

# INFOGR – Computer Graphics

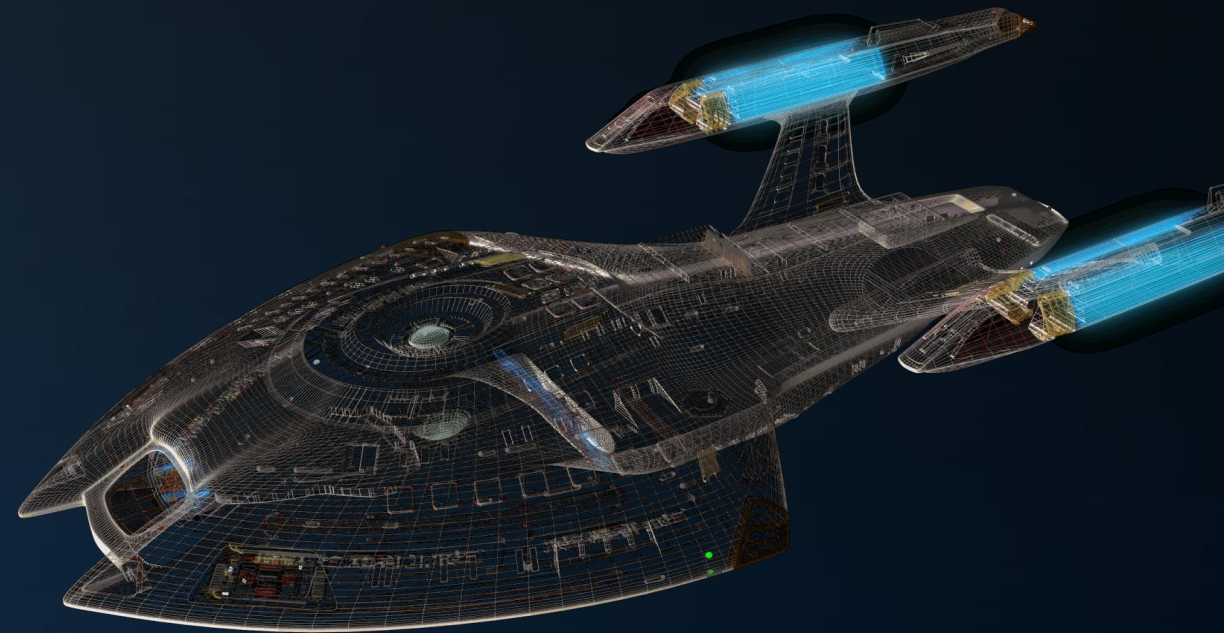
Jacco Bikker - April-July 2015 - Lecture 1: “Introduction”

# Welcome!



# Today's Agenda:

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



# Introduction

```
ric  
& (depth < MAX  
t = inside / 1  
nt = nt / nc  
os2t = 1.0f  
D, N );  
)  
at a = nt - nc  
at Tr = 1 - (R  
Tr) R = (D * n  
E * diffuse;  
= true;  
efl + 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  
ive)
```

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





# Introduction





# Introduction

```
...ics  
& (depth < MAXDEPTH)  
...  
t = inside / 1  
nt = nt / nc  
os2t = 1.0f  
, N );  
)  
...  
at a = nt - nc  
at Tr = 1 - (R  
Tr) R = (D * n  
...  
E * diffuse;  
= true;  
...  
efl + refr)) &  
...  
, 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  
...  
andom walk - d  
ive)
```

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





# Introduction

```
...ics  
& (depth < MAXDEPTH) {  
    if (inside / 2 < 1) {  
        nt = nt / nc;  
        pos2t = 1.0f - nt;  
        D, N );  
    }  
    at a = nt - nc;  
    at Tr = 1 - (R * a);  
    R = (D * a + Tr * a);  
    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)
```



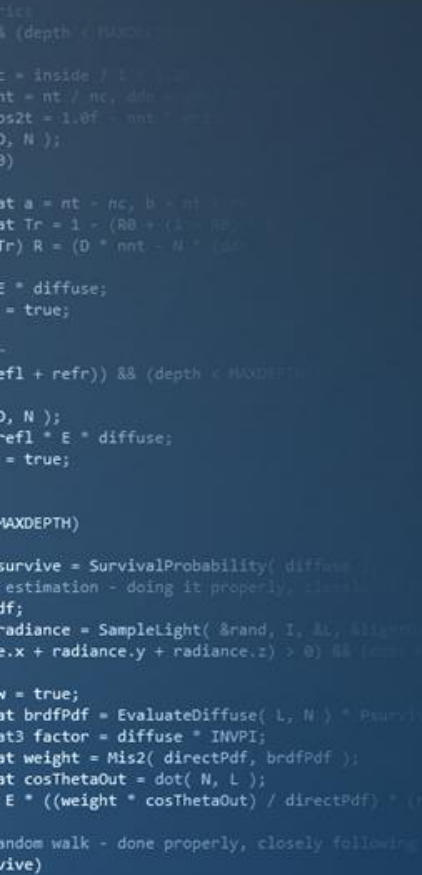


# Introduction

```
...ics  
& (depth < MAXDEPTH) {  
    ...  
    t = inside / 1.5; ...  
    nt = nt / nc; ...  
    os2t = 1.0f - nnt; ...  
    D, N );  
    ...  
    at a = nt - nc, b = nt; ...  
    at Tr = 1 - (RB + (1 - RB) * ...  
    Tr) R = (D * nnt - N * ...  
    ...  
    E * diffuse;  
    = true;  
    ...  
    ...  
    refl + refr)) && (depth < MAXDEPTH) {  
    ...  
    D, N );  
    refl * E * diffuse;  
    = true;  
    ...  
    MAXDEPTH)  
    survive = SurvivalProbability( diffuse, ...  
    estimation - doing it properly, closely ...  
    if;  
    radiance = SampleLight( &rand, I, &t, &light ...  
    e.x + radiance.y + radiance.z) > 0) && (depth < ...  
    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) * (radia ...  
    random walk - done properly, closely following ...  
    ve)  
    ...  
    at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, ...  
    survive;  
    pdf;  
    n = E * brdf * (dot( N, R ) / pdf);  
    sion = true;
```









# Introduction

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

// ...
at a = nt - nc, b = nt - nc;
at Tr = 1 - (RB + (1 - RB) * ...
Tr) R = (D * nnt - N * (d ...

// ...
E * diffuse;
= true;

// ...
efl + refr)) && (depth < MAXDEPTH)
{
    D, N );
    efl * 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 ...

// ...
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 wall
vive)

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



# Introduction

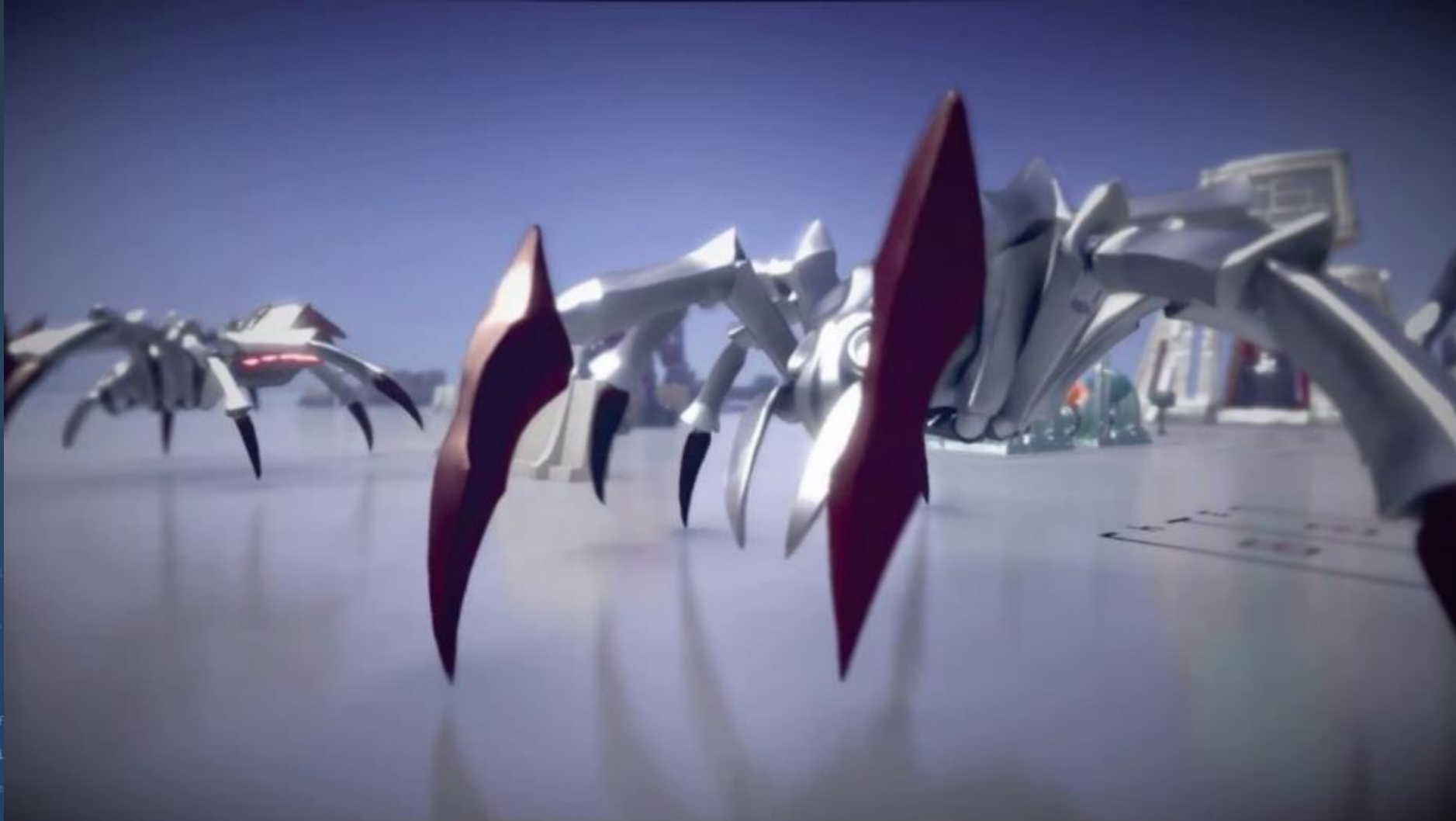




# Introduction



# Introduction

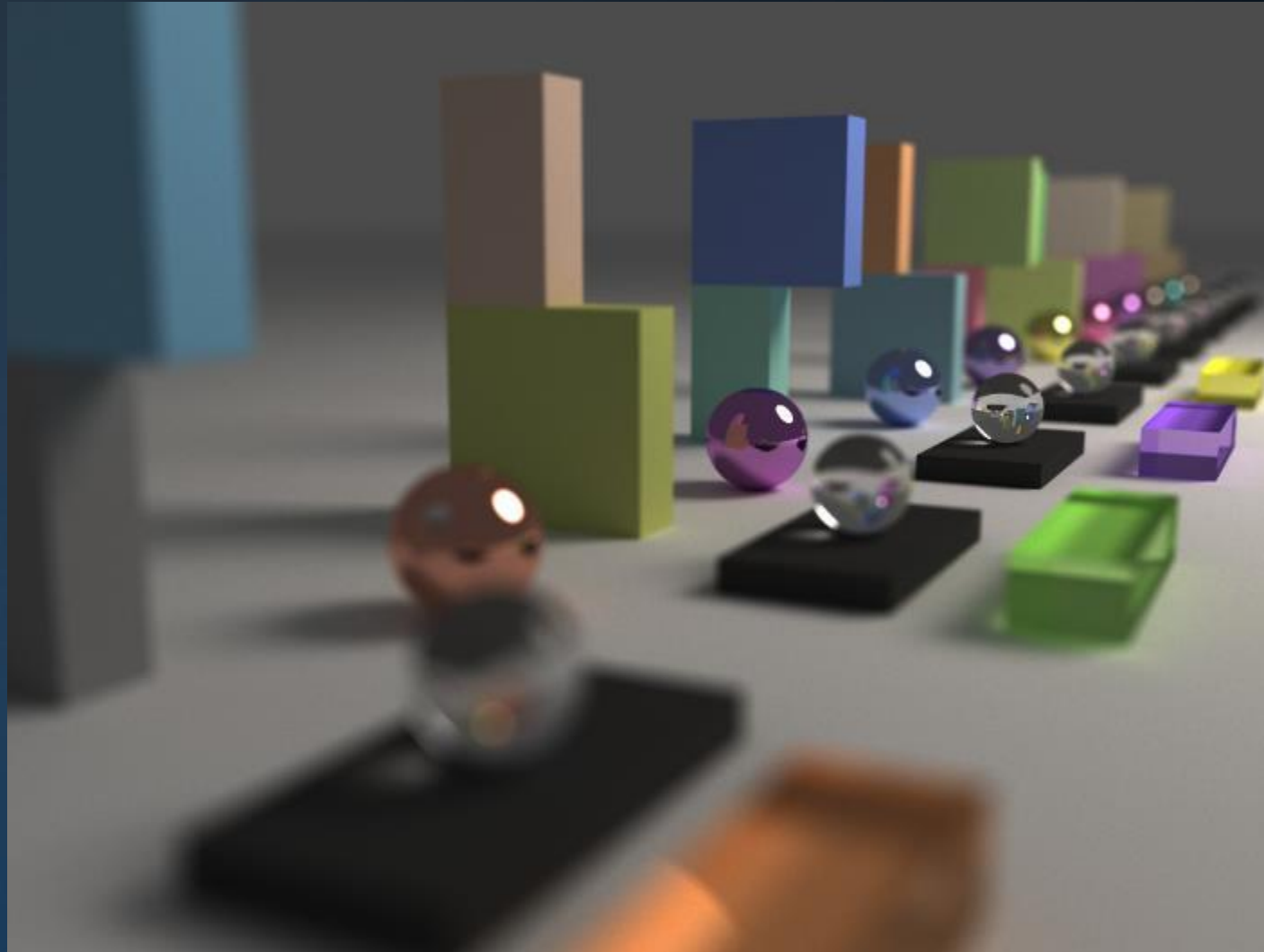




# Introduction



# Introduction





# Introduction

```

    if (depth < MAXDEPTH)
    {
        // Inside / Outside
        int nt = nt / nc, nde = nde / nc;
        double r2t = 1.0f - nnt * nde;
        double D, N );
    }

    // at a = nt - nc, b = nt - nc;
    // at Tr = 1 - (RB + (1 - RB) * r2t);
    // Tr) R = (D * nnt - N * nde);

    // E * diffuse;
    // = true;

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

    // MAXDEPTH)

    survive = SurvivalProbability( diffuse );
    // estimation - doing it properly, closely following wall
    if;
    radiance = SampleLight( &rand, I, &L, &light );
    // e.x + radiance.y + radiance.z) > 0) && (depth < MAXDEPTH)

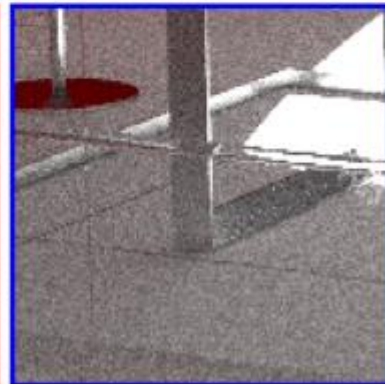
    // w = true;
    // at brdfPdf = EvaluateDiffuse( L, N ) * Psurvive;
    // at3 factor = diffuse * INVPI;
    // at weight = Mix2( directPdf, brdfPdf );
    // at cosThetaOut = dot( N, L );
    // E * ((weight * cosThetaOut) / directPdf) * (radiance);

    // random walk - done properly, closely following wall
    // survive)

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

```





VCM, 180 min.



# Introduction

```

    (depth < MAXDEPTH)
    {
        inside = inside + 1;
        nt = nt / nc; ddx = sqrt(1.0f - nt);
        cos2t = 1.0f - nnt; u = rand();
        N = N;
    }

    at a = nt - nc, b = nt + nc;
    at Tr = 1 - (R0 + (1 - R0) * u);
    Tr) R = (D * nnt - N * (ddx *
    E * diffuse;
    = true;

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

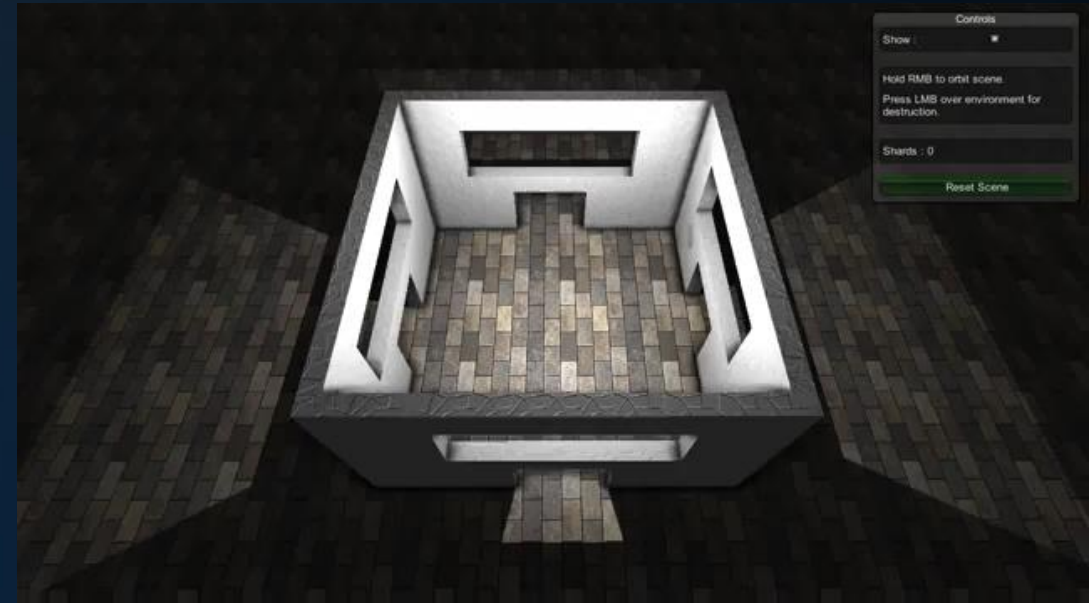
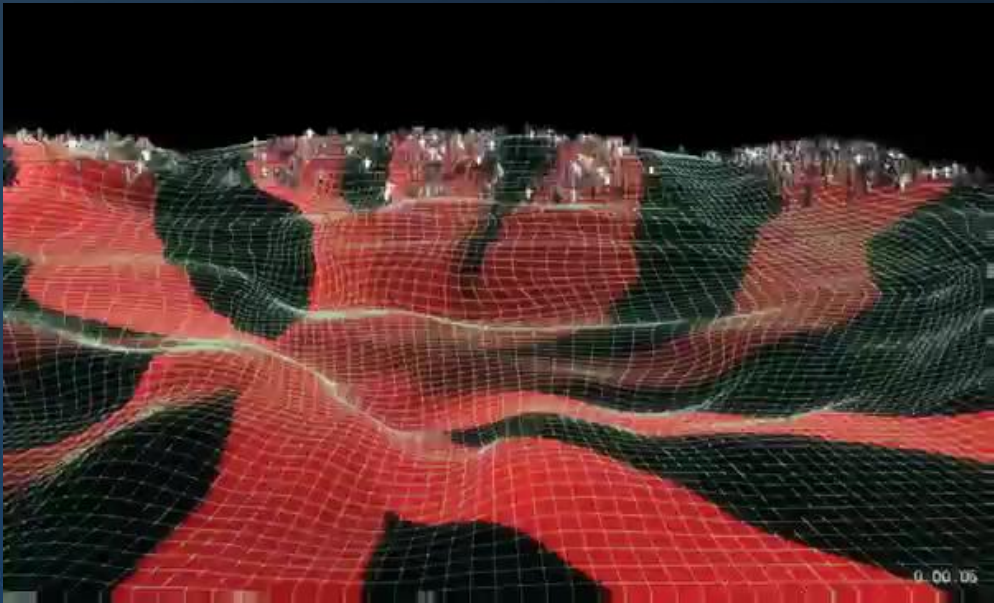
    MAXDEPTH)

    survive = SurvivalProbability( diffuse,
    estimation - doing it properly, classically
    df;

    radiance = SampleLight( &rand, I, &I, &light
    .x + radiance.y + radiance.z) > 0) && (max
    w = true;
    at brdfPdf = EvaluateDiffuse( L, N ) * Pdf;
    at3 factor = diffuse * INVPI;
    at weight = Mis2( directPdf, brdfPdf );
    at cosThetaOut = dot( N, L );
    E * ((weight * cosThetaOut) / directPdf) *
    random walk - done properly, closely following
    ve)

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

```



# Introduction

Computer Graphics 2015:

Looking for realism (in several wrong places):

## 1. Rasterization

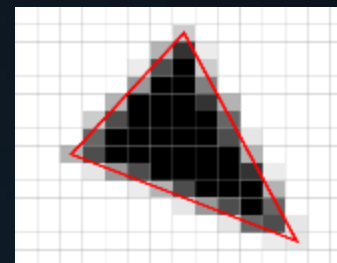
- Geometry
- Textures, shaders
- Clipping, culling
- Post processing
- ...

## 2. Ray tracing

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

## 3. Mathematics

- Vectors
- Matrices
- Transformations





# Introduction

Language: English,  
because of reasons.

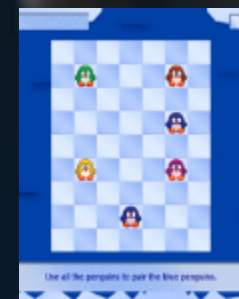
Prerequisites: C#.

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

13 lectures (due to Liberation Day, Ascension Day and retakes).

Supporting practica in all lecture weeks:

- On Tuesdays,
- In BBG-112, -175, -106, -109, -103



# Introduction

Supporting tutorials in all lecture weeks:

- On Thursdays
- In BBG-083, -169, -165 and -079.

Exams:

- Mid-term: May 21<sup>st</sup>.
- End of term: June 23<sup>rd</sup>.
- Retake: July 9<sup>th</sup>.

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.

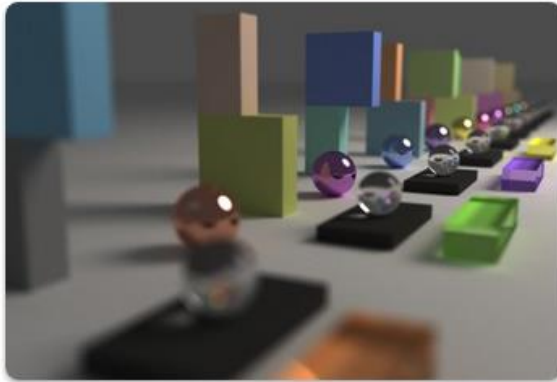




# Graphics

UNIVERSITEIT UTRECHT - INFORMATION AND COMPUTING SCIENCES

academic year 2014/15 – 4th period



## Navigation

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

Course Overview

Schedule

Practicals

Literature & Links

## News

# 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.

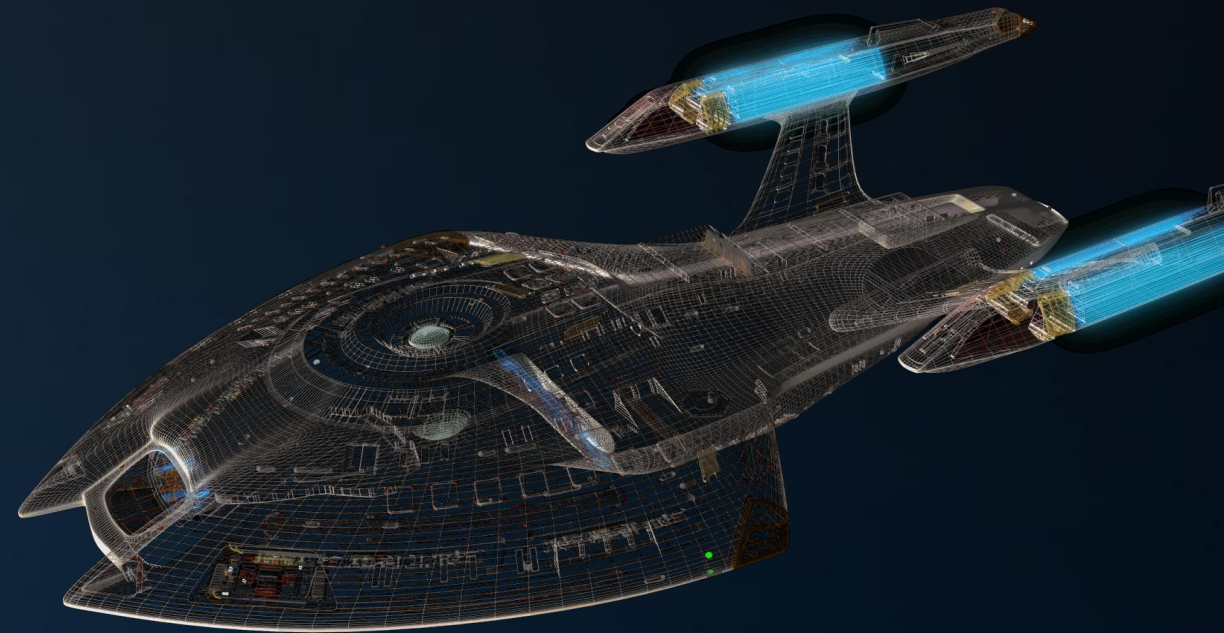
Regardless of your skill level and interests, make use of this course to improve.





# 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](mailto:bikker.j@gmail.com) / [j.bikker@uu.nl](mailto:j.bikker@uu.nl)

Office: BBL 425



Background:

Gamedev:

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

Academia:

- IGAD

Education:

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





## Team

### Teaching Assistants:

1. Forough Madehkhaksar
2. Coert van Gemeren
3. Anna Aljanaki



## Team

### Student Assistants:

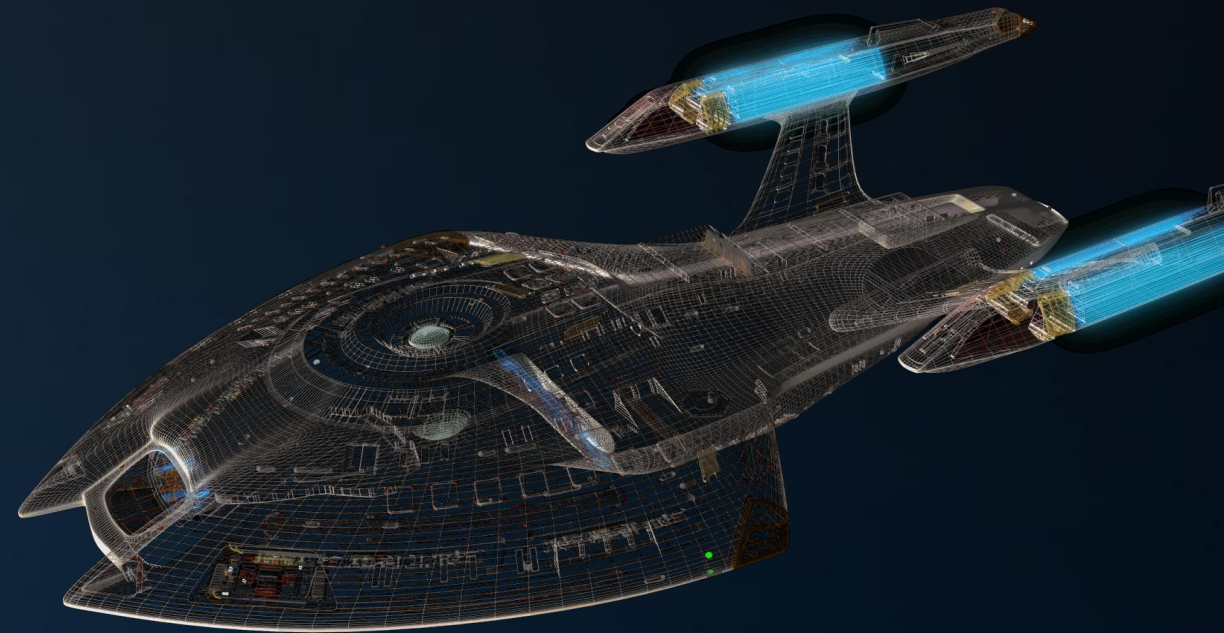
1. Tigran Gasparian
2. Jordi Vermeulen
3. Casper Schouls
4. Sander Vanheste
5. Jan Posthoorn





# Today's Agenda:

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



# Practical Details

## Assignment Overview:

- P1: Tutorial;
- P2: Basic shader programming;
- P3a: Advanced shader programming, or;
- P3b: Ray Tracing.

Final practicum grade is  $0.2 * P1 + 0.4 * P2 + 0.4 * \max(P3a, P3b)$ .

## 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  $\geq 6.0$  (after rounding); both T and P  $\geq 5.0$  (after rounding).





# Practical Details

How to hand in assignments:

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

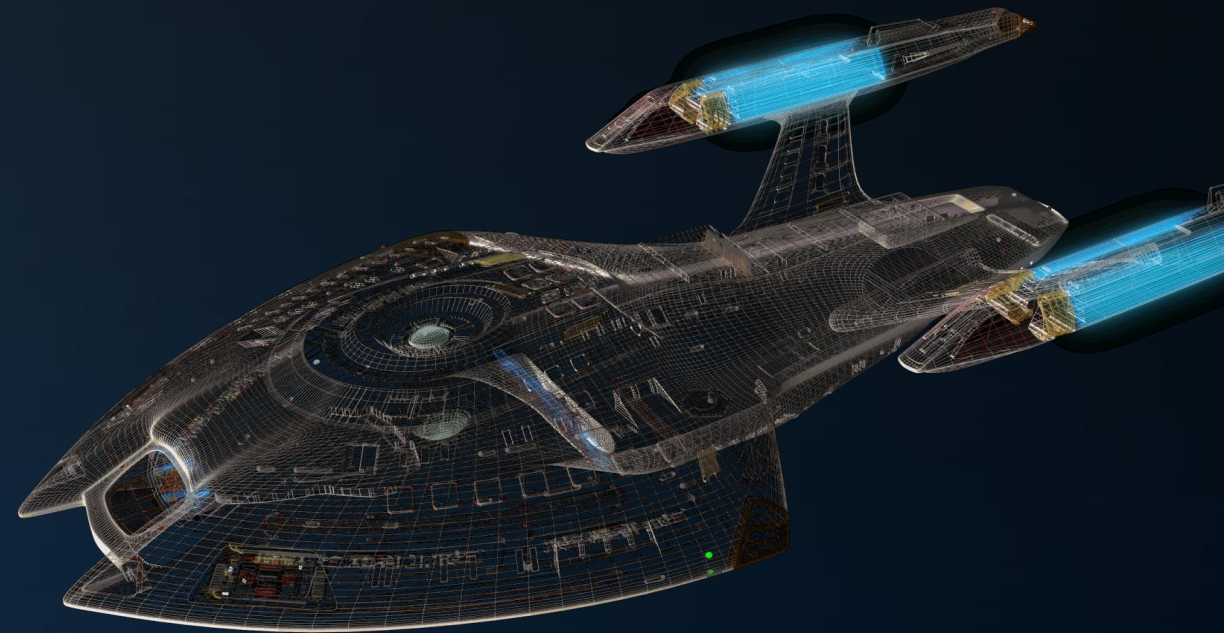
Retake:

- You must have submitted all programming assignments
- You must have participated in both exams
- Your total grade must be at least a 4.0 (after rounding)
- Retake covers whole course, and replaces  $\min(T1, T2)$ .



# 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 23<sup>th</sup>.

TA assistance is available on April 30<sup>th</sup> in rooms

BBG-083, -169, -165 and -079.

## PART 2: Programming assignment

P1 (XNA tutorial) is now available from the website.

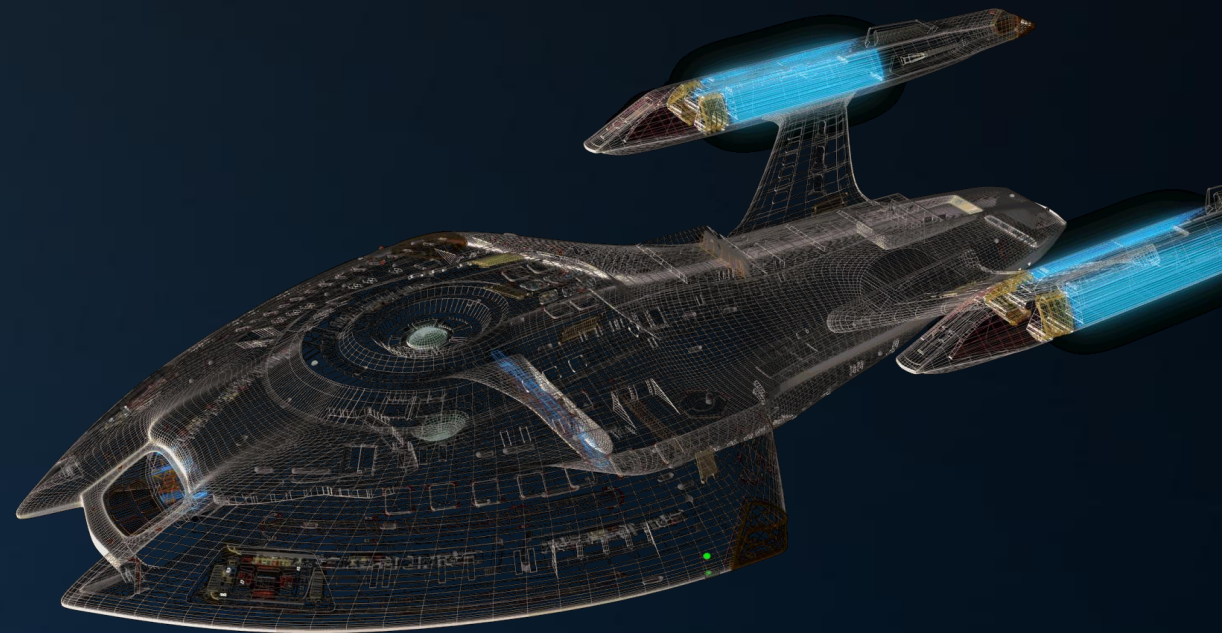
Assistance is available on Tuesday, April 28<sup>th</sup> in rooms

BBG-112, -175, -106, -109 and -103.



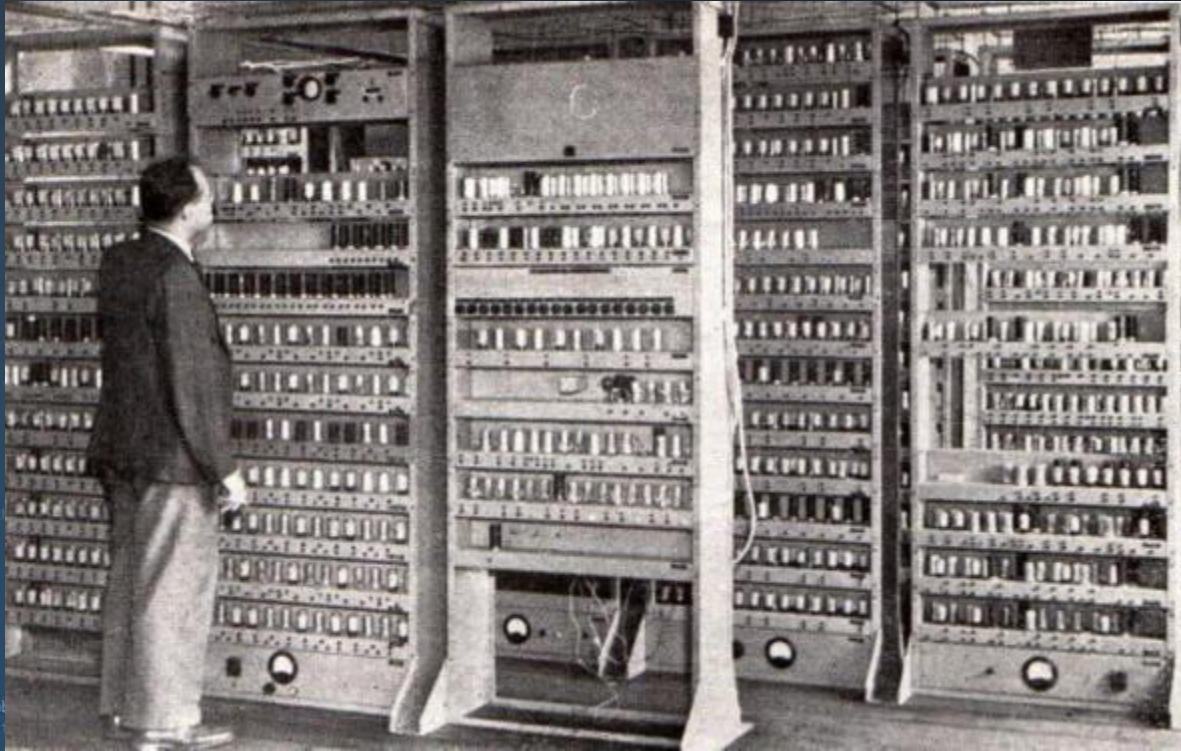
# Today's Agenda:

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





# Field Study



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

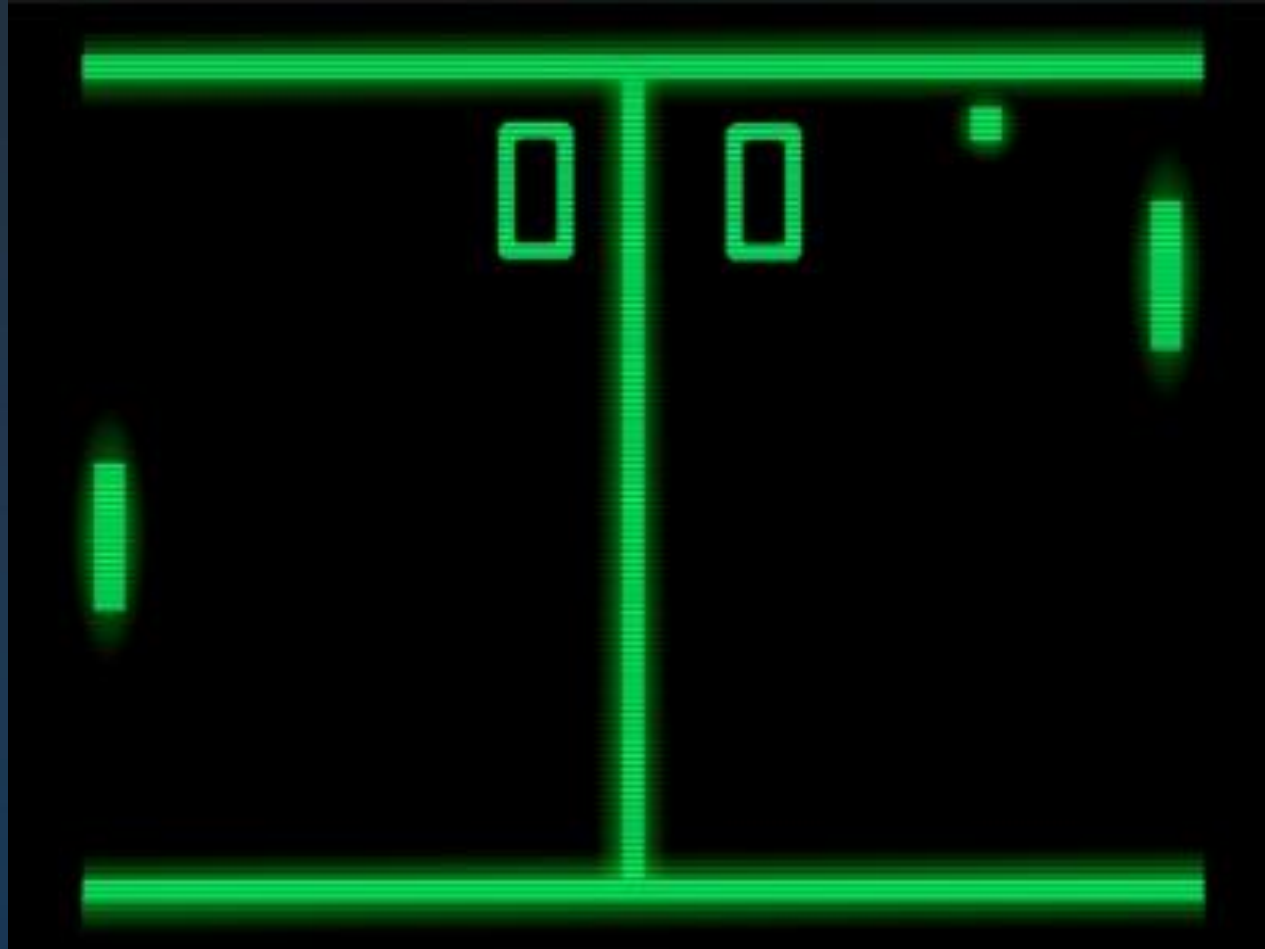


# Field Study

```

    if (depth < MAXDEPTH)
    {
        Vec inside = L;
        Vec nt = nc, nde = n;
        Vec pos2t = 1.0f - nnt * nnt;
        Vec D, N;
        Vec a = nt - nc, b = nt - nc;
        Vec Tr = 1 - (R0 + (1 - R0) * nnt);
        Vec R = (D * nnt - N * (pos2t));
        Vec E * diffuse;
        Vec refl;
        Vec refl + refr)) && (depth < MAXDEPTH)
        Vec D, N;
        Vec refl * E * diffuse;
        Vec refl;
        Vec MAXDEPTH)
        Vec survive = SurvivalProbability( diffuse );
        Vec estimation - doing it properly, closely following wall (survive)
        Vec if;
        Vec radiance = SampleLight( &rand, I, &t, &light);
        Vec e.x + radiance.y + radiance.z) > 0) && (e.x + radiance.y + radiance.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) * (radiance);
        Vec random walk - done properly, closely following wall (survive)
        Vec survive)
        Vec;
        Vec brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf );
        Vec survive;
        Vec pdf;
        Vec n = E * brdf * (dot( N, R ) / pdf);
        Vec sion = true;
    }

```





# Field Study



1981



1982



1982



# Field Study



1985



1987



1990



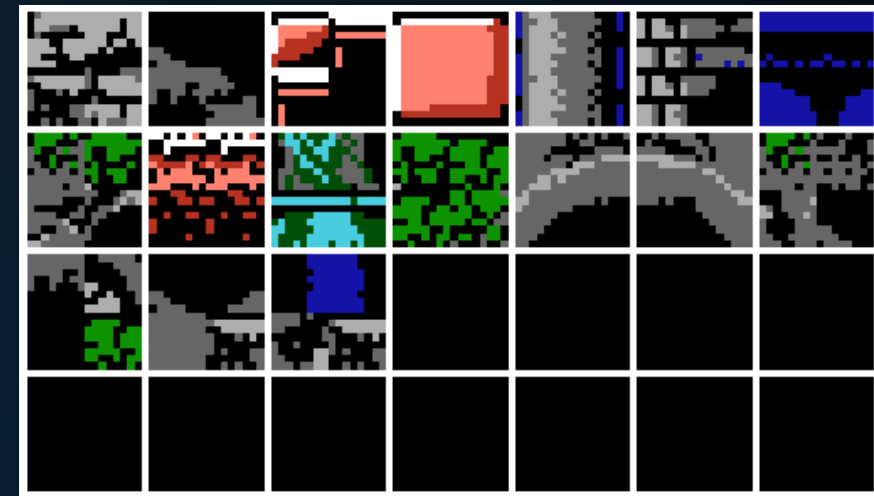
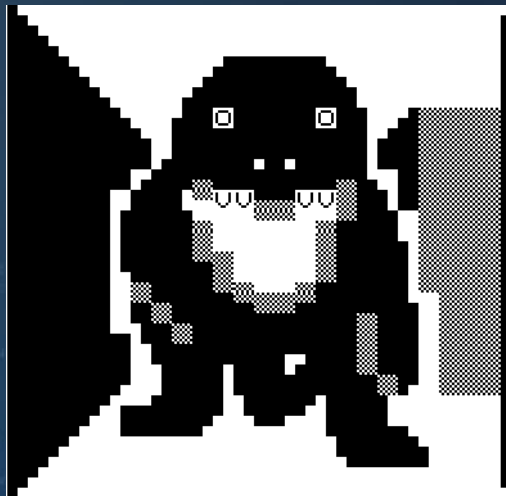


# Field Study

Early graphics:

2D, with limitations

- Tiles
- Few colors
- Sprites



# Field Study



```
at brdfPdf = EvaluateDiffuse( L, N ) * Pdf;
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 (survive)

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







# History of Graphics







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

























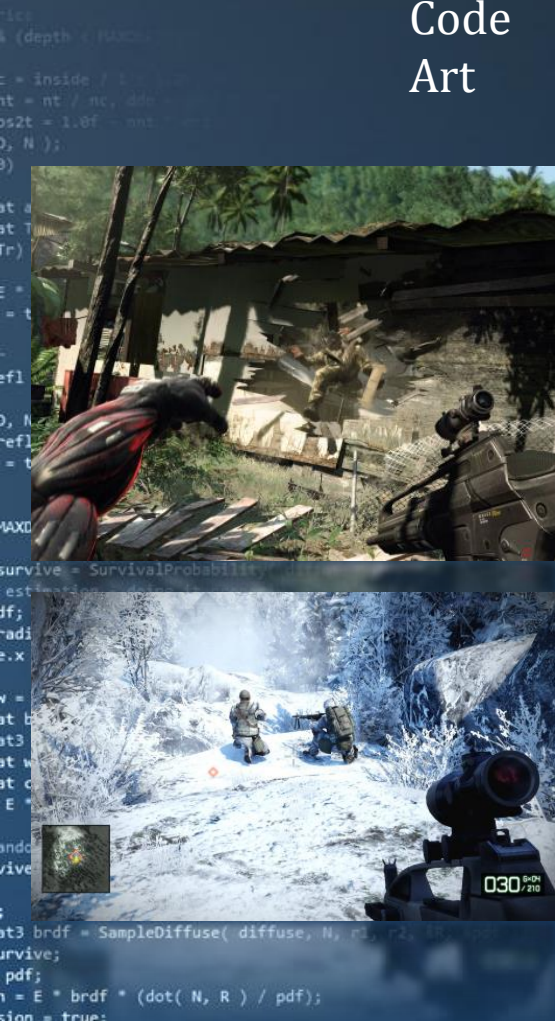




# Field Study

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



# State of the Art

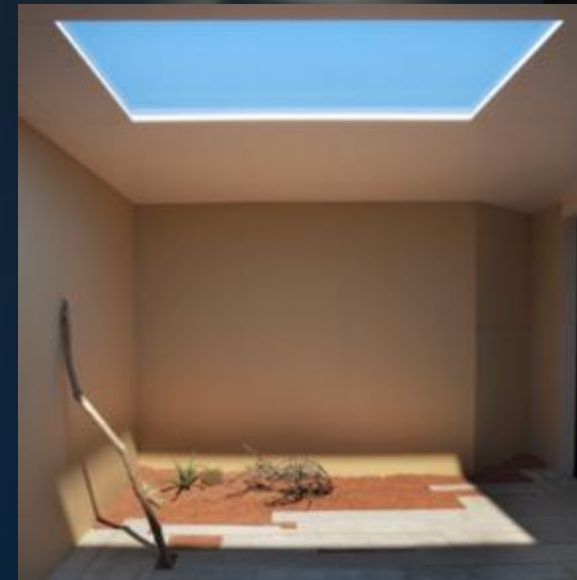
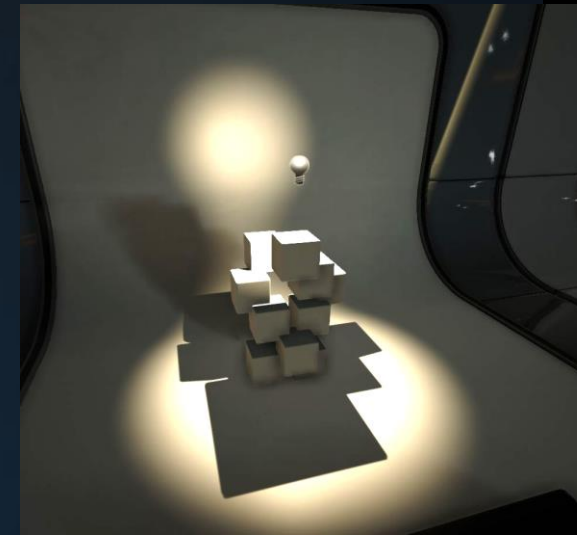
## Industry example: Unreal Engine 4

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

```

...
    & (depth < MAXDEPTH)
{
    // Inside / Outside
    int nt = nc; addo = 0;
    float os2t = 1.0f - nnt;
    float D, N;
    ...
    float a = nt - nc, b = nt - nc;
    float Tr = 1 - (RB + (1 - RB) * a);
    float R = (D * nnt - N * (1 - D));
    ...
    E * diffuse;
    = true;
    ...
    refl + refr)) && (depth < MAXDEPTH)
{
    D, N;
    refl * E * diffuse;
    = true;
    ...
    MAXDEPTH)
{
    survive = SurvivalProbability( diffuse );
    estimation - doing it properly, closely following walls (survive)
    if;
    radiance = SampleLight( &rand, I, &t, &lightmap );
    radiance.x + radiance.y + radiance.z > 0) && (depth < MAXDEPTH)
{
    w = true;
    float brdfPdf = EvaluateDiffuse( L, N ) * Psurvive;
    float3 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 (survive)
    survive)
    ...
    float3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf );
    survive;
    pdf;
    n = E * brdf * (dot( N, R ) / pdf);
    n = 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

# State of the Art

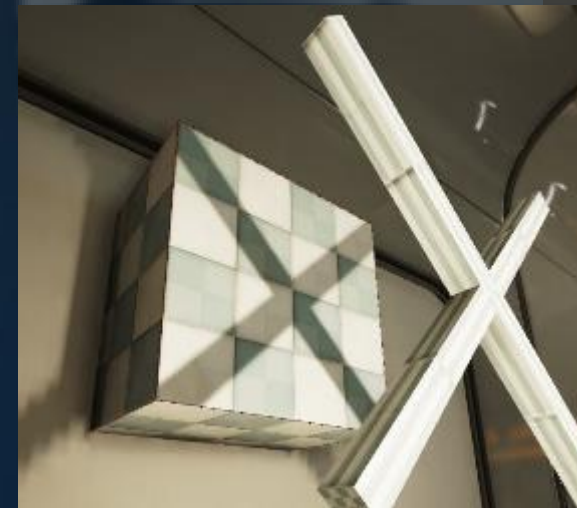
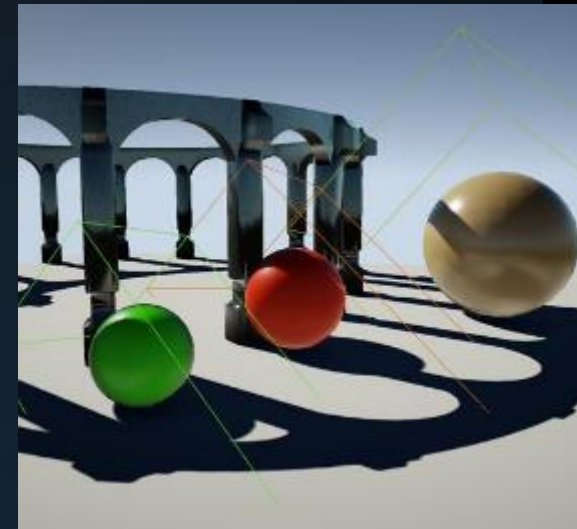
## Industry example: Unreal Engine 4

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

```

...ics
& (depth < MAXDEPTH)
{
    // Inside / Outside
    int nt = nt / nc; add = add / nc;
    float r2t = 1.0f - nnt * nnt;
    float D, N );
    // ...
    float a = nt - nc, b = nt * nc;
    float Tr = 1 - (R0 + (1 - R0) * r2t);
    float R = (D * nnt - N * (1 - R0));
    // ...
    E * diffuse;
    // ...
    refl + refr)) && (depth < MAXDEPTH)
    {
        D, N );
        refl * E * diffuse;
        // ...
        MAXDEPTH)
    {
        survive = SurvivalProbability( diffuse );
        // estimation - doing it properly, closely following wall
        if (survive)
        {
            radiance = SampleLight( &rand, I, &L, &align, &align );
            float x = radiance.x + radiance.y + radiance.z;
            if (x > 0) && (x < 1)
            {
                w = true;
                float brdfPdf = EvaluateDiffuse( L, N ) * Psurvive;
                float3 factor = diffuse * INVPI;
                float weight = Mix2( directPdf, brdfPdf );
                float cosThetaOut = dot( N, L );
                E * ((weight * cosThetaOut) / directPdf) * (radiance);
            }
            // random walk - done properly, closely following wall
            survive)
        }
        // ...
        float3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf );
        survive;
        pdf;
        n = E * brdf * (dot( N, R ) / pdf);
        // ...
        // ...
    }
}

```



- [-] **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](#)





```

    (depth < MAXDEPTH)
    {
        // Inside / Left side of the sphere
        int = nt / nc, ddx = dot(N, N);
        float cos2t = 1.0f - nnt * ddx;
        float D, N *);
        // Inside / Right side of the sphere
        int a = nt - nc, b = nt * nc;
        float Tr = 1 - (RB + (1 - RB) * cos(a * 2 * PI));
        float R = (D * nnt - N * ddx) * cos(b * 2 * PI);
        // Inside / Refractive index
        float E * diffuse;
        // Inside / Refractive index
        float = true;
        // Inside / Refractive index
        float refl + refr)) && (depth < MAXDEPTH)
        {
            // Inside / Refractive index
            float D, N *);
            // Inside / Refractive index
            float refl * E * diffuse;
            // Inside / Refractive index
            float = true;
            // Inside / Refractive index
            float MAXDEPTH)
            {
                // Inside / Refractive index
                float survive = SurvivalProbability( diffuse, r,
                // Inside / Refractive index
                // estimation - doing it properly, clearly
                float df;
                // Inside / Refractive index
                float radiance = SampleLight( &rand, l, &t, &ll,
                // Inside / Refractive index
                float ex + radiance.y + radiance.z) > 0) && (depth <
                // Inside / Refractive index
                float w = true;
                // Inside / Refractive index
                float brdfPdf = EvaluateDiffuse( L, N ) * Pdf;
                // Inside / Refractive index
                float at3 factor = diffuse * INVPI;
                // Inside / Refractive index
                float at weight = Mix2( directPdf, brdfPdf );
                // Inside / Refractive index
                float at cosThetaOut = dot( N, L );
                // Inside / Refractive index
                float E * ((weight * cosThetaOut) / directPdf);
                // Inside / Refractive index
                // random walk - done properly, closely follow
                // Inside / Refractive index
                // survive)
                // Inside / Refractive index
                //
                // Inside / Refractive index
                float at3 brdf = SampleDiffuse( diffuse, N, r1,
                // Inside / Refractive index
                // survive;
                // Inside / Refractive index
                float pdf;
                // Inside / Refractive index
                float n = E * brdf * (dot( N, R ) / pdf);
                // Inside / Refractive index
                // vision = true;
            }
        }
    }
}

```

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



# State of the Art

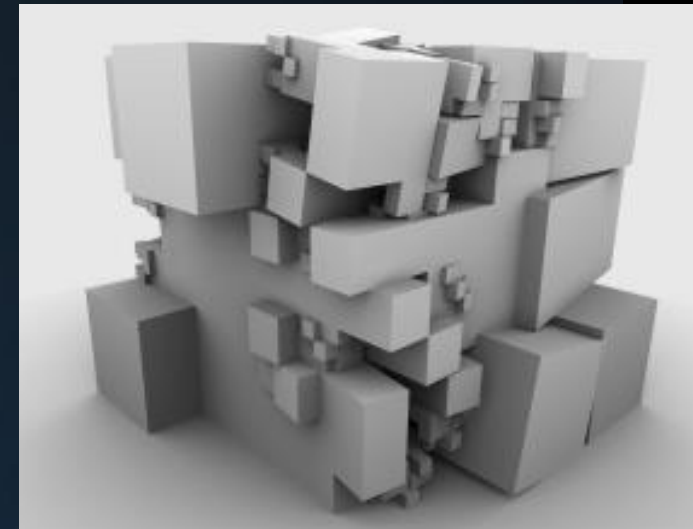
## Industry example: Unreal Engine 4

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

```

...
    & (depth < MAXDEPTH)
{
    ...
    nt = nt / nc; add = ...
    pos2t = 1.0f - nnt; ...
    D, N );
    ...
    at a = nt - nc, b = nt - nc;
    at Tr = 1 - (RB + (1 - RB) * ...
    Tr) R = (D * nnt - N * (D ...
    ...
    E * diffuse;
    ...
    = true;
    ...
    refl + refr)) && (depth < MAXDEPTH)
{
    D, N );
    refl * E * diffuse;
    ...
    = true;
    ...
    MAXDEPTH)
{
    survive = SurvivalProbability( diffuse );
    estimation - doing it properly, closely following wall
    if;
    radiance = SampleLight( &rand, I, &t, &light ...
    e.x + radiance.y + radiance.z) > 0) && (cosThetaOut > 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 wall
    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](#)





# 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

```

...
    & (depth < MAXDEPTH)
{
    ...
    nt = nt / nc; add = ...
    pos2t = 1.0f - nnt; ...
    D, N );
    ...
    at a = nt - nc, b = nt - nc;
    at Tr = 1 - (RB + (1 - RB) * ...
    Tr) R = (D * nnt - N * (D ...
    ...
    E * diffuse;
    ...
    = true;
    ...
    refl + refr)) && (depth < MAXDEPTH)
{
    D, N );
    refl * E * diffuse;
    ...
    = true;
    ...
    MAXDEPTH)
{
    survive = SurvivalProbability( diffuse );
    estimation - doing it properly, closely following walls (survive)
    if(
    radiance = SampleLight( &rand, I, &t, &lightmap );
    e.x + radiance.y + radiance.z) > 0) && (survive)
    ...
    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 walls (survive)
    survive)
    ...
    at3 brdf = SampleDiffuse( diffuse, N, r1, r2, &R, &pdf );
    survive;
    pdf;
    n = E * brdf * (dot( N, R ) / pdf);
    ...
    sion = true;
    ...

```



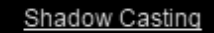
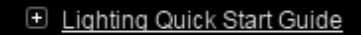
- [-] **Engine Features**
  - [-] Graphics
    - Rendering Overview
    - [-] Lighting and Shadows
      - + Lighting Quick Start Guide
      - + Types of Lights
        - Shadow Casting
      - [-] Light Mobility
        - Movable Lights
        - Static Lights
        - Stationary lights
    - + Lightmass 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



- 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**





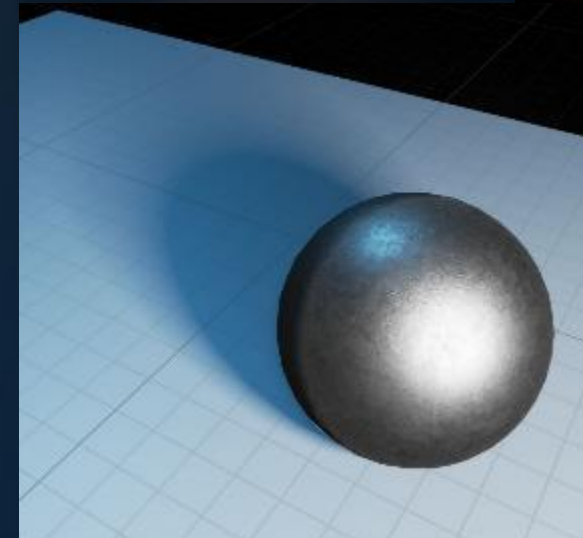
# State of the Art

## 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
        int nt = nc;
        float nnt = nt / nc;
        float nnt2 = 1.0f - nnt * nnt;
        float D = N * N;
        float R = (D * nnt - N * N) * (D * nnt - N * N);
        float a = nt - nc;
        float b = nt + nc;
        float Tr = 1 - (R0 + (1 - R0) * R);
        float R = (D * nnt - N * N) * (D * nnt - N * N);
        float E * diffuse;
        = true;
        refl + refr)) && (depth < MAXDEPTH)
        D, N );
        refl * E * diffuse;
        = true;
        MAXDEPTH)
        survive = SurvivalProbability( diffuse );
        estimation - doing it properly, closely following walls
        if;
        radiance = SampleLight( &rand, I, &L, &lightmap );
        e.x + radiance.y + radiance.z) > 0) && (depth < MAXDEPTH)
        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](#)



- 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 2015 - Lecture 1: “Introduction”

## END of “Introduction”

next lecture: “Graphics Fundamentals”

