rics & (depth < ∧∞coo

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

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

= * diffuse = true;

efl + refr)) && (depth < MAXDEPT

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

AXDEPTH)

survive = SurvivalProbability(diffuse)
estimation - doing it properly, closed
if;
radiance = SampleLight(&rand, I, &L, &liet)

e.x + radiance.y + radiance.z) > 0) 88 (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)

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

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

/INFOMOV/ Optimization & Vectorization

J. Bikker - Sep-Nov 2019 - Lecture 14: "Grand Recap"

Welcome!



nics & (depth < Notion⊺

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

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

= * diffuse; = true;

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

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

AXDEPTH)

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

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

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

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

Today's Agenda:

- Grand Recap
- Exam
- Now What



ics & (depth < ₩00000

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

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

= * diffuse; = true;

• •fl + refr)) && (depth < MAXDEPIII

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

AXDEPTH)

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

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) * (PS

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

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

Today's Agenda:

- Grand Recap
- Exam
- Now What

Get ready for the ride

SEHWARZENEGGER

of your life.





INFOMOV – Lecture 14 – "Digest & Recap"

Recap

at Tr = 1 - (R0 + () Fr) R = (D ⁺ nnt -)

D, N); refl * E * diffuse;

AXDEPTH)

survive = SurvivalProbability(
 estimation - doing it properl
if; e.x + radiance.y + radiance.z

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

/ive)

at3 brdf = SampleDiffuse(diffu pdf; n = E * brdf * (dot(N, R) / pdf)























INFOMOV – Lecture 14 – "Digest & Recap"

Recap – lecture 1



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











f:

/ive)

urvive; pdf;

sion = true:

at3 brdf = SampleDiffuse(diffuse, N, r1, r2, 8R,

1 = E * brdf * (dot(N, R) / pdf);

Recap – lecture 1 Intel VTune Amplifier XE 2015 Basic Hotspots Hotspots by CPU Usage viewpoint (change) ? 🗢 Analysis Target 🕺 🛆 Analysis Type 🔛 Collection Log 🛛 🛍 Summary 🥝 Bottom-up 🗣 Caller/Callee 🕏 Top-down Tree 🛃 Tasks 🕨 • • • × Data Of Interest (CPU Metrics) Grouping: Function / Call Stack rViewing ∉ 1 of 49 selected stack(s) CPU Time-* 22.8% (1.029s of 4.507s) Function / Call Stack Effective Time by Utilization \gg Spin Overhead Time Time 🔲 Idle 📕 Poor 📒 Ok 📳 Ideal 🔲 Over SystemProceduralFire...on - fireobject.cpp SystemProceduralFir...fireobject.cpp:1459 FireObject::ProcessFireCollisionsRange 3.444s 0s 0s SystemProceduralFire...fireobject.cpp:1377 ⊞ <a>FireObject::FireCollisionCallback 3.025s 0s 0s Smoke.exe!Parallel...managertbb.cpp:573 0s 0s NtWaitForSingleObject 3.406s Smoke.exe![TBB paral...- parallel_for.h:212 4.507s 0s 05 ♀ **-** ► × Smoke.exeltbb::inter...- parallel_for.h:150 tmpl84.00a_2013 - Microsoft Visual Studio (Administrator) FILE EDIT VIEW PROJECT BUILD DEBUG NSIGHT TOOLS CODE-BUILDER ANALYZE WINDOW HELP Release - 🕨 II = 😘 💪 🗟 k Sign in Win32 31.4s 31.5s 31.6s 31.7s 31.8s 31.9s 3 Ruler Area plate 150701(3).vsp + × Template 150701(2).vsp 🗹 🎮 Frame Current View: Function Detai - 🔻 🕆 = 🦖 👇 💾 😭 🖽) ∩` (∰) Assumptions Thread (D -Tmpl8::Game::Simulate Running G Solution 'tmpl84.00a_2013' Template.exe 🔽 🏨 CPU Time 🔺 🖪 Template External Deper Spin and Overhead... Current function Called functions Calling functions استأسب فالله readme.txt CPU Sample ++ game.cpp Bottom of Stack 🔽 🞮 Tasks ata = nt -▶ 🖻 game.h CPU Usage ++ surface.cpp 67.4% Isurface.h Any Thread Any Module Any Utilization - 🗣 ++ template.cpp E template.h 1ode: on 💌 Loop Mode: Functions only Related Views: Caller/Callee Functions Performance metric: Inclusive Samples % _ [[] × Averages Cal Stacks Function Code View % Exclusive % Inclusive Module Source File d: \water \game. % Calls Module d: \water \gam Tmpl8::Game::Tick Tmpl8::Game::Init 8.70s 0.77s 91.88% 8.12% water Impl8:::Game::DrawTriangle 1.330 1.33e 8.80% 8.80% water d: lwater loame 1.196 1.196 7.86% 2.87% 7.86% water drop[i].pos += drop[i].pos - prev_pos; efl + refr)) && (depth pl8::Game::RenderZSprite d: \water \game mpl8::Game::RenderWaterSurface 0.436 1.766 11.67% water water d:\water\oame npl8::Surface::Clear 0.11s 0.11s 0.75% 0.75% d:\water\purfa drop[i].pos += gravity * 0.25f; mpl8::Game::RenderDebugIn 0.03s 0.05s water d: \water \game Crysis 2), N); 0.13% 0.10% 0.02% 0.01% 0.01% 0.01% 0.13% 0.10% 0.02% 0.01% water water d: \water \surfac d: \water \game. Tmpl8::Surface::Plot 0.02s 0.02s 0.02s mpl8::Game::DownSca for (int step = 0; step < 3; step++)</pre> refl * E * diffuse; **Gameplay Performance** Impl8::Game::TmeSmooth 0.00s 0.00s 0.00s 0.25s water water d: \water \game mpl8::Surface::AddLine 0.00s d:\water\surfao 0.00% 1.66% 0.01% 99.93% water d: \water \temple [006ADCD0] 0.00s 0.00s for (int j = i + 1; j < DROPCOUNT; j++)</pre> 5.4 % 0.00% f:\dd\vctools\cr tmainCRTStartup 0.00s 15.08s water 0.00% DL_main 0.00s 15.03s 99.57% water d: \water \templa 40 Tmpl8::Game::Tick 0.00s 13.88s 91.96% water d:\water\pame. 25.8 % float dist = length(drop[i].pos - drop[j].pos); (AXDEPTH) Tmpl8::Game::DrawBoat Tmpl8::Game::GlowLine 0.00s 0.00s 0.00% 0.01% water water d: \water \game. (dist < (DROPRADIUS * 2) 33.7 % 35 0.019 d:\water\game. $\neg M_{n}$ survive = SurvivalProbabil 1.9 % Source Log drop[i].pos += direction * (DROPRADIUS * 2 - dist) * 0.02f; 0.5 % int step = 0; step < 3; step</pre> drop[j].pos -= direction * (DROPRADIUS * 2 - dist) * 0.02f; S 25 // simulation step 3a - satisfy constrai for (int j = i + 1; j < DROPCOUNT; j++</pre> Child Calls % Cals radiance = SampleLight(&r float dist = (drop(i).pos - drop(j).pos).Length();
if (dist < (DROFRADIUS * 2))</pre> 20 e.x + radiance.y + radianc if (drop[i].pos.y > 20) drop[i].pos.y = 19.99f - drop[i].pos.z * 0.0001f; vector3 direction = (drop[i].pos - drop[j].pos).Normalizec drop[i].pos 4= direction * (DROPRADIUS * 2 - dist) * 0.021 drop[j].pos -= direction * (DROPRADIUS * 2 - dist) * 0.021 15 if (drop[i].pos.x < -20) drop[i].pos.x = -19.99f + drop[i].pos.z * 0.0001f; if (drop[i].pos.x > 20) drop[i].pos.x = 19.99f - drop[i].pos.z * 0.0001f; v = true; if (drop[i].pos.z < -20) drop[i].pos.z = -19.99f + drop[i].pos.z * 0.0001f; / simulation step 3b - esticity constraints - evide walls (dampell)pear > 201 dept)lpear > 10.396 - dept13 pears + 0.0001 (dampell)pears + -001 dept[1]pears = 10.396 + drep[1]pears + 0.001 (dampell)pears + 001 dept[1]pears = 10.396 + drep[1]pears + 0.0001 (dampell)pears + -001 dept[1]pears = 10.396 + drep[1]pears + 0.0001 at brdfPdf = EvaluateDiffu if (drop[i].pos.z > 20) drop[i].pos.z = 19.99f - drop[i].pos.z * 0.0001f; 0.1 % at3 factor = diffuse * IN\ 100 % - 4 at weight = Mis2(directPdf at cosThetaOut = dot(N, L Source file: d: \water \game.cpp Line 97 E * ((weight * cosThetaOut) / dire andom walk - done properly, c

6

Recap – lecture 2

t

tics & (depth < MoxOS

nt = nt / nc, do ps2t = 1.0f - nn D, N); D)

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

* diffuse; = true;

. fl + refr)) && ((

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

(AXDEPTH)

survive = SurvivalProb
estimation - doing it
if;
radiance = SampleLight
e.x + radiance.y + rad

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

andom walk - done properly, closely follow /ive)

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

Red = u4 & (255 << 16); Green = u4 & (255 << 8); Blue = u4 & 255;

E

Ne need to go deeper

F

E.

Е

<pre>fldz xor ecx, ecx fld dword ptr ds:[405290h] mov edx, 28929227h fld dword ptr ds:[40528Ch] push esi mov esi, 0C350h = 50000 add ecx, edx mov eax, 91D2A969h xor edx, 1773/352h shr ecx, 1 mul eax, edx fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr ecx, 1 mul eax, etx shr etx, 0Eh dec esi jne tobetimed<0>+1Fh</pre>		
<pre>xor ecx, ecx fld dword ptr ds:[405290h] mov edx, 28929227h fld dword ptr ds:[40528Ch] push esi mov esi, 0C350h = 50000 add ecx, edx mov eax, 91D2A969h xor edx, 17737352h shr ecx, 1 mul eax, edx fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx, 0Eh dec esi jne tobetimed<0>+1Fh</pre>	fldz	
<pre>fld dword ptr ds:[405290h] mov edx, 28929227h fld dword ptr ds:[40528Ch] push esi mov esi, 0C350h = 50000 add ecx, edx mov eax, 91D2A969h xor edx, 17737352h shr ecx, 1 mul eax, edx fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx, 0Eh dec esi jne tobetimed<0>+1Fh</pre>	xor ecx, ecx	
<pre>mov edx, 28929227h fld dword ptr ds:[40528Ch] push esi mov esi, 0C350h = 50000 add ecx, edx mov eax, 91D2A969h xor edx, 17737352h = 28763 (!! xor edx, 17737352h shr ecx, 1 mul eax, edx fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr ecx, 1 mul eax, edx shr ecx, 1 mul eax, edx shr edx 0Eh dec esi jne tobetimed<0>+1Fh</pre>	fld dword ptr ds:[405290h]	
<pre>fld dword ptr ds:[40528Ch] push esi mov esi, 0C350h = 50000 add ecx, edx mov eax, 91D2A969h xor edx, 17737352h shr ecx, 1 mul eax, edx fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx_ 0Eh dec esi jne tobetimed<0>+1Fh</pre>	mov_edx, 28929227h	
<pre>push esi mov esi, 0C350h = 50000 add ecx, edx mov eax, 91D2A969h = 2⁴⁶/28763 (!! xor edx, 17737352h shr ecx, 1 mul eax, edx fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx 0Eh dec esi jne tobetimed<0>+1Fh</pre>	fld dword ptr ds:[40528Ch]	
<pre>mov esi, 0C350h = 50000 add ecx, edx mov eax, 91D2A969h = 246 28763 (!! xor edx, 17737352h shr ecx, 1 mul eax, edx fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx, 0Eh dec esi jne tobetimed<0>+1Fh</pre>	push esi	
<pre>add ecx, edx mov eax, 91D2A969h xor edx, 17737352h shr ecx, 1 mul eax, edx fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr ecx, 4 mul eax, edx shr edx, 0Eh</pre>	mov esi, 0C350h) = 50000	
<pre>mov eax. 91D2A969h xor edx, 1773/352D shr ecx, 1 mul eax, edx fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1), st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx, 0Eh dec esi jne tobetimed<0>+1Fh</pre>	add ecx, edx 2 ⁴⁶	/
<pre>xor edx, 17737352D 28763 shr ecx, 1 mul eax, edx fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx, 0Fh dec esi jne tobetimed<0>+1Fh</pre>	mov eax, $91D2A969h$ = $-20767a$	
<pre>shr ecx, 1 mul eax, edx fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx, 0Eh dec esi jne tobetimed<0>+1Fh</pre>	xor edx, 17737352h 28763	
<pre>mul eax, edx fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx_ 0Eh dec esi jne tobetimed<0>+1Fh</pre>	shr ecx, 1	
<pre>fld st(1) faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx_ 0Eh dec esi jne tobetimed<0>+1Fh</pre>	mul eax, edx	
<pre>faddp st(3), st mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx_0Eh dec esi jne tobetimed<0>+1Fh</pre>	fld st(1)	
<pre>mov eax, 91D2A969h shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx_ 0Eh dec esi jne tobetimed<0>+1Fh</pre>	<pre>faddp st(3), st</pre>	
<pre>shr edx, 0Eh add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx, 0Eh dec esi jne tobetimed<0>+1Fh</pre>	mov eax, 91D2A969h	
<pre>add ecx. edx fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx_ 0Eh dec esi jne tobetimed<0>+1Fh</pre>	shr edx, 0Eh	
<pre>fmul st(1),st xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx_ 0Eh dec esi jne tobetimed<0>+1Fh</pre>	add ecx, edx	
<pre>xor edx, 17737352h shr ecx, 1 mul eax, edx shr edx, 0Fh dec esi jne tobetimed<0>+1Fh</pre>	<pre>fmul st(1),st</pre>	
<pre>shr ecx, 1 mul eax, edx shr edx_ 0Fh dec esi jne tobetimed<0>+1Fh</pre>	xor edx, 17737352h	
<pre>mul eax, edx shr edxOFh dec esi jne tobetimed<0>+1Fh</pre>	shr ecx, 1	
shr edx, OFh dec esi jne tobetimed<0>+1Fh	mul eax, edx	
dec esi jne tobetimed<0>+1Fh	shr edx, OFh	NO5 .
🤍 jne tobetimed<0>+1Fh// 🛛 😓 💺	dec esi	
	jne tobetimed<0>+1Fh	



INFOMOV – Lecture 14 – "Digest & Recap"

Recap – lecture 3

Τ0

T1

tics ≹j(depth < NoCOS

c = inside ? 1 (1) ht = nt / nc, ddn bs2t = 1.0f - nnt (D, N); 3)

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

* diffuse; = true;

. fl + refr)) 88 (dent)

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

AXDEPTH)

survive = SurvivalProbability(dif .estimation - doing it properly If; radiance = SampleLight(&rand, I,

e.x + radiance.y + radiance.z) > 0)

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

andom walk - done properly, close ⁄ive)

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



L1 I-\$ Τ0 L2 \$ L1 D-\$ T1 L3 \$ Τ0 L1 I-\$ L2 \$ L1 D-\$ T1 L1 I-\$ Τ0 L2 \$ L1 D-\$ T1

	Slot U	Slot 1	
000			
001			
002			
003			
004			
005			
006		Making analy March	10121-0101
007		a start war	1.1.1
008		A Later	
009		A COLORED AND	
00A 🗌			
00B			
00C 🛛		L'VOLA	R40 0.
00D [
00D			
00F			
		Construction of the second second	

ot 2	slot 3



INFOMOV – Lecture 14 – "Digest & Recap"

Recap – lecture 4

ics
{ (depth < Multi
: = inside } 1
it = nt / nc, c
solt = 1.0f - r
0, N);
})</pre>

t Tr = 1 r) R = (D *

= true;

fl + refr)) &8

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

AXDEPTH)

survive = Surviv
estimation - do
if;

radiance = SampleLight(&rand, I, e.x + radiance.y + radiance.z) > 0

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 Soul /ive)

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





_ 🗆 🗙



... hem het cadeau geven dat hij écht wil.







Recap – lecture 5 & 6

ata = nt -



radiance = SampleLight(&rand,



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







SIMD Basics

Other instructions:

m128 c4 = _mm_div_ps(a4, b4);	<pre>// component-wise division</pre>
m128 d4 = _mm_sqrt_ps(a4);	<pre>// four square roots</pre>
m128 d4 = _mm_rcp_ps(a4);	<pre>// four reciprocals</pre>
m128 d4 = _mm_rsqrt_ps(a4);	<pre>// four reciprocal square roots</pre>

Agner

_m128 d4 = _mm_max_ps(a4, b4); _____m128 d4 = __mm__min__ps(a4, b4);

Keep the assembler-like syntax in mind:

"Automate vectorization is the easiest way of generating SIMD code, and I would recommend to use this method when it orks. Automatic vectorization, av fail or produce suboptimal, de in the following cases: • when the a prithm is too complex. when data han to be re-arranged in der to fit into vectors and it is not obvious to a compiler how to be this or when other parts of the code needs to be a nged to han the re-arranged data. when it is not know. In the control of the which data sets are bigger or smaller than the vecto. ize. ompiler whether the size of a data set is when it is not known to . a multiple of the vector r not. when the algorithm in twest Us to functions that are defined elsewhere or cannot e inlined d which are not readily available in vector versions. when the algorithm involves many bunches that are not easily vectorized.

- when floating point operations have to be peordered or transformed and it is particular whether the compiler whether these transformations are per ssible with respect to precision, over low, etc.
- wher anctions are implemented with lookup ta

(!)



Recap – lecture 7

nics ≹j(depth < MoCOS

t = inside } 1 | 1 | 1 ht = nt / nc, ddn | 1 s2t = 1.0f - nnt | n 2, N); 3)

at a = nt - nc, b = nt = $\frac{1}{10}$ at Tr = 1 - (R0 + (1 - R0) Tr) R = (D = nnt - N = (d0)

= * diffuse; = true;

efl + refr)) && (depth < NAXDEPTION

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

AXDEPTH)

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

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

andom walk - done properly, close /ive)

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















Recap – lecture 8

ics & (depth < Mox⊡s

c = inside f | ht = nt / nc, ddh os2t = 1.0f - nnt O, N); D)

nt a = nt - nc, b = nc nt Tr = 1 - (R0 + (1 - Rc ir) R = (D ⁼ nnt - N ⁼ (dd

= * diffuse; = true;

-•fl + refr)) && (depth < NAX

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

AXDEPTH)

survive = SurvivalProbability(diffus estimation - doing it properly close ff; radiance = SampleLight(&rand, I, &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 3
/ive)

, H33 Brdf = SampleDiffuse(diffuse, N, r1, r2, &R, &pdf urvive; .pdf; n = E * brdf * (dot(N, R) / pdf); sion = true:



POTATO SALAD

Some good cooks sprinkle grated pimiento cheese on this

4 cups diced cooked potatoes 1 cup sliced celery 3 hard-cooked eggs, cut up 1/2 cup finely cut onion or sliced green onions 1/4 cup sliced radishes 1 cup mayonnaise 1 tablespoon vinegar 1 taspoon prepared mustard 11/2 to 2 teaspoons salt 1/8 teaspoon pepper Lettuce

Mix all the ingredients in a bowl. Cover and refrigerate several hours so flavors can blend. Serve on crisp lettuce. Makes 6 servings.











Recap – lecture 9 & 10

nics & (depth < Moo

t = inside } l | | | | ht = nt / nc, ddn | | os2t = 1.0f - nnt | | >, N); ≥)

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

= * diffuse; = true;

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

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

AXDEPTH)

survive = SurvivalProbability(diffuse



urvive;

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





INFOMOV – Lecture 14 – "Digest & Recap"

Recap – lecture 11





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



INFOMOV – Lecture 14 – "Digest & Recap"

SCHWARZENEGGER

Get ready for the ride of your life.

SCHWARZENEGGER

TOTAL RECAP

TOTAL RECAR

S C H W A R Z E N E G G E R

GE

Get ready for the ride of your life.

Recap – Lecture 14

ics & (depth < NOCCS⊂

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

= * 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, &light) 2.x + radiance.y + radiance.z) > 0) &&

w = 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 in /ive)

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

Recap

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

(AXDEPTH)

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

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

andom walk - done properly, closely follo /ive)

at3 brdf = SampleDiffuse(diffuse, N, r1, r2, &R, I urvive; pdf; 1 = E * brdf * (dot(N, R) / pdf); sion = true



"Dear Charles,

In other words:

We will talk a lot about the 'C' in O(N).

* disclaimer: 'that has not been optimized by an expert'.



At the end of this course:

You will know how to speed up critical code by a factor 2x to 10x (and more).

- You will be able to do this to virtually any program*.
- Your understanding of higher level optimization approaches will increase.
- You will be able to apply these principles to new / alien hardware.
- You will have a more intimate relationship with your computer.

nics & (depth < Notion⊺

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

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

= * diffuse; = true;

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

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

AXDEPTH)

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

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

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

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

Today's Agenda:

- Grand Recap
- Exam
- Now What



nics & (depth < MoxDea

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

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

= * diffuse = true;

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

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

AXDEPTH)

survive = SurvivalProbability(diffuse estimation - doing it properly, closed Hf; radiance = SampleLight(&rand, I, &L, &L)

e.x + radiance.y + radiance.z) > 0) 88 (do 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 Soul /ive)

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

What to Study

1. Slides

- 2. Literature on the website and in the slides:
 - Modern Microprocessors: a 90 minute guide, see lecture 2 slides or <u>click here</u>
 - What Every Programmer Should Know About Memory (just the <u>vellow bits</u>)
 - Gallery of Processor Cache Effects (<u>link</u>)
 - Game Programming Patterns Data Locality
 - Data-Oriented Design (Or Why You Might Be Shooting Yourself in the Foot With OOP)
 - The Neglected Art of Fixed Point Arithmetic
 - Cache-oblivious Algorithms and Data Structures (just the <u>vellow bits</u>)
 - A Survey of General-Purpose Computation on Graphics Hardware

3. <u>2016/2017/2018</u> exams

4. Skills you picked up with the practical assignments



nics & (depth < NODEs

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

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

= * diffuse; = true;

efl + refr)) && (depth < MAXDEPIN

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

AXDEPTH)

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

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) * (rad

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

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

Example Questions

You may bring a dictionary to the exam. You may answer in Dutch, if you wish. You may **not** bring notes to the exam. You may bring pizza to the exam.

CPUs and GPUs have fundamentally different core strategies for dealing with

latencies such as memory access time. What are these strategies?



tics & (depth < ₩XDDD

: = inside ? 1 (1.1) ht = nt / nc, ddn (1.1) ps2t = 1.0f - nnt (1.1) p, N); ∂)

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

= * diffuse; = true;

efl + refr)) && (depth < MAXDEPIN

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

AXDEPTH)

w = 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) * (red

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

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

Example Questions

You may bring a dictionary to the exam. You may answer in Dutch, if you wish. You may **not** bring notes to the exam. You may bring pizza to the exam.

Why is the theoretical peak performance of a GPU typically much higher

than that of a CPU?



nics & (depth < Modes

= = inside } 1 | | | | ht = nt / nc, ddn | | ps2t = 1.0f - nnt | n D, N); ∂)

at a = nt - nc, b = nt at Tr = 1 - (R0 + (1 - R0 Fr) R = (D = nnt - N - (000

= * diffuse = true;

efl + refr)) && (depth < MAXDEPTH

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

AXDEPTH)

survive = SurvivalProbability(diffuse estimation - doing it properly, if; radiance = SampleLight(&rand, I, %L, %L)

e.x + radiance.y + radiance.z) > 0) && (d. 100

v = true; at brdfPdf = EvaluateDiffuse(L, N) Poundive 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 Small /ive)

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

Example Questions

What is DMA?

You may answer in Dutch, if you wish. You may **not** bring notes to the exam. You may bring pizza to the exam.

You may bring a dictionary to the exam.





nics & (depth < NODEs

t = inside 7 1 1 1 0 nt = nt / nc, ddn us2t = 1.0f - nnt ∩ n D, N); 2)

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

= * diffuse; = true;

efl + refr)) && (depth < MAXDEPIN

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) 88

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) * (rad

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

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

Example Questions

Explain the concept of streaming processing.

nics & (depth < Monder

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

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

= * diffuse = true;

efl + refr)) && (depth < MAXDEPTH

D, 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) 88

w = 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)

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

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

Example Questions

What or who is NUMA?

You may bring a dictionary to the exam. You may answer in Dutch, if you wish. You may **not** bring notes to the exam. You may bring pizza to the exam.



24



tics & (depth < Moose

t = inside 7 1 1 1 0 ht = nt / nc, ddh us2t = 1.0f - nnt 1 n D, N); ∂)

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

= * diffuse = true;

efl + refr)) && (depth < MONDEPTH

D, 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) 88

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) * (rac

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

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

Example Questions

Explain what false sharing is.

You may bring a dictionary to the exam. You may answer in Dutch, if you wish. You may **not** bring notes to the exam. You may bring pizza to the exam.

25

nics & (depth < NODEs

= = inside ? 1 (1.1) ht = nt / nc, ddh ps2t = 1.0f - nnt (2, N); 2)

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

= * diffuse; = true;

efl + refr)) && (depth < MAXDEPTH

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

AXDEPTH)

survive = SurvivalProbability(diffuse .estimation - doing it properly, closed if; radiance = SampleLight(&rand, I, &l, align)

e.x + radiance.y + radiance.z) > 0) && (double w = 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) * (rad

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

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

Example Questions

How does a GPU handle conditional code?





), N); refl * E * diffuse;

AXDEPTH)

survive = SurvivalProbability(diffu radiance = SampleLight(&rand, I, &L,

e.x + radiance.y + radiance.z) > 0) 88

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) = (Pa

andom walk - done properly, closely followi /ive)

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

Example Questions

Why does OpenCL have a native_sqrt as well as an sqrtf?



nics & (depth < Modes

: = inside ? 1 () 0 nt = nt / nc, ddn () ps2t = 1.0f - nat () 2, N); 2)

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

= * diffuse; = true;

efl + refr)) && (depth < MAXDEPTH

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) 88

w = true; at brdfPdf = EvaluateDiffuse(L, N) Paurologi 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 Smool /ive)

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

Example Questions

Do modern systems still use SRAM? Why / why not?



tics & (depth < Moder

: = inside 7 1 (177) ht = nt / nc, ddh ps2t = 1.0f - nnt (2, N); 2)

at a = nt - nc, b = 11 at Tr = 1 - (R0 + 1 - 11 Ir) R = (D ⁺ nnt - N

= * diffuse; = true;

efl + refr)) && (depth < MAXDEPIN

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 && (dot)

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) * (rad

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

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

Example Questions

You may bring a dictionary to the exam. You may answer in Dutch, if you wish. You may **not** bring notes to the exam. You may bring pizza to the exam.

How many bits are needed for a 128KB 8-way set associative cache,

assuming a cache line size of 128 bytes?



nics & (depth < NODEs

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

at a = nt - nc, b = nt = ntat Tr = 1 - (R0 + (1 - Rc) Tr) R = (D = nnt - N = 100

= * diffuse; = true;

efl + refr)) && (depth < MAXDEPIN

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

AXDEPTH)

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

w = 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) * (red

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

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

Example Questions

You may bring a dictionary to the exam. You may answer in Dutch, if you wish. You may **not** bring notes to the exam. You may bring pizza to the exam.

Is self-modifying code possible on a modern processor? Under what

conditions?



nics & (depth < Notion⊺

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

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

= * diffuse; = true;

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

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

AXDEPTH)

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

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

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

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

Today's Agenda:

- Grand Recap
- Exam
- Now What





estimation - doing it properly the if; adiance = SampleLight(&rand, I, &L, &Light) e.x + radiance.y + radiance.z) > 0) && (000

survive = SurvivalP

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

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

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







nics ≹j(depth < ™0000

: = inside ? 1 1 1 1 ht = nt / nc, ddn bs2t = 1.0f - nnt " n D, N); D)

at a = nt - nc, b = nt at Tr = 1 - (R0 + (1 - R0 Ir) R = (D = nnt - N = (d00)

= * diffuse; = true;



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









ics
& (depth < MAL
& (depth < MAL
c = inside ?
ht = nt / nc,
ss2t = 1.0f), N);
</pre>

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

* diffus = true;

fl + refr))

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

AXDEPTH)

survive = Surviva estimation - doi df; radiance = Sample e.x + radiance.y

w = true; at brdfPdf = Eval at3 factor = diffuse ______, at weight = Mis2(directPdf, brdfPdf); at cosThetaOut = dot(N, L); E * ((weight * cosThetaOut) / directPdf

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

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







nics & (depth < Moccs

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

= * diffuse; = true;

efl + refr)) && (depth < MADEPTION

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

AXDEPTH)

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

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 /ive)

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

COMPUTER PROGRAMMER



What my friends think I do



What my spouse thinks I do



What my mom thinks I do



What I think I do



What society thinks I do



What I actually do



tics ≹ (depth < Max0)

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

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

= * diffuse; = true;

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

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

AXDEPTH)

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

w = 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)

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

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



/INFOMOV2019/

