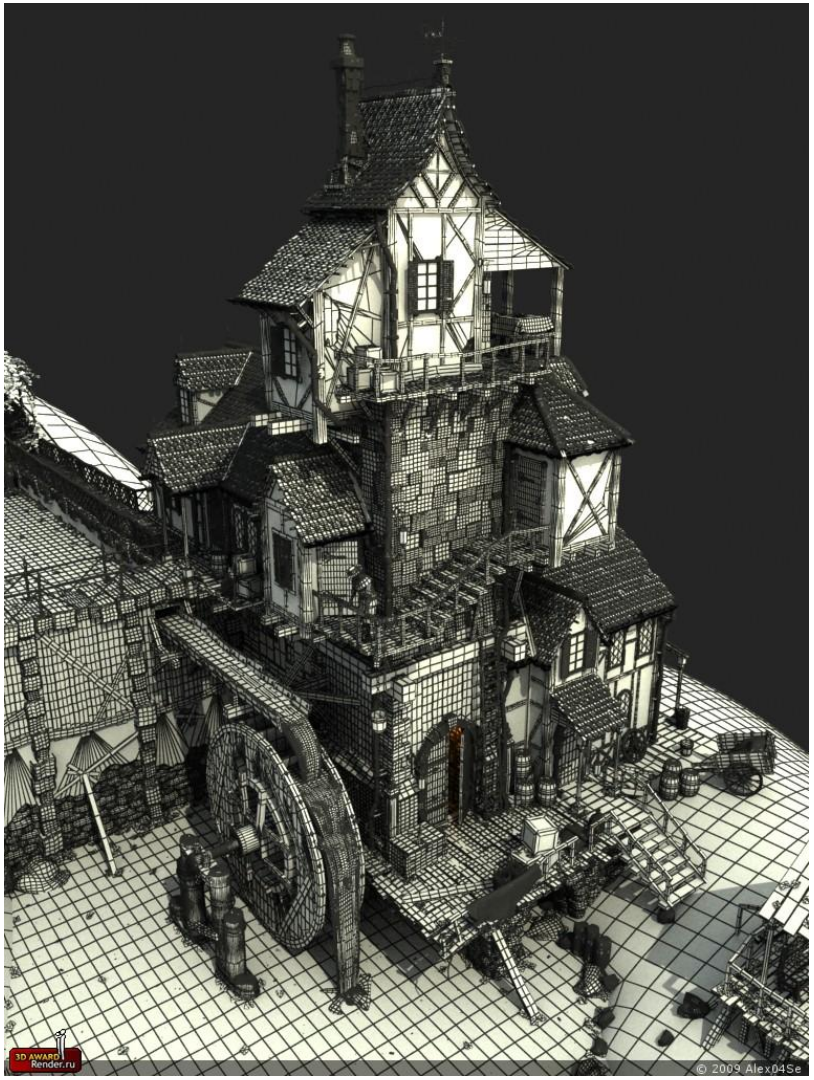
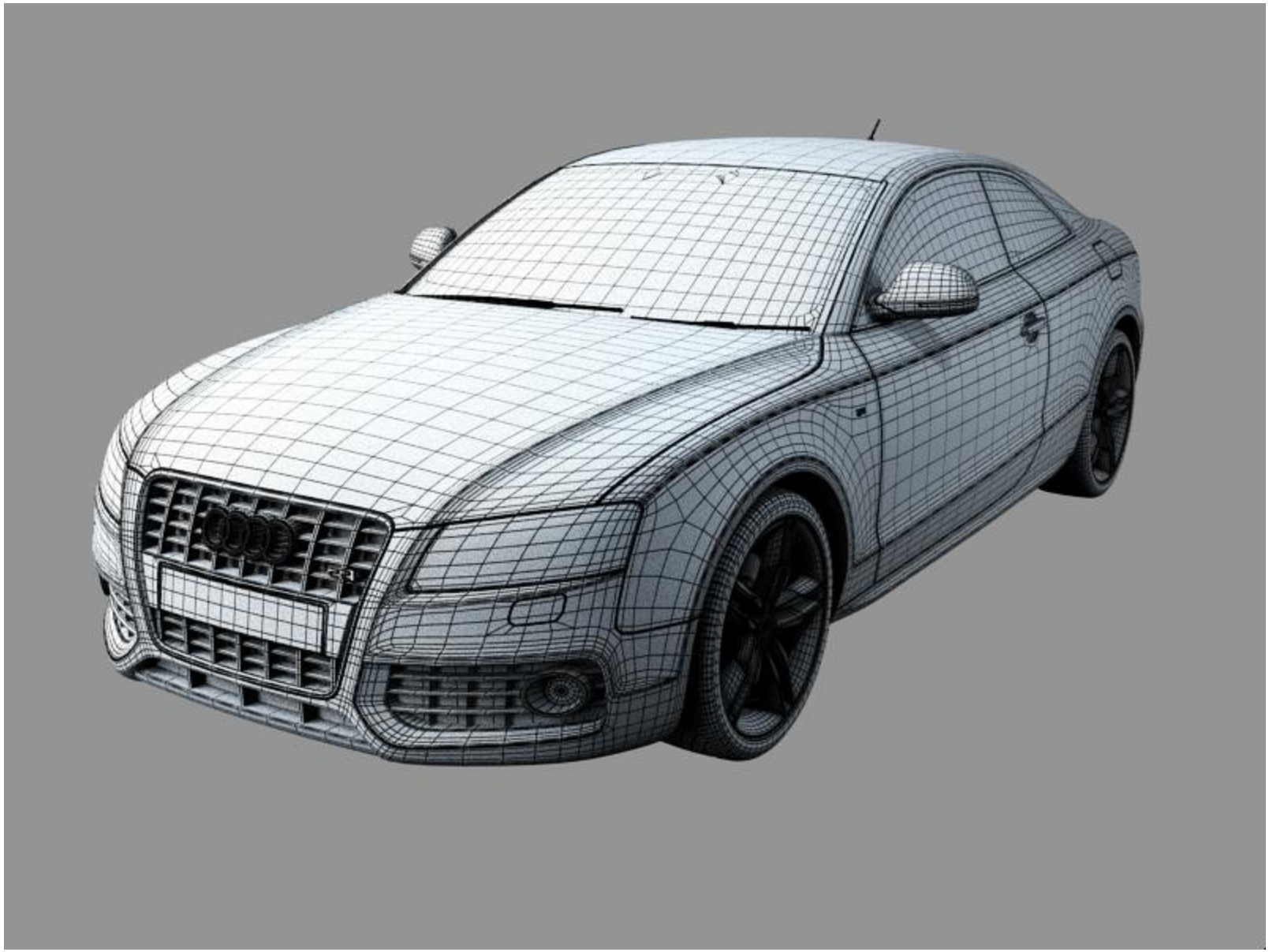
Computer graphics

Михайлов Александр

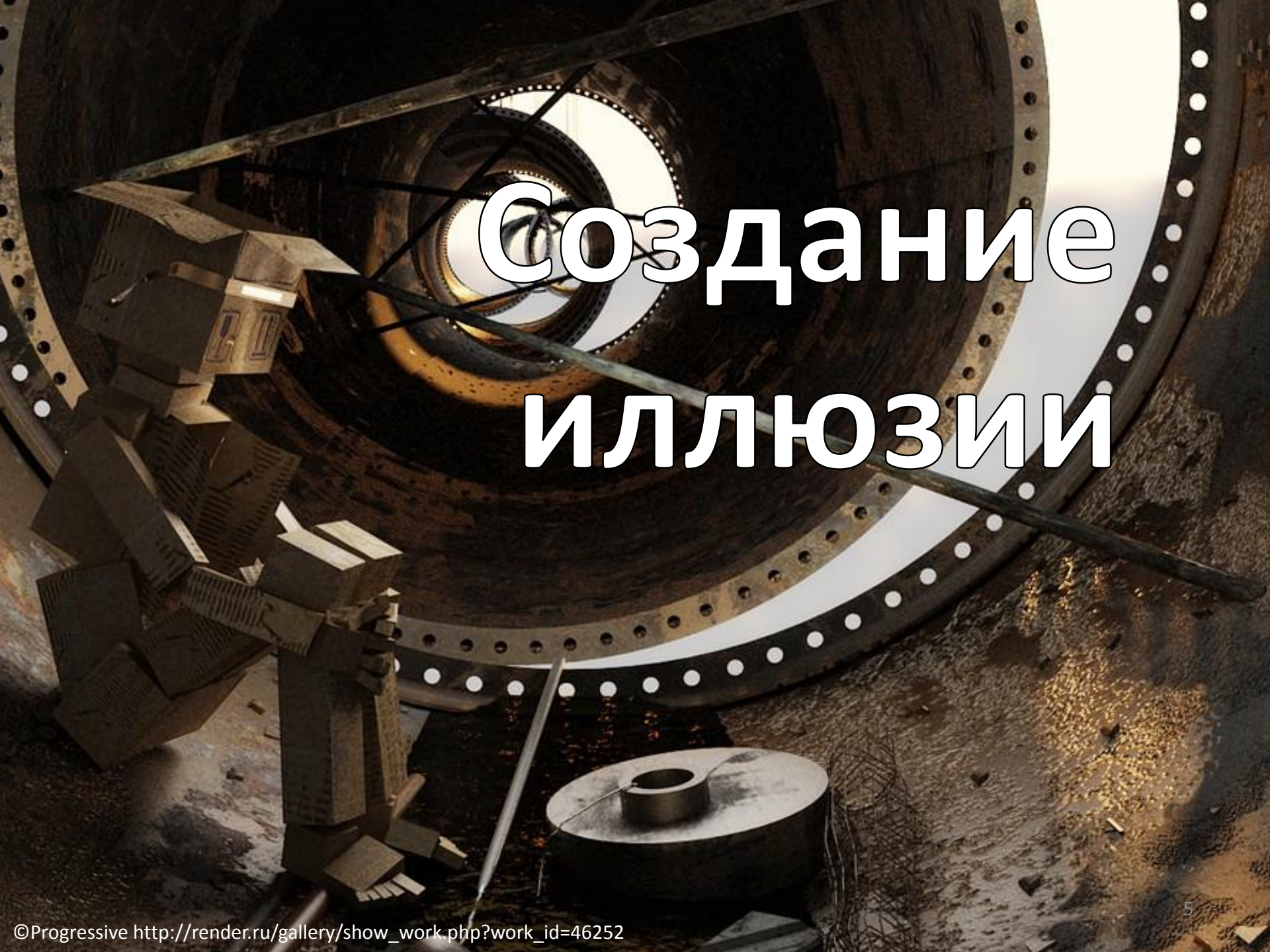
СПбАУ, 2011

Задача CG

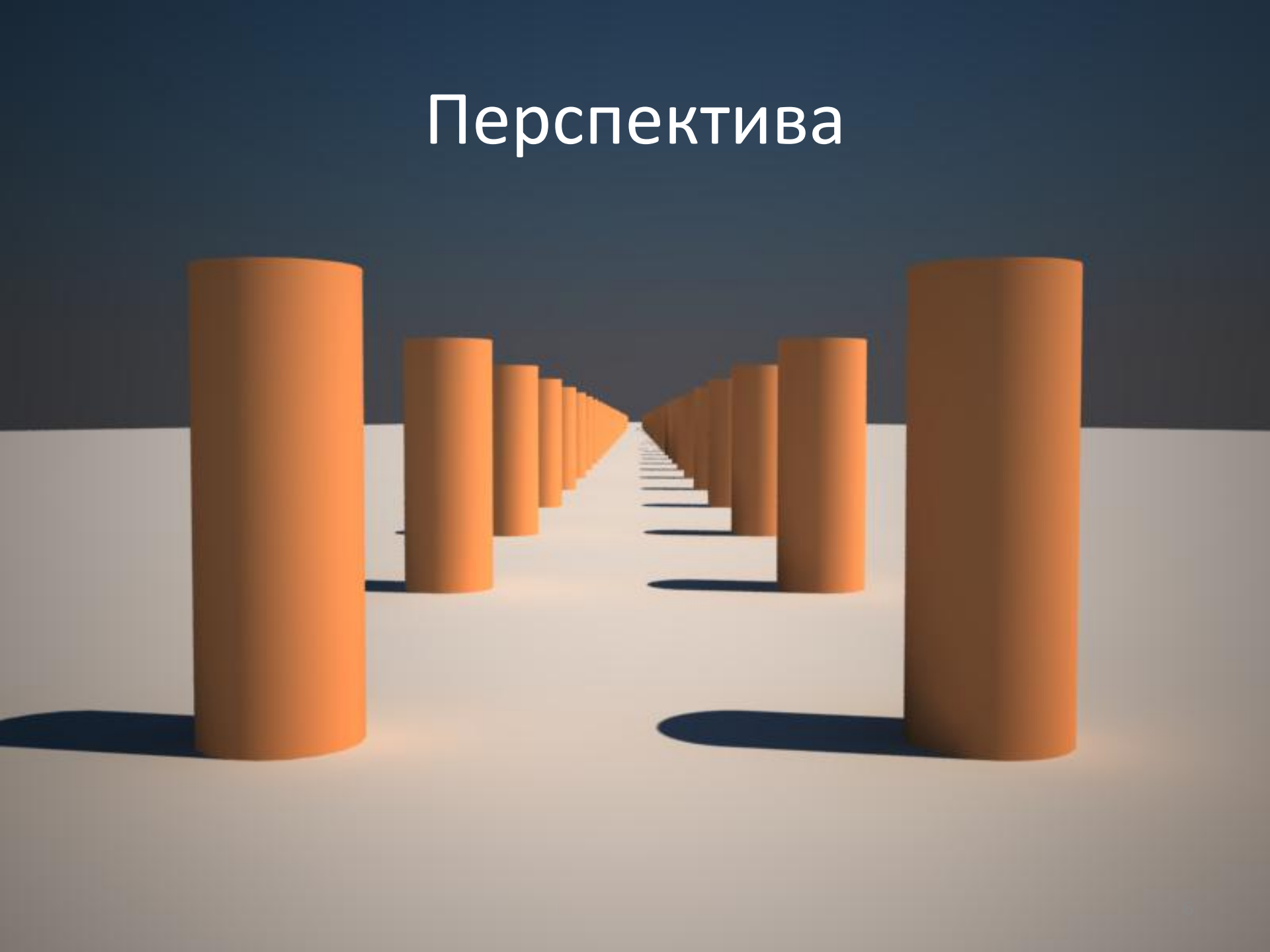




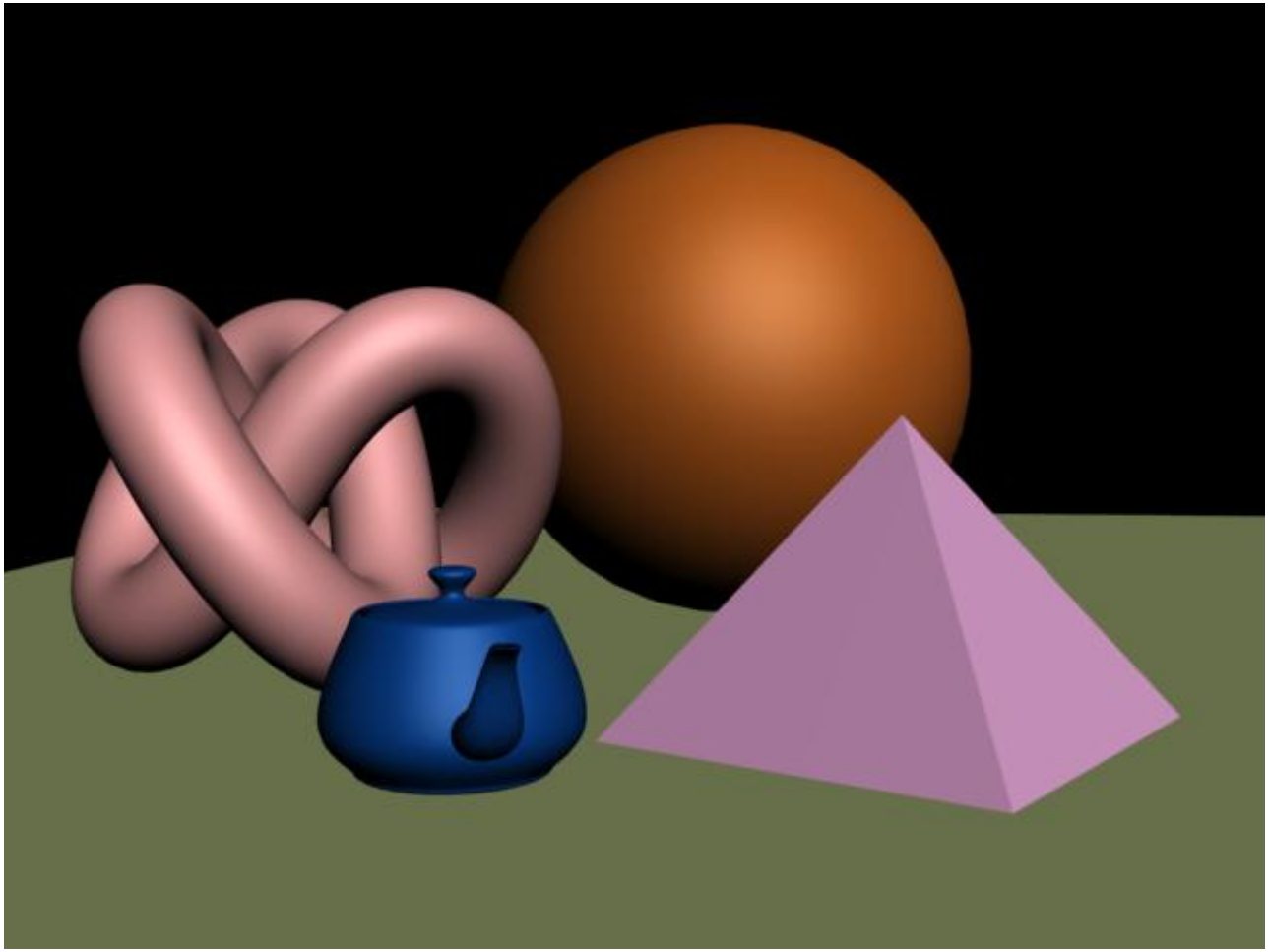
Teapotahedron



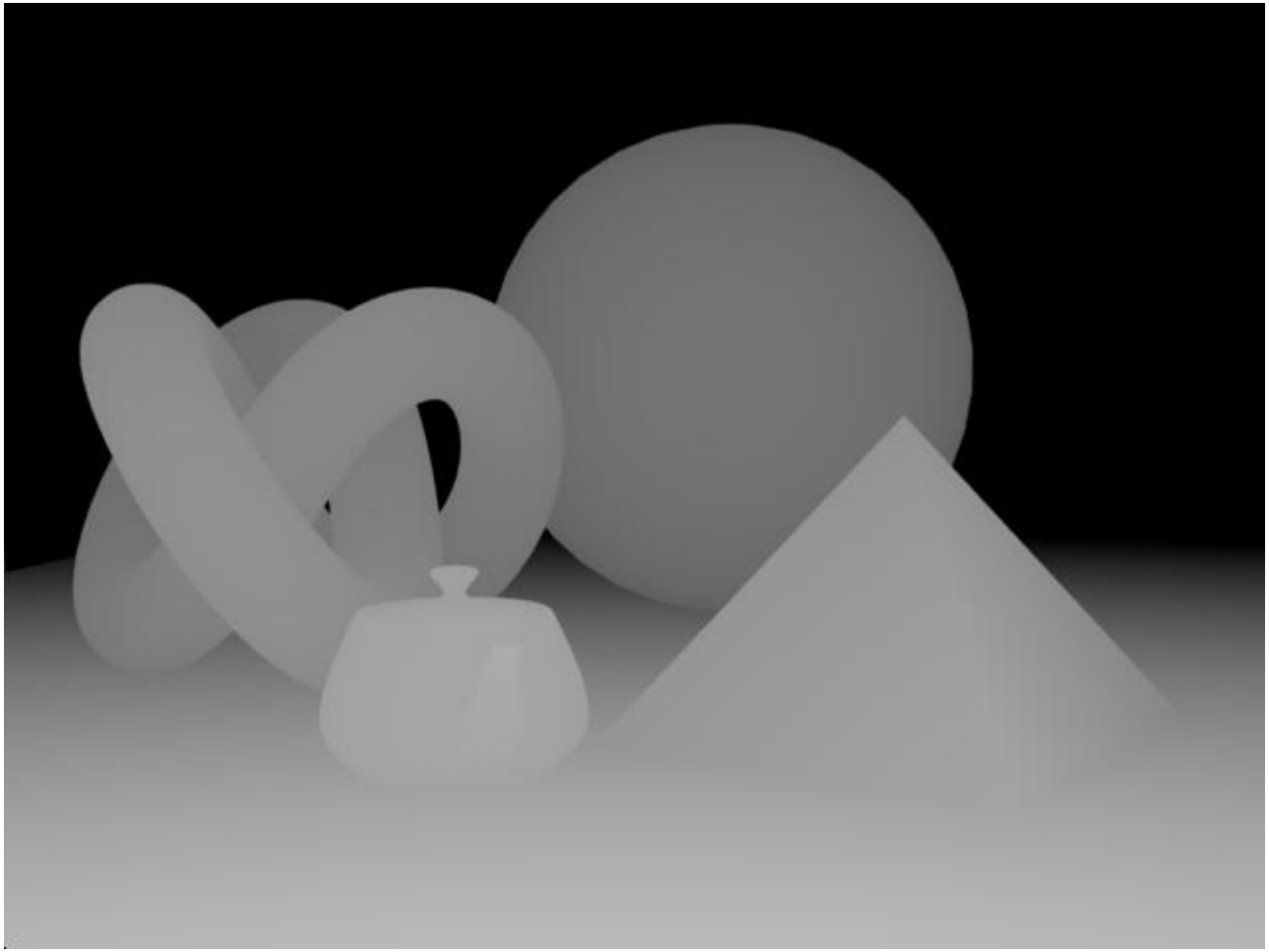
Перспектива



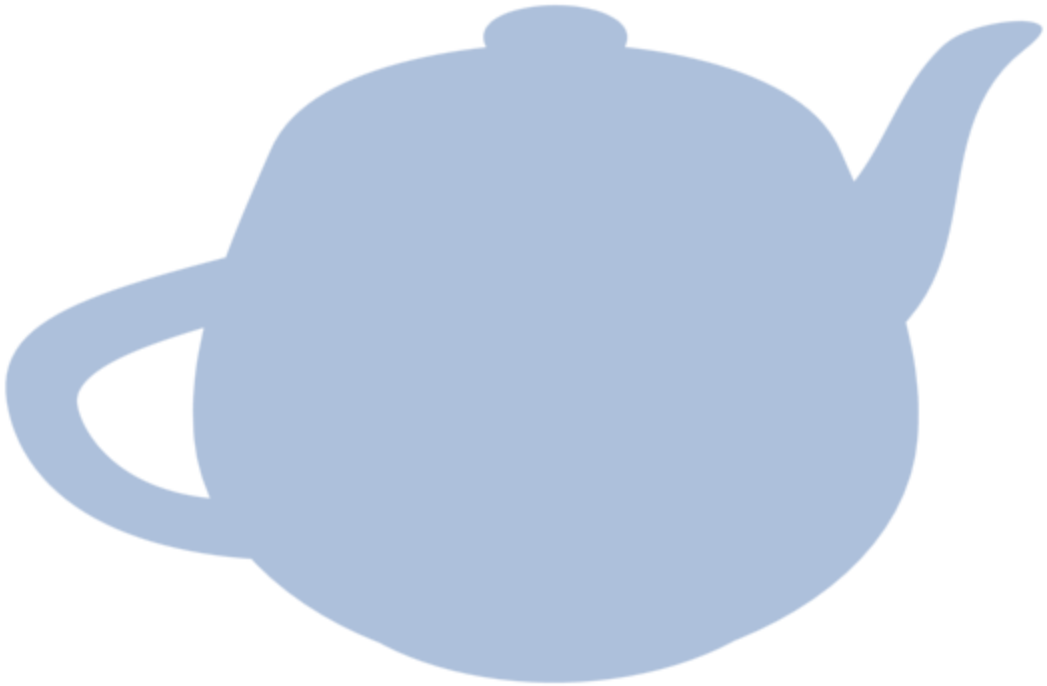
Заслонение



Z-буфферизация



Освещение



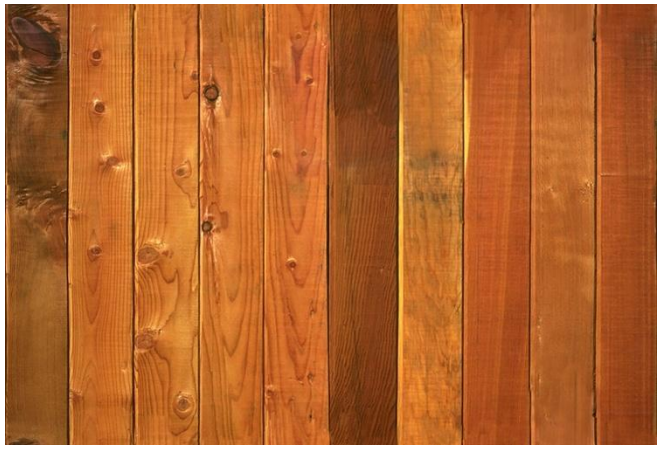
Освещение

Текстурирование

© MEX Солнечный http://render.ru/gallery/show_work.php?work_id=62918

Пример

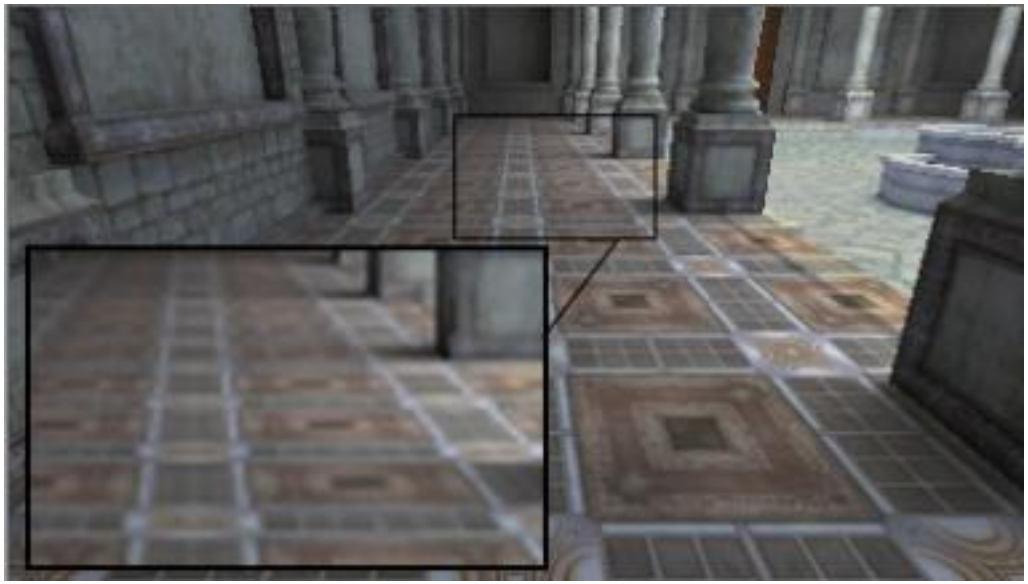
+



=

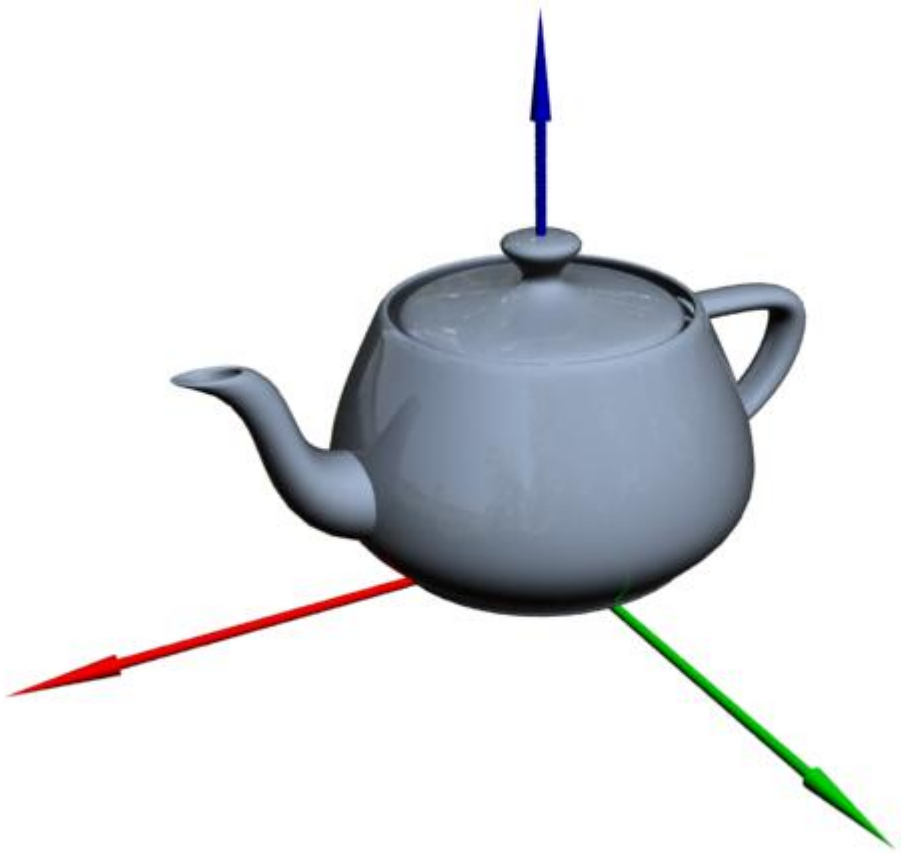
Текстурные координаты

Фильтрация текстур

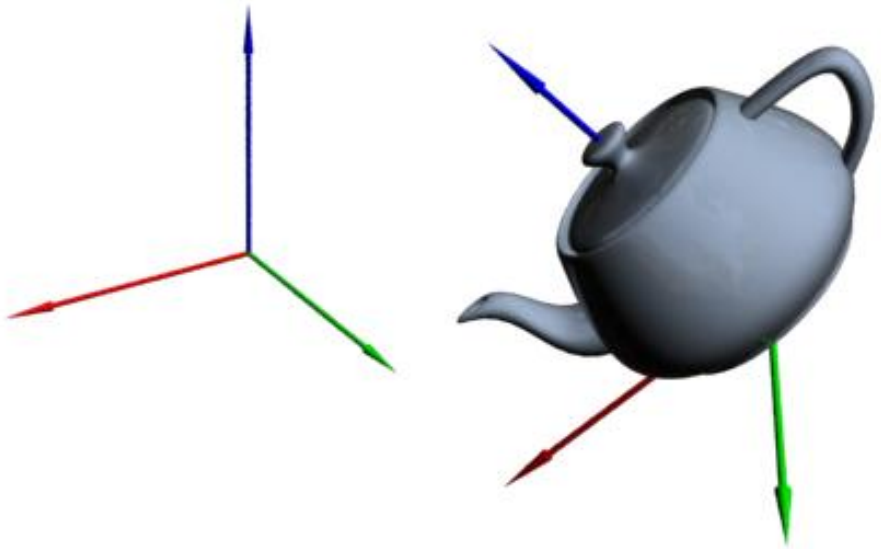


Модельно-видовые преобразования

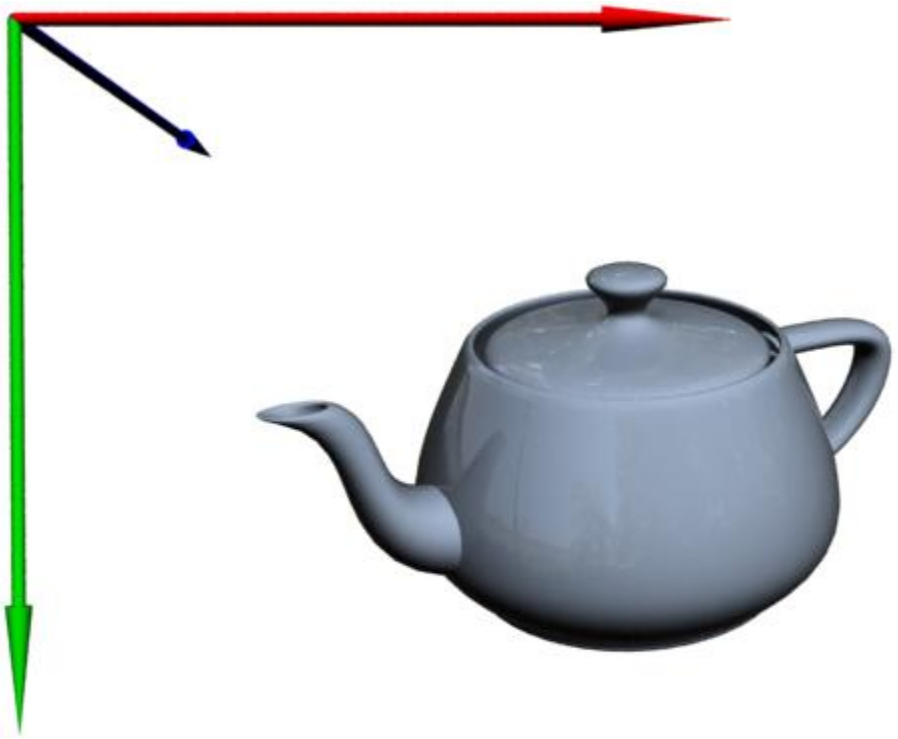
Model space



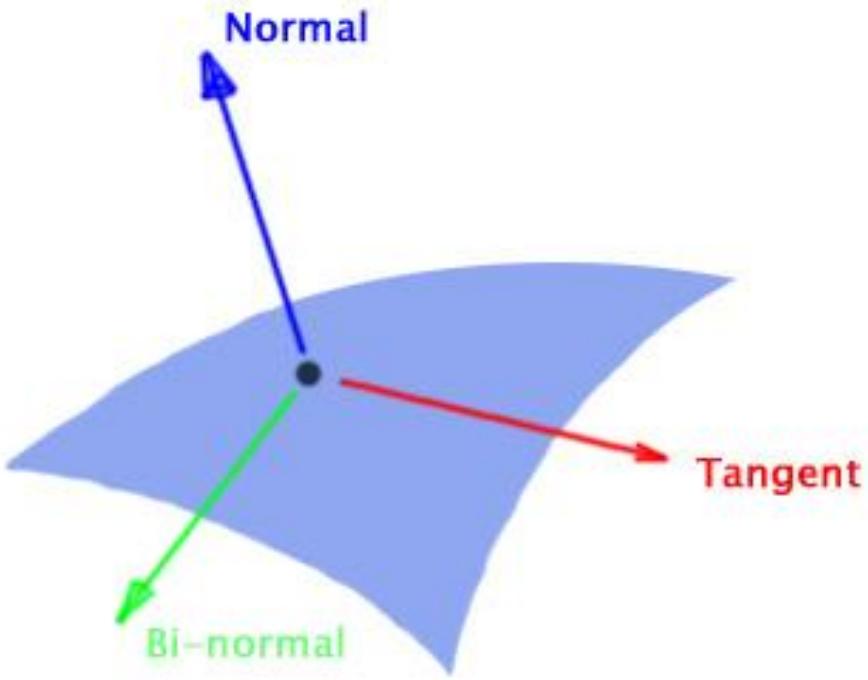
World space



Screen space



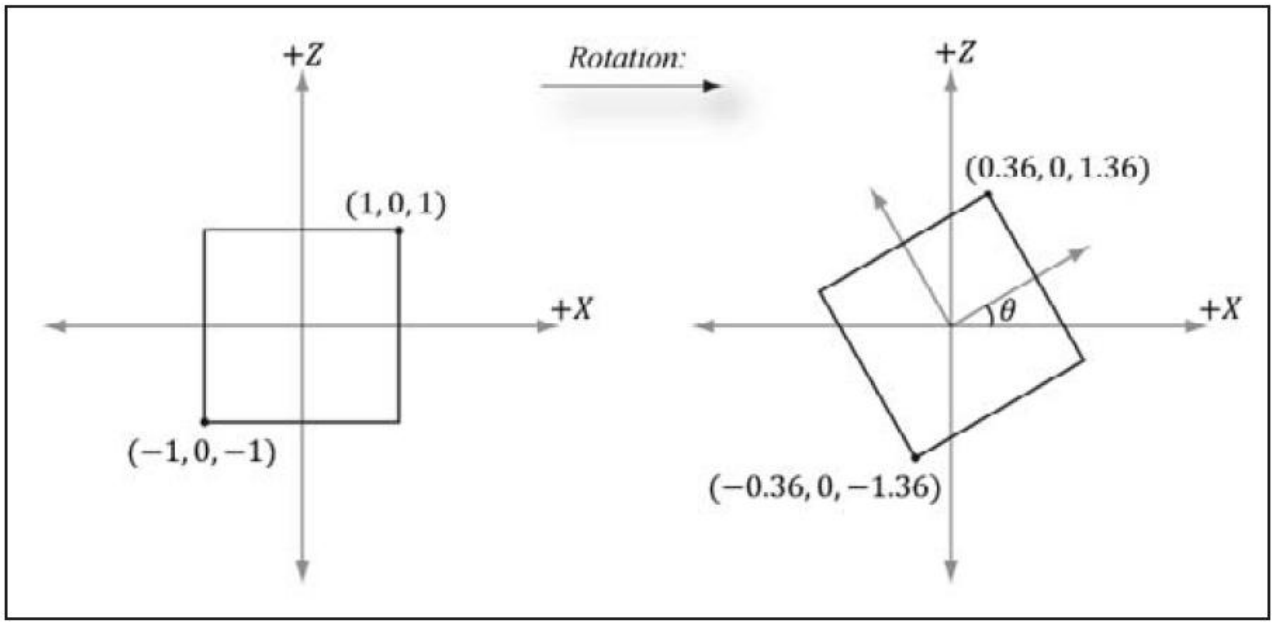
Tangent space



Преобразования координат

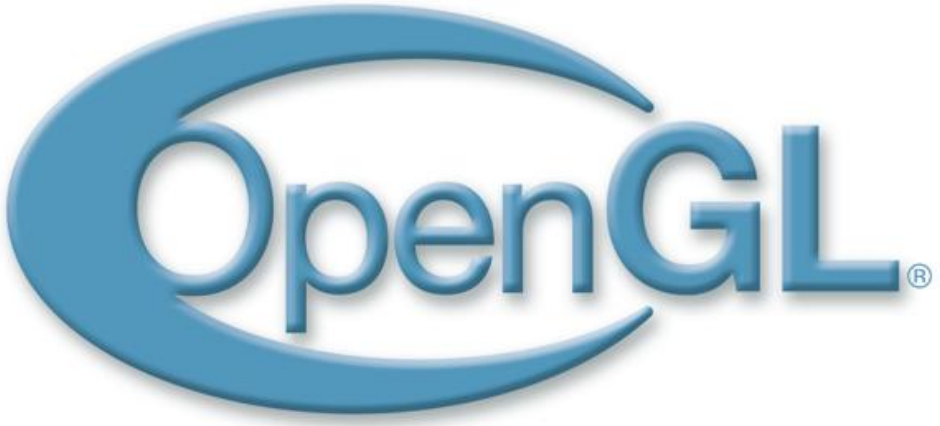
$$[-1, 0, -1, 1]\begin{bmatrix} \frac{\sqrt{3}}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{1}{2} & 0 & \frac{\sqrt{3}}{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \approx [-0.36, 0, -1.36, 1]$$

$$[1, 0, 1, 1]\begin{bmatrix} \frac{\sqrt{3}}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{1}{2} & 0 & \frac{\sqrt{3}}{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \approx [0.36, 0, 1.36, 1]$$



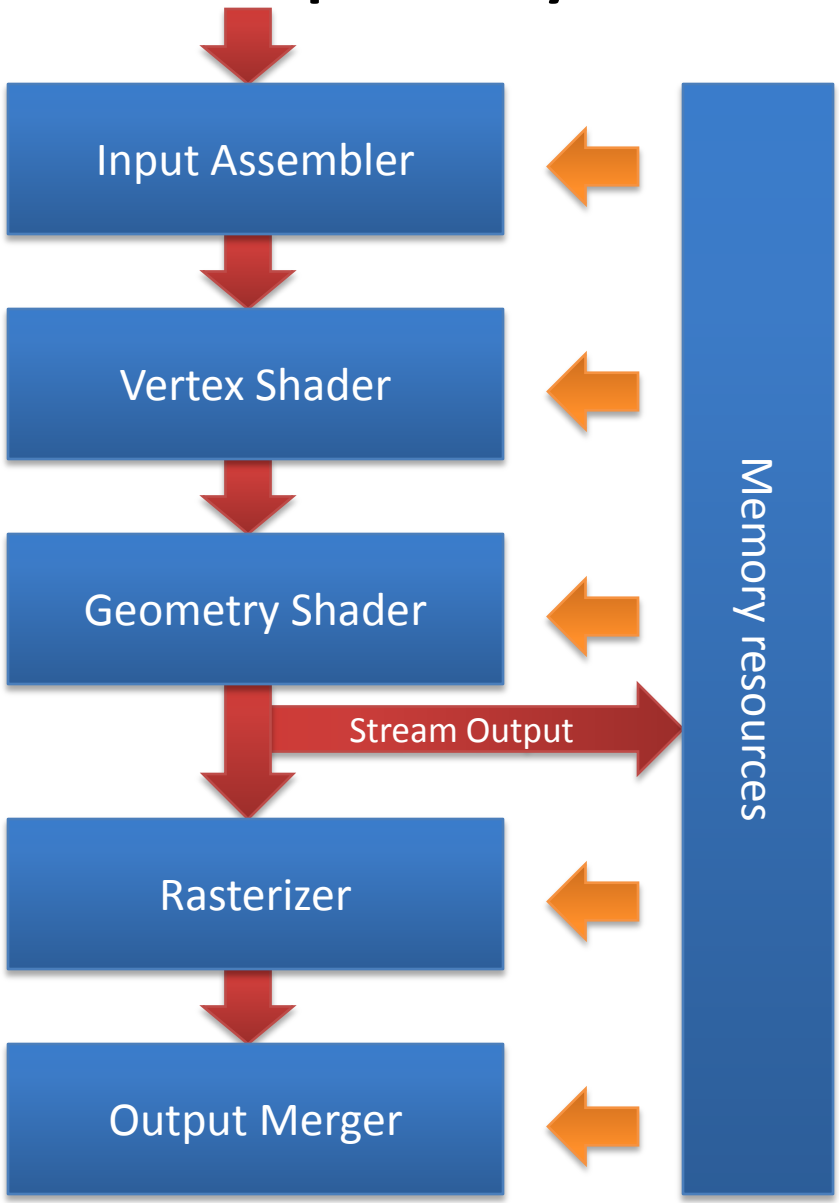
GAPI

vs



Что лучше?

Конвейер визуализации



Языки программирования шейдеров

DirectX	OpenGL
HLSL	GLSL
Nvidia Cg	

Vertex Shader

Вход: вершина(position, normal, texcoord…) в пространстве модели (*model space*)

Выход: вершина в пространстве экрана (*Screen space*)

```
for(int i = 0; i < vertices.size(); ++i) {
   vertices[i] = VertexShader(vertices[i]);
}
```

Vertex Shader

```
struct VS_INPUT
{
    float4 Pos      : POSITION;
    float3 Norm     : NORMAL;
    float2 Tex      : TEXCOORD0;
};
```

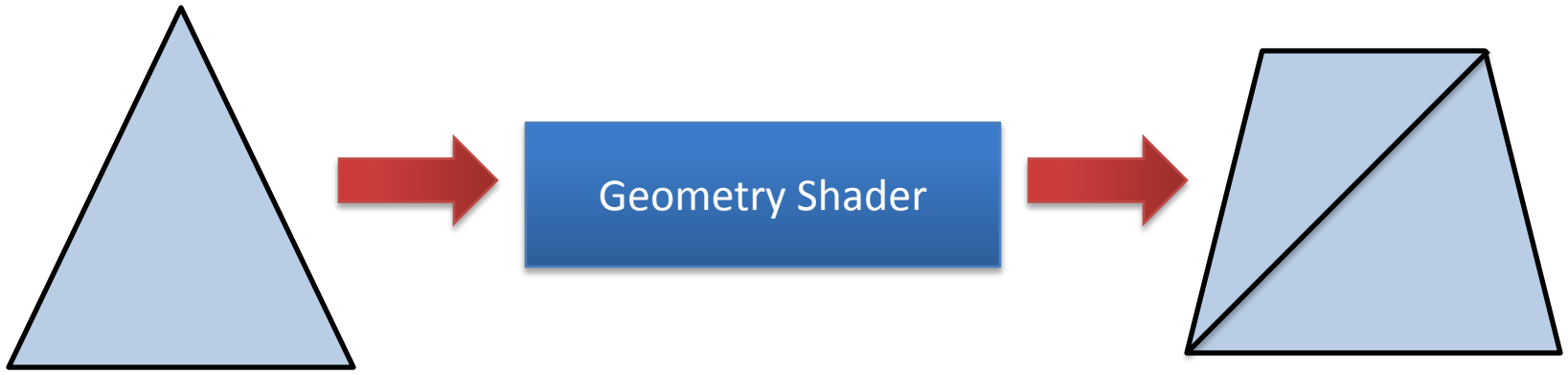
```
struct GSPS_INPUT
{
    float4 Pos : SV_POSITION;
    float3 Norm : TEXCOORD0;
    float2 Tex : TEXCOORD1;
};
```

```
GSPS_INPUT VS( VS_INPUT input )
{
    GSPS_INPUT output;

    output.Pos = mul( input.Pos, WorldVeiwProj );
    output.Norm = mul( input.Norm, World );
    output.Tex = input.Tex;

    return output;
}
```

Geometry Shader

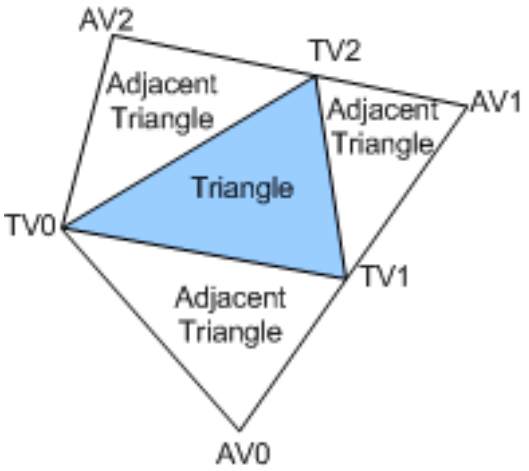
Вход: Примитив

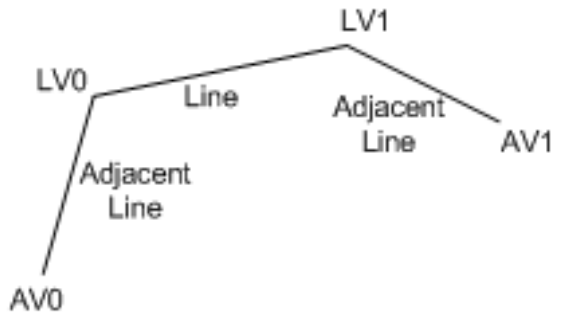
Выход: Примитив(ы)



Geometry Shader

- Примитивы

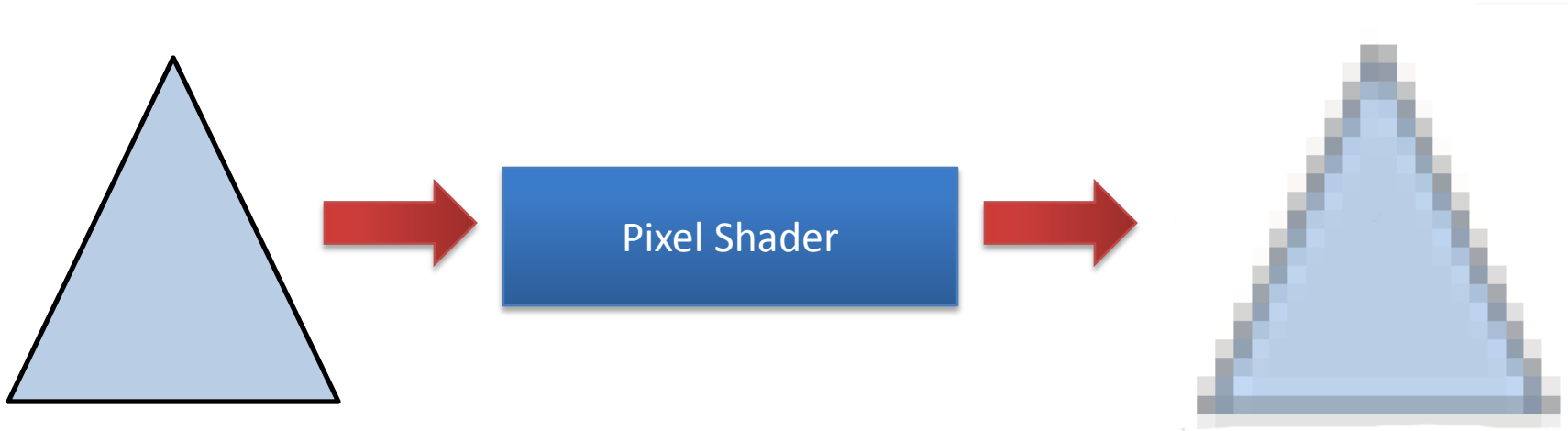




Pixel Shader

Вход: вершина(position, normal, texcoord…) в пространстве экрана (*screen space*)

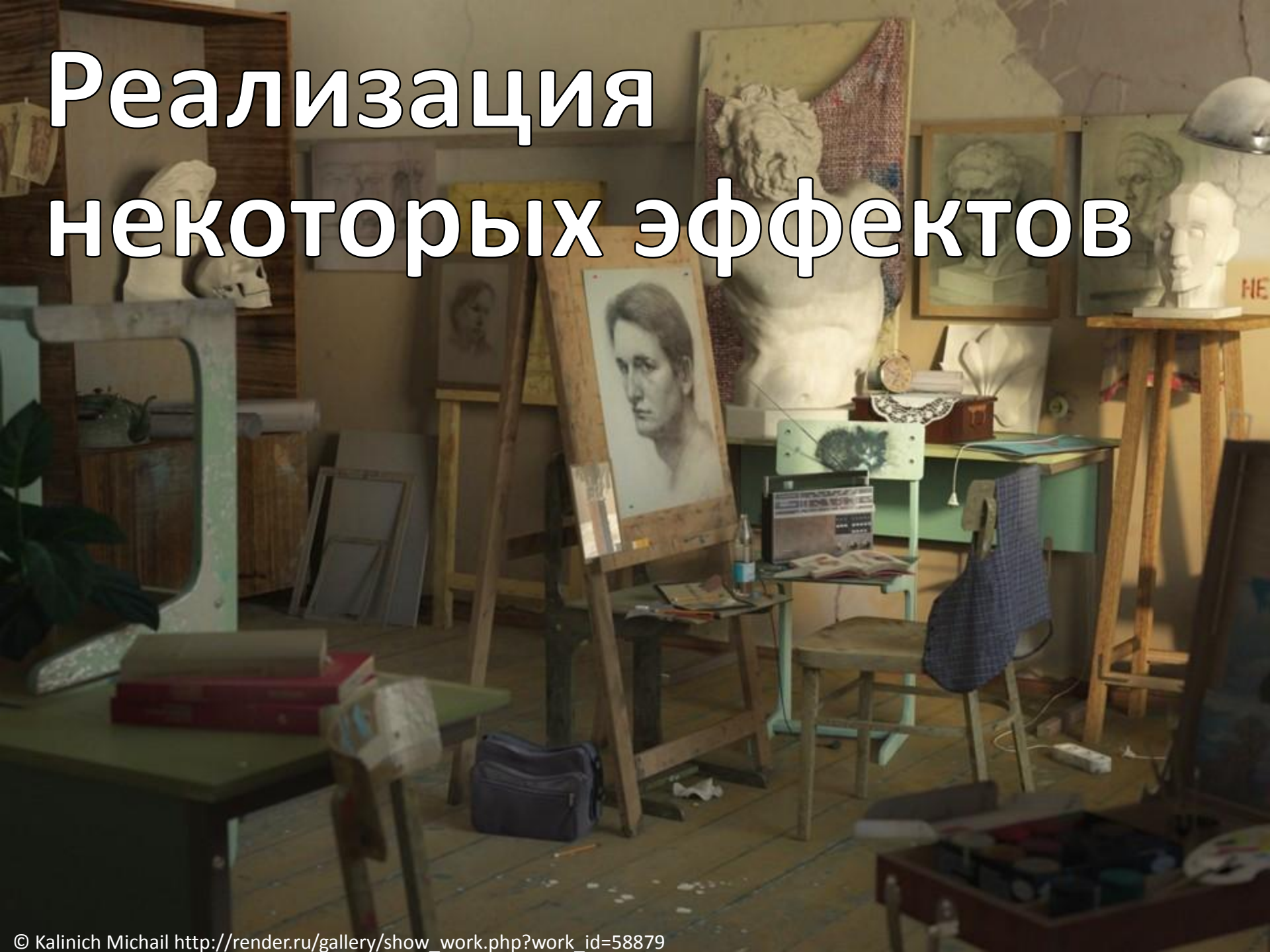
Выход: цвет пикселя



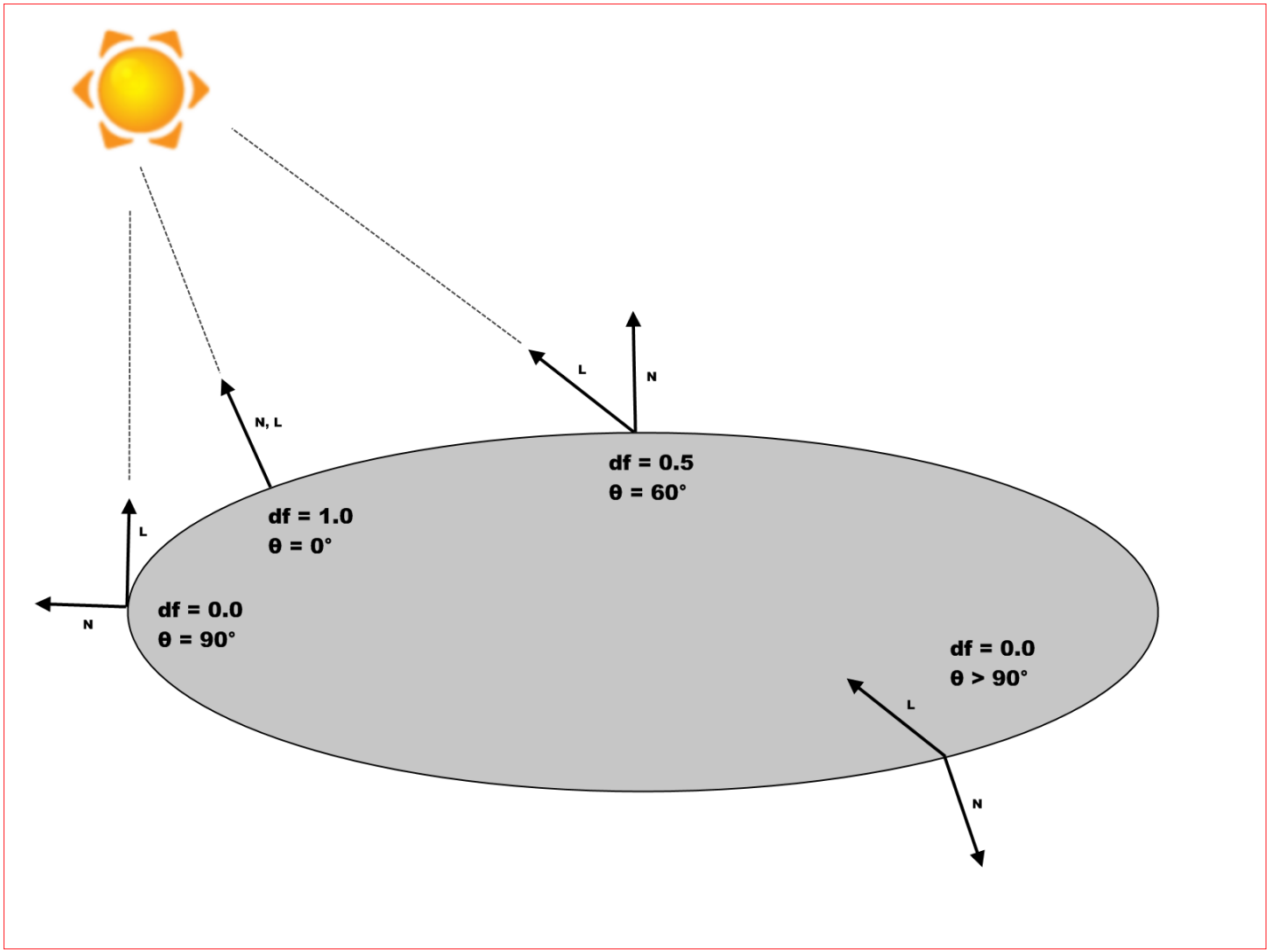
Pixel Shader example

```
struct GSPS_INPUT
{
    float4 Pos : SV_POSITION;
    float3 Norm : TEXCOORD0;
    float2 Tex : TEXCOORD1;
};

float4 PS( GSPS_INPUT input) : SV_Target
{
    // Calculate lighting assuming light color is <1,1,1,1>
    float fLighting = saturate( dot ( input.Norm, vLightDir ) );

    // Load the diffuse texture and multiply by the lighting amount
    float4 cDiffuse = g_txDiffuse.Sample( samLinear, input.Tex ) * fLighting;
    cDiffuse.a = 1;

    // return diffuse
    return cDiffuse;
}
```

Реализация некоторых эффектов

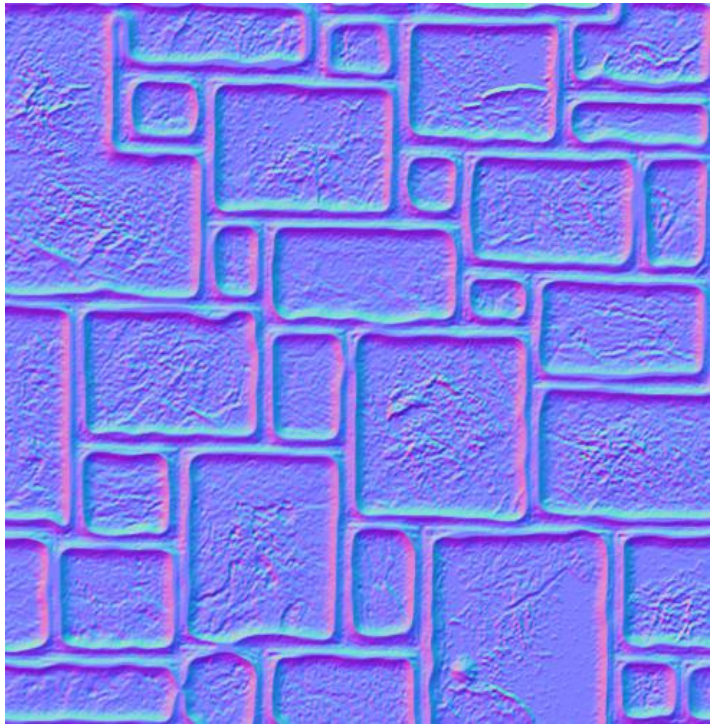


Диффузная модель освещения



Normal mapping

Normal mapping



Normal mapping. Реализация

```
struct GSPS_INPUT
{
    float4 Pos   : SV_POSITION;
    float2 Tex   : TEXCOORD1;
    float3 Light : TEXCOORD2;
};

float4 PS( GSPS_INPUT input) : SV_Target
{
    float3 cNormal = g_txNormal.Sample(samLinear, input.Tex);
    float fLighting = saturate( dot (cNormal , input.Light) );

    float4 cDiffuse = g_txDiffuse.Sample( samLinear, input.Tex ) * fLighting;
    cDiffuse.a = 1;

    // return diffuse
    return cDiffuse;
}
```

Normal mapping. Реализация

```
struct VS_INPUT
{
    float4 Pos      : POSITION;
    float3 Norm     : NORMAL;
    float2 Tex      : TEXCOORD0;
    float3 Binorm   : TEXCOORD1;
    float2 Tangent  : TEXCOORD2;
};

struct GSPS_INPUT
{
    float4 Pos   : SV_POSITION;
    float3 Light: TEXCOORD2;
    float2 Tex   : TEXCOORD1;
};

GSPS_INPUT VS( VS_INPUT input )
{
    GSPS_INPUT output;

    output.Pos = mul( input.Pos, WorldViewProj );
    output.Tex = input.Tex;
    float3 normal   = mul( input.Norm,    World);
    float3 binormal = mul( input.Binorm,  World);
    float3 tangent  = mul( input.Tangent, World);
    float3x3 tbn = float3x3(normal, binormal, tangent);

    output.Light = mul(vLightDir , tbn);

    return output;
}
```

Что почитать?

- **Introduction to 3D Game Programming with DirectX 10 [Frank D.Luna]**
- http://www.unchainedgeometry.com/jbloom/pdf/homog-coords.pdf
- http://developer.amd.com/gpu_assets/Iourcha-Yang-Pomianowski-AA(HPG2009).pdf
- GPU Gems [all]
- http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.10.8322&rep=rep1&type=pdf
- http://developer.nvidia.com/
- http://developer.amd.com/
- http://msdn.microsoft.com/en-us/directx/default
- http://developer.amd.com/media/gpu_assets/Tatarchuk-POM.pdf

Спасибо за внимание!