

INFOGR – Computer Graphics

Jacco Bikker - April-July 2016 - Lecture 1: “Introduction”

Welcome!



Today's Agenda:

- Topic Introduction
- Course Introduction
- Team
- Practical Details
- Assignments
- Field Study
- State of the Art



Introduction



Just Cause 3



Introduction



HALO 5



Introduction



GTA V



Introduction

```
ics  
& (depth < MAXDEPTH)  
    t = inside / 2  
    nt = nt / nc  
    pos2t = 1.0f / (D * N);  
    D, N);  
    0)  
    at a = nt - nc  
    at Tr = 1 - (R * Tr)  
    Tr) R = (D * n  
    E * diffuse;  
    = true;  
    (refl + refr)) &  
    D, N);  
    refl * E * dif  
    = true;  
    MAXDEPTH)  
    survive = Surv  
    estimation -  
    df;  
    radiance = Sam  
    e.x + radiance  
    w = true;  
    at brdfPdf = E  
    at3 factor = d  
    at weight = Mi  
    at cosThetaOut  
    E * ((weight  
    random walk - d  
    vive)
```



Homeworld Remastered



Introduction

```

ics
& (depth < MAXDEPTH)
{
    // Inside / Outside
    int nt = nt / nc;
    float r0s2t = 1.0f - nnt;
    float r0, N );
    // ...
    float a = nt - nc, b = nt - nc;
    float Tr = 1 - (R0 + (1 - R0) * r0s2t);
    float R = (D * nnt - N * (1 - Tr));
    // ...
    E * diffuse;
    // ...
    refl + refr)) && (depth < MAXDEPTH)
    {
        // ...
        D, N );
        refl * E * diffuse;
        // ...
        MAXDEPTH)
        survive = SurvivalProbability( diffuse );
        // estimation - doing it properly, closely
        if;
        radiance = SampleLight( &rand, I, &t, &light );
        // ...
        w = true;
        float brdfPdf = EvaluateDiffuse( L, N ) * Psurvive;
        float3 factor = diffuse * INVPI;
        float weight = Mis2( directPdf, brdfPdf );
        float cosThetaOut = dot( N, L );
        E * ((weight * cosThetaOut) / directPdf) * (radiance);
        // random walk - done properly, closely following survival
        // ...
        float3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R );
        survive;
        pdf;
        n = E * brdf * (dot( N, R ) / pdf);
        // ...
    }
}

```

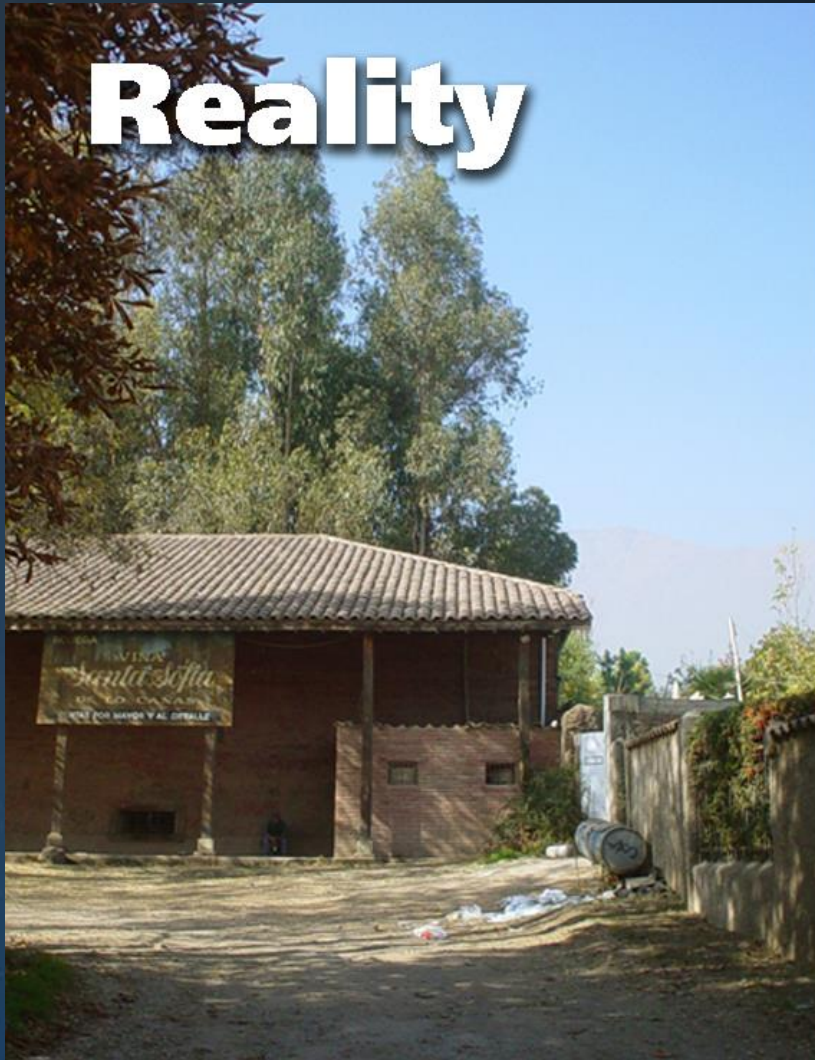


Crysis

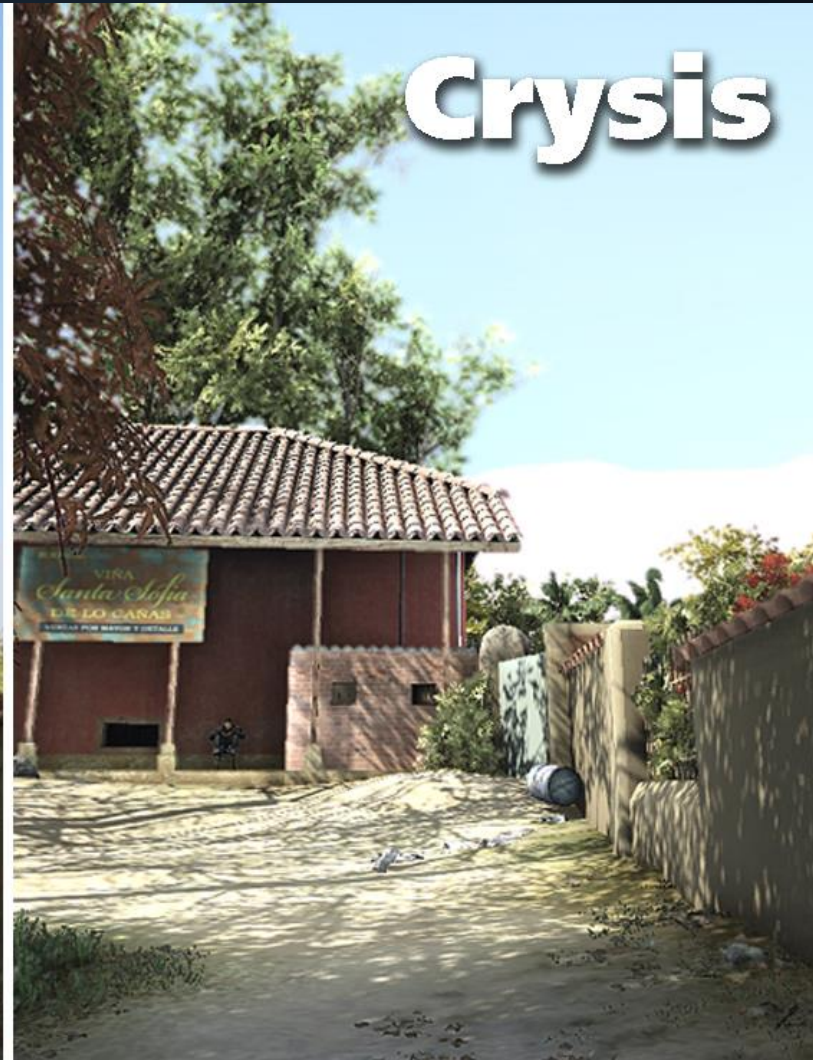


Introduction

Reality



Crysis



Crysis



Introduction

```
...ics
& (depth < MAXDEPTH)
{
    // Inside / Outside
    nt = nt / nc; nnt = nnt / nc;
    pos2t = 1.0f - nnt;
    D, N );
}

// Russian roulette
float a = nt - nc, b = nt / nc;
float Tr = 1 - (RB + (1 - RB) * pos2t);
float R = (D * nnt - N * pos2t) / Tr;

// Russian roulette
E * diffuse;
= true;

// Russian roulette
refl + refr)) && (depth < MAXDEPTH)
{
    D, N );
    refl * E * diffuse;
    = true;

    MAXDEPTH)

survive = SurvivalProbability( diffuse );
estimation - doing it properly, closely following
if;
radiance = SampleLight( &rand, I, &t, &light );
e.x + radiance.y + radiance.z) > 0) && (max(0,
w = true;
at brdfPdf = EvaluateDiffuse( L, N ) * Psurvive;
at3 factor = diffuse * INVPI;
at weight = Mix2( directPdf, brdfPdf );
at cosThetaOut = dot( N, L );
E * ((weight * cosThetaOut) / directPdf) * (radiance
random walk - done properly, closely following
survive)

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



Lord of the Rings



Introduction

```
...ics
& (depth < MAXDEPTH)
{
    // Inside / Outside
    nt = nt / nc; rddo = rddo / nc;
    pos2t = 1.0f - nnt * nnt;
    D, N );
}

// Inside / Outside
at a = nt - nc, b = nt + nc;
at Tr = 1 - (R0 + (1 - R0) * pos2t);
Tr) R = (D * nnt - N * pos2t);

E * diffuse;
= true;

// Inside / Outside
efl + refr)) && (depth < MAXDEPTH)
{
    D, N );
    refl * E * diffuse;
    = true;

MAXDEPTH)

survive = SurvivalProbability( diff
estimation - doing it properly, c
if;
radiance = SampleLight( &rand, 1, &
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

random walk - done properly, closely
vive)

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



Star Wars



Introduction

```
...ics
& (depth < MAXDEPTH)
{
    // Inside / Outside
    int nt = nc; nco = nci;
    pos2t = 1.0f - nnt; nnt = nnt * 0.5f;
    D, N );
}

// Russian roulette
float a = nt - nc, b = nt * nc;
float Tr = 1 - (RB + (1 - RB) * a);
float R = (D * nnt - N * (1 - Tr));

// Diffuse
E * diffuse;
= true;

// Refractive
refl + refr)) && (depth < MAXDEPTH)
{
    D, N );
    refl * E * diffuse;
    = true;

    MAXDEPTH)

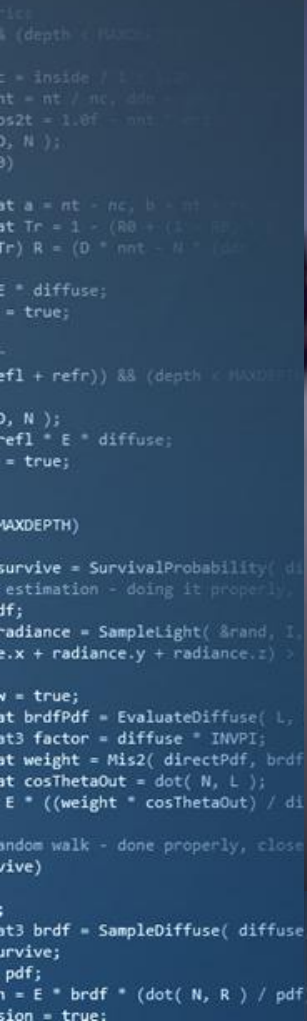
survive = SurvivalProbability( diffuse );
estimation - doing it properly, closely following
if;
radiance = SampleLight( &rand, I, &t, &align,
e.x + radiance.y + radiance.z > 0) && (e.x + radiance.y + radiance.z > 0)
{
    w = true;
    at brdfPdf = EvaluateDiffuse( L, N ) * Psurvive;
    at3 factor = diffuse * INVPI;
    at weight = Mix2( directPdf, brdfPdf );
    at cosThetaOut = dot( N, L );
    E * ((weight * cosThetaOut) / directPdf) * (radiance
    random walk - done properly, closely following
    survive)

    at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &P, &align,
    survive;
    pdf;
    n = E * brdf * (dot( N, R ) / pdf);
    sion = true;
}
```



Zootopia





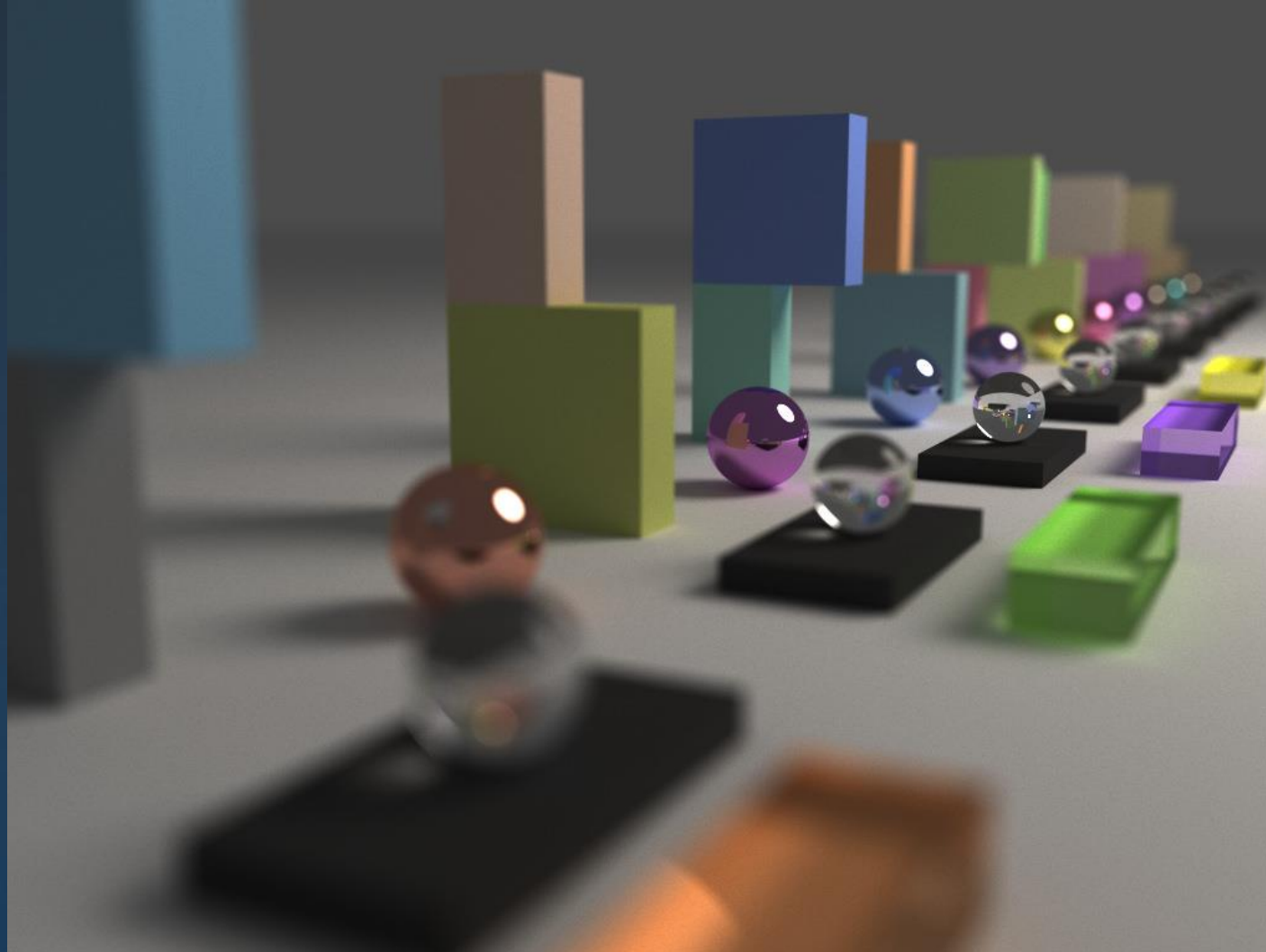
Introduction



Just Cause 3



Introduction

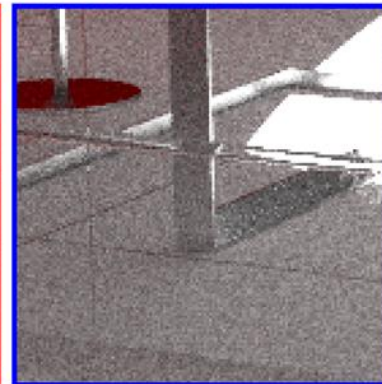
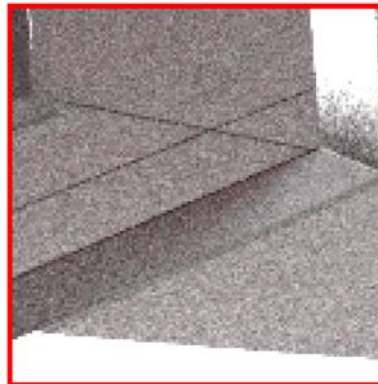
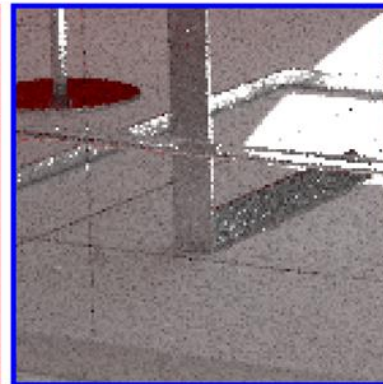
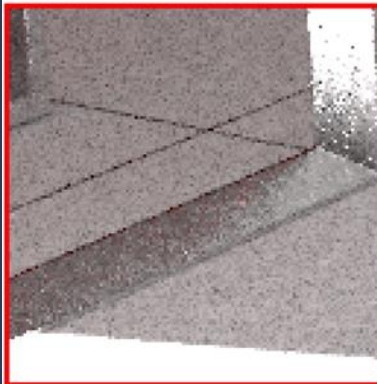




Introduction

```

    if (depth < MAXDEPTH)
    {
        // Inside the glass
        nt = inside / 1.5;
        nc = nt / (nc + 1);
        n2t = nt * nt;
        r2t = 1.0f - n2t;
        D = N;
        R = (D * n2t - N * r2t);
        // Diffuse reflection
        E = diffuse;
        refl = true;
        // Refraction
        ref1 + refr)) && (depth < MAXDEPTH)
    {
        D, N );
        refl * E * diffuse;
        = true;
    }
    MAXDEPTH)
    survive = SurvivalProbability( diffuse );
    estimation - doing it properly, closely following
    if;
    radiance = SampleLight( &rand, I, &L, &align, &
    e.x + radiance.y + radiance.z) > 0) && (rand <
    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
    random walk - done properly, closely following the
    survive)
    at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &
    survive;
    pdf;
    n = E * brdf * (dot( N, R ) / pdf);
    sion = true;
    
```



MCPPM, 180 min.

VCM, 180 min.

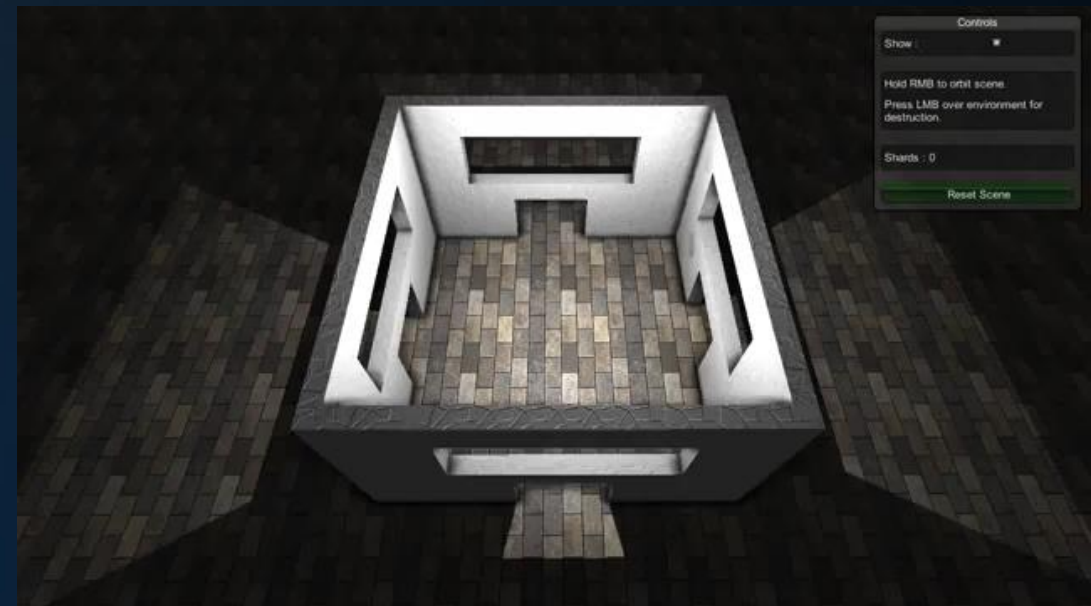
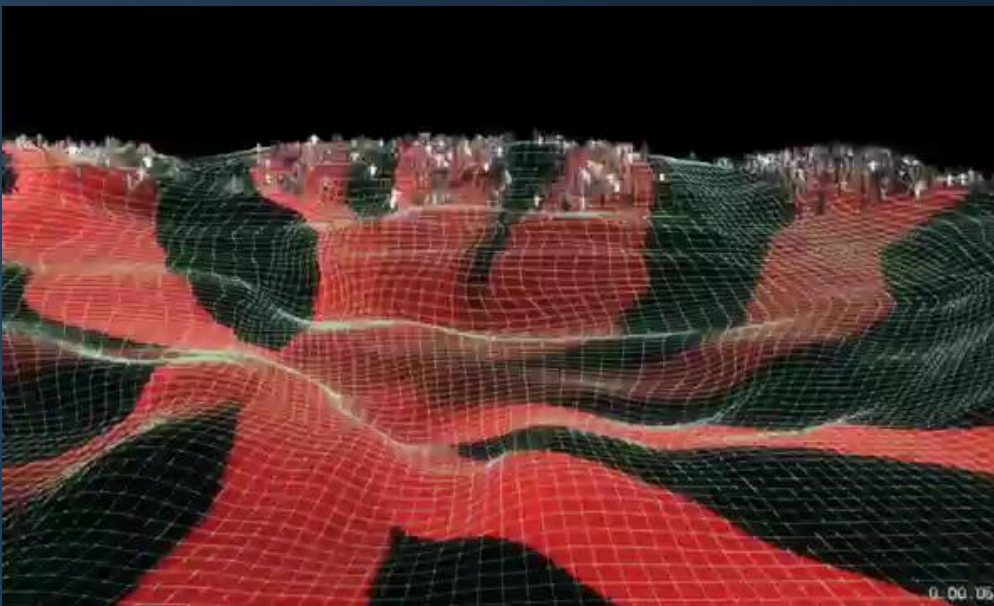


Introduction

```

    (depth < MAXDEPTH)
    {
        int inside = 1;
        int nt = nt / nc;
        double ddn = dot(N, N);
        double cos2t = 1.0f - nnt / ddn;
        double t = (0, N);
        double a = nt - nc;
        double b = nt + nc;
        double Tr = 1 - (R0 + (1 - R0) * cos(a));
        double R = (0 * nnt - N * ddn) / (b - a);
        double E = diffuse;
        double refl = true;
        double refl + refr)) && (depth < MAXDEPTH)
        {
            double N;
            double refl = E * diffuse;
            double refl = true;
            double MAXDEPTH)
            {
                double survive = SurvivalProbability( diffuse );
                double estimation - doing it properly, classically
                if;
                double radiance = SampleLight( &rand, I, &l, &light
                double x + radiance.y + radiance.z) > 0) && (max
                double w = true;
                double brdfPdf = EvaluateDiffuse( L, N ) * Pearc
                double t3 factor = diffuse * INVPI;
                double weight = Mis2( directPdf, brdfPdf );
                double cosThetaOut = dot( N, L );
                double E * ((weight * cosThetaOut) / directPdf) *
                double random walk - done properly, closely followin
                double survive)
                {
                    double t3 brdf = SampleDiffuse( diffuse, N, r1, r2
                    double survive;
                    double pdf;
                    double n = E * brdf * (dot( N, R ) / pdf);
                    double sion = true;

```



Introduction

Computer Graphics 2016:

Looking for realism (in several wrong places):

1. Rasterization

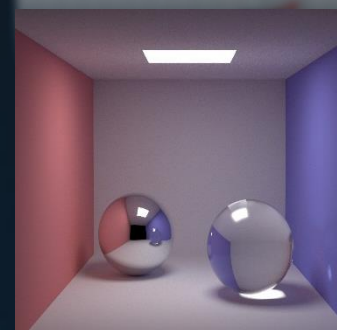
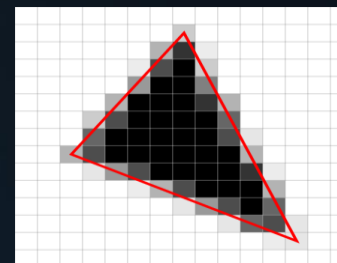
- Geometry
- Textures, shaders
- Clipping, culling
- Post processing
- ...

2. Ray tracing

- Ray/triangle intersections
- Bounding volume hierarchy
- Snell, Fresnel, Beer
- Whitted, Cook, Kajiya
- ...

3. Mathematics

- Vectors
- Matrices
- Transformations



Introduction

Language: Dutch,
because of reasons.

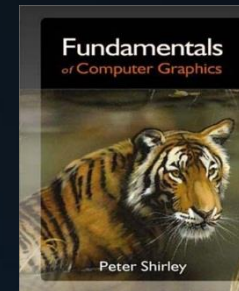
Prerequisites: C#.

Literature: Fundamentals of Computer Graphics (3rd edition), by
Peter Shirley and Steve Marschner (or 4th, or 2nd, or 1st).

15 lectures.

Supporting working colleges in all lecture weeks except the first:

- On Tuesdays,
- In many different rooms – see schedule.



Introduction

Exams:

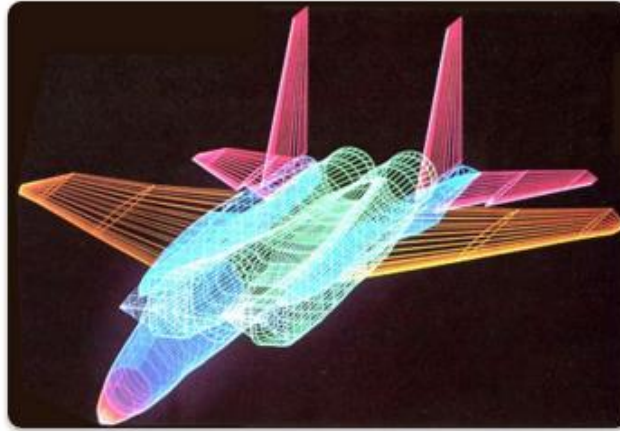
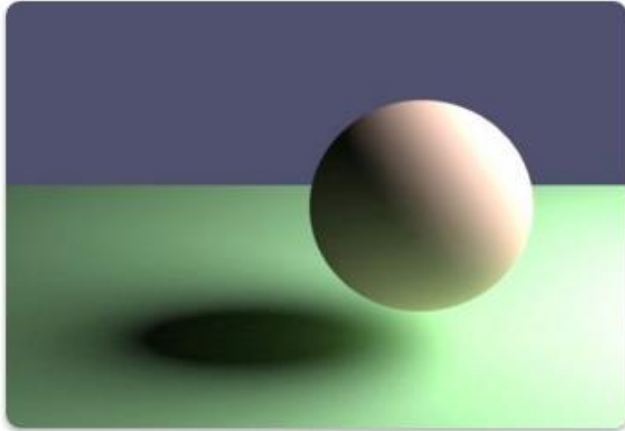
- Mid-term: May 24th.
- End of term: June 30th.
- Retake: July 14th.

Attendance:

You are not required to attend any of the lectures / tutorials / practica (i.e., if you are here, it's because you want to).*

*Obviously, attendance is highly recommended.





<http://www.cs.uu.nl/docs/vakken/gr>

News

Lectures (Topics, Slides)

Tutorials

Exam & Grading

Course Overview

Schedule

Practica

Literature & Links

<https://infogr2016.slack.com/messages/general/details/>

Apps Bookmarks google gmail news tech games misc coding download tools INFOGR # ag | Slack Thesis Tigran

Graphics 2015/2...

jbikker

CHANNELS (2)

general

random

DIRECT MESSAGES (13)

slackbot

aquila169

extrabb

gerbenaalvanger

hugo.hogenbirk

marijns95

mthq

snookik

sp

yorick

+ Invite People

#general

13 members | Company-wide announcements and work-based matters



Search

Send [this link](#) to your team to invite them

April 22nd



jbikker 2:36 PM

joined #general



jbikker 2:48 PM

set the channel purpose: Discuss INFOGR related topics here.



hugo.hogenbirk 3:19 PM

joined #general. Also, @aquila169 joined, @extrabb joined,



mthq 2:45 PM

Kan je je ook aanmelden voor deze slack met een @students.uu.nl adres?



jbikker 5:45 PM

Ik had m ingesteld op uu.nl, hoe voeg ik domains toe?



jbikker 5:56 PM

About #general



Channel Details

Purpose

Discuss INFOGR related topics here.

Created by you on April 22nd

Pinned Items



1/13 Members



jbikker



aquila169



datdutchdude



extrabb

<https://infogr2016.slack.com>

Graphics (INFOGR)

[New Topic](#) 

Search this forum...



0 topics • Page 1 of 1

Announcements

Replies

Views

Last post



General forum rules

by **Arjan Egges** » Tue Mar 01, 2016 3:25 pm » in General

0

1319

by **Arjan Egges** →
Tue Mar 01, 2016 3:25 pm

Display topics from previous:

All Topics ▾

Sort by

Post time ▾

Descending ▾

Go

[New Topic](#) 

0 topics • Page 1 of 1

[Return to Board Index](#)

Jump to ▾

Who is online

Users browsing this forum: 1

<https://www.projects.science.uu.nl/gmt>

Forum permissions

You **cannot** post new topics in this forumYou **cannot** reply to topics in this forumYou **cannot** edit your posts in this forumYou **cannot** delete your posts in this forumYou **cannot** post attachments in this forum

Introduction

Course characteristics:

This is a very intensive course. Be sure to keep up, i.e. don't miss lectures.

Be aware that this course will be attended by a diverse student population:

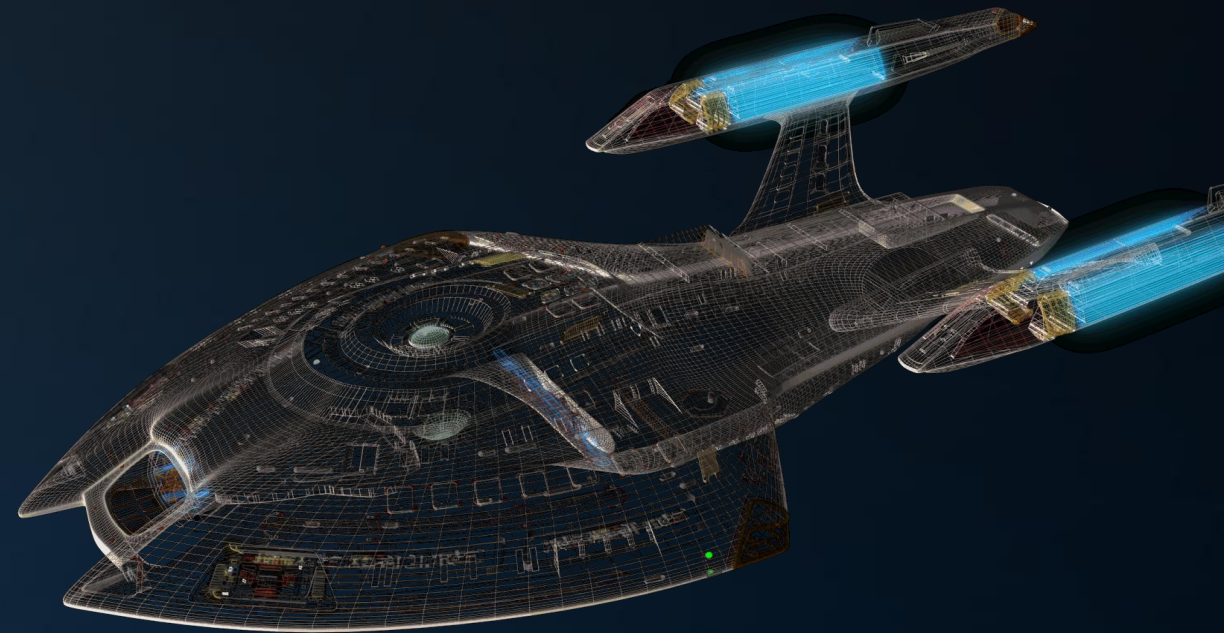
- Math-savvy students;
- Programming gurus;
- Game people;
- Informatics guys.

Regardless of your skill level and interests, make use of this course to improve.



Today's Agenda:

- Topic Introduction
- Course Introduction
- Team
- Practical Details
- Assignments
- Field Study
- State of the Art



Team

Lecturer:

Jacco Bikker

bikker.j@gmail.com / j.bikker@uu.nl

Office: BBL 424



Background:

Gamedev:

- Lost Boys
- Davilex
- Green Dino
- Overloaded
- Vanguard

Academia:

- IGAD

Education:

- HBO
- Doctoral
(Delft; Ray Tracing in Games, 2012)



Team

Student Assistants:

1. Gerben Aalvanger
2. Hugo Hogenbirk
3. Casper Schouls
4. Bruno dos Santos Carvalhal
5. Sam van der Wal



Today's Agenda:

- Topic Introduction
- Course Introduction
- Team
- Practical Details
- Assignments
- Field Study
- State of the Art



Practical Details

Assignment Overview:

- P1: Tutorial;
- P2: Ray Tracing;
- P3: Rasterization.

Final practicum grade is $0.2 * P1 + 0.4 * P2 + 0.4 * P3$.

Exam overview:

- T1: Mid-term exam;
- T2: Final exam.

Final exam grade is $0.5 * T1 + 0.5 * T2$.

Final grade: $(2T + P) / 3$

Passing criteria:

Final Grade ≥ 6.0 (after rounding); both T and P ≥ 5.0 (after rounding).



Practical Details

How to hand in assignments:

- <http://www.cs.uu.nl/docs/submit>

```

    if (depth < MAXDEPTH)
    {
        int nc = inside / L * 4;
        int nt = nt / nc, ddt = 0;
        double r1 = 0, r2 = 0, r3 = 0;
        double cos2t = 1.0f / nnt;
        for (int i = 0; i < nt; i++)
        {
            double r = (D * i + 0.5) / nnt;
            int a = nt - nc, b = nt - a;
            int Tr = 1 - (RR + (1 - RR) * r);
            int R = (D * nnt - N * (ddt + Tr));
            double E = diffuse;
            bool = true;
            // ...
            refl + refr)) && (depth < MAXDEPTH))
            {
                D, N );
                refl * E * diffuse;
                = true;
            }
        }
    }
    MAXDEPTH)

    survive = SurvivalProbability( diffuse );
    // estimation - doing it properly, cleaner
    if;
    radiance = SampleLight( &rand, I, &L, &align, &pdf );
    (e.x + radiance.y + radiance.z) > 0) && (cosThetaOut > 0))
    {
        w = true;
        brdfPdf = EvaluateDiffuse( L, N ) * Psurf;
        at3 factor = diffuse * INVPI;
        at3 weight = Mis2( directPdf, brdfPdf );
        at cosThetaOut = dot( N, L );
        E * ((weight * cosThetaOut) / directPdf) * (radiance
    }

    random walk - done properly, closely following Monte Carlo
    (survive)
};
at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf );
survive;
pdf;
n = E * brdf * (dot( N, R ) / pdf);
tion = true;

```



Practical Details

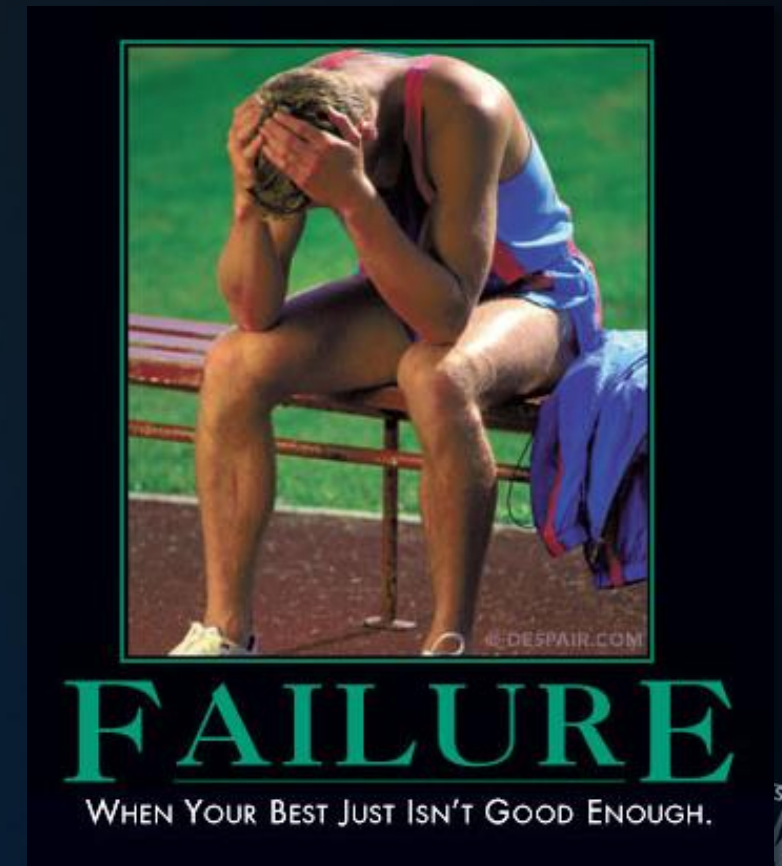
Retake: only if you failed the course, and scored at least a 4.0 (before rounding).

Retake / Theory:

- Retake covers all theory and replaces $\min(T1, T2)$.

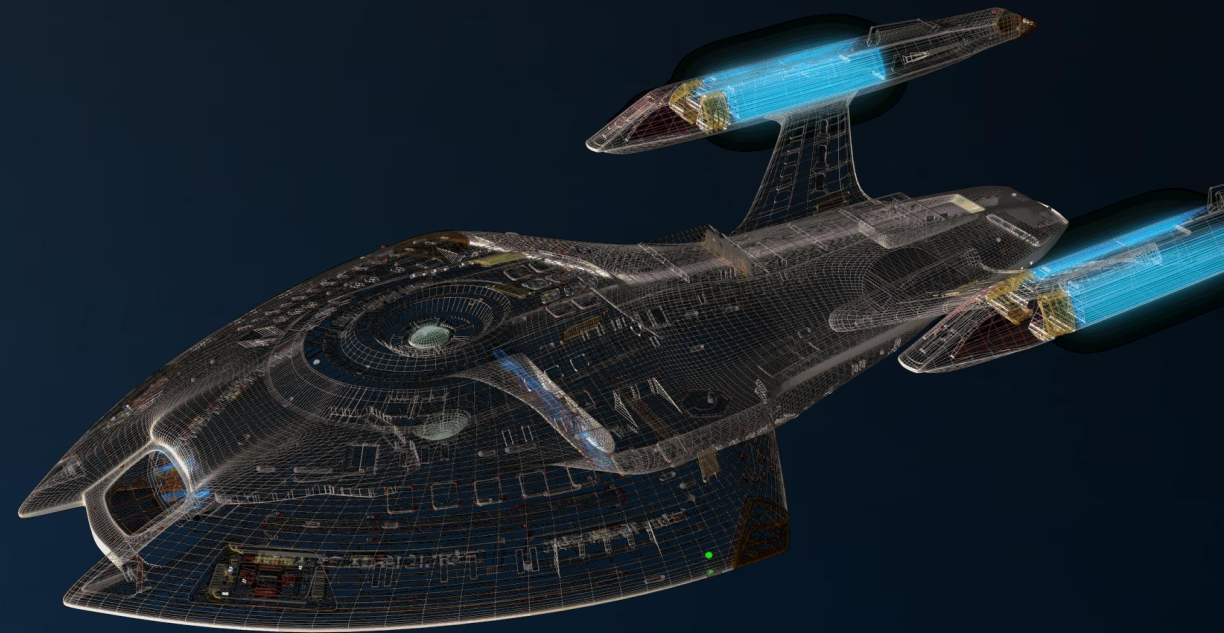
Retake / Practical:

- Retake replaces $\min(P1, P2, P3)$.
Topic will be assigned individually.



Today's Agenda:

- Topic Introduction
- Course Introduction
- Team
- Practical Details
- Assignments
- Field Study
- State of the Art



Assignments

PART 1: Mathematics

Tutorial 1 will be available on Thursday, April 28th.

PART 2: Programming assignment

P1 (OpenTK Tutorial) is now available from the website.

Assistance is available on Tuesday, May 3rd in rooms
BBG-079, -083, -109 and -112.

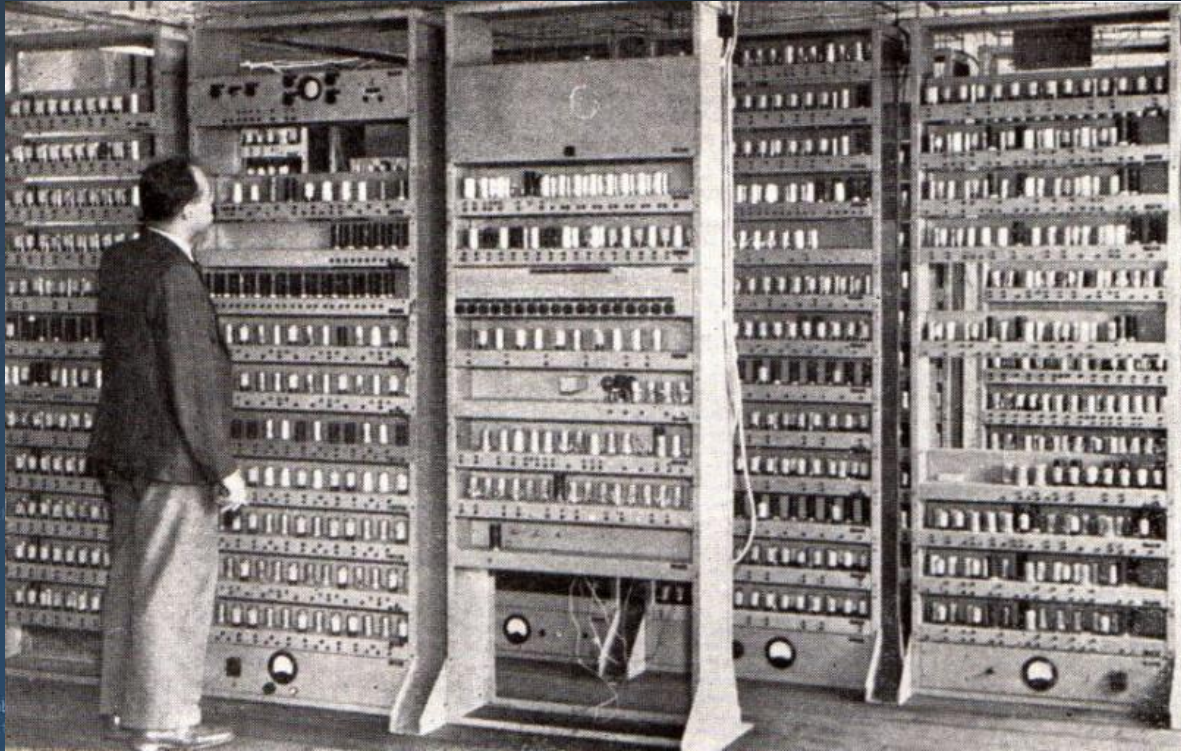


Today's Agenda:

- Topic Introduction
- Course Introduction
- Team
- Practical Details
- Assignments
- Field Study
- State of the Art



Field Study



A. S. Douglas. Noughts and Crosses. EDSAC, 1952.

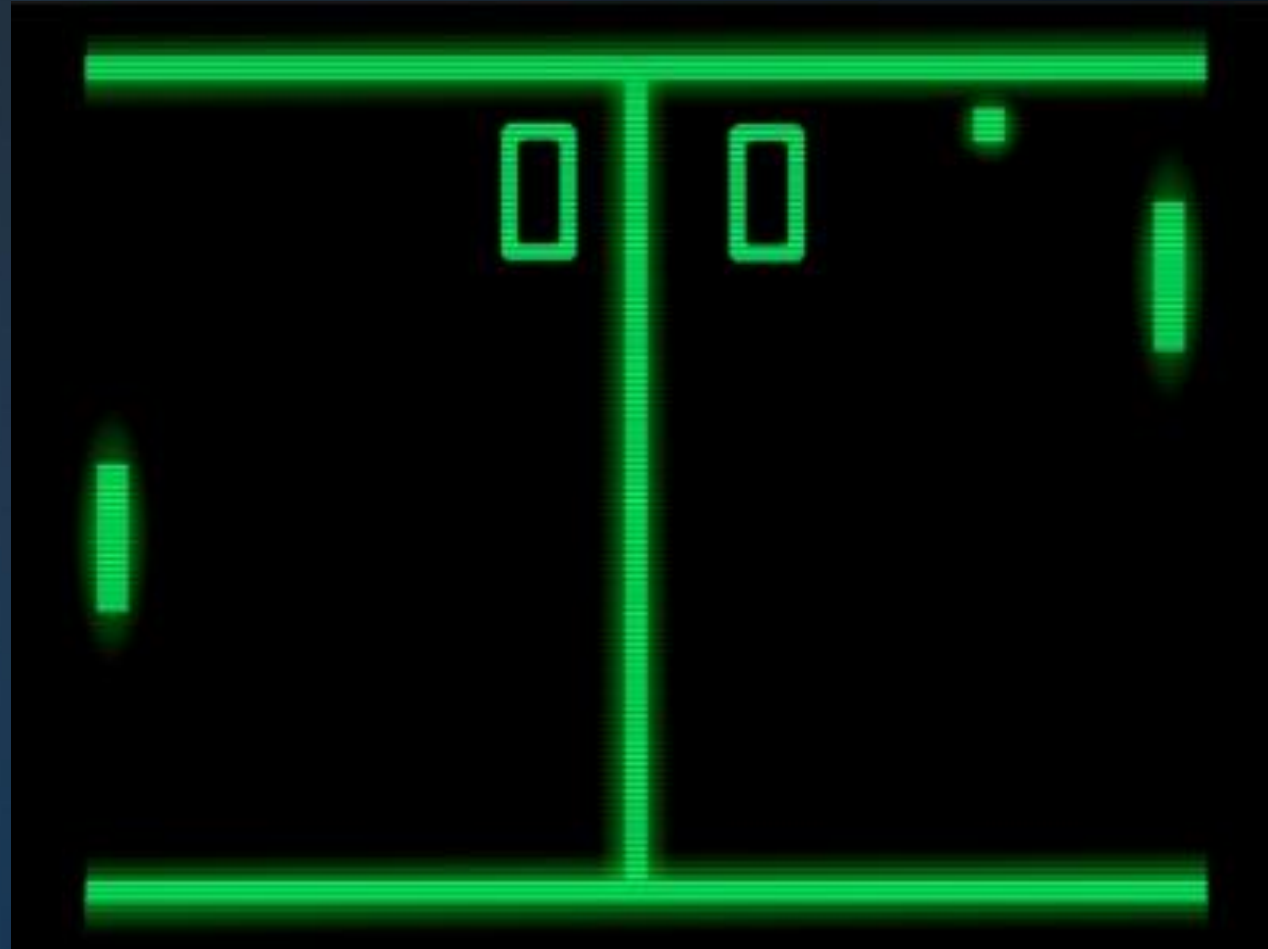


Field Study

```

    if (depth < MAXDEPTH)
    {
        Vec inside = L;
        Vec outside = L;
        Vec nt = nt / nc, nde = nde / nc;
        Vec pos2t = 1.0f - nnt * nnt;
        Vec D, N;
        Vec a = nt - nc, b = nt - nc;
        Vec Tr = 1 - (RB + (1 - RB) * pos2t);
        Vec R = (D * nnt - N * (pos2t));
        Vec E * diffuse;
        Vec refl;
        Vec refl + refr;
        Vec D, N;
        Vec refl * E * diffuse;
        Vec refl;
        Vec MAXDEPTH);
        Vec survive = SurvivalProbability( diffuse );
        Vec estimation - doing it properly, closely following walls (survive);
        Vec radian = SampleLight( &rand, I, &t, &light );
        Vec e.x + radian.y + radian.z > 0) && (e.x + radian.y + radian.z > 0);
        Vec v = true;
        Vec brdfPdf = EvaluateDiffuse( L, N );
        Vec factor = diffuse * INVPI;
        Vec weight = Mis2( directPdf, brdfPdf );
        Vec cosThetaOut = dot( N, L );
        Vec E * ((weight * cosThetaOut) / directPdf) * (radian);
        Vec random walk - done properly, closely following walls (survive);
        Vec survive);
        Vec brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf );
        Vec survive;
        Vec pdf;
        Vec n = E * brdf * (dot( N, R ) / pdf);
        Vec n = true;
    }

```



Field Study



1981



1982



1982



Field Study



1985



1987



1990

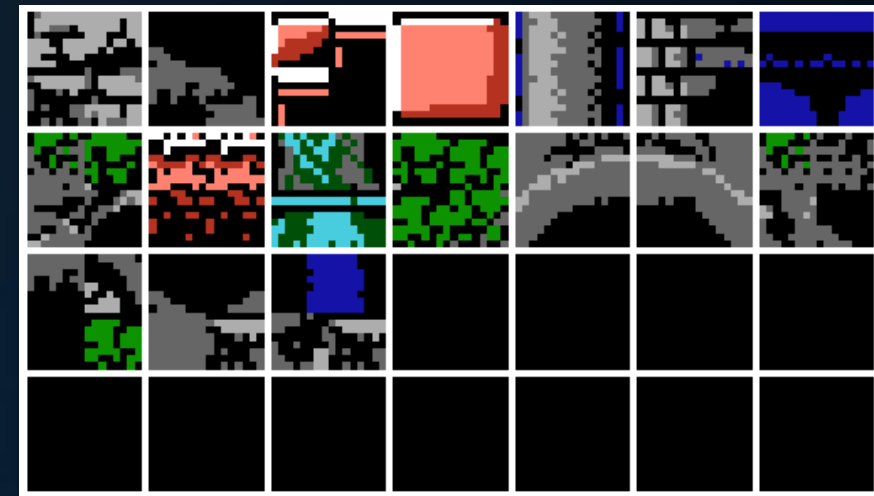
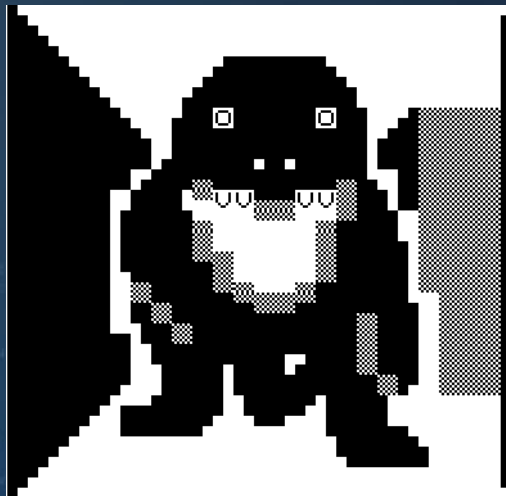


Field Study

Early graphics:

2D, with limitations

- Tiles
- Few colors
- Sprites



Field Study



```
at brdfPdf = EvaluateDiffuse( L, N ) = PdfDiffuse;
```

```
at3 factor = diffuse * INVPI;
```

```
at weight = Mix2( directPdf, brdfPdf );
```

```
at cosThetaOut = dot( N, L );
```

```
E * ((weight * cosThetaOut) / directPdf) * (radiance
```

```
andom walk - done properly, closely following wall (or
```

```
ive)
```

```
at3 brdf = SampleDiffuse( diffuse, N, r1, r2, BR, &pdf );
```

```
urvive;
```

```
pdf;
```

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

```
sion = true;
```







THE FLIP Channel

A500

OPERATIONS
MANUAL

AMIGA

Commodore 1084X
VIDEO MONITOR



TODO:

- BUY SHIP
- FIND BIG WOOD
- DEFEAT LECHUCK
- MARRY ELAINE!

Full Overview, 100%

VIDEO 1084X



50

AMMO

100%

HEALTH

2

3

4

5

6

7

ARMS



2%

ARMOR

BULL
SHEL
ROCK
CELL

50
0
0
0

200
50
50
300

History of Graphics





History of Graphics

```
...ics
& (depth < MAXDEPTH)
{
    nt = inside / 1.5;
    nt = nt / nc; add = 1.0f - nt;
    pos2t = 1.0f - nt;
    D, N );
}

at a = nt - nc; b = nt - nc;
at Tr = 1 - (RB + (1 - RB) *
Tr) R = (D * nt - N * (add

E * diffuse;
= true;

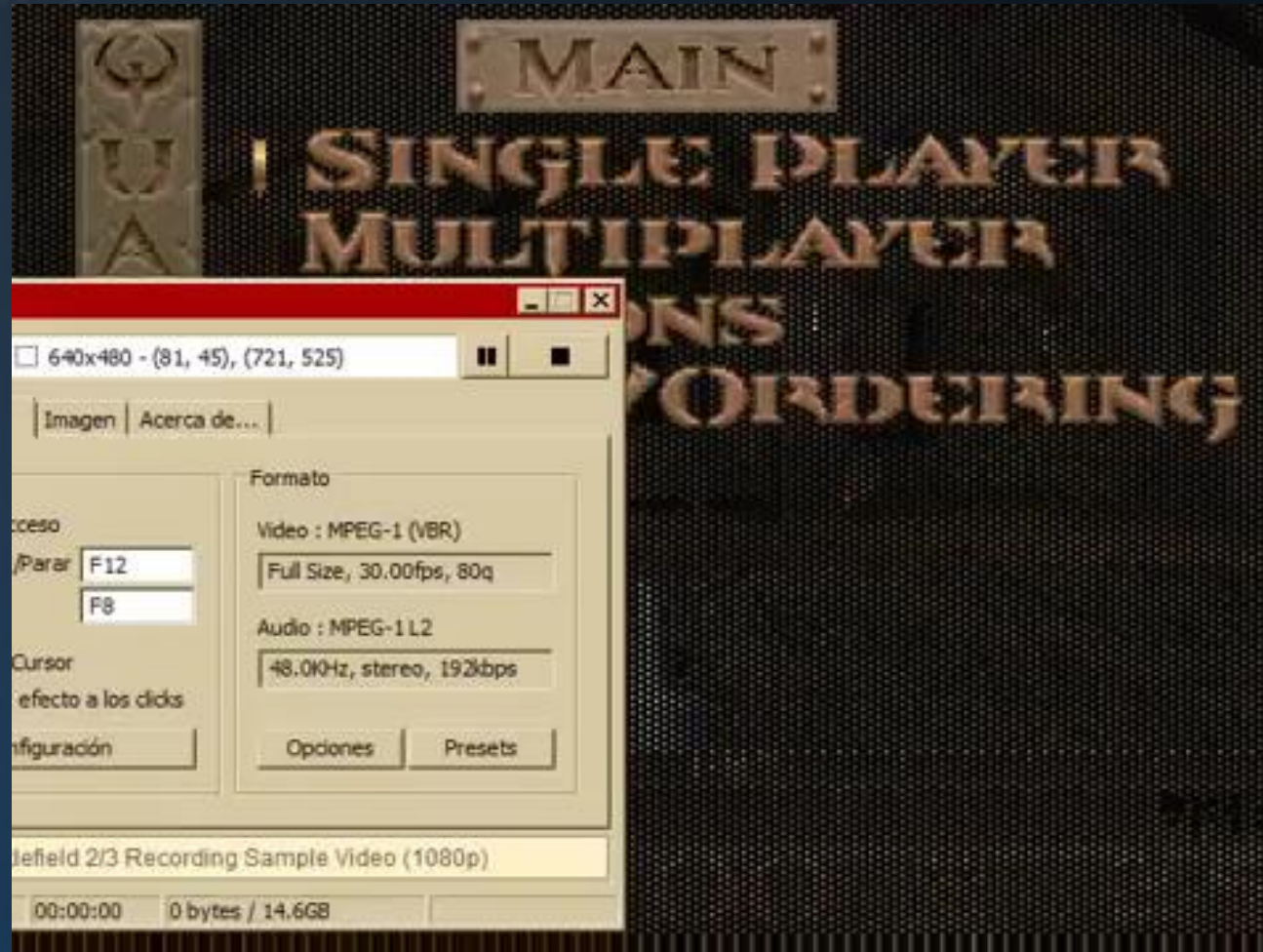
efl + refr)) && (depth < MAXDEPTH)
{
    D, N );
    refl * E * diffuse;
    = true;
}

MAXDEPTH)

survive = SurvivalProbability( diffuse );
estimation - doing it properly, closely following
if;
radiance = SampleLight( &rand, I, &t, &align,
e.x + radiance.y + radiance.z ) > 0) && (depth <
v = true;
at brdfPdf = EvaluateDiffuse( L, N ) * Psurvive;
at3 factor = diffuse * INVPI;
at weight = Mix2( directPdf, brdfPdf );
at cosThetaOut = dot( N, L );
E * ((weight * cosThetaOut) / directPdf) * (radiant

random walk - done properly, closely following
ive)

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















Field Study

Game production:

Code
Art

Crysis:

> 1M lines of code; 85k shaders

Unreal 3 engine:

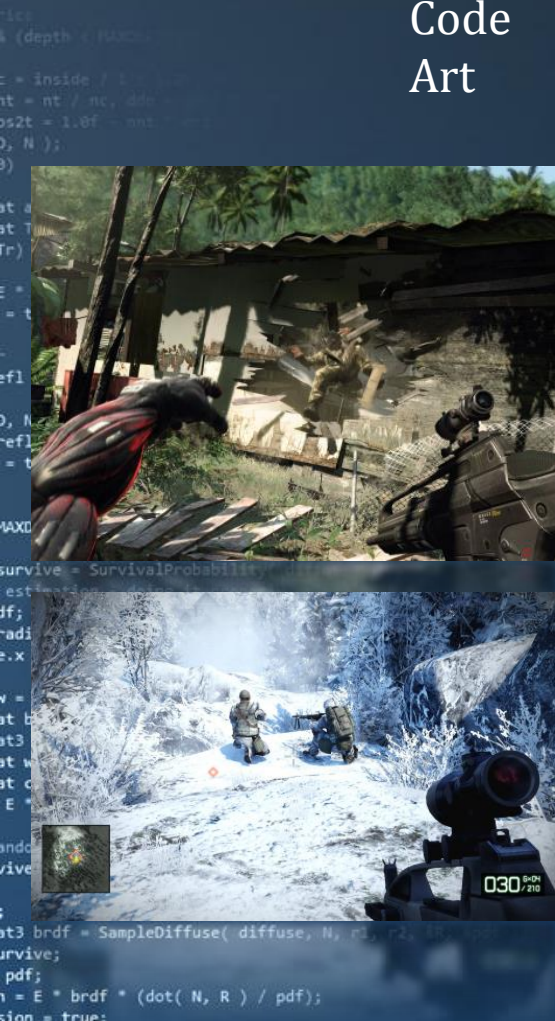
2M lines of code

Frostbite:

“10x Unreal 3”

Minecraft:

< 200k lines of code.



Field Study

History of graphics in games, digest

Initially fast progression:

- from 2D to 3D,
- from monochrome to true-color,
- from wireframe to shaded,
- from sparse to highly detailed.

But also:

- from reasonably efficient to produce to extremely labor-intensive.



State of the Art

Industry example: Unreal Engine 4

- Lights
- Shadows
- Reflections
- Ambient occlusion
- Light shafts
- Indirect lighting cache
- Ray traced soft shadows
- Bump mapping



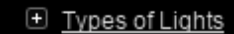
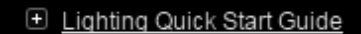
- [-] **Engine Features**
 - [-] **Graphics**
 - [Rendering Overview](#)
 - [-] **Lighting and Shadows**
 - + [Lighting Quick Start Guide](#)
 - + [Types of Lights](#)
 - [Shadow Casting](#)
 - [-] [Light Mobility](#)
 - [Movable Lights](#)
 - [Static Lights](#)
 - [Stationary lights](#)
 - + [Lightmass Global Illumination](#)
 - [Reflection Environment](#)
 - [Ambient Occlusion](#)
 - [Light Shafts](#)
 - [Light Functions](#)
 - [Ambient Cubemaps](#)
 - [Distance Field Ambient Occlusion](#)
 - [IES Light Profiles](#)
 - [Indirect Lighting Cache](#)
 - [Lit Translucency](#)
 - [Ray Traced Distance Field Soft Shadows](#)
 - [Light Propagation Volumes](#)
 - [Bump Mapping w/o Tangent Space](#)
 - + [Materials](#)
 - + [Post Process Effects](#)
 - + [Particle Systems](#)



- [-] Graphics

Rendering Overview

- [-] Lighting and Shadows



Shadow Casting

- Light Mobility

Movable Lights

Static Lights

Stationary lights

- +** Lightmass Global Illumination

Reflection Environment

Ambient Occlusion

Light Shafts

Light Functions

Ambient Cubemaps

Distance Field Ambient Occlusion

IES Light Profiles

Indirect Lighting Cache

Lit Translucency

Ray Traced Distance Field Soft

Shadows

Light Propagation Volumes

Bump Mapping w/o Tangent Space

- #### **Materials**

- ⊕ Post Process Effects

- ## ⊕ Particle Systems



- [-] Graphics

Rendering Overview

- Lighting and Shadows

- [+ Lighting Quick Start Guide](#)

- ### Types of Lights

- ### Shadow Casting

- ☐ Light Mobility

- ### Movable Lights

- ### Static Lights

- ### Stationary lights

- + Lightmass Global Illumination**

- ### Reflection Environment

- ### Ambient Occlusion

- ### Light Shafts

- ### Light Functions

- ## Ambient Cubemaps

- ### Distance Field Ambient Occlusion

- ## IES Light Profiles

- ### Indirect Lighting Cache

- ### Lit Translucency

- ## Ray Traced Distance Field Soft

- ## Shadows

- ### Light Propagation Volumes

- ### Bump Mapping w/o Tangent Space

- ## **+** Materials

- ⊕ Post Process Effects

- #### **Particle Systems**



State of the Art

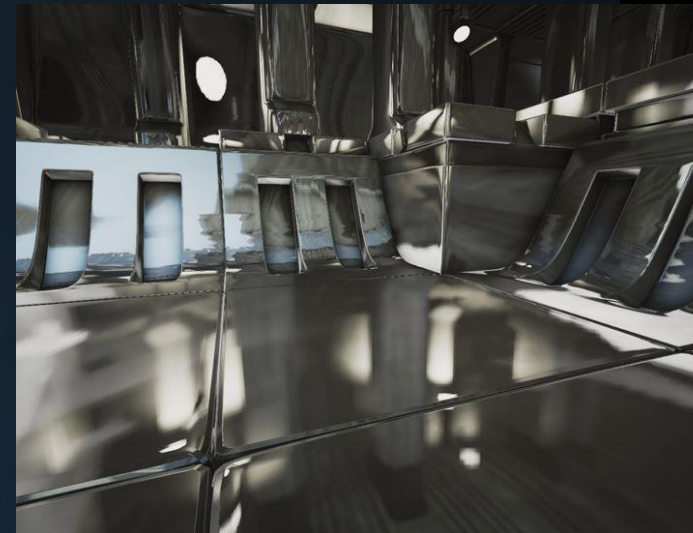
Industry example: Unreal Engine 4

- Lights
- Shadows
- **Reflections**
- Ambient occlusion
- Light shafts
- Indirect lighting cache
- Ray traced soft shadows
- Bump mapping

```

...
    if (depth < MAXDEPTH)
    {
        // Inside / Outside
        float nt = inside / nc;
        float nnt = nt * nt;
        float rns2t = 1.0f - nnt;
        float rns2b = nnt;
        float D, N;
        // ...
        float a = nt - nc, b = nt * nc;
        float Tr = 1 - (R0 + (1 - R0) * rns2t);
        float R = (D * nnt - N * rns2b);
        // ...
        E * diffuse;
        // ...
        refl + refr)) && (depth < MAXDEPTH)
        // ...
        D, N);
        refl * E * diffuse;
        // ...
        MAXDEPTH)
        survive = SurvivalProbability( diffuse );
        estimation - doing it properly, closely
        if;
        radiance = SampleLight( &rand, I, &L, &lightmap );
        e.x + radiance.y + radiance.z) > 0) && (e.x + radiance.y + radiance.z) > 0)
        // ...
        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) * (radiance);
        random walk - done properly, closely following walls
        survive)
        // ...
        at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf );
        survive;
        pdf;
        n = E * brdf * (dot( N, R ) / pdf);
        sion = true;
    }

```



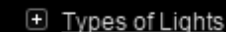
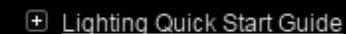
- [-] **Engine Features**
 - [-] **Graphics**
 - [Rendering Overview](#)
 - [-] **Lighting and Shadows**
 - [+] [Lighting Quick Start Guide](#)
 - [+] [Types of Lights](#)
 - [Shadow Casting](#)
 - [-] [Light Mobility](#)
 - [Movable Lights](#)
 - [Static Lights](#)
 - [Stationary lights](#)
 - [+] [Lightmass Global Illumination](#)
 - [Reflection Environment](#)
 - [Ambient Occlusion](#)
 - [Light Shafts](#)
 - [Light Functions](#)
 - [Ambient Cubemaps](#)
 - [Distance Field Ambient Occlusion](#)
 - [IES Light Profiles](#)
 - [Indirect Lighting Cache](#)
 - [Lit Translucency](#)
 - [Ray Traced Distance Field Soft Shadows](#)
 - [Light Propagation Volumes](#)
 - [Bump Mapping w/o Tangent Space](#)
 - [+] [Materials](#)
 - [+] [Post Process Effects](#)
 - [+] [Particle Systems](#)



- [-] Graphics

Rendering Overview

- ## Lighting and Shadows



Shadow Casting

- ☐ Light Mobility

Movable Lights

Static Lights

Stationary lights

☒ Lightmass Global Illumination

Reflection Environment

Ambient Occlusion

Light Shafts

Light Functions

Ambient Cubemaps

Distance Field Ambient Occlusion

IES Light Profiles

Indirect Lighting Cache

Lit Translucency

Ray Traced Distance Field Soft

Shadows

Light Propagation Volumes

Bump Mapping w/o Tangent Space

+ Materials

- ⊕ Post Process Effects

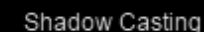
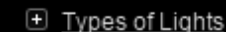
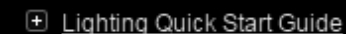
⊕ Particle Systems



- [-] Graphics

Rendering Overview

- ## Lighting and Shadows



- Light Mobility

Movable Lights

Static Lights

Stationary lights

☒ Lightmass Global Illumination

Reflection Environment

Ambient Occlusion

Light Shafts

Light Functions

Ambient Cubemaps

Distance Field Ambient Occlusion

IES Light Profiles

Indirect Lighting Cache

Lit Translucency

Ray Traced Distance Field Soft

Shadows

Light Propagation Volumes

Bump Mapping w/o Tangent Space

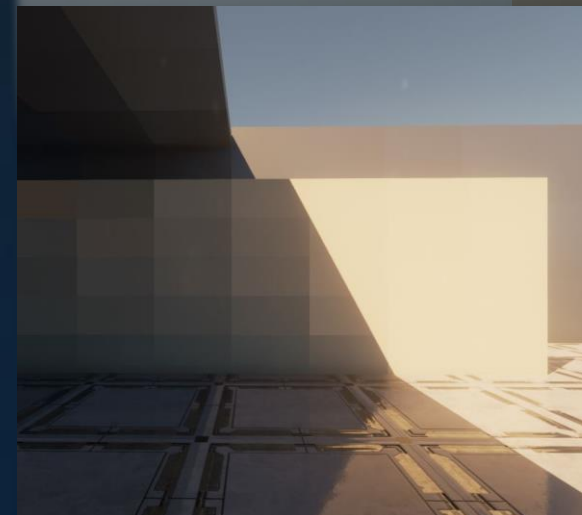
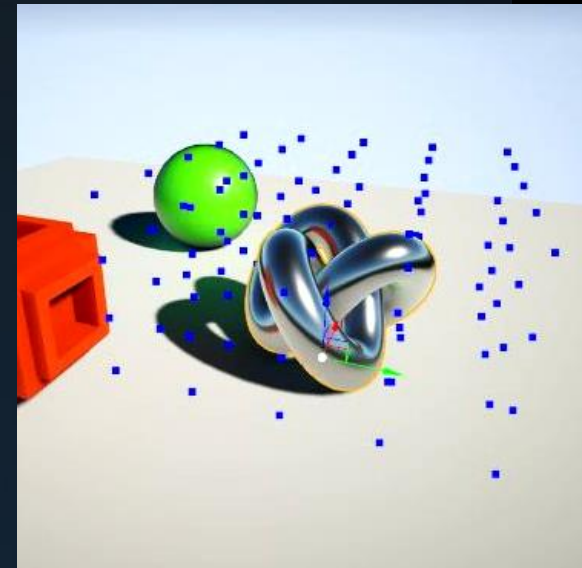
Materials

- ⊕ Post Process Effects

+ Particle Systems



- Lights
- Shadows
- Reflections
- Ambient occlusion
- Light shafts
- **Indirect lighting cache**
- Ray traced soft shadows
- Bump mapping



State of the Art

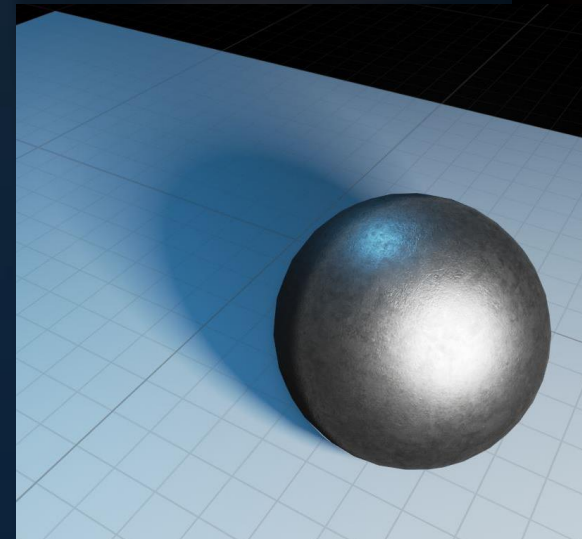
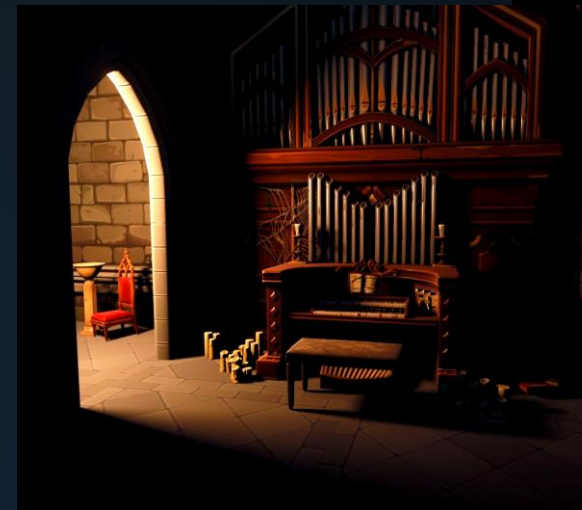
Industry example: Unreal Engine 4

- Lights
- Shadows
- Reflections
- Ambient occlusion
- Light shafts
- Indirect lighting cache
- **Ray traced soft shadows**
- Bump mapping

```

    if (depth < MAXDEPTH)
    {
        // Inside / Outside
        bool inside = (dot(N, L) < 0);
        float nt = nt / nc;
        float n2t = 1.0f - nt;
        float n2r = 1.0f - n2t;
        float r = (inside ? -1.0f : 1.0f);
        float D = D * n2r;
        float N = N * n2t;
        float L = L * n2t;
        float a = nt - nc;
        float b = nt + nc;
        float Tr = 1 - (R0 + (1 - R0) * a * b);
        float R = (D * nnt - N * (1 - D));
        float E = diffuse;
        bool refl = true;
        if (refl + refr) && (depth < MAXDEPTH)
        {
            // Refraction
            float D = D * n2r;
            float N = N * n2t;
            float L = L * n2t;
            float a = nt - nc;
            float b = nt + nc;
            float Tr = 1 - (R0 + (1 - R0) * a * b);
            float R = (D * nnt - N * (1 - D));
            float E = diffuse;
            bool refl = true;
        }
        if (depth < MAXDEPTH)
        {
            // Survival Probability
            float survive = SurvivalProbability( diffuse );
            // Estimation - doing it properly, closely following walls
            if (survive)
            {
                // Radiance
                float radiance = SampleLight( &rand, I, L, &light );
                float x = radiance.x + radiance.y + radiance.z;
                if (x > 0) && (depth < MAXDEPTH)
                {
                    // New
                    bool w = true;
                    float brdfPdf = EvaluateDiffuse( L, N ) * Psurvive;
                    float factor = diffuse * INVPI;
                    float weight = Mis2( directPdf, brdfPdf );
                    float cosThetaOut = dot( N, L );
                    float E = ((weight * cosThetaOut) / directPdf) * (radiance.x + radiance.y + radiance.z);
                    // Random walk - done properly, closely following walls
                    if (survive)
                    {
                        // Sample Diffuse
                        float brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf );
                        float survive = SurvivalProbability( diffuse );
                        float pdf;
                        float n = E * brdf * (dot( N, R ) / pdf);
                        bool refl = true;
                    }
                }
            }
        }
    }

```



Engine Features

Graphics

Rendering Overview

Lighting and Shadows

Lighting Quick Start Guide

Types of Lights

Shadow Casting

Light Mobility

Movable Lights

Static Lights

Stationary lights

Lightmass Global Illumination

Reflection Environment

Ambient Occlusion

Light Shafts

Light Functions

Ambient Cubemaps

Distance Field Ambient Occlusion

IES Light Profiles

Indirect Lighting Cache

Lit Translucency

Ray Traced Distance Field Soft Shadows

Light Propagation Volumes

Bump Mapping w/o Tangent Space

Materials

Post Process Effects

Particle Systems

- Lights
- Shadows
- Reflections
- Ambient occlusion
- Light shafts
- Indirect lighting cache
- Ray traced soft shadows
- **Bump mapping**

- Lights
- Shadows
- Reflections
- Ambient occlusion
- Light shafts
- Indirect lighting cache
- Ray traced soft shadows
- **Bump mapping**



State of the Art

Modern rendering in games:

Stacking algorithms that solve part of the problem:

Shadows

Reflections

Participating media

Indirect light

Designed to ‘look good’, not to be (necessarily) correct

Each partial solution comes with parameters and limitations

But: well-suited for today’s hardware.



Next week:

Foundation



INFOGR – Computer Graphics

Jacco Bikker - April-July 2016 - Lecture 1: “Introduction”

END of “Introduction”

next lecture: “Graphics Fundamentals”

