tic: (depth < NA

= 1051de / 1 ht = nt / nc, dde 552t = 1.8f - nnt 5, N ); 3)

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

= diffuse; = true;

efl + refr)) && (depth k HAADIII

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

AXDEPTH)

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) \* 1

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

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

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

Get ready for the ride of your life.



tic: ≰ (depth < ⊡

: = inside / L it = nt / nc, dde os2t = 1.01 - ... ), N ); 3)

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

= diffuse = true;

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

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

AXDEPTH)

survive = SurvivalProbability difference estimation - doing it properion if; adiance = SampleLight( @rand I =.x + radiance.y + radiance.z) > 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)

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

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

# INFOGR – Computer Graphics

Jacco Bikker - April-July 2015 - Lecture 13: "Grand Recap"

# Welcome!



## RECAP

tica **i (depth** k 1000

= inside / : it = nt / nc, dda ss2t = 1.0f - nn; 3) N );

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

= diffuse = true;

efl + refr)) && (depth & HANDIT

D, N ); refl \* E \* diffuse; = 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 ) \* Pour 1 st3 factor = diffuse \* INVPI; st weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* 0000

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

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

Lecture 2: Rasters, Vectors, Colors

#### Math:

Vectors: magnitude, Pythagoras, linear (in)dependency, normalization, positions versus vectors, scalars, bases, Cartesian coordinate system, orthonormal, dot product (and its relation to the cosine), cross product.

#### Concepts:

Raster, frame rate, vertical retrace, 'frame-less', RGB colors, 16-bit, palletized, HDR.









tice ⊾ (depth ∈ RAS

: = inside / l ht = nt / nc, ddo os2t = 1.0f - nnt -D, N ); B)

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

= diffuse; = true;

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

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

AXDEPTH)

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

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

sndom walk - done properly, closely following vive)

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

### Tutorial 1

#### Make sure you are able to:

- Show that the scalar product of vectors is commutative and associative;
- Show the relation between magnitude and the dot of a vector with itself;
- Show that for two random vectors  $\vec{a}$  and  $\vec{b}$ ,  $\vec{a} \times \vec{b} = -(\vec{b} \times \vec{a})$  (exercise 6).
  - Turn 2D coordinates into screen coordinates and vice versa (exercise 8).



## RECAP

11c) 4 (depth - 114)

= inside / 1 it = nt / nc, ddo ss2t = 1.0f = ont ), N ); 3)

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

= diffuse; = true;

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

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

AXDEPTH)

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)

; pt3 brdf = SampleDiffuse( diffuse, N, F1, F2, R, F3, pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

#### Lecture 3: Geometry & Textures

#### Math:

Slope-intersect, implicit curves, functions, mappings, general implicit line form (and its relation to the normal), half spaces, parametric curves, SOHCAHTOA, implicit circles, implicit planes, parametric circles / spheres / planes.

#### **Concepts:**

Procedural textures, texture mapping, clamping and tiling, oversampling, undersampling, bilinear interpolation, MIP-mapping, trilinear interpolation.









tica ⊾ (depth < 10.5

: = inside / l ht = nt / nc, ddo os2t = 1.0f - nnt -D, N ); B)

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

= diffuse; = true;

efl + refr)) 88 (depth is MANDER

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

AXDEPTH)

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

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

sndom walk - done properly, closely following vive)

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

### Tutorial 2

#### Make sure you are able to:

- Turn a slope-intersect representation into parametric / implicit and vice versa;
- Calculate the normal for a pair of (linear independent) vectors;
- Calculate the distance of a point to a sphere.



tice k (depth < 10.5

= inside / 1 it = nt / nc, dde 552t = 1.0f = nnt 5, N ); 3)

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

= diffuse = true;

⊆ efl + refr)) && (depth < HAODE

D, N ); •efl \* E \* diffuse; = true;

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property if; radiance = SampleLight( &rand, I &:x + radiance.y + radiance.r) = 0.000

v = true;

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

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

; pt3 brdf = SampleDiffuse( diffuse, N, r1, r2, NR, bp3 pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

### Lecture 5: Engine Fundamentals

#### Math:

Matrices: coefficients, diagonal matrices, the identity and zero matrix; matrix addition, matrix/scalar, matrix/vector and matrix/matrix multiplication, distributive, associative, commutative, transpose, inverse, determinant, Laplace, Sarrus, cofactors, adjoint, (uniform) scaling, shearing, projection, reflection, rotation, linear transforms, transforming normals.

#### **Concepts:**

Rendering pipeline, scenegraph, object space, camera space, screen space, connectivity data, fragments.







tic: K (depth < 100⊂

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

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

= diffuse; = true;

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

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

#### AXDEPTH)

survive = SurvivalProbability difference estimation - doing it properly ff; 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 following : /ive)

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

#### Tutorial 3

Make sure you are able to:

- Multiply two matrices;
- Calculate the determinant of a matrix;
- Construct a scaling matrix;
- Transform a normal;
- Construct a matrix with translation.



## RECAP

tica k (depth < 10.5

= inside / 1 it = nt / nc, dde ss2t = 1.0f = nnt ), N ); 3)

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

= diffuse; = true;

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

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

AXDEPTH)

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

w = true; st brdfPdf = EvaluateDiffuse( L, N) Part st3 factor = diffuse \* INVPI; at weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* 100

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

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

#### Lecture 6: Projection & Rasterization

#### Math:

View frustum, camera space, orthographic view volume, canonical view volume, perspective projection, homogeneous coordinates, homogenization.

#### **Concepts:**

Linear perspective, fish eye lens, parallel projection, perspective projection, rasterization, connectivity data, triangle strips, normal interpolation, pervertex shading, per-pixel shading, light reflection, barycentric coordinates.









tice ⊾(depth < 1935

: = inside / l it = nt / nc, ddo os2t = 1.0f - nnt -O, N ); B)

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

= diffuse; = true;

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

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

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it properly if; adiance = SampleLight( %rand, I, M) 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)

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

### Tutorial 4

Make sure you are able to:

- Construct a 'look-at' matrix using *E*,  $\vec{V}$  and  $\vec{up}$ ;
- Combine multiple affine transforms into one;
- Transform a 3D vector using a 4 × 4 matrix (including homogenization).



tic: K (depth < NA

= = inside / 1 nt = nt / nc, dds os2t = 1.0f = nnt 0, N ); 0)

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

= diffuse = true;

: :fl + refr)) && (depth ic HACOLI)

D, N ); refl \* E \* diffuse; = 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 a /ive)

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

#### Lecture 7: Visibility

#### Concepts:

Painter's, overdraw, BSP traversal (back-to-front, front-to-back), z-buffer, values in the z-buffer, z-fighting, Sutherland-Hodgeman clipping, n-gons, guard bands, back-face culling, frustum culling, hierarchical bounding volume culling, culling using a grid, portals: visibility, mirrors, 'portals'.







## RECAP

tica ≰ (depth < 10.5

= inside / 1 it = nt / nc, dde ss2t = 1.0f - nnt 3, N ); 3)

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

= diffuse; = true;

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

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

AXDEPTH)

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

w = true; st brdfPdf = EvaluateDiffuse( L, N ) Pours) st3 factor = diffuse \* INVPI; at 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, pdf; n = E \* brdf \* (dot( N, R ) / pdf); sion = true:

#### Lecture 8: Ray Tracing Intro

#### Math:

Rendering equation, ray equation, setting up a world space screen plane, ray setup, ray/plane and ray/sphere intersection, distance attenuation, N dot L.

#### Concepts:

The "God Algorithm": light transport in nature, ray tracing versus rasterization, convex / concave, reflection and shadows in a rasterizer, global data, ray optics, Fresnel, Snell, Whitted-style (recursive) ray tracing.







fic: ⊾ (depth < 1925

: = inside / 1 it = nt / nc, dde os2t = 1.0f - nnt -O, N ); B)

st a = nt - nc, b = nt - ncst Tr = 1 - (R0 + 1)Tr) R = (0 \* nnt - N)

E \* diffuse; = true;

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

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

#### AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property ff; radiance = SampleLight( %rand, I e.x + radiance.y + radiance.z) = 0.0000

v = true;

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

sndom walk - done properly, closely following vive)

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

### Tutorial 5

Make sure you are able to:

- Construct the virtual screen plane given *E*,  $\vec{V}$ ,  $\vec{up}$  and the FOV;
- Construct a (normalized) ray through a pixel on this screen plane;
- Intersect the ray with planes and spheres;
- Calculate normals for the intersection points;
- Calculate the reflection of a ray using an intersection point and its normal.



## RECAP

tic: ⊾(depth ⊂ NA

= inside / L it = nt / nc, dde -552t = 1.0f - nnt -5, N ); 3)

st a = nt - nc, b - nt st Tr = 1 - (R8 + (1 Tr) R = (D \* nnt - N

= diffuse; = true;

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

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

AXDEPTH)

survive = SurvivalProbability( difference estimation - doing it property if; radiance = SampleLight( &rand, I 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)

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

#### Lecture 9: Shading Models

Math:

Clamped cosine, irradiance: integrating over hemisphere, steradians.

#### Concepts:

Light transport: emitters, surfaces and materials, sensors; IES lights, absorption, scattering, directional lights, irradiance, material properties, optical discontinuities, exitance, radiance, pinhole camera, aperture, shading, BRDF, Phong, 'ambient', physically based rendering.

Questions?









tic: K (depth < 100

= inside / 1 it = nt / nc, dde ss2t = 1.01 - nnt 5, N ); 3)

st a = nt - nc, b - nt + + st Tr = 1 - (R8 + (1 - 1) Tr) R = (D \* nnt - N

= diffuse = true;

⊂ efl + refr)) && (depth < HAODUT

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

#### AXDEPTH)

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

w = true; st brdfPdf = EvaluateDiffuse( L, N ) \* 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)

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

#### Lecture 10: Ground Truth

#### **Concepts:**

Distributed ray tracing, glossy reflections, soft shadows, umbra, penumbra, area lights, shadow maps, contact shadows, visibility integral, Monte-Carlo, stochastic soft shadows, variance / noise, stochastic reflections, stratification, depth of field, motion blur, dispersion, anti-aliasing, ray tree, indirect light, path tracing.









## RECAP

"Le: € (depth ( 192

= = inside / 1 nt = nt / nc, dds os2t = 1.0f = nnt 0, N ); 0)

= diffuse; = true;

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

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

#### AXDEPTH)

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

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

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

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

#### Lecture 11: Accelerate

#### **Concepts:**

Required ray tracing performance, grids / nested grids / octrees / kDtrees (and their (dis)advantages), the bounding volume hierarchy, BVH construction, BVH traversal, BVH size bounds, BVH depth, good BVHs: SAH, construction termination, packet traversal.







## RECAP

tic: K (depth < 100

= inside / 1 it = nt / nc, ddo ss2t = 1.8f - nnt 3, N ); 3)

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

= diffuse = true;

efl + refr)) && (depth is Hout

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

AXDEPTH)

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

v = true; t brdfPdf = EvaluateDiffuse( L, N.) Provident st3 factor = diffuse \* INVPI; st weight = Mis2( directPdf, brdfPdf ); st cosThetaOut = dot( N, L ); E \* ((weight \* cosThetaOut) / directPdf) \* 100

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

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

#### Lecture 12: Post Processing

#### **Concepts:**

Post processing, camera / sensor behavior, lens flares, vignetting, chromatic aberration, noise / grain, HDR bloom and glare, tone mapping / exposure control, color correction / grading, gamma, gamma correction, depth of field, circle of confusion, ambient occlusion, screen space AO, bilateral filtering, screen space reflections, limitations of screen space approaches.









SCHWARZENEGGER

Get ready for the ride of your life.

E(HAP

TOTAL RECAP

**TOTAL RECAP** 

TOTA

### RECAP

tice (depth < 1935

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

= diffuse; = true;

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

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

AXDEPTH)

survive = SurvivalProbability difference estimation - doing it properly if; adiance = SampleLight( &mand, I .x + radiance.y + radiance.r) > 0\_\_\_\_\_\_

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

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

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

## SCHWARZENEGGER SCHWARZENEGGER

Ge

Get ready for the ride of your life.

## What's Next?



- efl + refr)) && (depth
- ), N ); efl \* E \* diffuse; = true;

#### AXDEPTH)

- survive = SurvivalProbability( dif if: adiance = SampleLight( &rand, I. 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)

andom walk - done properly, closely felle vive)

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

### **Upcoming Attractions:**

- One more tutorial: *right after this lecture.*
- Final Exam: Tuesday June 23, 08:30 (EDUC-GAMMA)
- P3 deadline: Tuesday June 30, 23:59
- Retake Exam: Thursday July 9, 13:30 (EDUC-ALFA)

#### Master:

- **Optimization & Vectorization** 
  - **Advanced Graphics**



tic: k (depth < NA

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

st  $a = nt - nc_1 b - nt$ st Tr = 1 - (R0 + (1 - 1))Tr ) R = (0 = nnt - 1)

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

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

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

AXDEPTH)

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

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

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

# **INFOGR – Computer Graphics**

Jacco Bikker - April-July 2015 - Lecture 13: "Grand Recap"

next up: "Final Exam"

