuse = true;

-:fl + refr)) && (depth k HANDIII

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

#### AXDEPTH)

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

v = true; at brdfPdf = EvaluateDiffuse( L, N.) \* Provident st3 factor = diffuse \* INVPI; st weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

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

; pt3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, brd pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### Rotating the Camera

Rotation, "The Hard Way":

- Setup a camera matrix
- Apply this matrix to the screen center.





ic: (depth c NA

: = inside / l ht = nt / nc, dde bs2t = 1.8f - nnt 2, N ); 3)

st a = nt - nc, b = nt + st Tr = 1 - (R0 + (1 fr) R = (D \* nnt - N \*

= \* diffuse; = true;

efl + refr)) && (depth is HANDIIII)

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

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property if; radiance = SampleLight( %rand, I &:x + radiance.y + radiance.z) = 0)

v = true; t brdfPdf = EvaluateDiffuse( L, N, ) \* Process st3 factor = diffuse \* INVPI; st weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \*

andom walk - done properly, closely following /ive)

; st3 brdf = SampleDiffuse( diffuse, N, r1, r2, UR, so urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### Rotating the Camera

Rotate right, "The Easy Way":

target = pos + V; target += C \* right; V = normalize( target - pos );

### Rotate up:

target = pos + V; target += C \* up; V = normalize( target - pos );





tice € (depth (⊂1935

= inside / 1 it = nt / nc, ddo -552t = 1.0f - nnt -5, N ); 8)

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

= diffuse; = true;

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

D, N ); ~efl = E = diffuse = true;

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property if; radiance = SampleLight( &rand, I e.x + radiance.y + radiance.r) > 0

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pour st3 factor = diffuse \* INVPI; st weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* 000

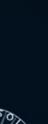
andom walk - done properly, closely following -/ive)

; pt3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, brd pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### Framerate

You can make your camera rotate at a constant speed by factoring in frame time.

- 1. Measure the time (in milliseconds) it takes to render a frame
- 2. For the next frame, multiply movement by this number



tica € (depth < 14.55

: = inside / 1 it = nt / nc, dde os2t = 1.0f - nn: 0, N ); 0)

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

= diffuse; = true;

-: efl + refr)) && (depth < HADIIII

D, N ); ref1 \* E \* diffu = true;

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property if; adiance = SampleLight( &rand, I. .x + radiance.y + radiance.z)

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pauro st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* read

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

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

## Today's Agenda:

- Recap
- End of the Primary Ray
- Normals
- The Camera
- Assignment P2



INFOGR – Lecture 4 – "Ray Tracing (2)"

## Assignment P2

tic: K (depth < 100

: = inside / L it = nt / nc, dde os2t = 1.0f - nnt 0, N ); 0)

st a = nt - nc, b - m st Tr = 1 - (R0 + (... Tr) R = (0 \* nnt - n \*

= diffuse = true;

-:fl + refr)) && (depth & MADIII

D, N ); ~efl \* E \* diffus = true;

#### AXDEPTH)

v = true; t brdfPdf = EvaluateDiffuse( L, N\_) = Punct st3 factor = diffuse \* INVPI; st weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

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

; t3 Brdf = SampleDiffuse( diffuse, N, Fl, F2, SR, Spath prvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); Sion = true:

### New Challenge

### Smallest executable.



1k-sw-raytrace'em all by Tristar & Red Sector Inc. (2004)



INFOGR – Lecture 4 – "Ray Tracing (2)"

## Assignment P2

Ø.

### Use That Debug Output!

: = inside / 1 it = nt / nc. os2t = 1.0f D, N ); B)

nt a = nt nt Tr = 1 -'r) R = (D \*

= diffus = true;

. fl + refr)) 88

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

AXDEPTH)

survive = SurvivalP estimation - doing if; radiance = SampleLight(Smand, I e.x + radiance.y + radiance.z)

v = true; t brdfPdf = EvaluateDiffuse( L, N.) \* Proc.) st3 factor = diffuse \* INVPI; st weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

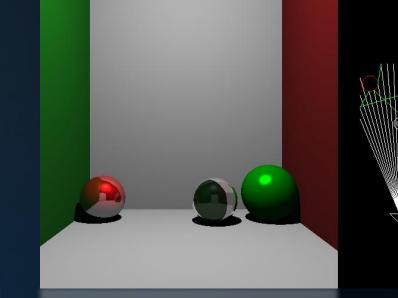
andom walk - done properly, closely following : /ive)

; pt3 brdf = SampleDiffuse( diffuse, N, F1, F2, UR, pr pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

#### PRIXITIVES, 11 LIGATS: 2 PRIXARE RAES: 2821---FOV: -3

VOGSLE ALL FERVILES UNTA VILSE KES VVOGSLE 10 LIGAVILES OFF VVOGSLE 20 FACUSS OFF VVOGSLE 30 FEFETVICAL OFF VVOGSLE 30 FEFETVICAL OFF VVOGSLE 51 OEFFACUSAL OFF

\*242525: 8 RELOSE TURE (TOTAL): 0.0077 S PASSE TURESCOR: 0.10, 0.10, 0.10 CRE FOS: 0.0000, -0.0000, -5.000 CRE FOS: 0.0000, -0.0000, -5.000 CRE ROTATION: 0.0000, -0.000 CRE SUB: 0.0000, -0.000





ice (depth c HA

: = inside / 1 it = nt / nc, dde os2t = 1.01 - nnt D, N ); D)

st a = nt - nc, b - nt st Tr = 1 - (80 + (1 Tr) R = (D \* nnt - N \*

= diffuse = true;

-:fl + refr)) && (depth k HANDERT

D, N ); ~efl \* E \* diffuse; = true;

AXDEPTH)

v = true; at brdfPdf = EvaluateDiffuse( L, N ) \* Pour st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); at cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf)

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

; st3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, st urvive; pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

# **INFOGR – Computer Graphics**

Jacco Bikker - April-July 2016 - Lecture 4: "Ray Tracing (2)"

# END of "Ray Tracing (Part 2)"

next lecture: "Ray Tracing (Part 3)"

