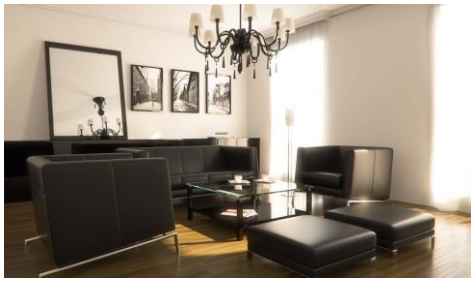
Освещение в играх и графические процессоры

к.ф.-м.н. Владимир Фролов (vfrolov@graphics.cs.msu.ru)
ВМиК МГУ, И.П.М. им. Келдыша РАН

Глобальное освещение

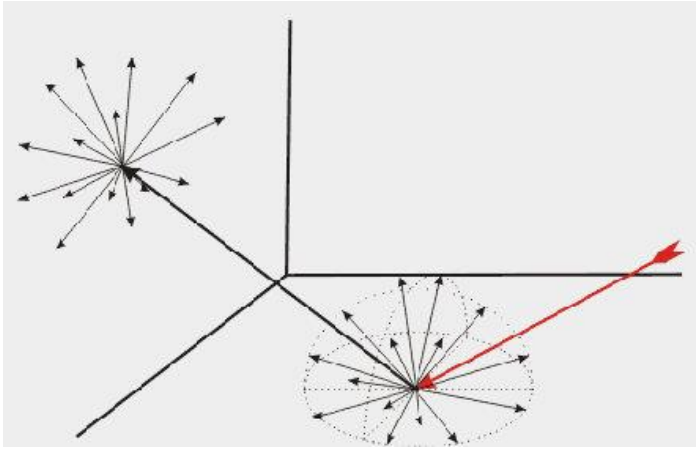


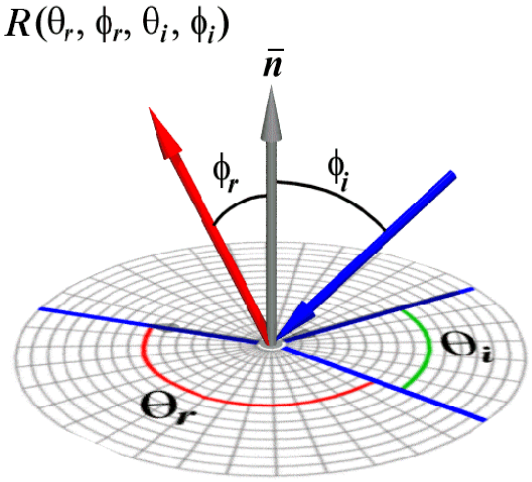
Интеграл освещенности

$$I(\varphi_r, \theta_r) = \int_{\varphi_i} \int_{\theta_i} L(\varphi_i, \theta_i) R(\varphi_i, \theta_i, \varphi_r, \theta_r) \cos(n, l_{\varphi_i, \theta_i}) d\varphi_i d\theta_i$$

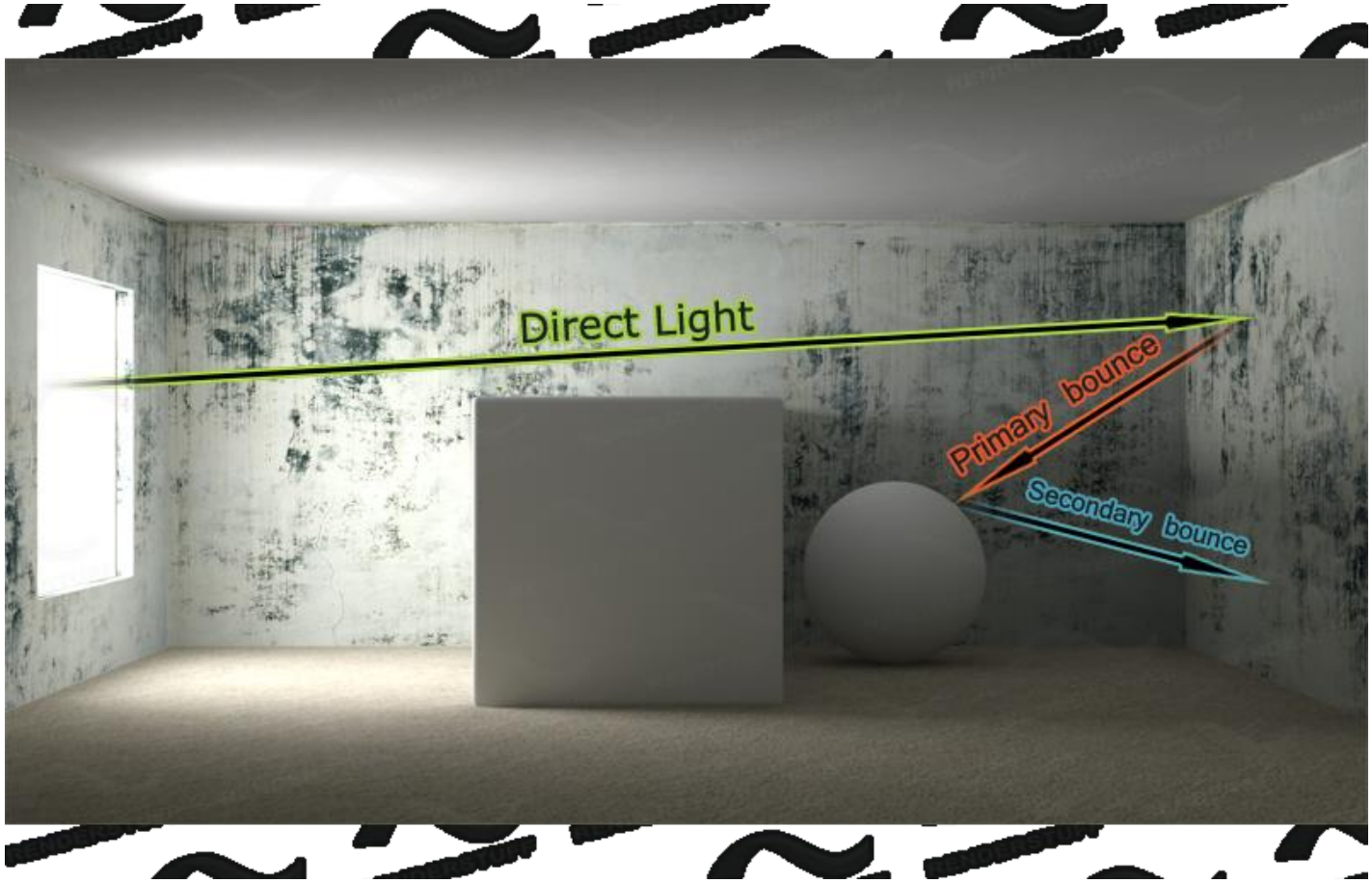
$n - нормаль$

$l_{\varphi_i, \theta_i} - вектор$





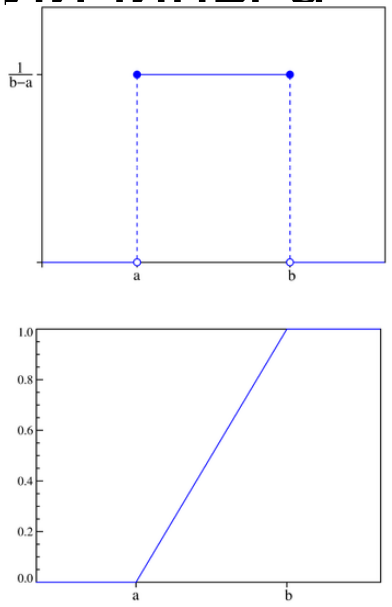
Глобальное освещение



$$I(\varphi_r, \theta_r) = \int_{\varphi_i} \int_{\theta_i} L(\varphi_i, \theta_i) R(\varphi_i, \theta_i, \varphi_r, \theta_r) \cos(n, l_{\varphi_i, \theta_i}) d\varphi_i d\theta_i$$

Метод Монте-Карло

- u – случ. Величина, равн. Распред. на [a,b]
- p(u) – плотность вероятности случ. величины u
- Тогда f(u) – тоже случ. величина.



$$Ef(u) = \int_a^b f(x)\,p(x)\,dx$$

$$p(x) = \frac{1}{b-a}$$

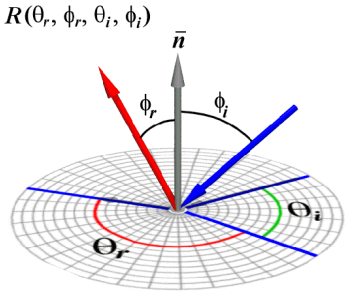
$$\int_a^b f(x)dx = (b-a)Ef(u) \approx \frac{b-a}{N}\sum f(u_i)$$

Метод Монте-Карло

$$I(\varphi_r, \theta_r) = \int_{\varphi_i} \int_{\theta_i} L(\varphi_i, \theta_i) R(\varphi_i, \theta_i, \varphi_r, \theta_r) \cos(n, l_{\varphi_i, \theta_i}) d\varphi_i d\theta_i$$

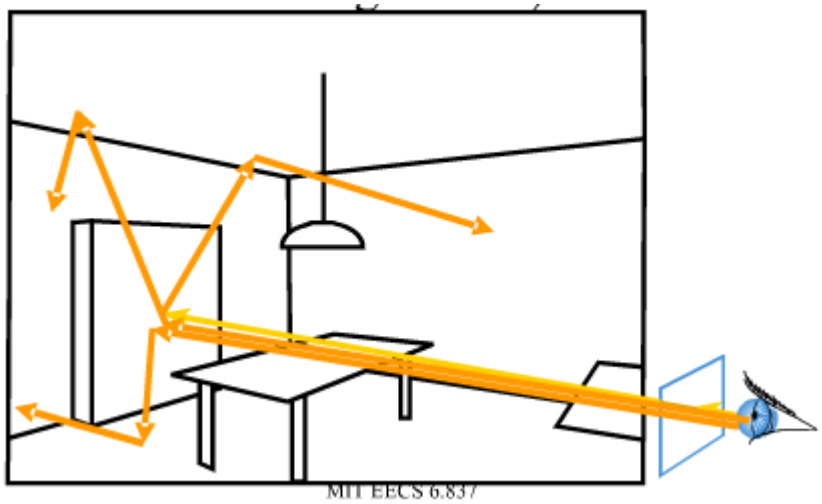
$$\int_a^b f(x)dx = (b-a)Ef(u) \approx \frac{b-a}{N} \sum f(u_i)$$

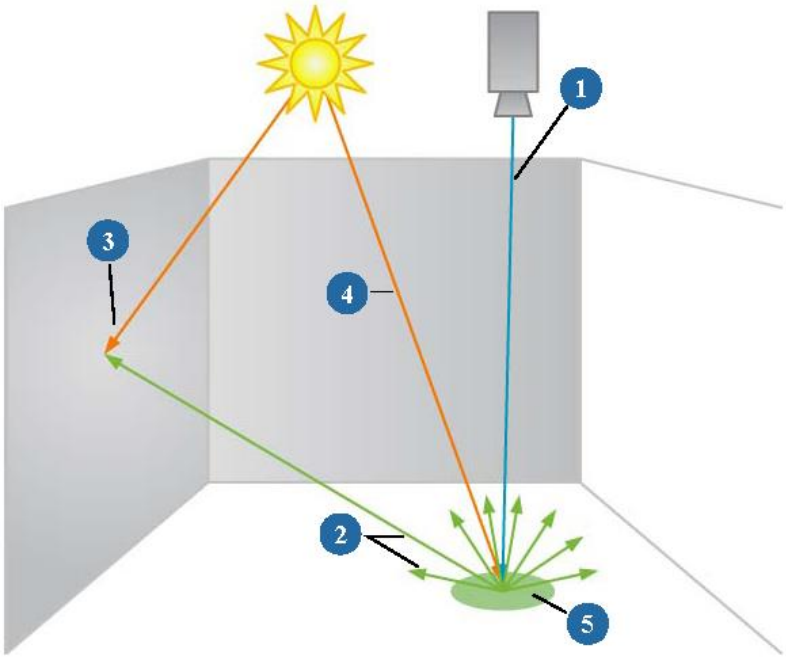
$$I(\varphi_r, \theta_r) \approx \frac{\sum_i L(\varphi_i, \theta_i) R(\varphi_i, \theta_i, \varphi_r, \theta_r) \cos(n, l_{\varphi_i, \theta_i})}{N}$$



Трассировка путей

$$I(\varphi_r, \theta_r) \approx \frac{\sum_i L(\varphi_i, \theta_i) R(\varphi_i, \theta_i, \varphi_r, \theta_r) \cos(n, l_{\varphi_i, \theta_i})}{N}$$





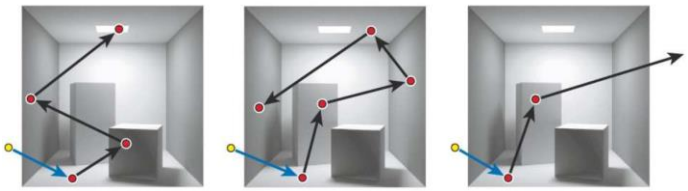
$$I(\varphi_r, \theta_r) \approx \frac{\sum_i L(\varphi_i, \theta_i) R(\varphi_i, \theta_i, \varphi_r, \theta_r) \cos(n, l_{\varphi_i, \theta_i})}{N}$$

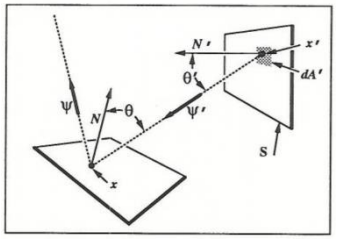
Исходные параметры: ray - Луч, depth - глубина рекурсии

Результат: Монте-Карло выборка значения освещенности

Function *TracePath(ray: Ray; depth : Integer) : Integer* **is**

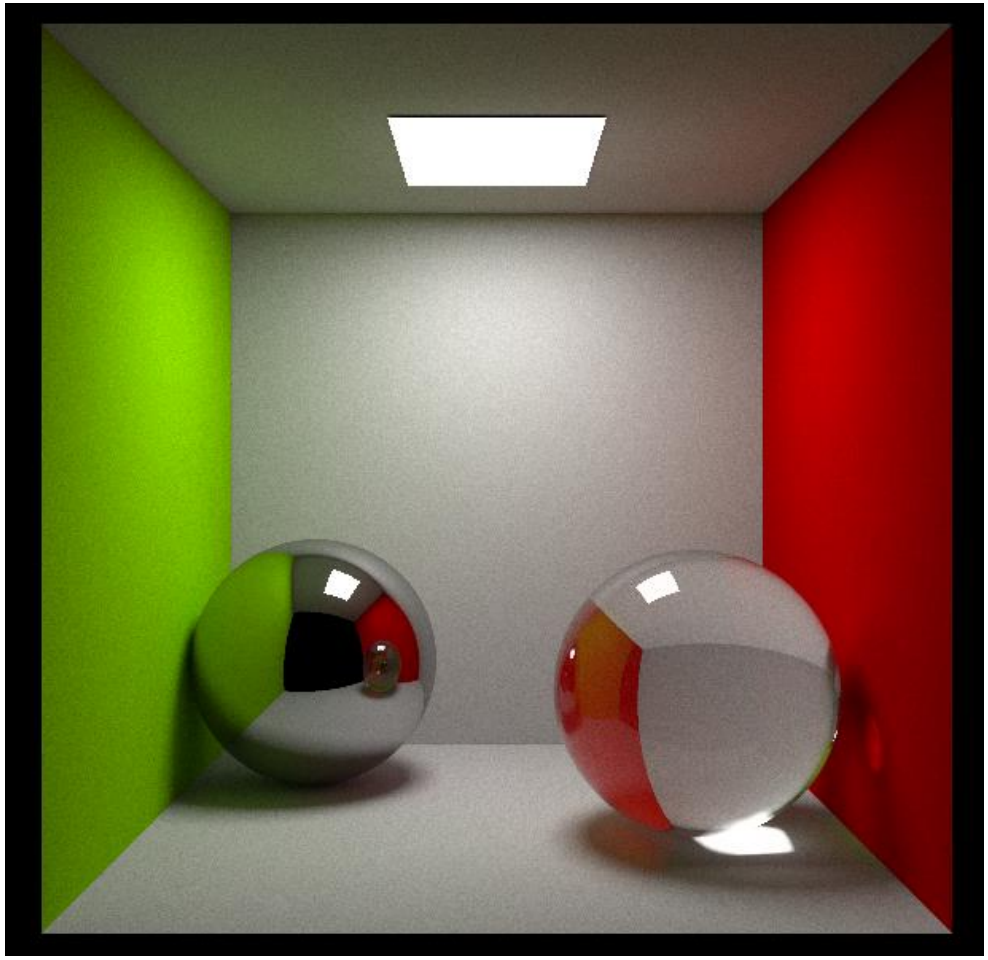
```
if depth == MAX_DEPTH then
    return Black;
end
hit ← RaySceneIntersection(ray)
if not hit.exists then
    return Black;
end
m ← hit.material
if m.IsLight then
    return m.emittance;
end
newRay.O ← hit.pos
newRay.D ← RandomUnitVectorInHemisphereOf(hit.norm)
cosTheta ← dot(newRay.D, hit.norm)
BRDF ← m.reflectance * cosTheta
return BRDF * TracePath(newRay, depth + 1) ;
end
```





Алгоритм 2: Простейший вид Монте-Карло трассировки путей. I

Трассировка путей



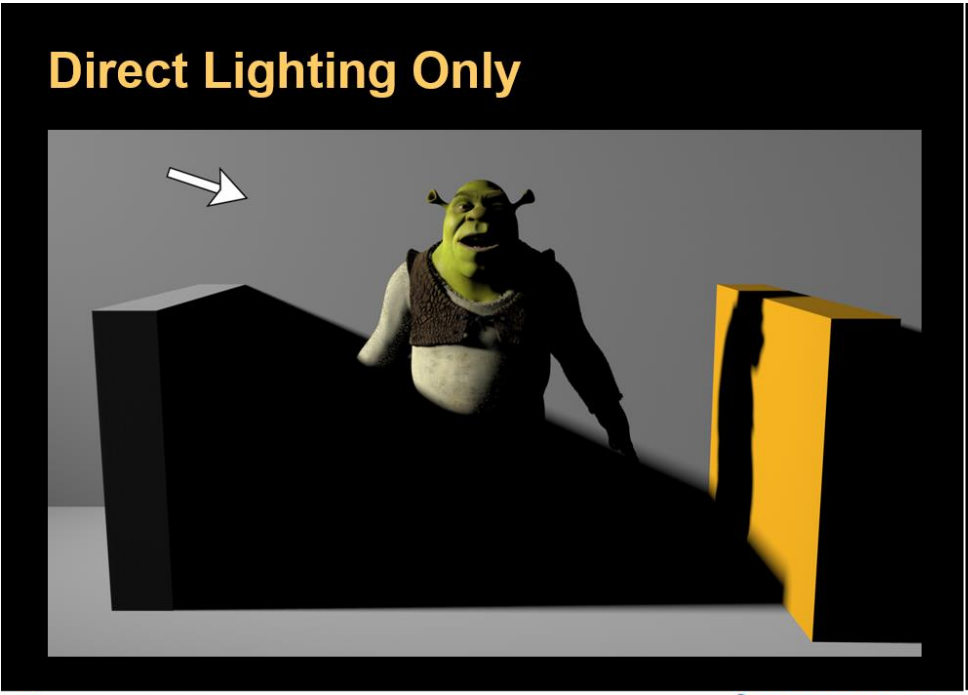

кузин павел
hydra renderer

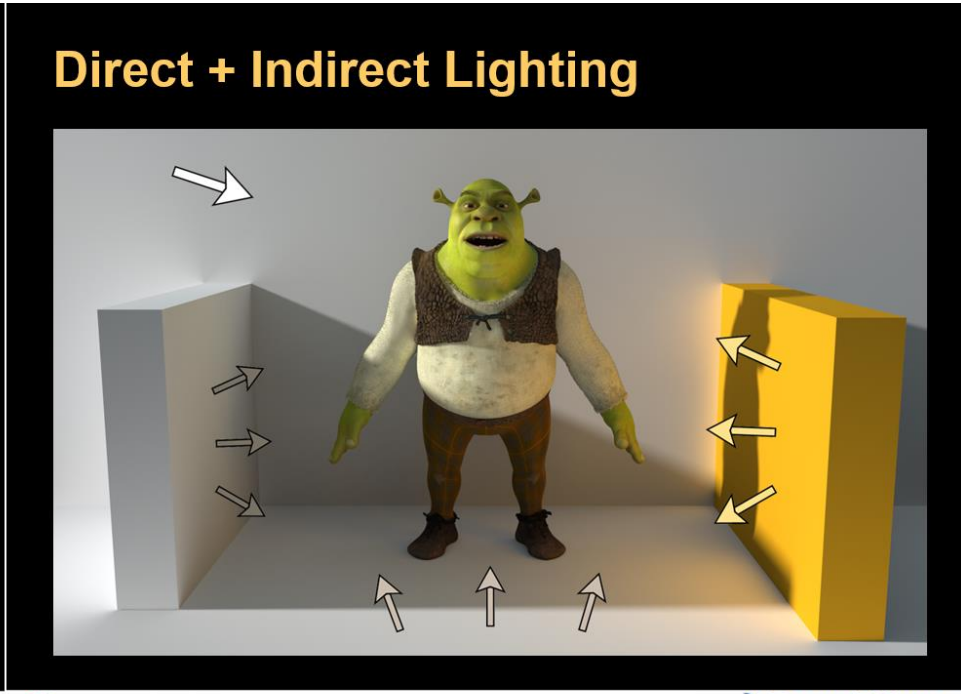
Глобальное освещение в играх

- От глобального освещения к локальному
- Приближённые методы глобального:
- Карты освещённости
- Сферические гармоники
  - Они же “volumetric lightmaps”
  - Оно же “radiance cache”
- Излучательность
- Light Propagation Volumes
- Voxel Cone Tracing
- …

Локальное освещение

$$I(\varphi_r, \theta_r) = \int_{\varphi_i} \int_{\theta_i} L(\varphi_i, \theta_i) R(\varphi_i, \theta_i, \varphi_r, \theta_r) \cos(n, l_{\varphi_i, \theta_i}) d\varphi_i d\theta_i$$

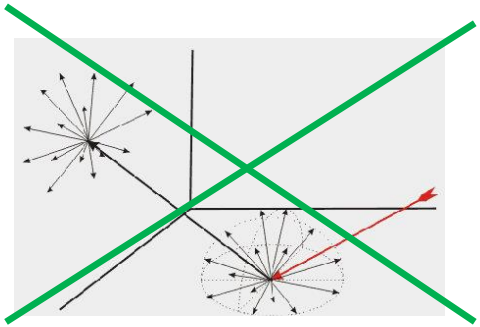


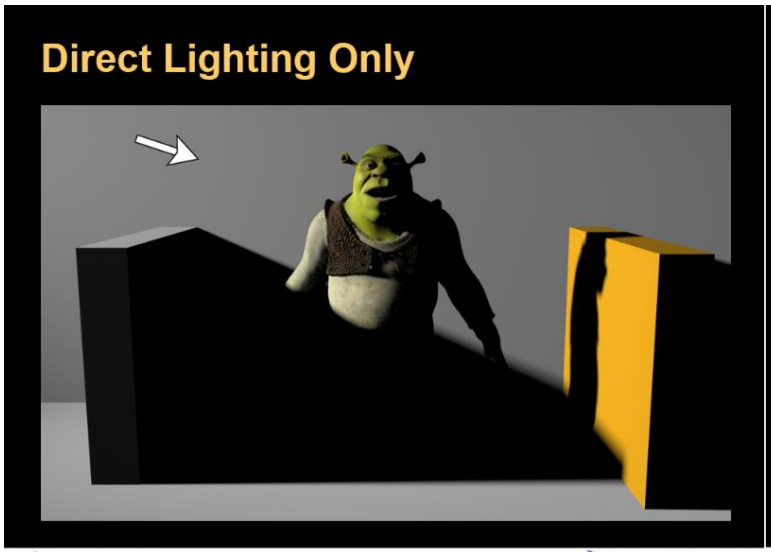


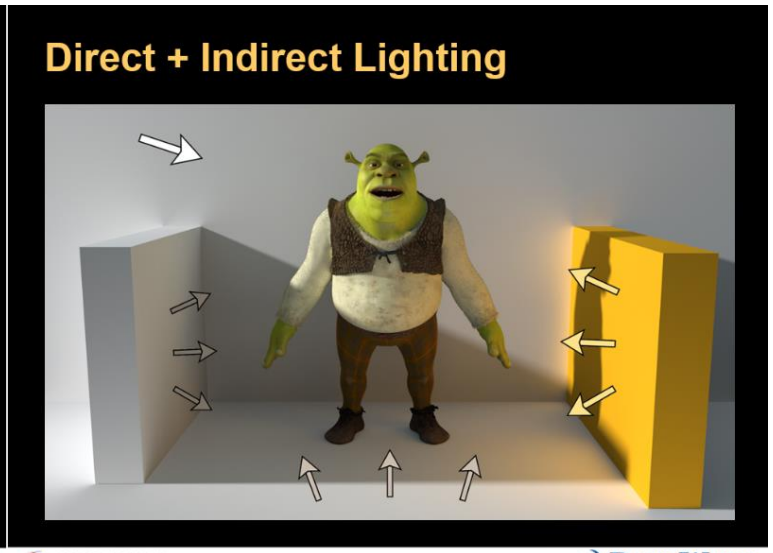
Локальное освещение

$$I(\varphi_r, \theta_r) = \int_{\varphi_i} \int_{\theta_i} L(\varphi_i, \theta_i) R(\varphi_i, \theta_i, \varphi_r, \theta_r) \cos(n, l_{\varphi_i, \theta_i}) d\varphi_i d\theta_i$$

- L – яркость источника
- R – модель материала
- Тень нужно считать отдельно







Локальное освещение

```glsl
//Blinn-Phong reflection model wiath attenuation; Fragment Shader
//To keep it simple didn't add ambient and emissive lights;
//only diffuse and specular with white intensity
#version 330
layout(location = 0) out vec4 out_color;

uniform vec3 light_position;
uniform vec3 eye_position;

uniform int material_shininess;

uniform float material_kd;
uniform float material_ks;

//attenuation coefficients
uniform float att_kC;
uniform float att_kL;
uniform float att_kQ;

in vec3 world_pos;
in vec3 world_normal;
```

```glsl
vec3 L = normalize( light_position - world_pos);//light direction
vec3 V = normalize( eye_position - world_pos);//view direction

float LdotN = max(0, dot(L,world_normal));

float diffuse = material_kd * LdotN;

//attenuation
float d = distance(light_position, world_pos);
float att = 1.0 / (att_kC + d * att_kL + d*d*att_kQ);

float specular = 0;

if(LdotN > 0.0)
{
vec3 H = normalize(L + V );//Halfway(Blinn-Phong)
specular = material_kd * pow(max(0, dot(H, world_normal)), material_shininess);
}

float light = att * diffuse + att * specular;
```

http://in2gpu.com/2014/06/19/lighting-vertex-fragment-shader/

$$I(\varphi_r,\theta_r)=\int_{\varphi_i}\int_{\theta_i} L(\varphi_i,\theta_i)R(\varphi_i,\theta_i,\varphi_r,\theta_r)\cos(n,l_{\varphi_i,\theta_i})d\varphi_i d\theta_i$$

```glsl
//Blinn-Phong reflection model wiath attenuation; Fragment Shader
//To keep it simple didn't add ambient and emissive lights;
//only diffuse and specular with white intensity
#version 330
layout(location = 0) out vec4 out_color;

uniform vec3 light_position;
uniform vec3 eye_position;

uniform int material_shininess;

uniform float material_kd;
uniform float material_ks;

//attenuation coefficients
uniform float att_kC;
uniform float att_kL;
uniform float att_kQ;

in vec3 world_pos;
in vec3 world_normal;
```

```glsl
vec3 L = normalize( light_position - world_pos);//light direction
vec3 V = normalize( eye_position - world_pos);//view direction

float LdotN = max(0, dot(L,world_normal));

float diffuse = material_kd * LdotN;

//attenuation
float d = distance(light_position, world_pos);
float att = 1.0 / (att_kC + d * att_kL + d*d*att_kQ);

float specular = 0;

if(LdotN > 0.0)
{
vec3 H = normalize(L + V );//Halfway(Blinn-Phong)
specular = material_kd *pow(max(0, dot(H, world_normal)), material_shininess);
}

float light = att * diffuse + att * specular;
```

1.0

http://in2gpu.com/2014/06/19/lighting-vertex-fragment-shader/

Полный поток Ф; Вт;
Flux;

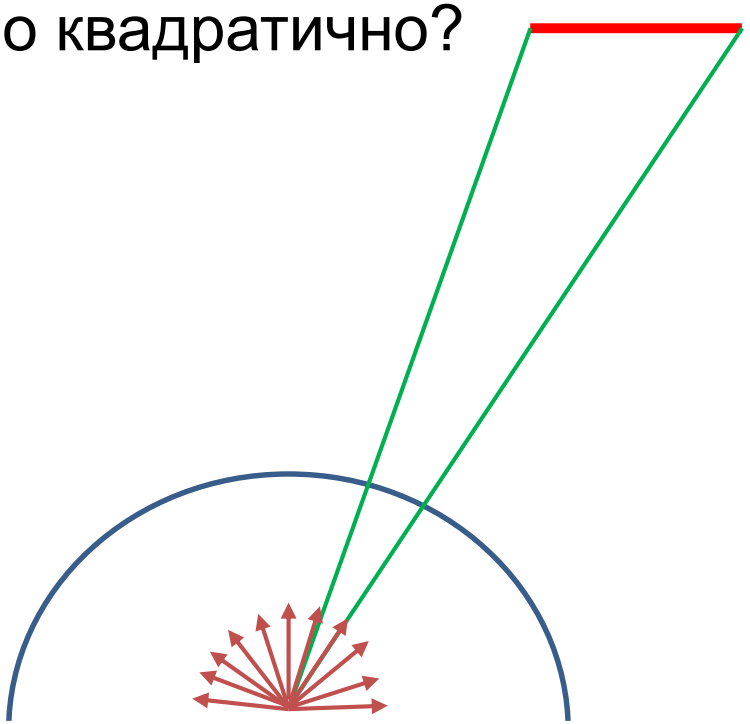
Сила света I; Вт / Ст;
Radiant intensity;

Освещённость E; Вт/м^2;
Irradiance;

Яркость L; Вт/(Ст*м^2);
Radiance;

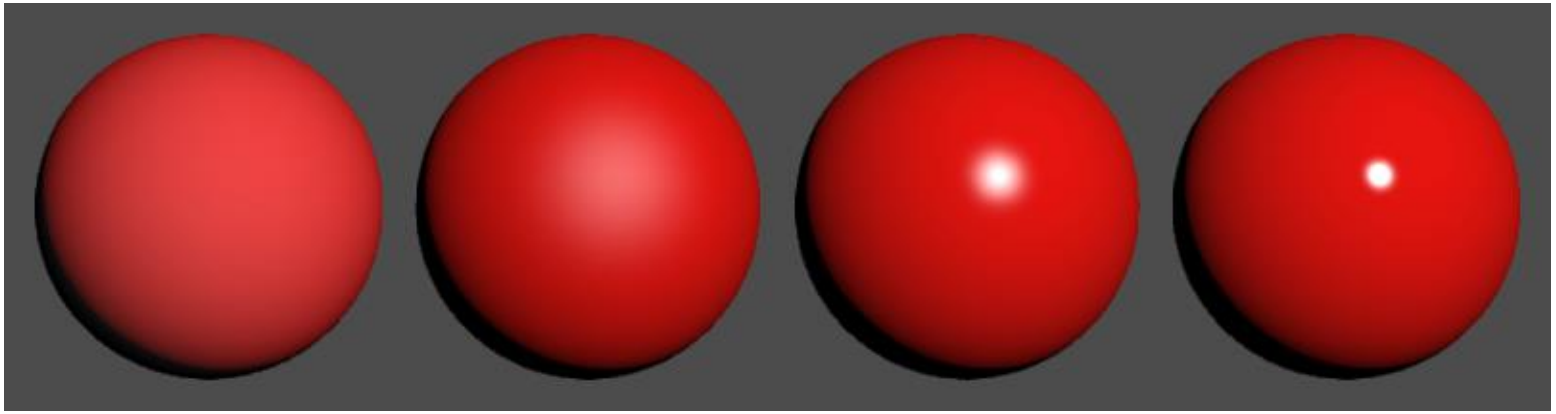
Локальное освещение

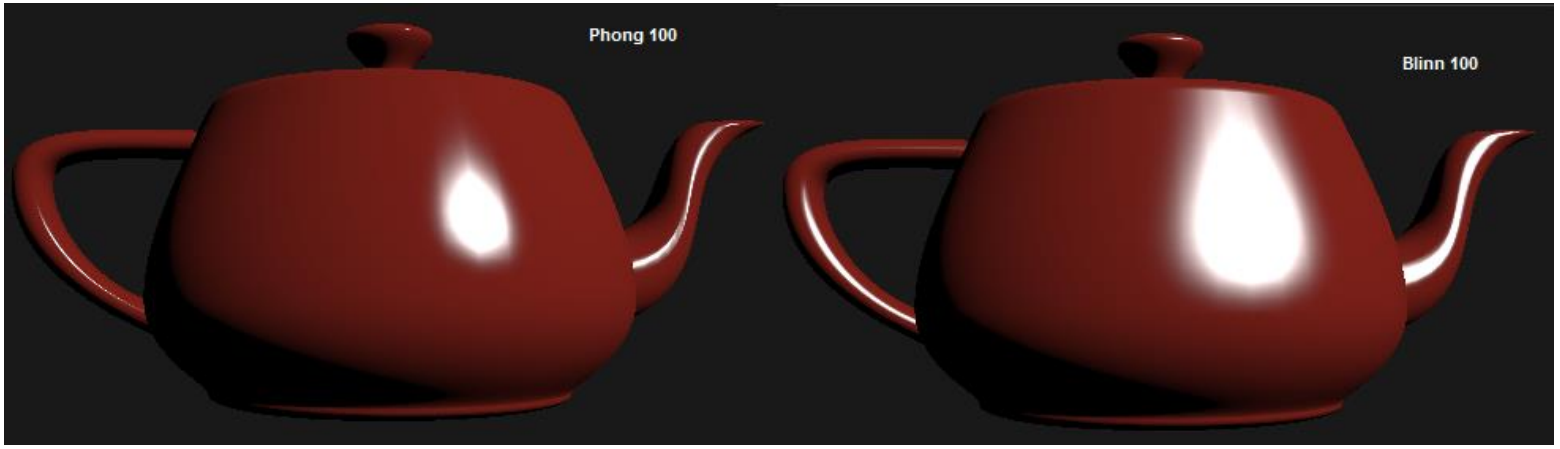
- Почему свет затухает в локальных моделях?
- Почему именно квадратично?



$$I(\varphi_r, \theta_r) = \int_{\varphi_i} \int_{\theta_i} L(\varphi_i, \theta_i) R(\varphi_i, \theta_i, \varphi_r, \theta_r) \cos(n, l_{\varphi_i, \theta_i}) d\varphi_i d\theta_i$$

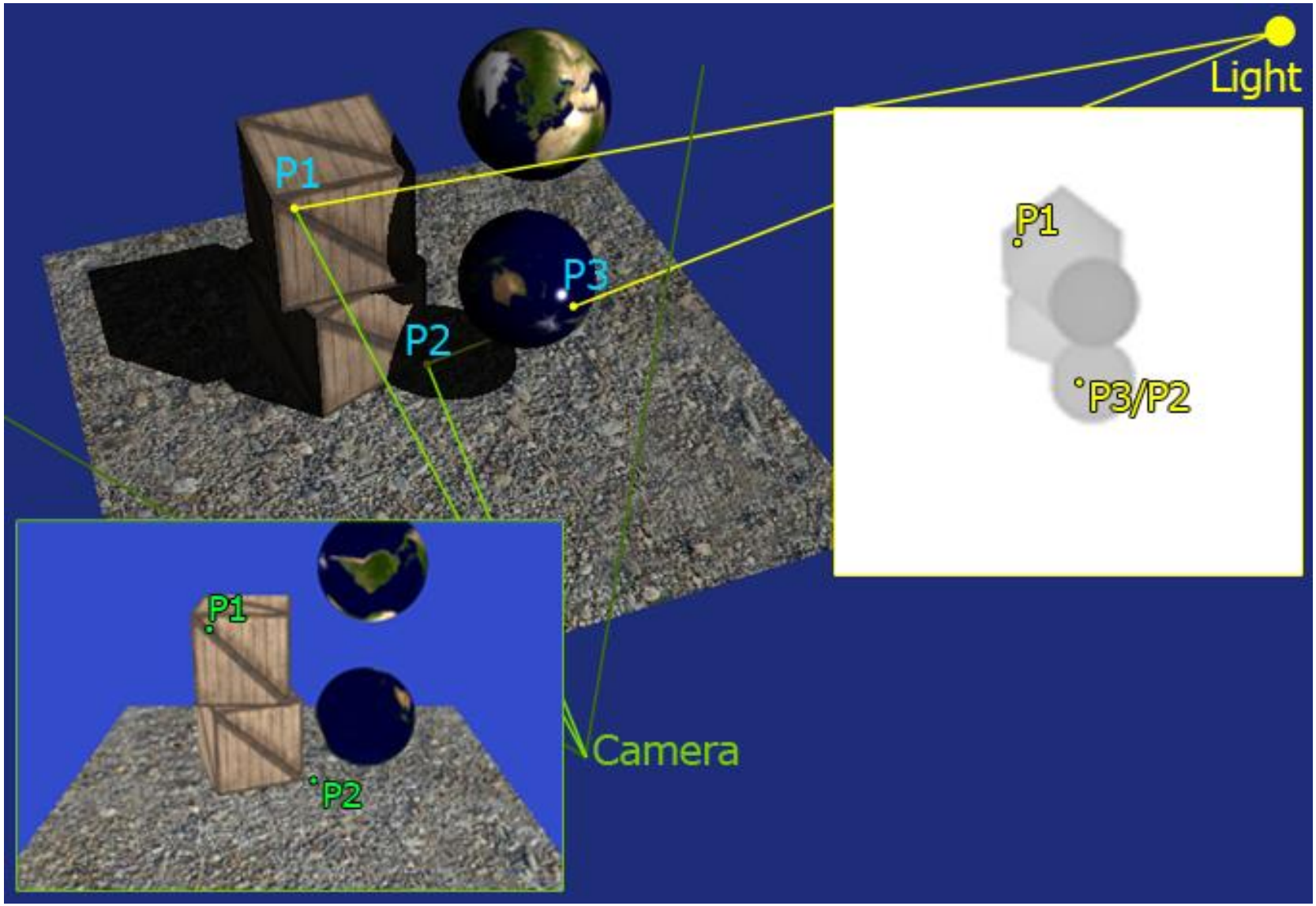
Локальное освещение





$$I(\varphi_r, \theta_r) = \int_{\varphi_i} \int_{\theta_i} L(\varphi_i, \theta_i) R(\varphi_i, \theta_i, \varphi_r, \theta_r) \cos(n, l_{\varphi_i, \theta_i}) d\varphi_i d\theta_i$$

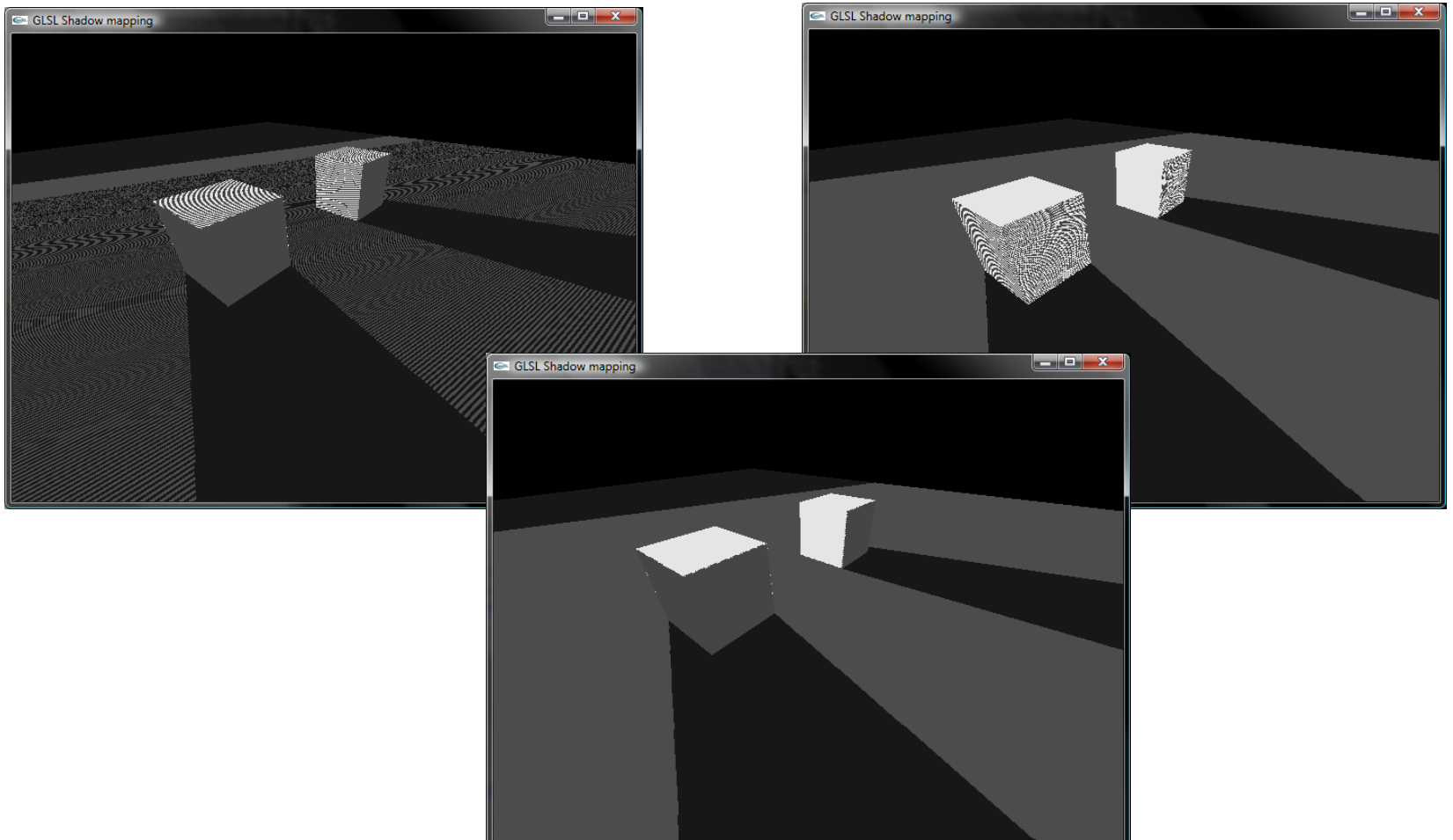
Карты теней



Карты теней (артефакты)

- Z-fighting

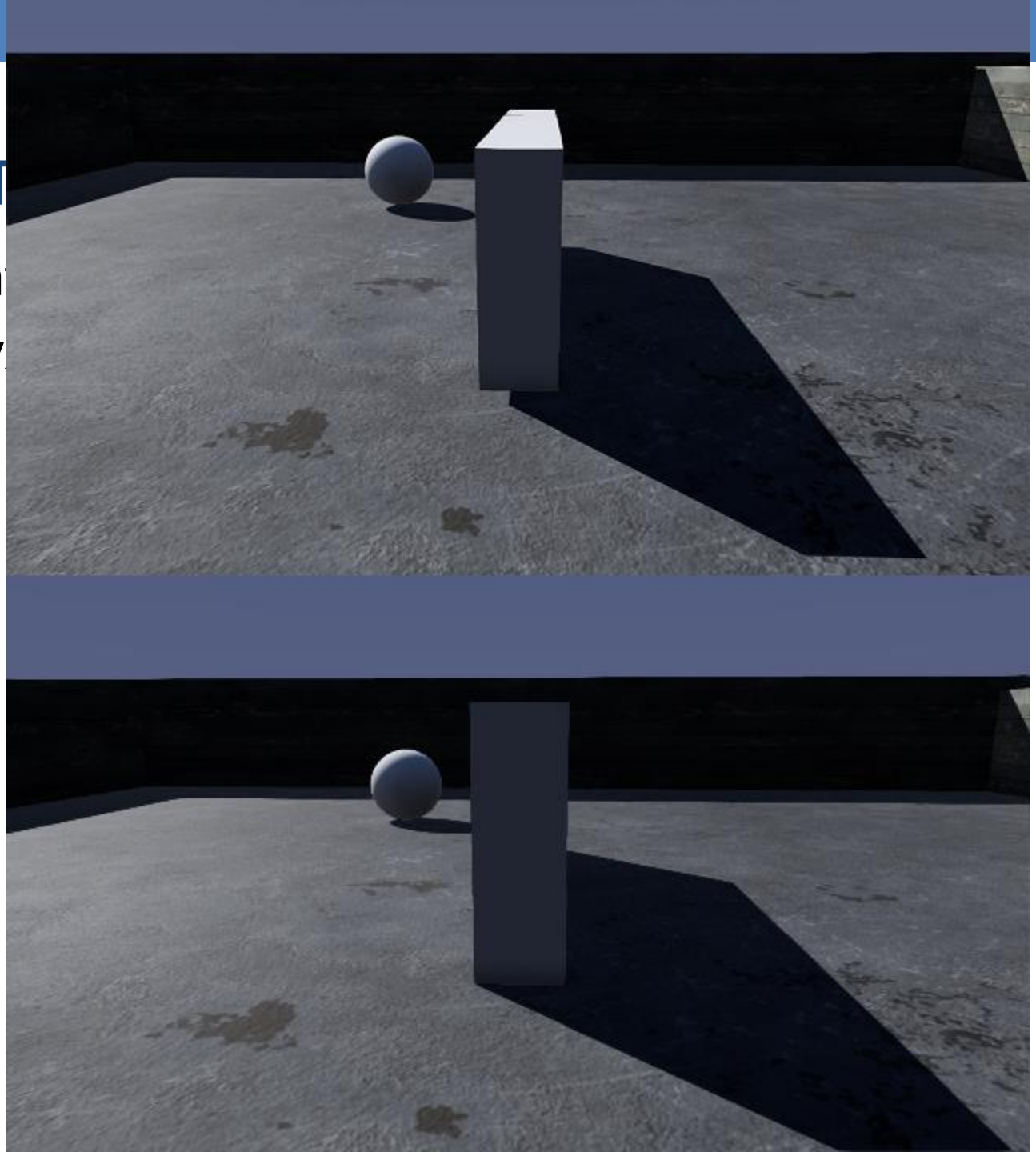
✓ =>нужно добавить смещение



Карт

- Z-figh

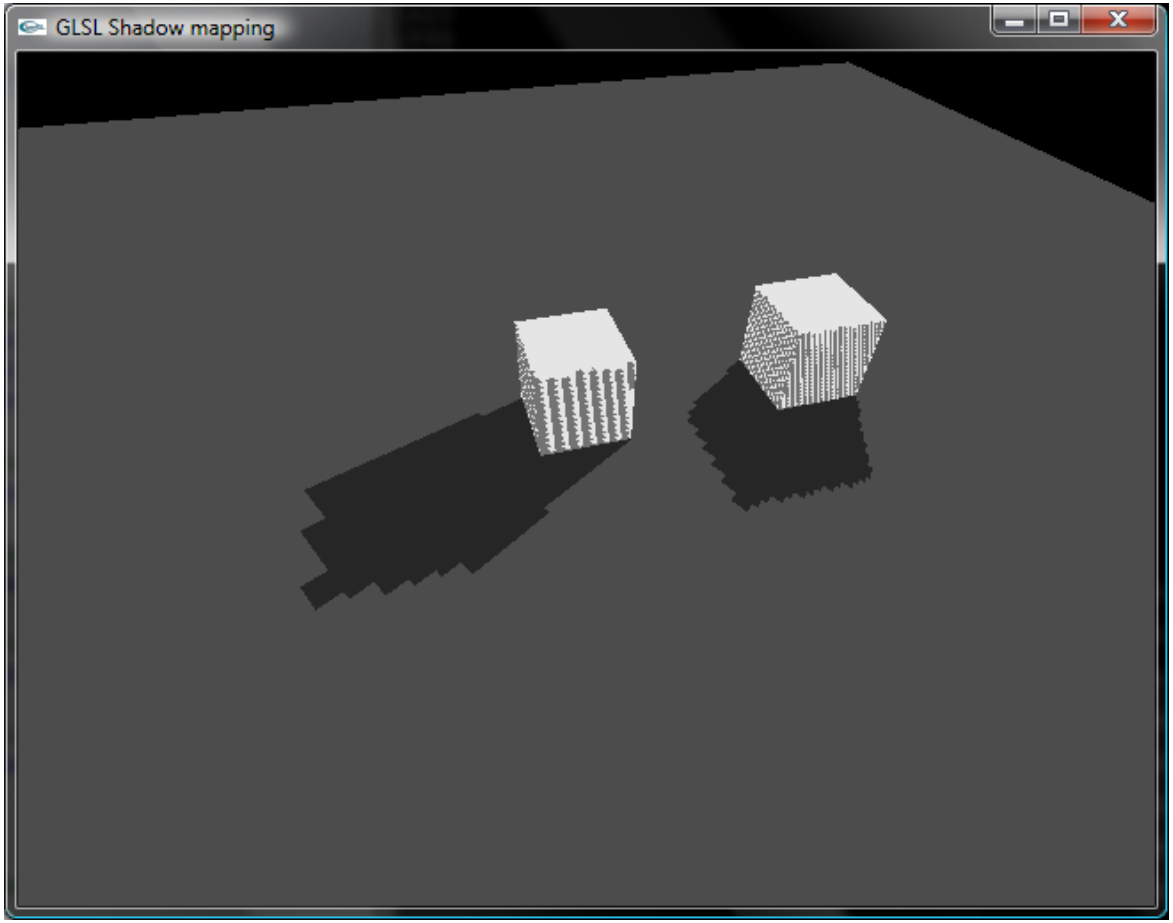
✓ =>ну



Карты теней (артефакты)

- Алиасинг

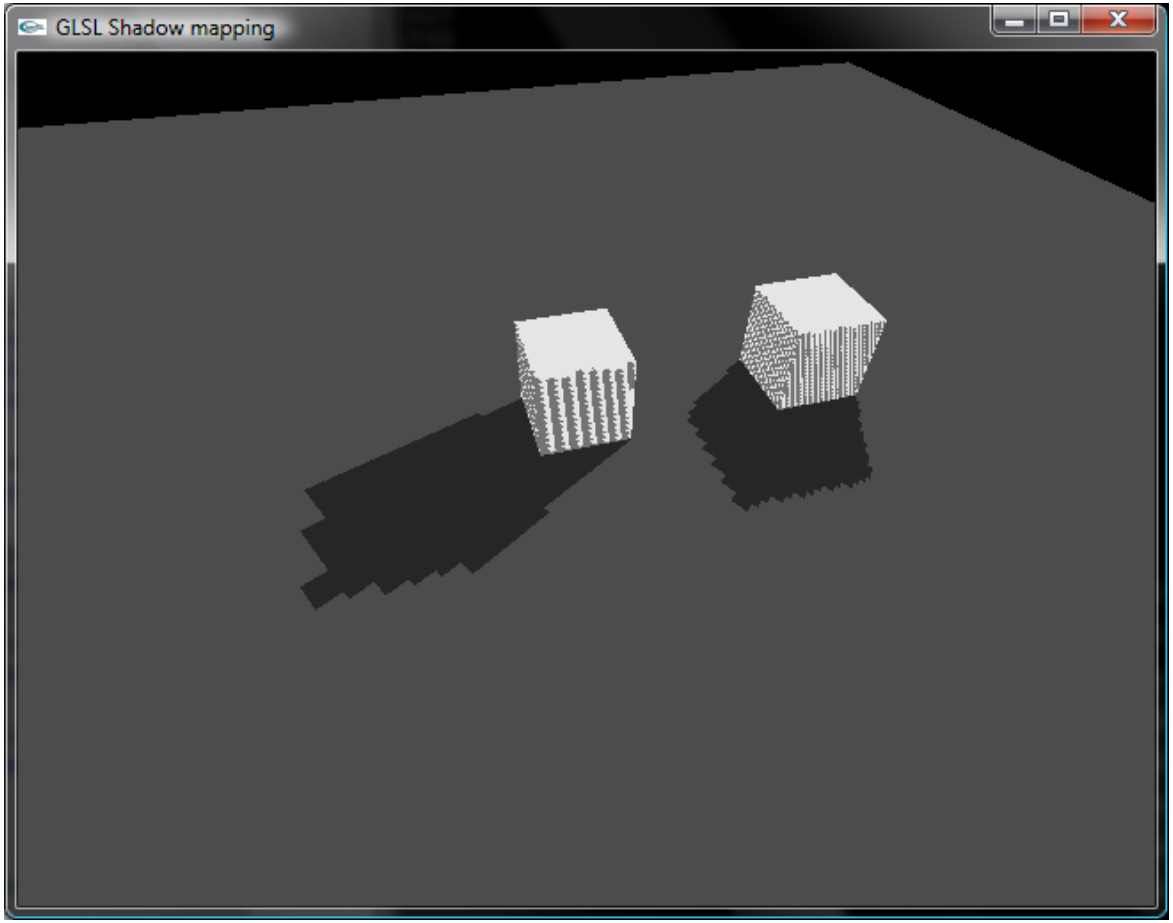
✓ => что нужно сделать?



Карты теней (артефакты)

- Алиасинг

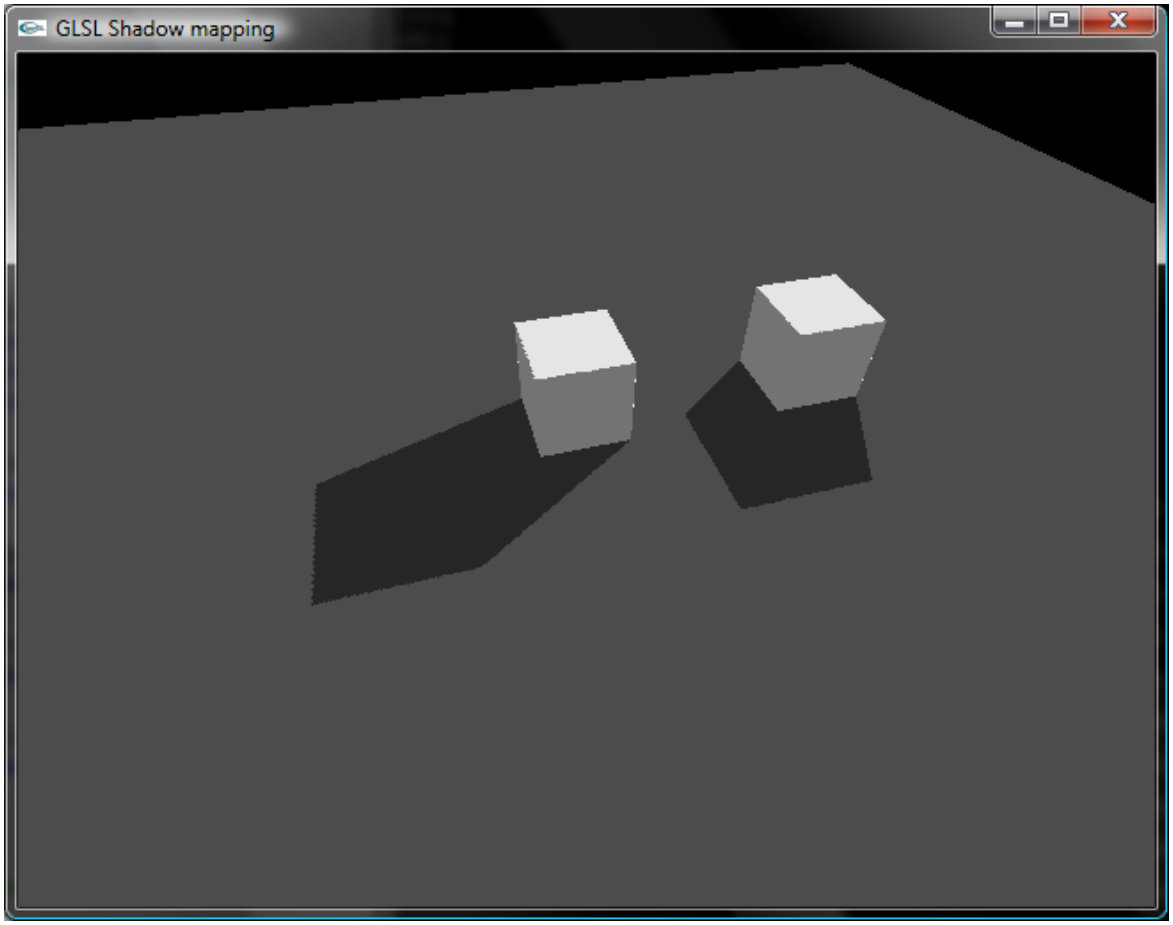
✓ => что нужно сделать?



Карты теней (артефакты)

- Алиасинг

✓ => можно увеличить разрешение, но !!!!!!!

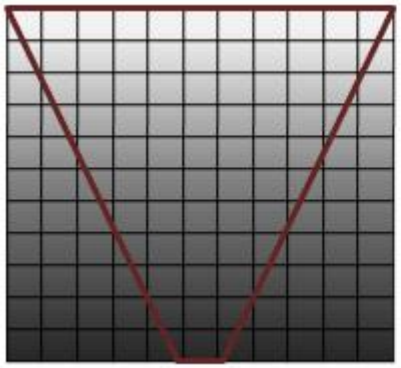


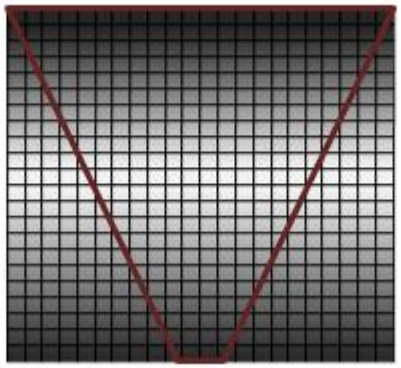

FPS: 91

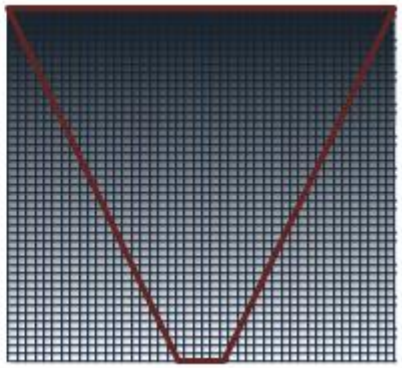

FPS: 128

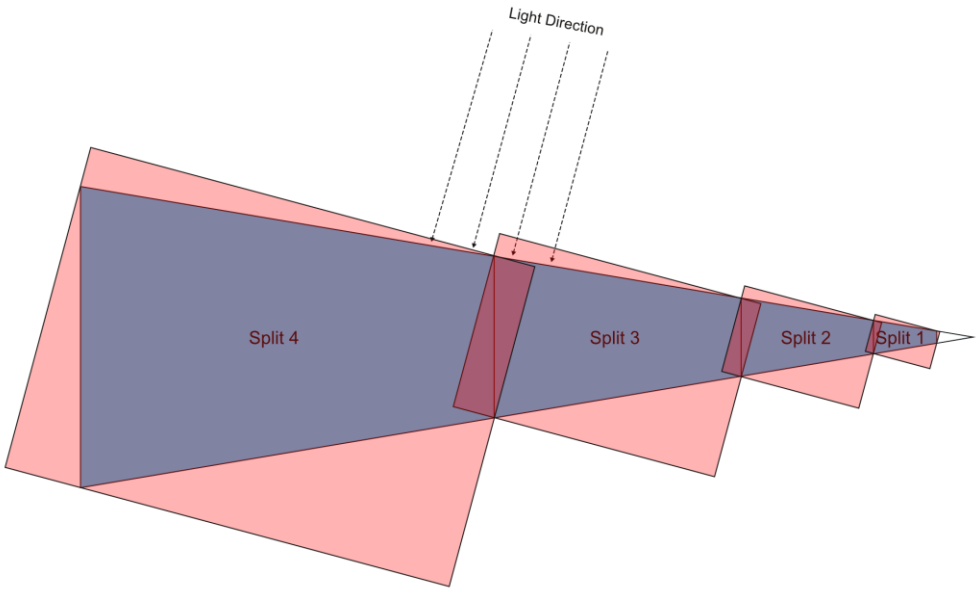
Карты теней (артефакты)

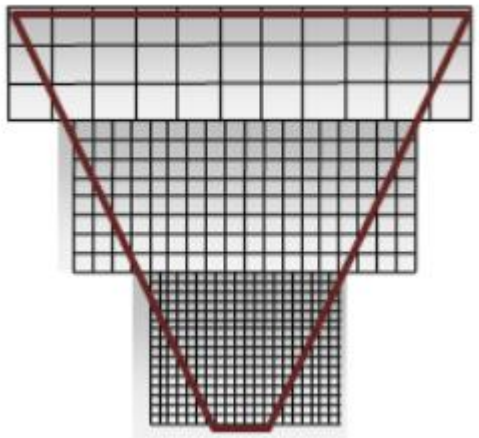
- Алиасинг









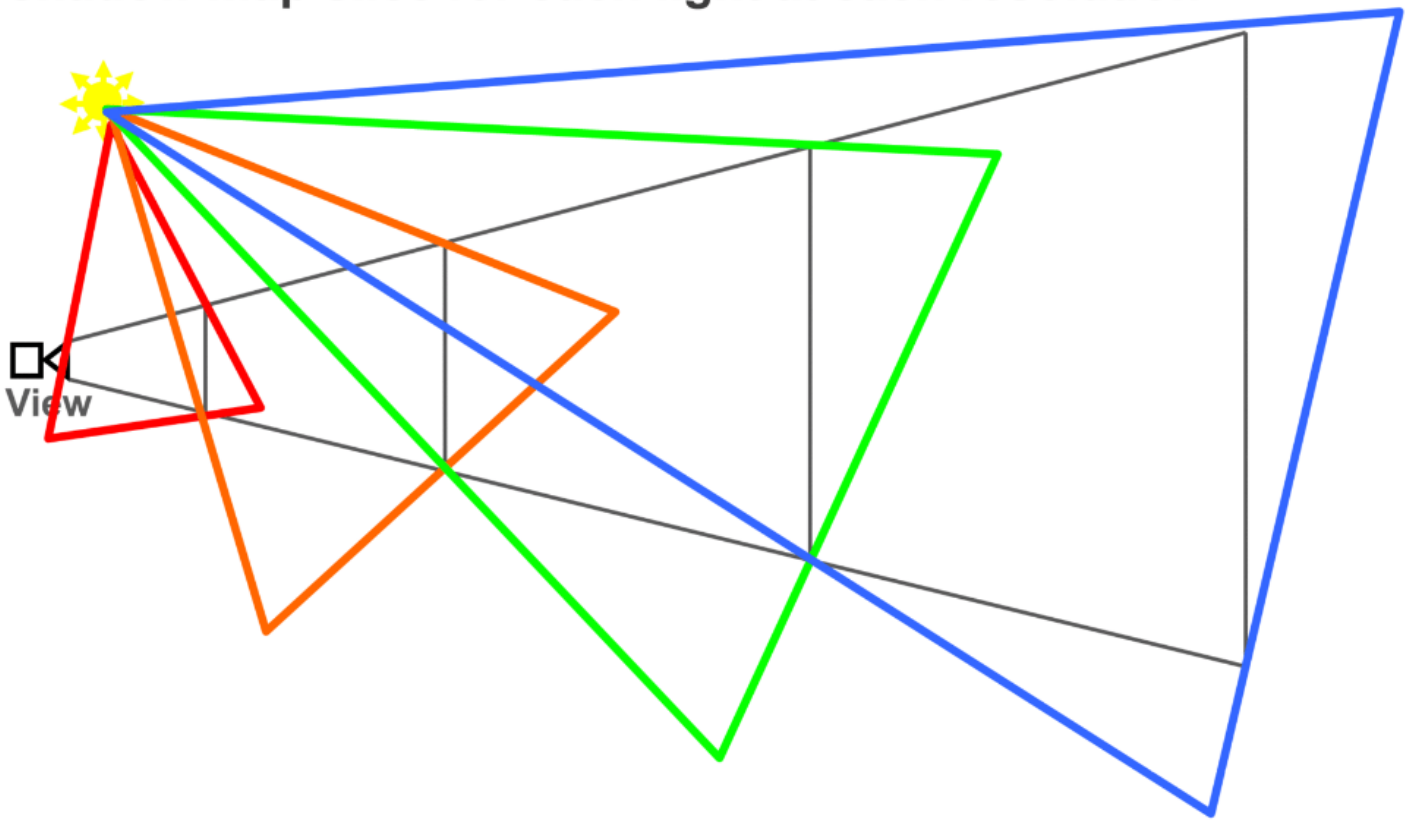


Карты теней (артефакты)

Cascaded Shadow Maps

State of the art non-ray tracing technique

- **Render a shadow map slice for each light at each resolution**

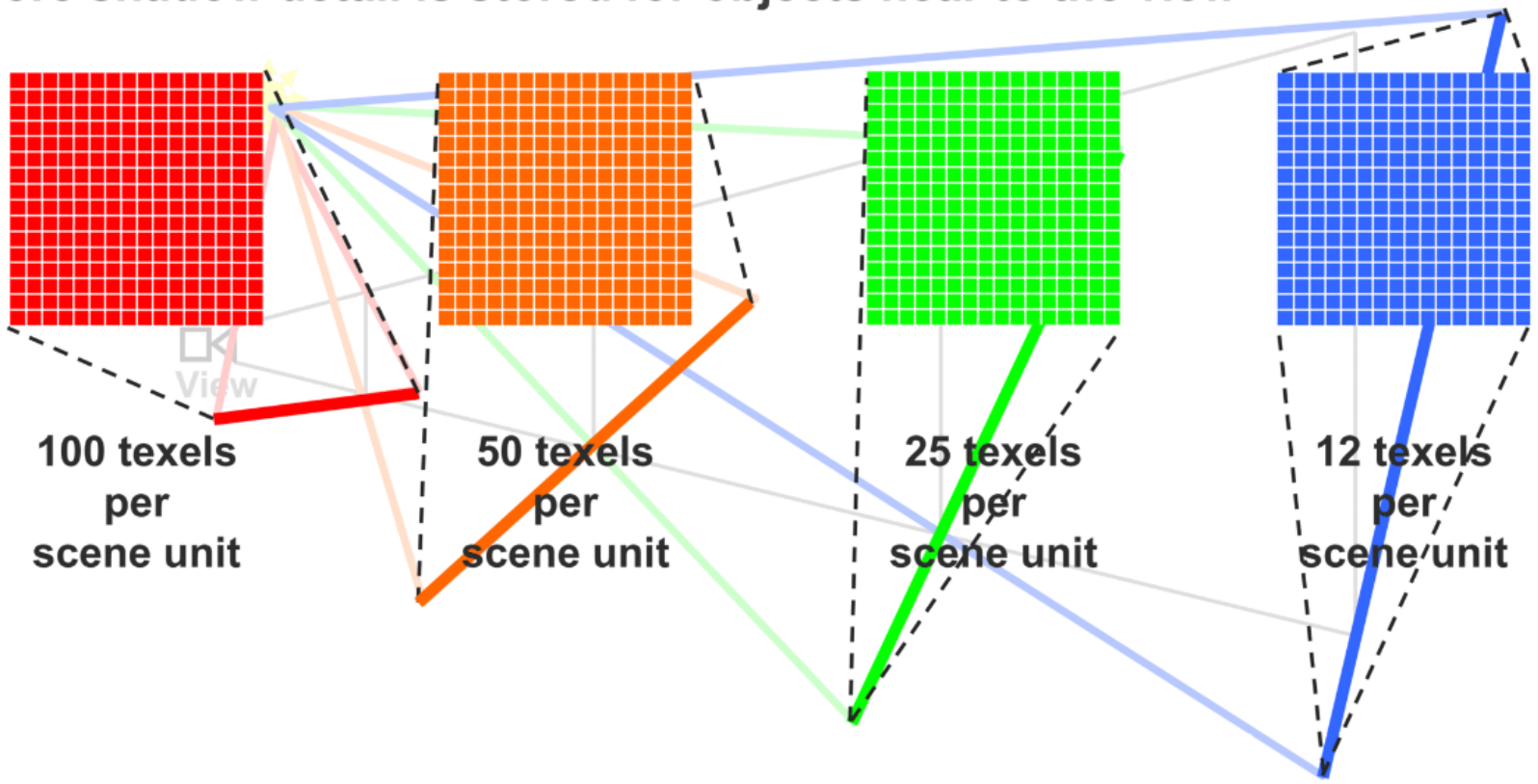


Карты теней (артефакты)

Cascaded Shadow Maps

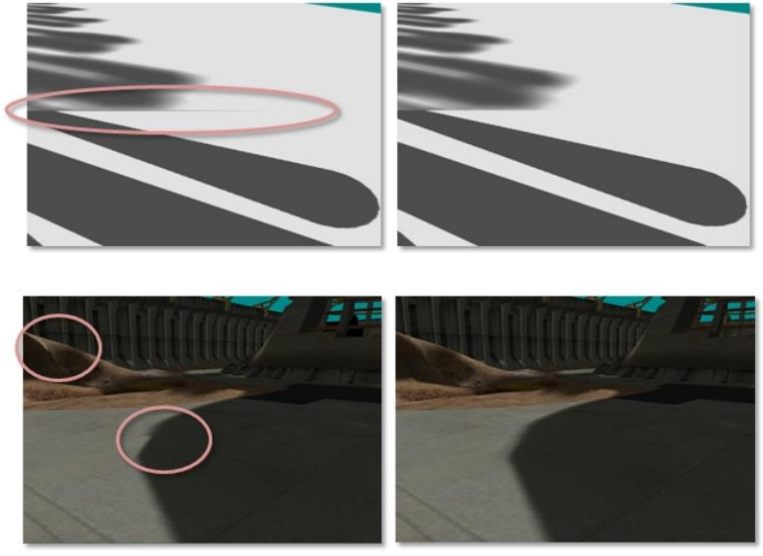
State of the art non-ray tracing technique

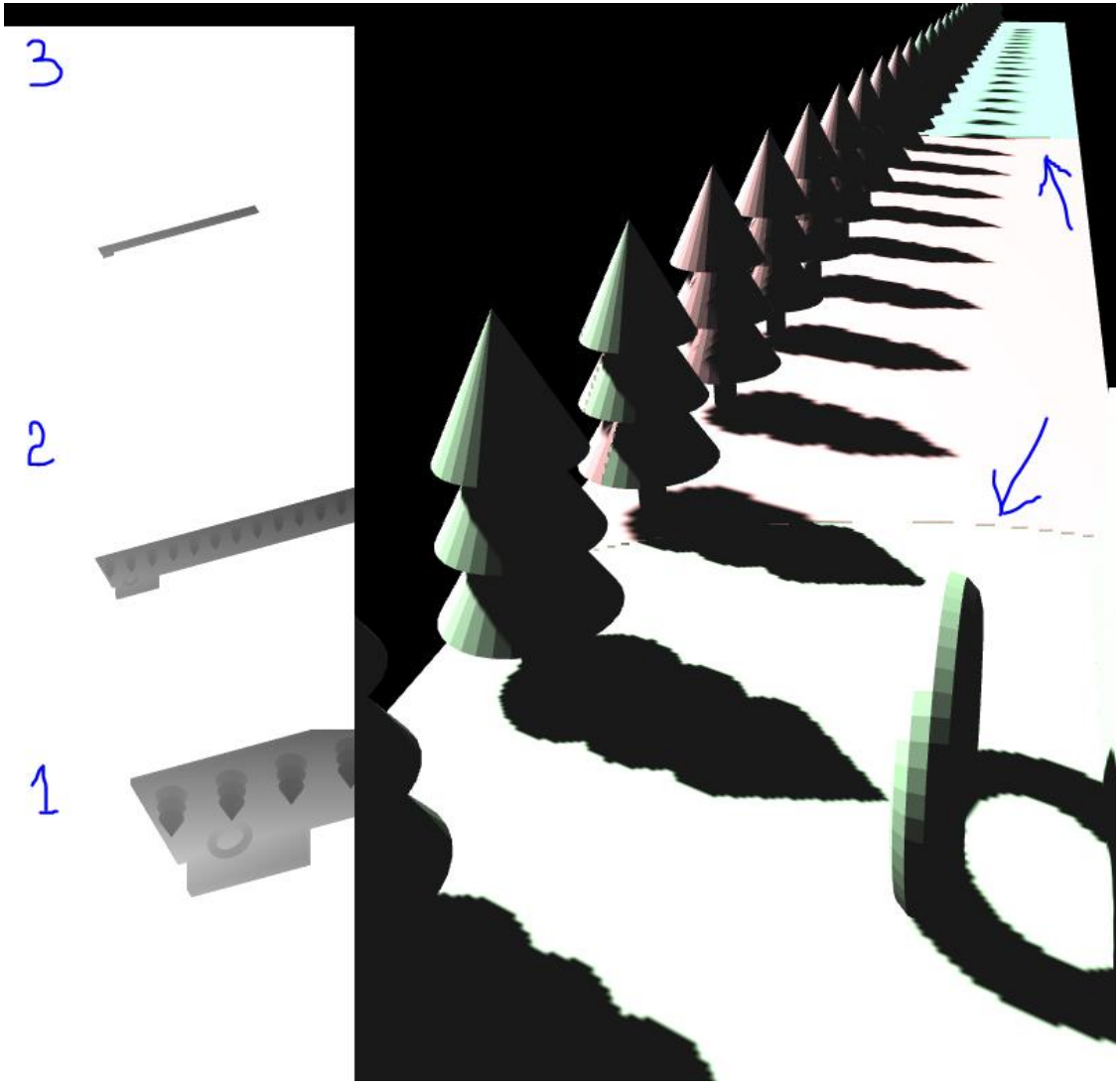
- More shadow detail is stored for objects near to the view



Карты теней (артефакты)

- Каскадные карты теней

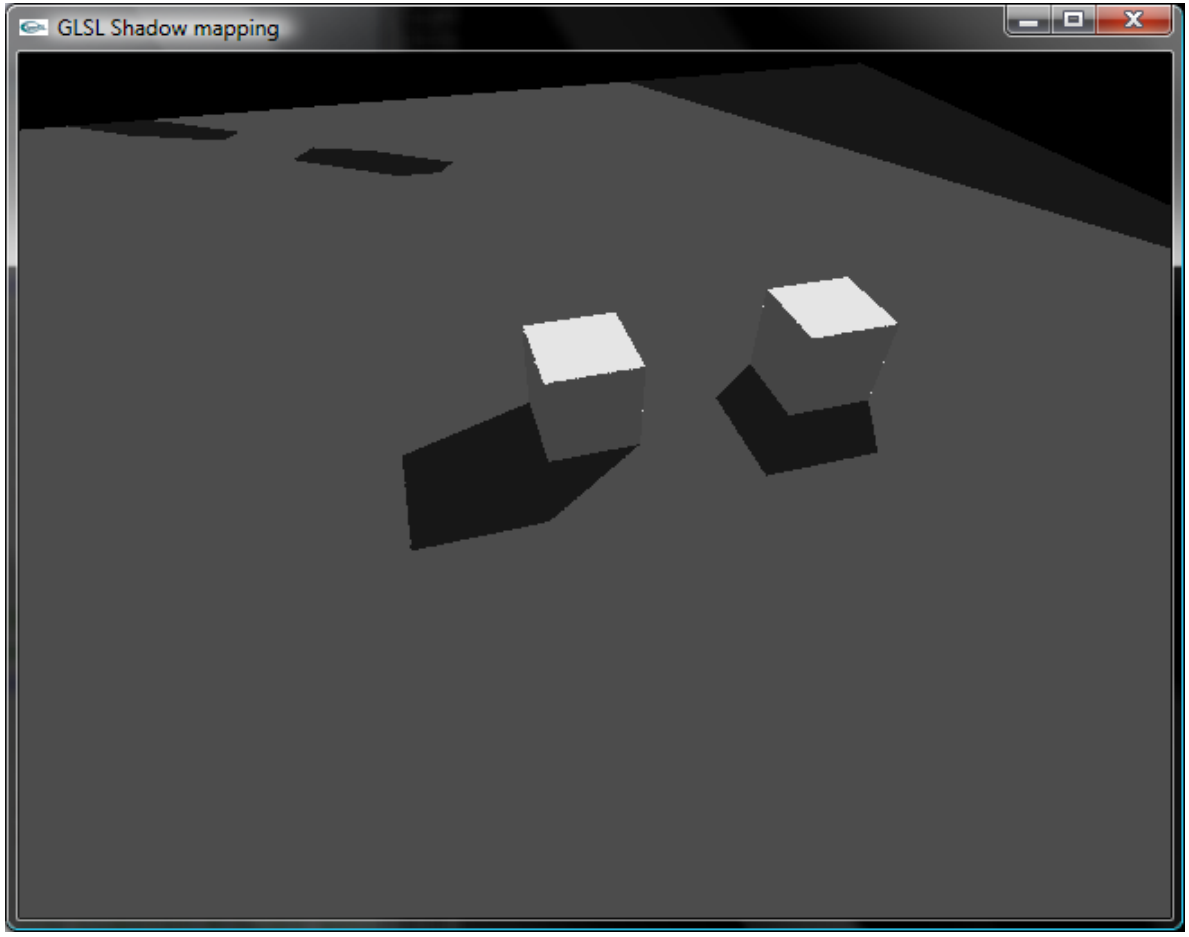




Карты теней (артефакты)

- Тени за пределами light frustum

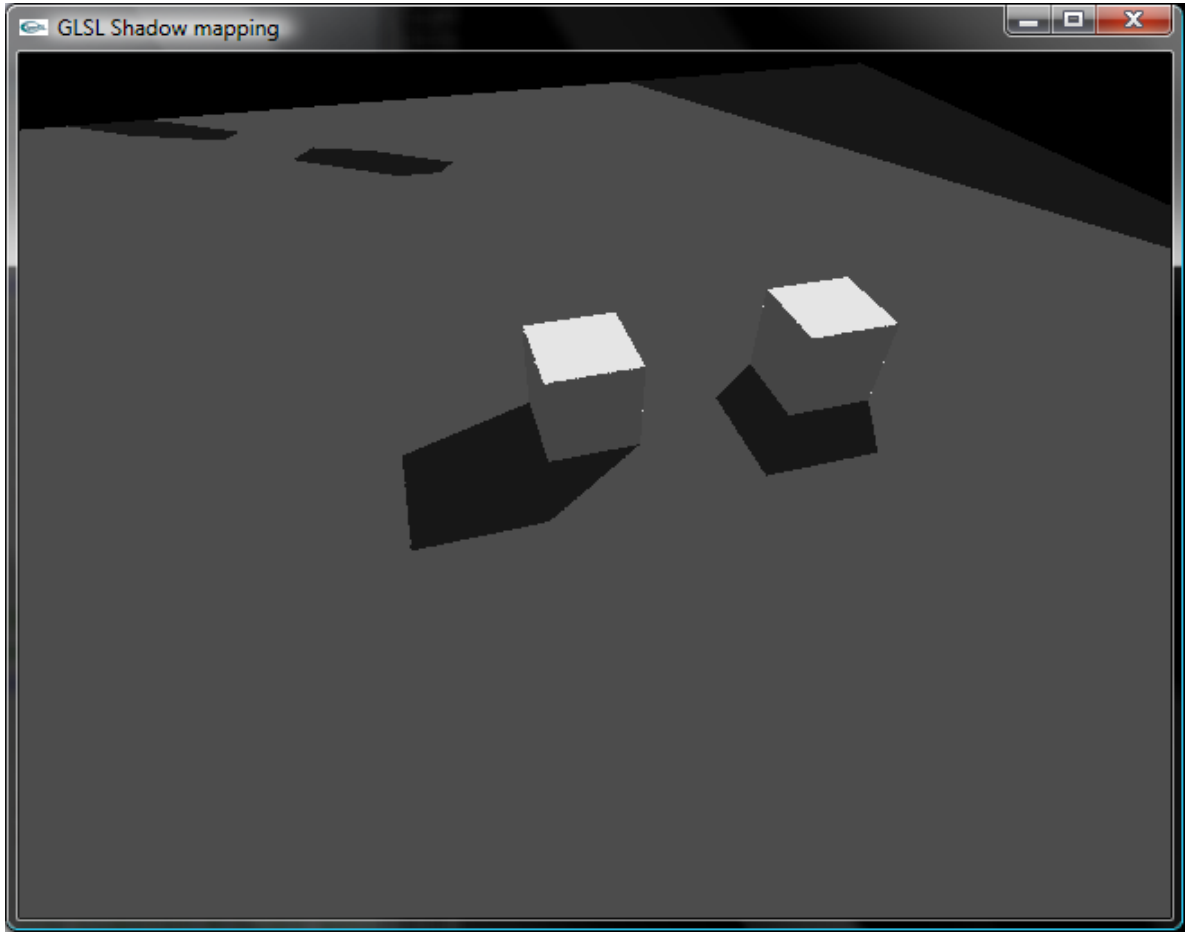
✓ =>почему появляется такой артефакт?



Карты теней (артефакты)

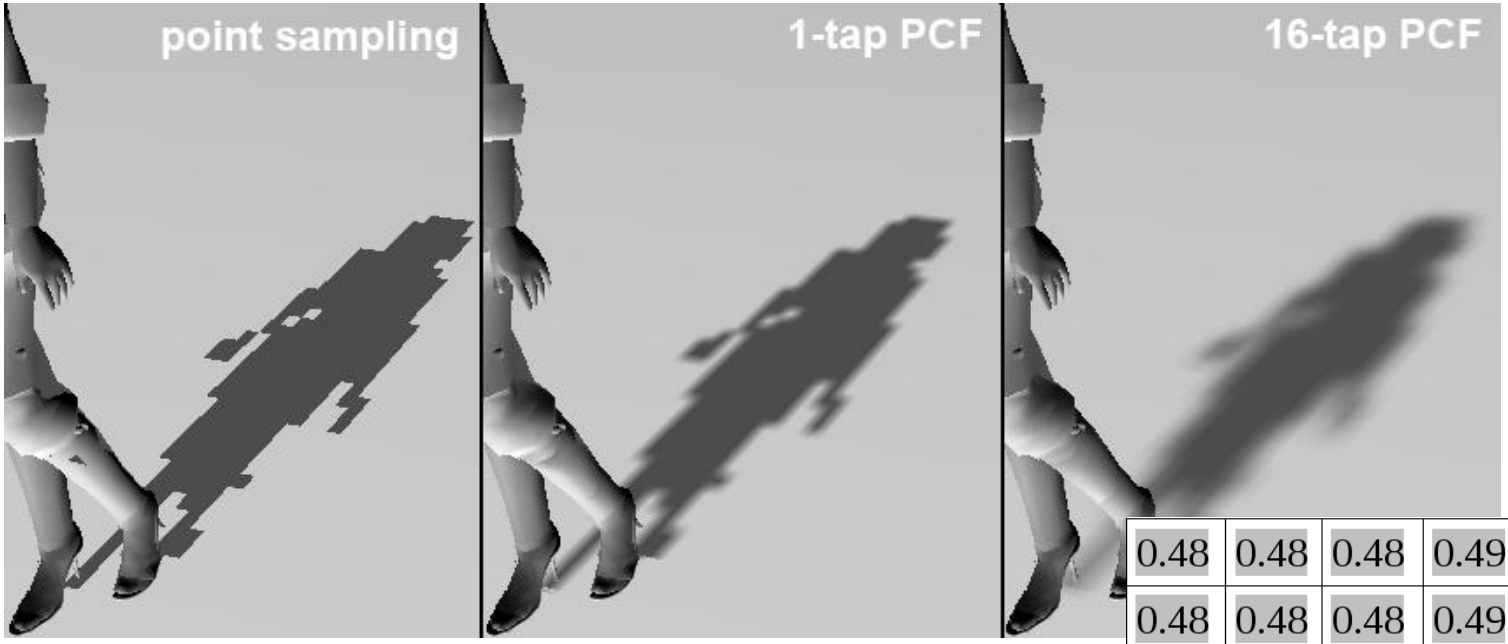
- Тени за пределами light frustum

✓glTexParameteri => GL_CLAMP



Карты теней (мягкие тени)

- Percentage-Closer Filter

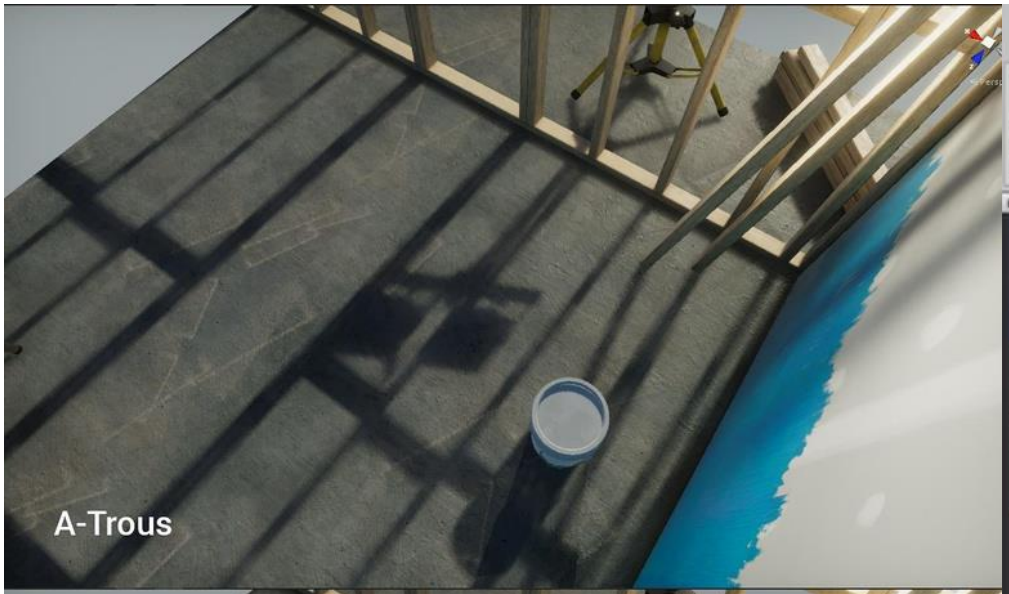


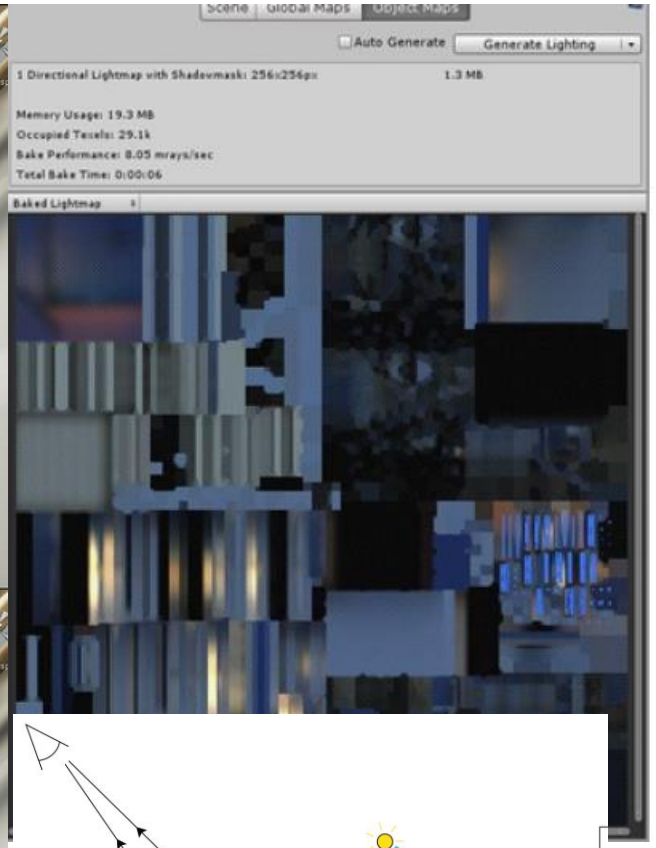
0.48	0.48	0.48	0.49	0.5	0.5	0.51
0.48	0.48	0.48	0.49	0.5	0.5	0.51
0.48	0.48	0.49	0.49	0.5	0.51	0.51
0.49	0.49	0.49	0.5	0.51	0.51	0.51
0.49	0.49	0.5	0.51	0.51	0.51	0.51
0.49	0.5	0.5	0.51	0.51	0.51	0.51
0.49	0.5	0.51	0.51	0.51	0.51	0.51

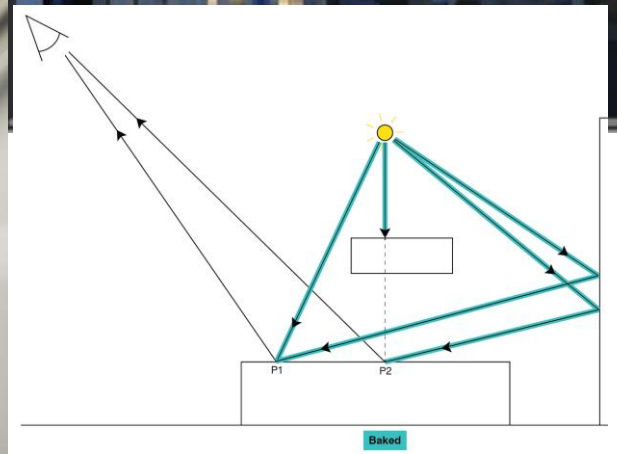
Глобальное освещение в играх

- От глобального освещения к локальному
- Приближённые методы глобального:
- Карты освещённости
- Сферические гармоники
  - Они же “volumetric lightmaps”
  - Оно же “radiance cache”
- Излучательность
- Light Propagation Volumes
- Voxel Cone Tracing
- …

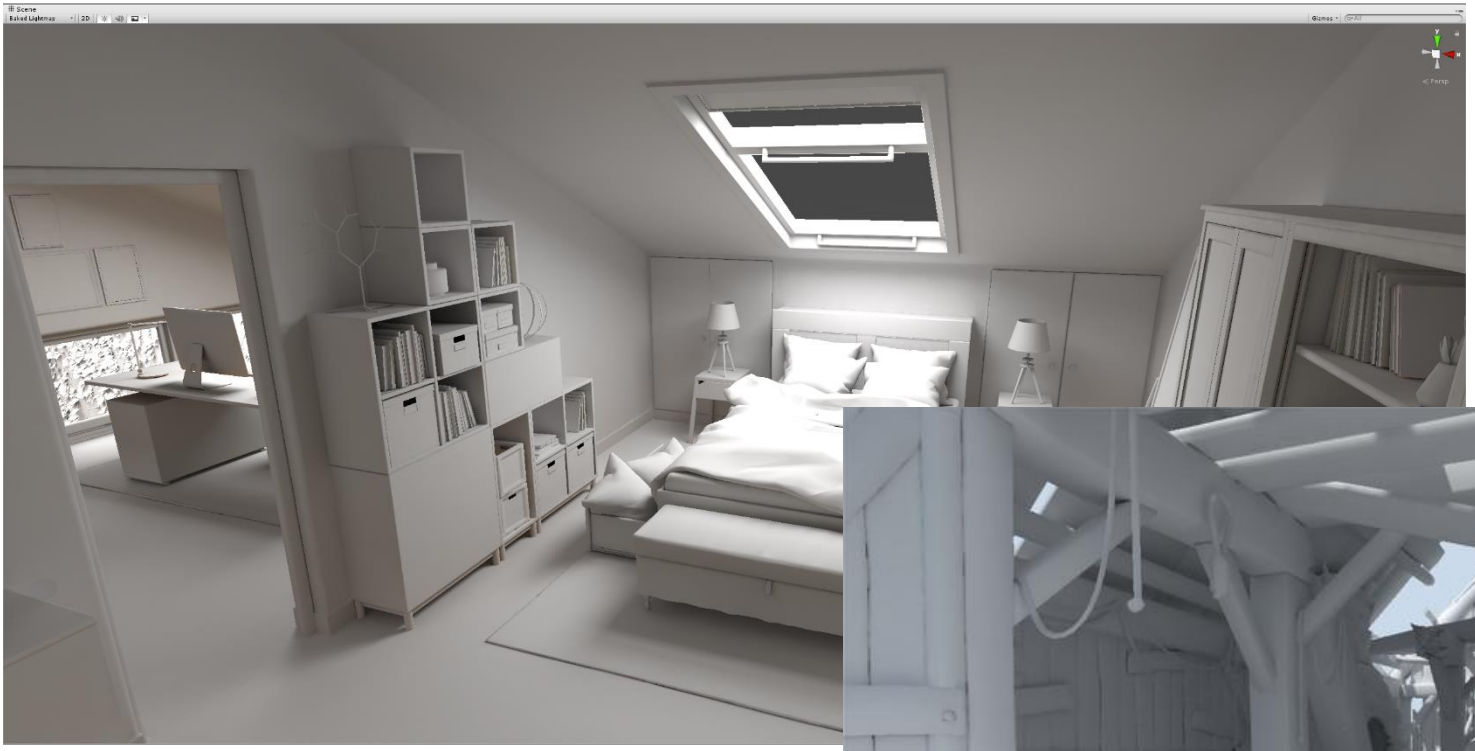
Карты освещённости

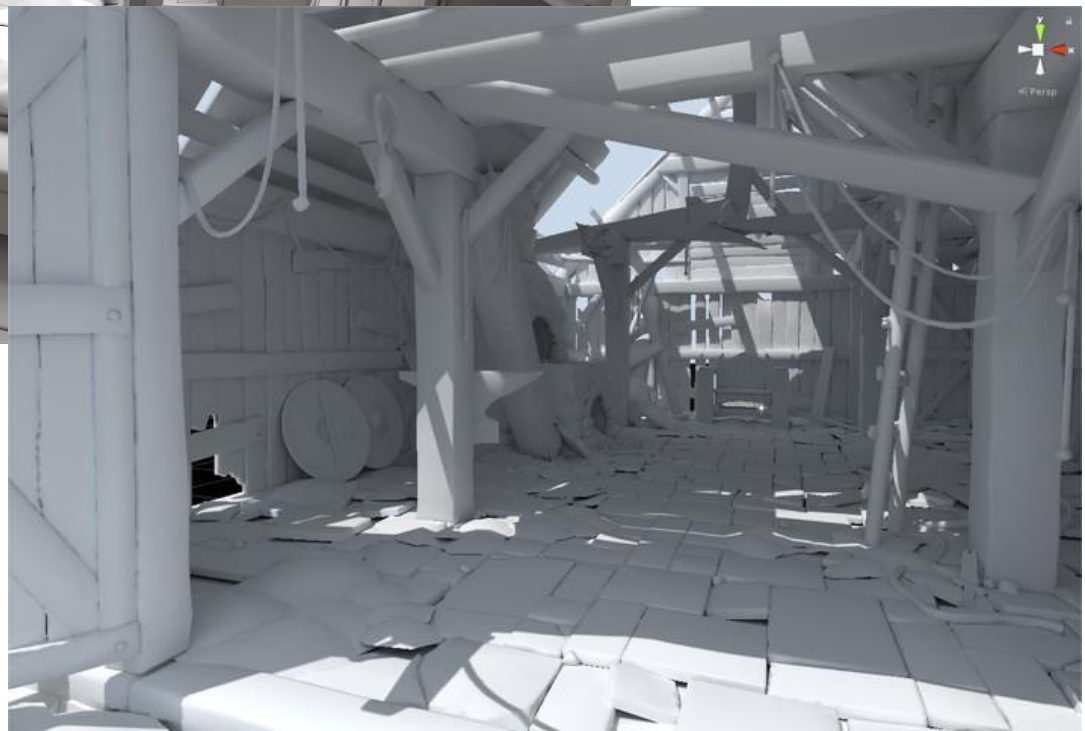


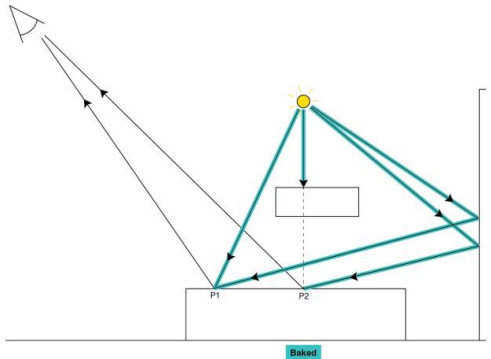




Карты освещённости





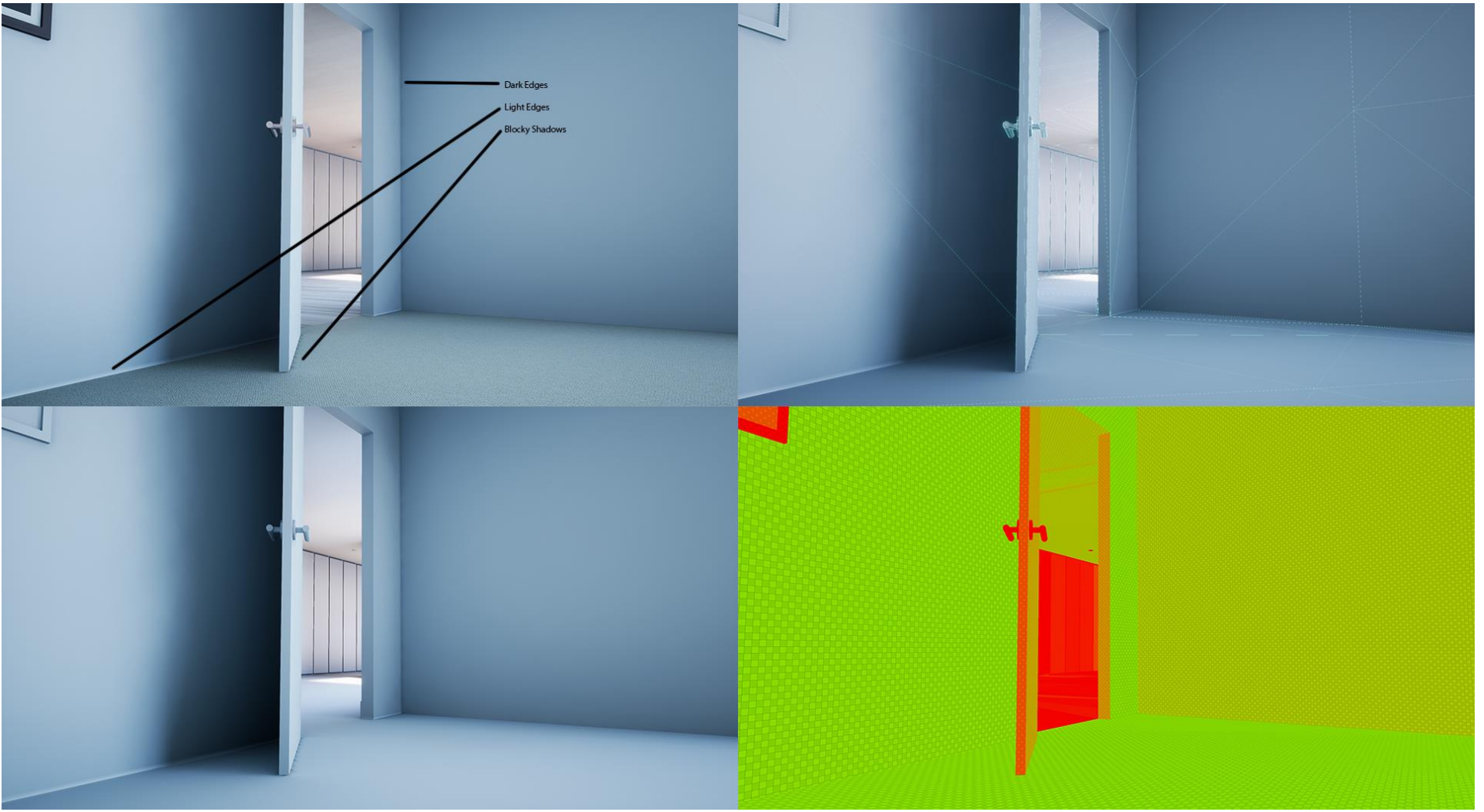


Карты освещённости

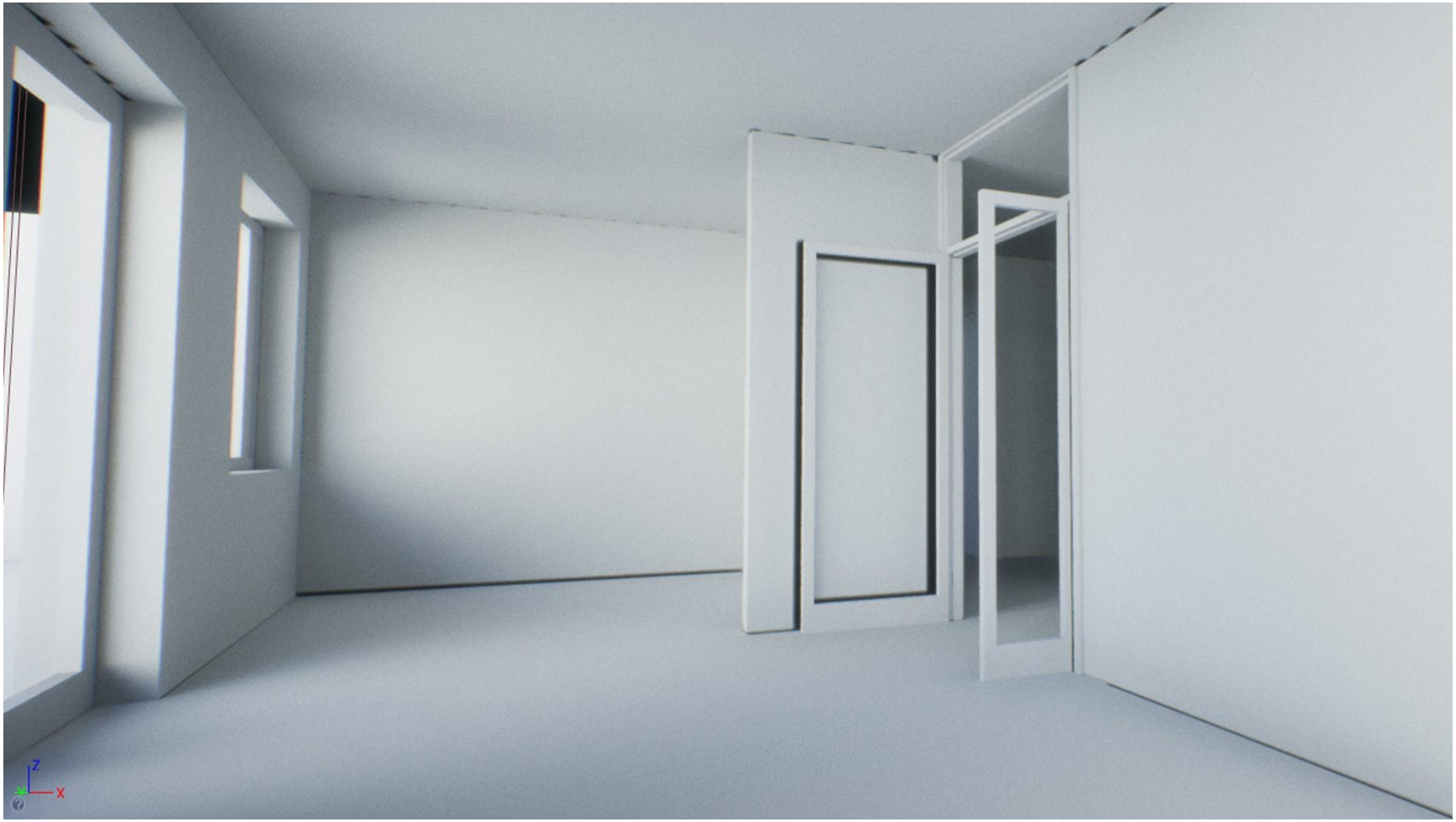
- (+) очень дёшего
- (+) просто
- (+) предрассчёт => точность

- (-) только Ламберт
- (-) нужна хорошая развёртка
- (-) много памяти
- (-) артефакты
- (-) освещение статично

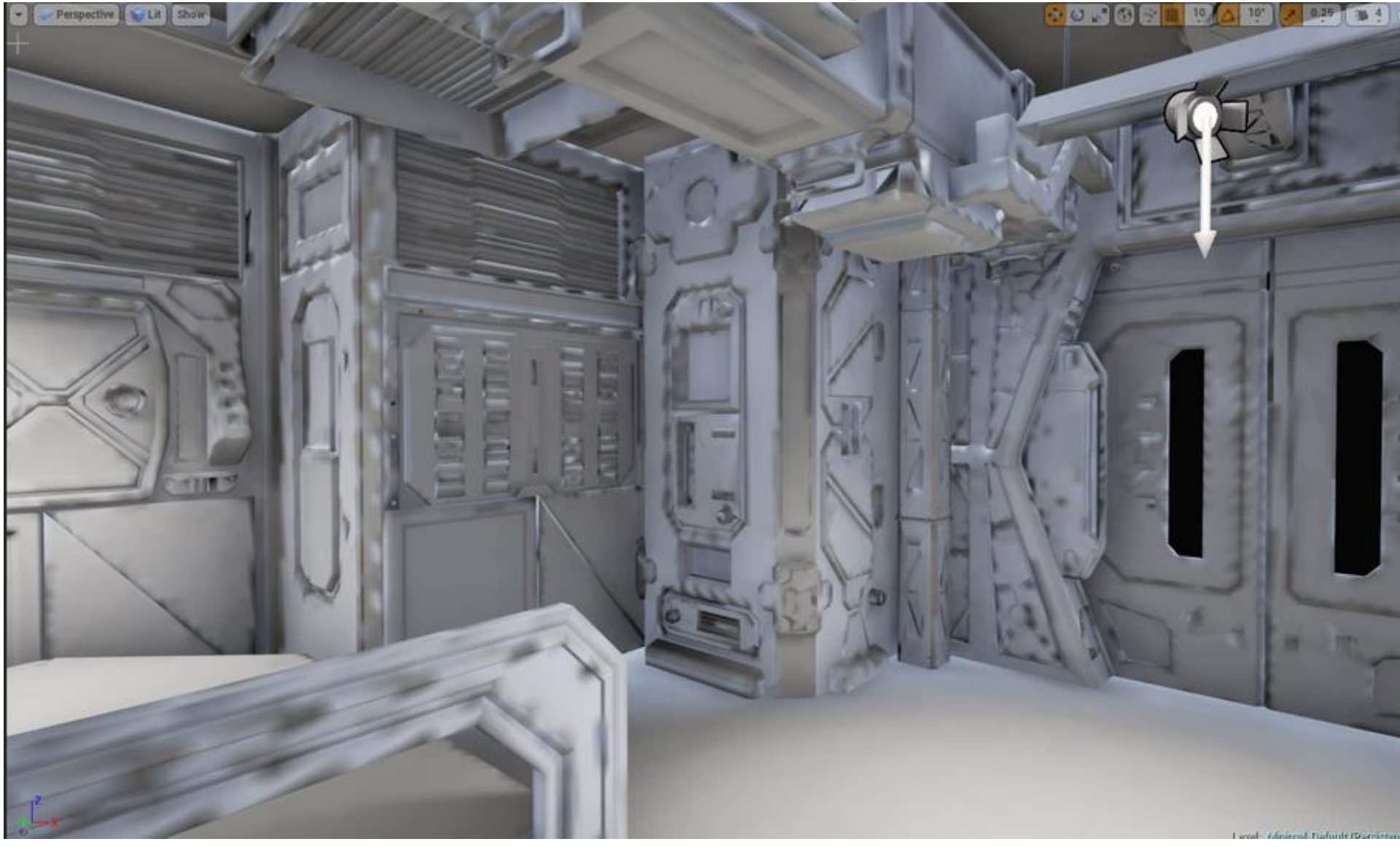
Карты освещённости (артефакты)



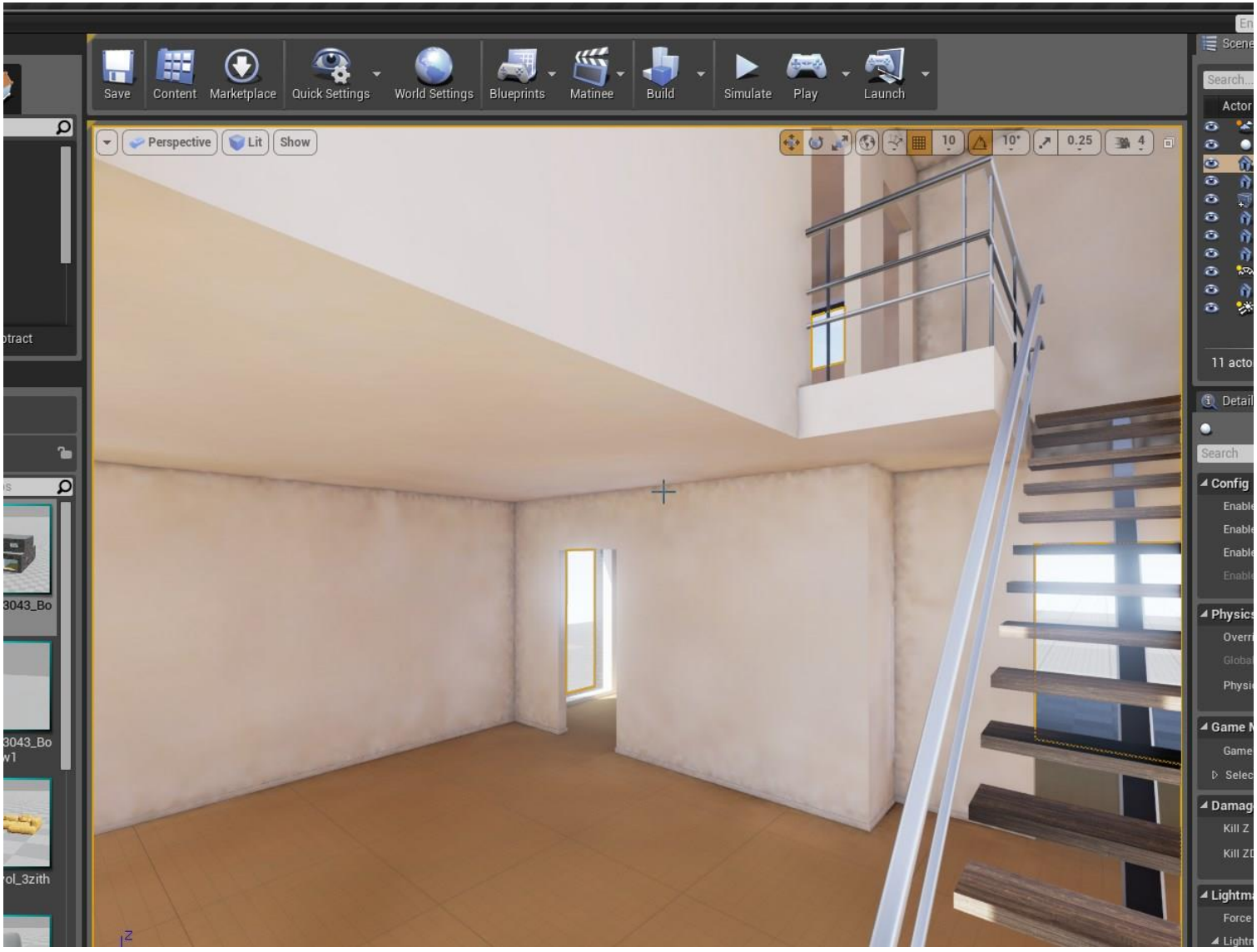
Карты освещённости (артефакты)



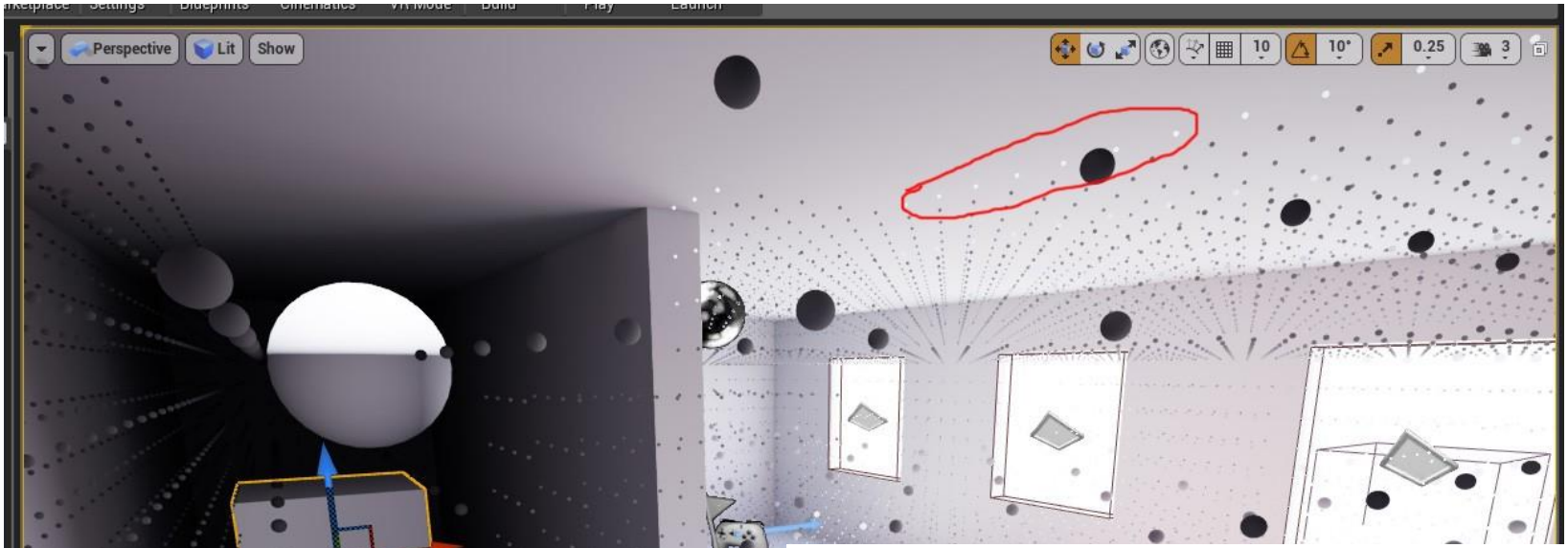
Карты освещённости (артефакты)



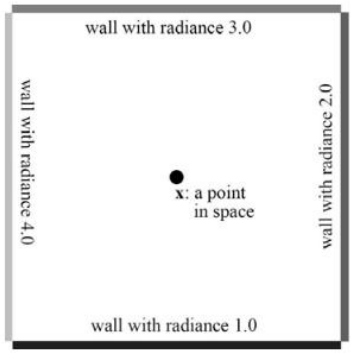
Карты освещённости (артефакты)

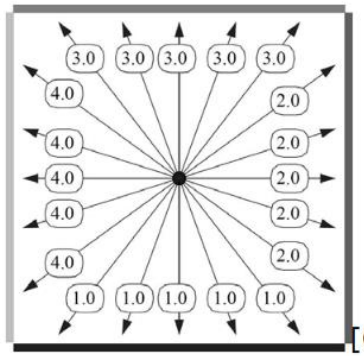


Сферические гармоники



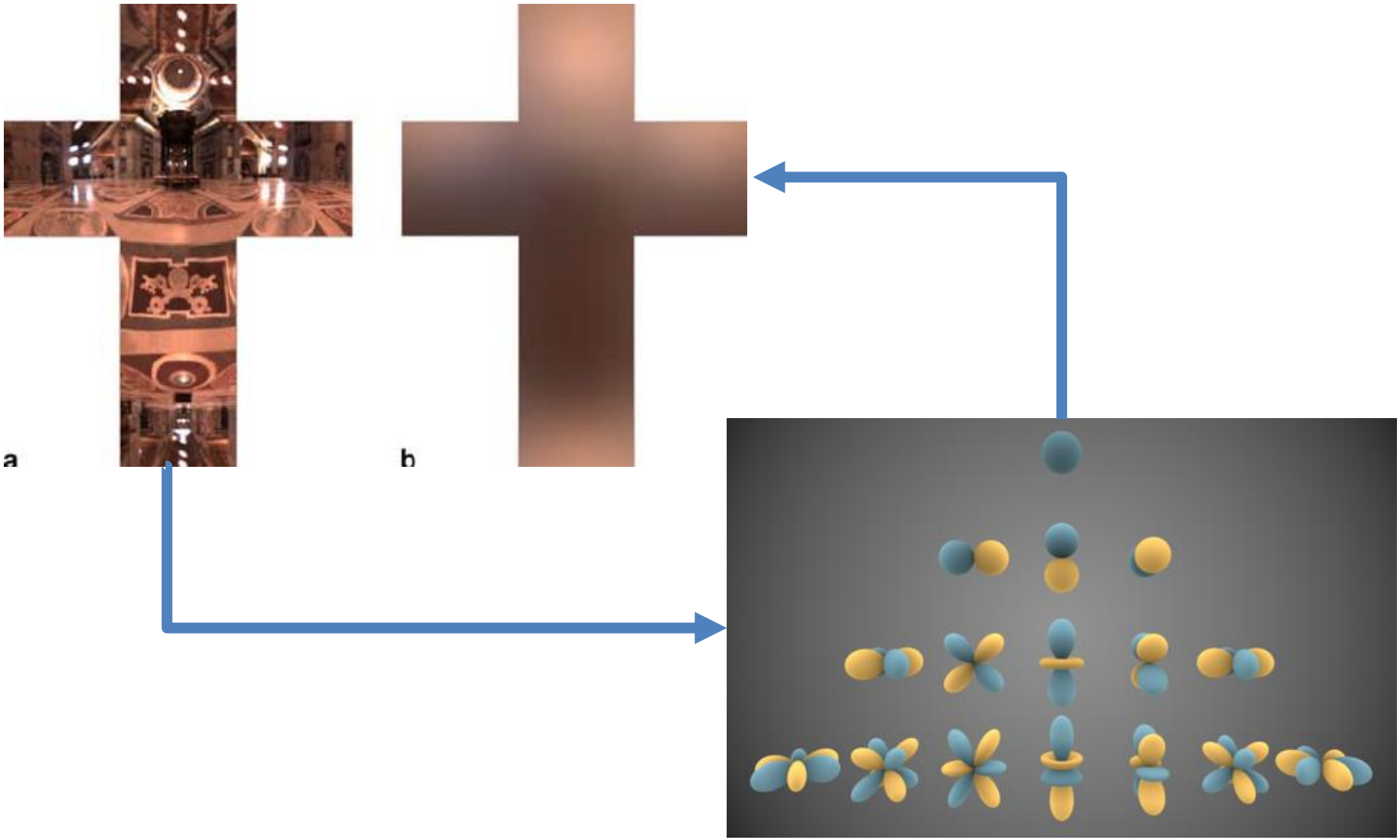
Radiance



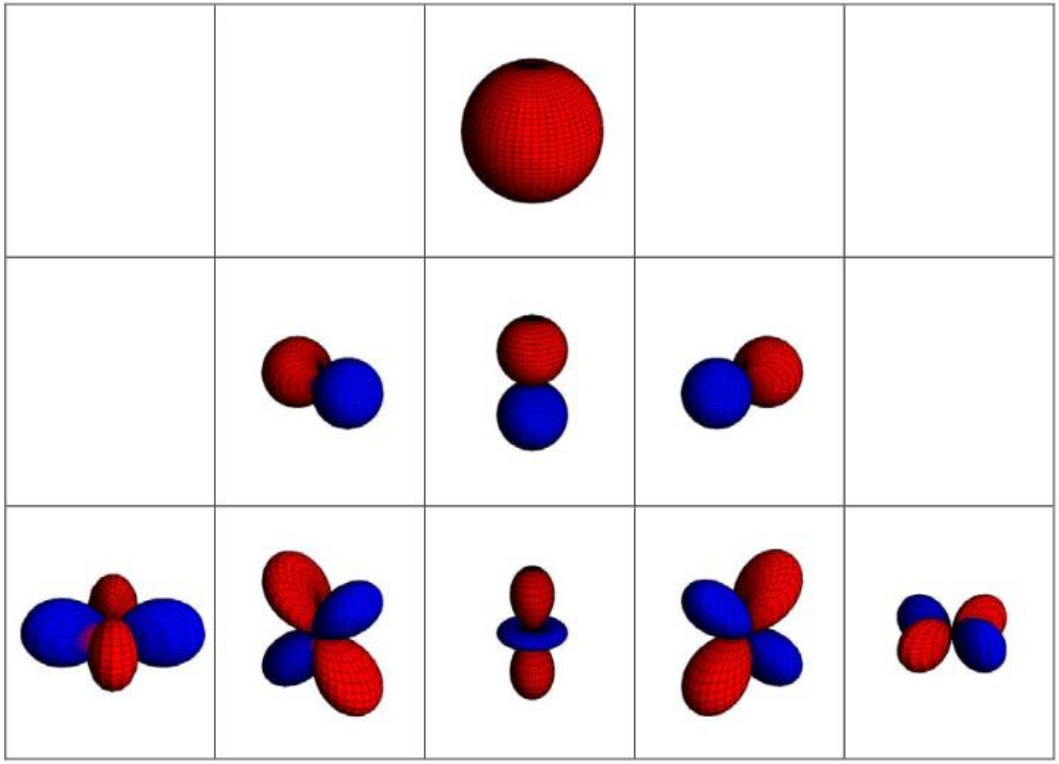


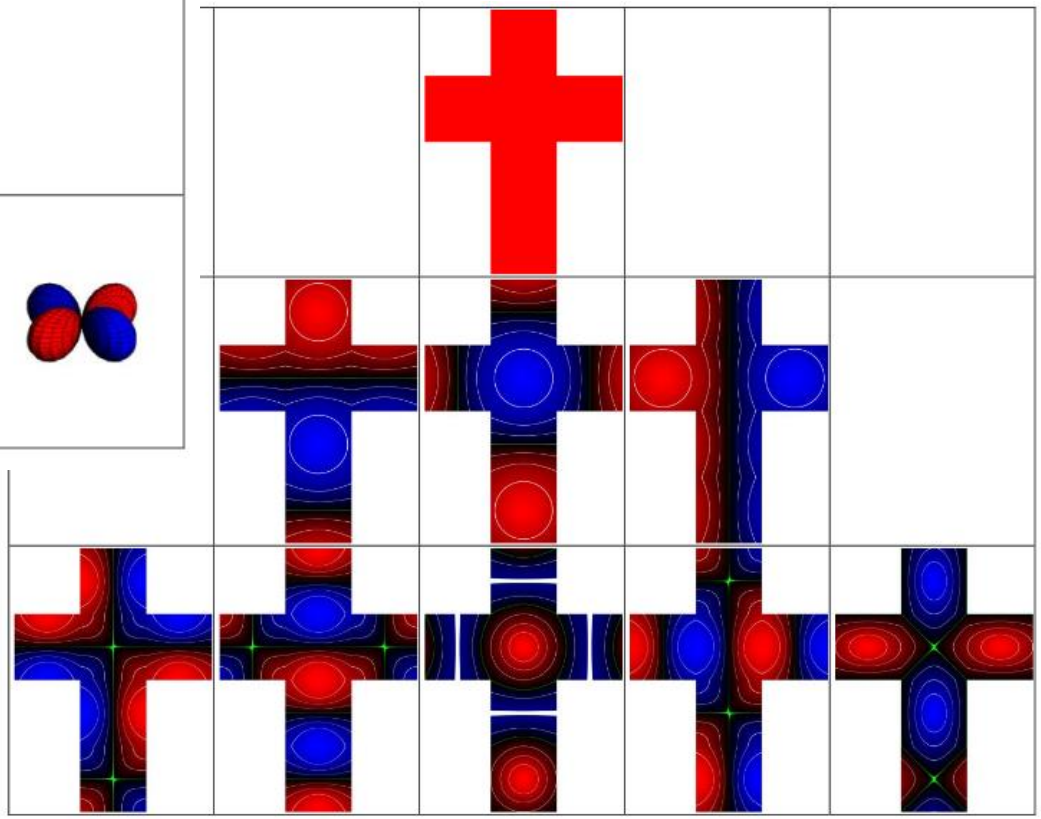
[Greger98]

Сферические гармоники

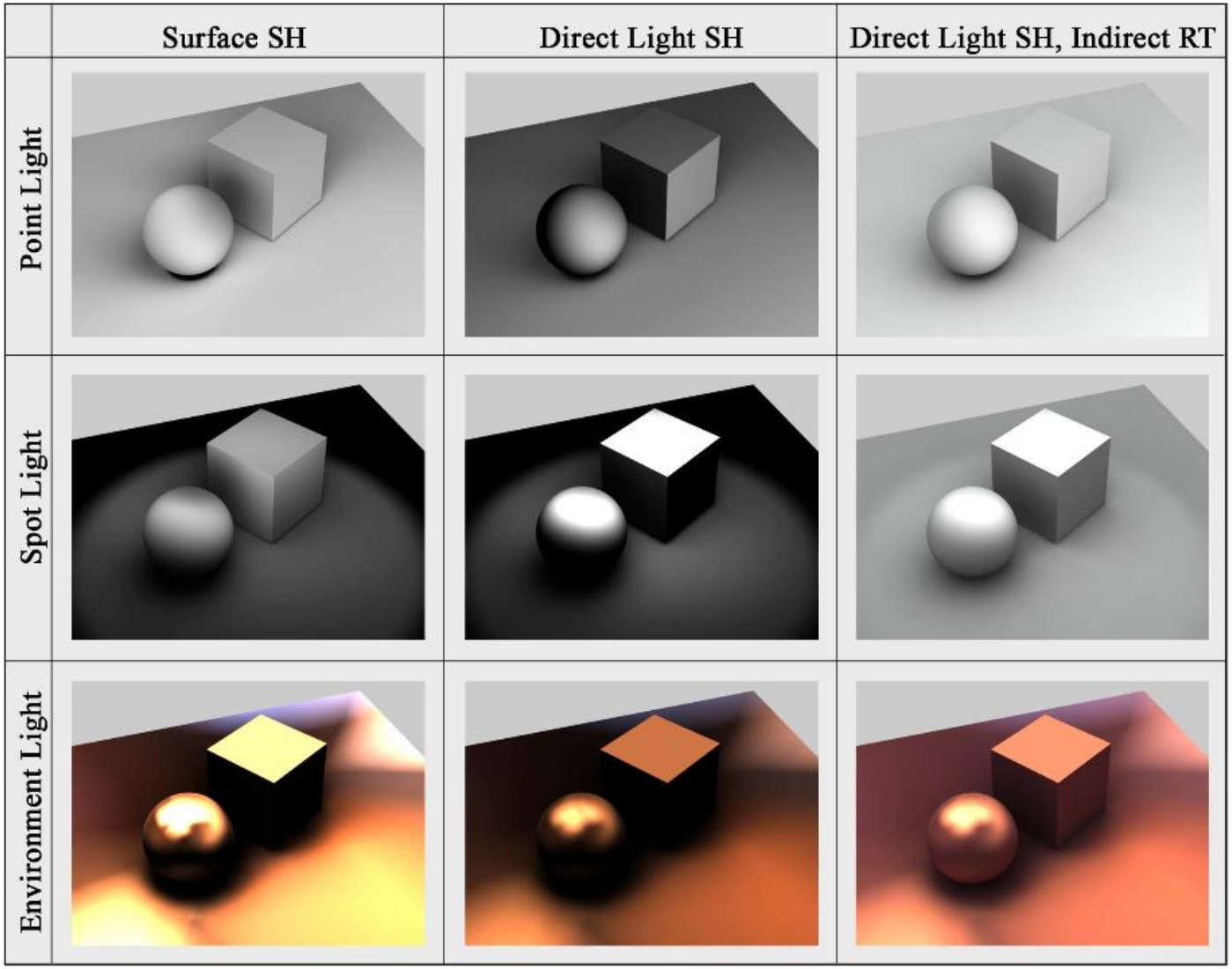


Сферические гармоники

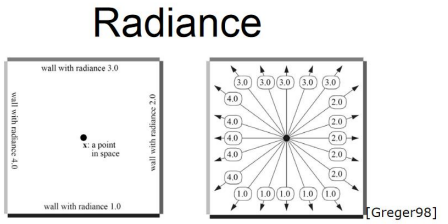




Сферические гармоники



Сферические гармоники



- (+/-) относительно дёшего
- (+) предрассчёт => точность
- (+) динамическое освещение
- (+/-) глосси отражения

- (-/+) много памяти
- (-) сложно (много нюансов)
- (-) нужна хорошая расстановка проб
- (-) артефакты / протечки

Излучательность

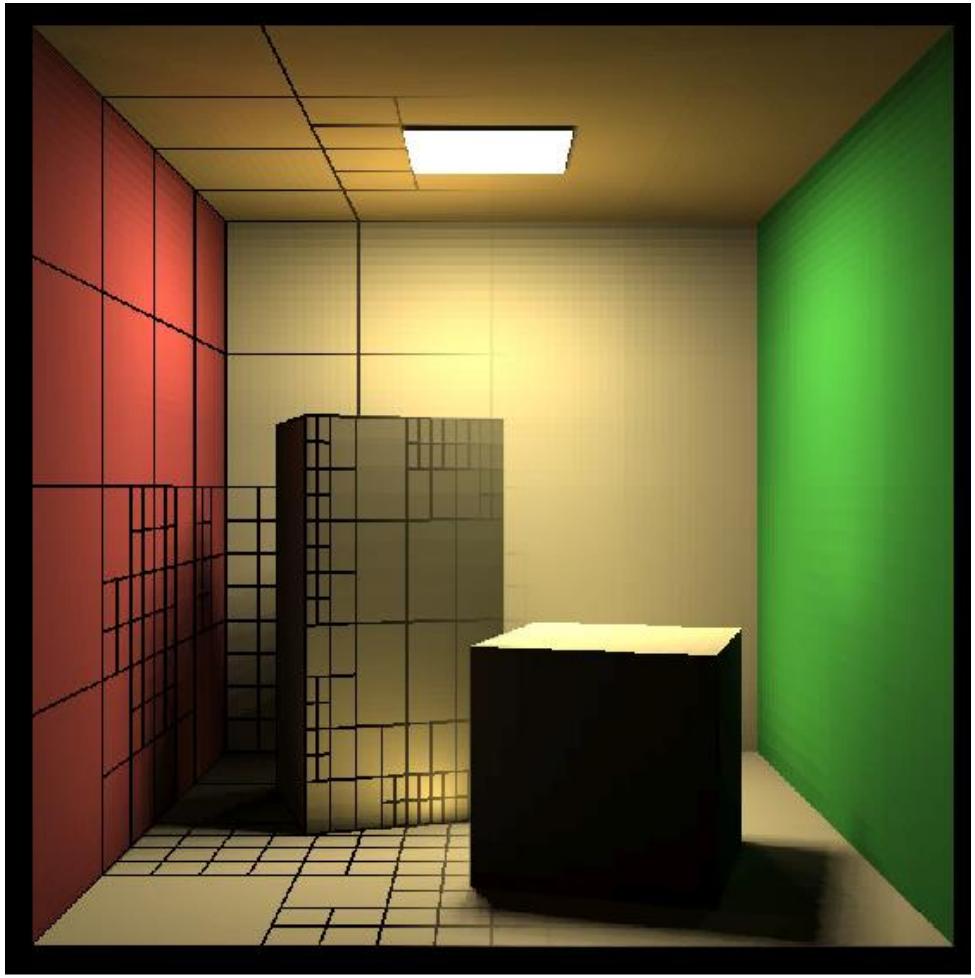
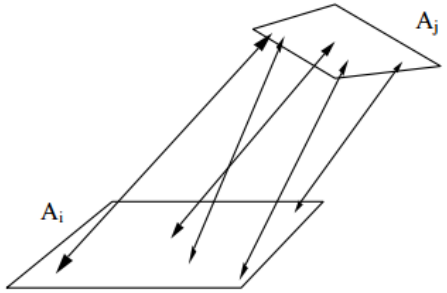
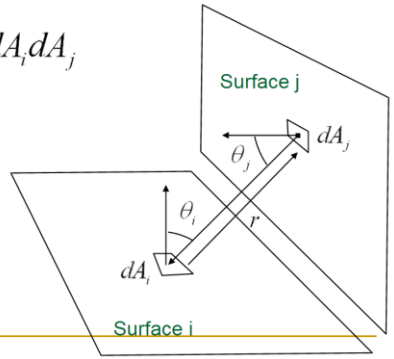


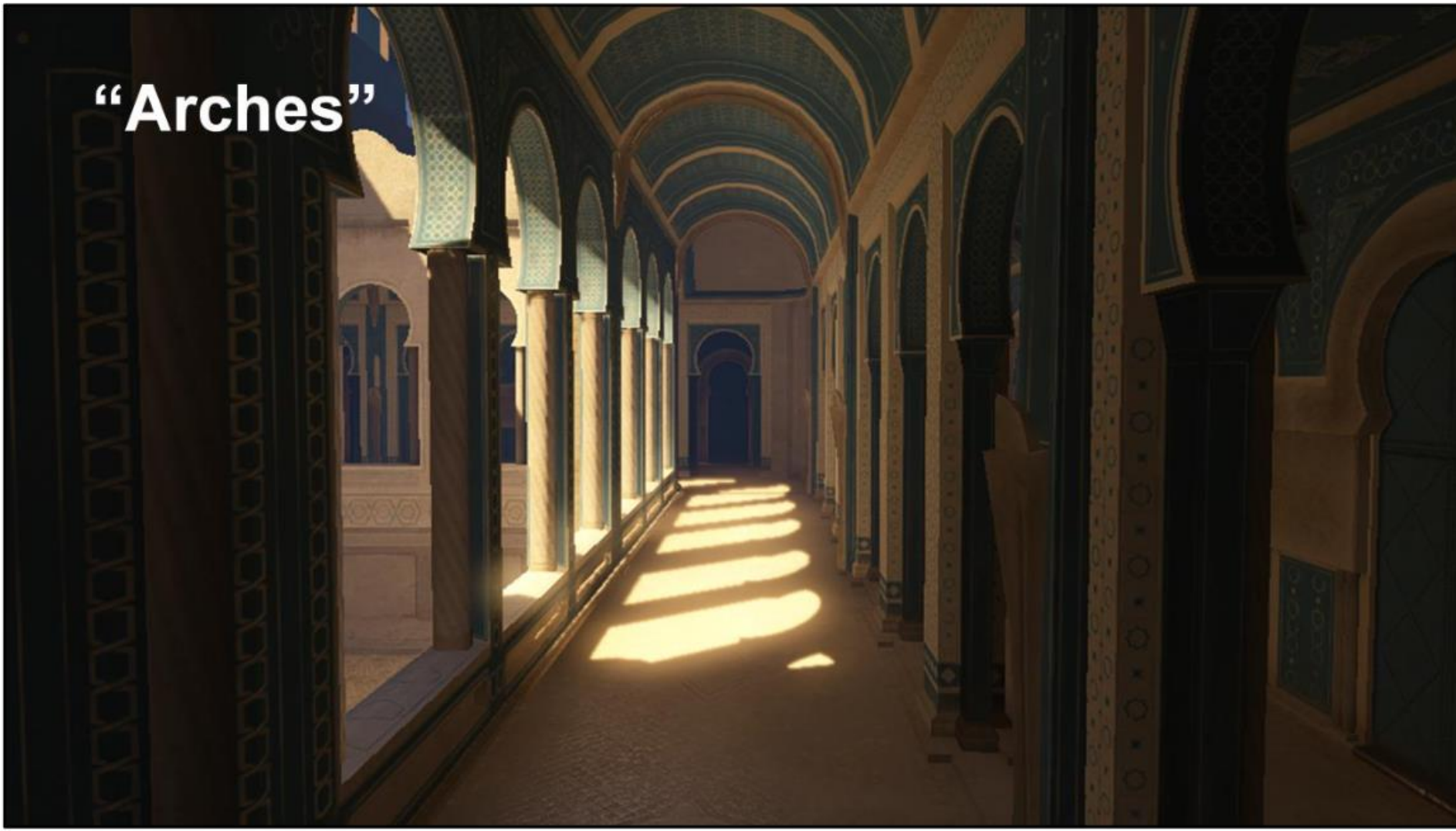
Figure 3. Radiosity solution using adaptive



$$F_{ij} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos\theta_i \cos\theta_j}{\pi |r|^2} dA_i dA_j$$



Излучательность



Излучательность

Излучательность

Излучательность

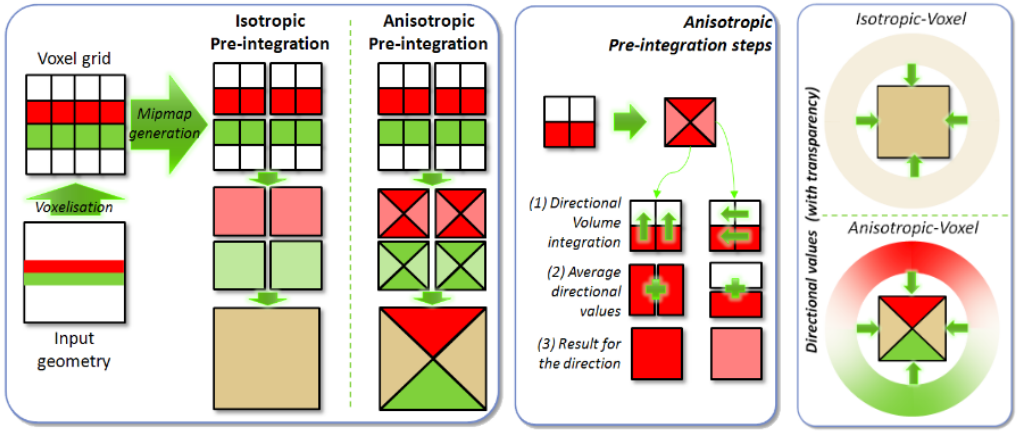
Излучательность

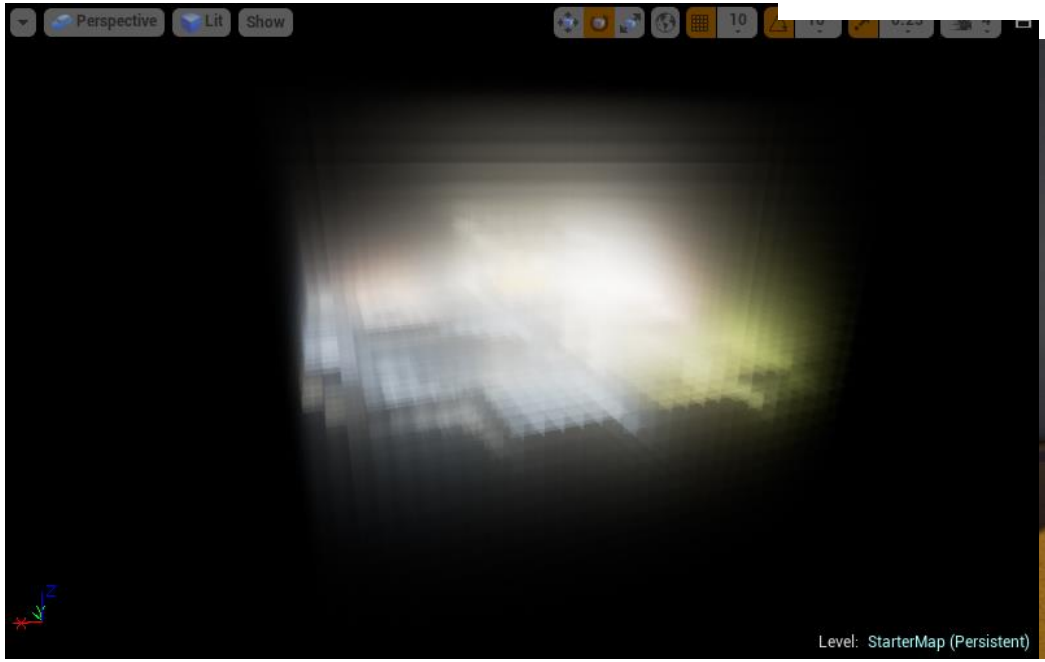
- (+) относительно дёшего
- (+) предрассчёт => точность
- (+) динамическое освещение
- (+) относительно компактный
- (+) не даёт артефактов
  (на упрощённой модели)

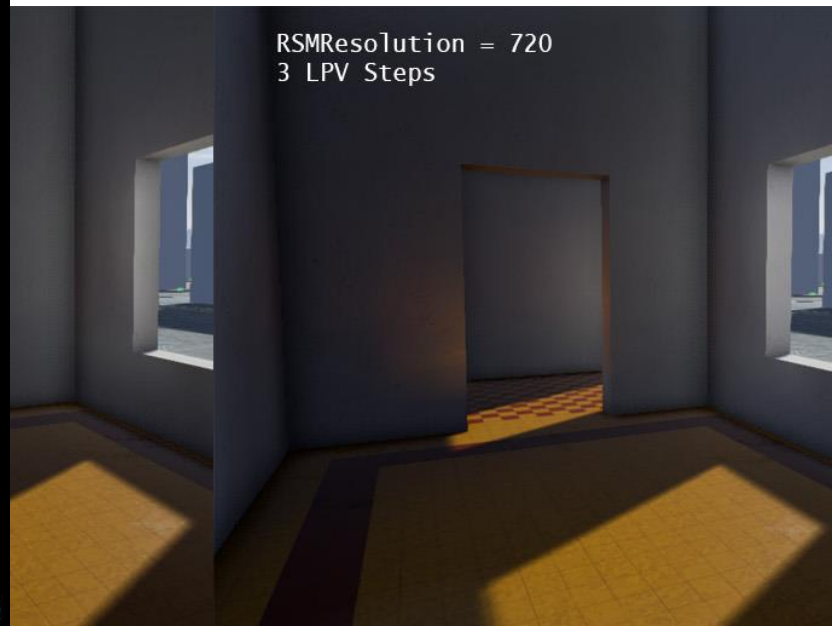
- (-) только лаберт
- (-) нужна апроксимация сцены
- (-) O(N*N) где N – количество площадок

Light Propagation Volumes

- Есть много памяти
- Дорого, сложно ...
- И плохо
- Cascaded LPV







Light Propagation Volumes

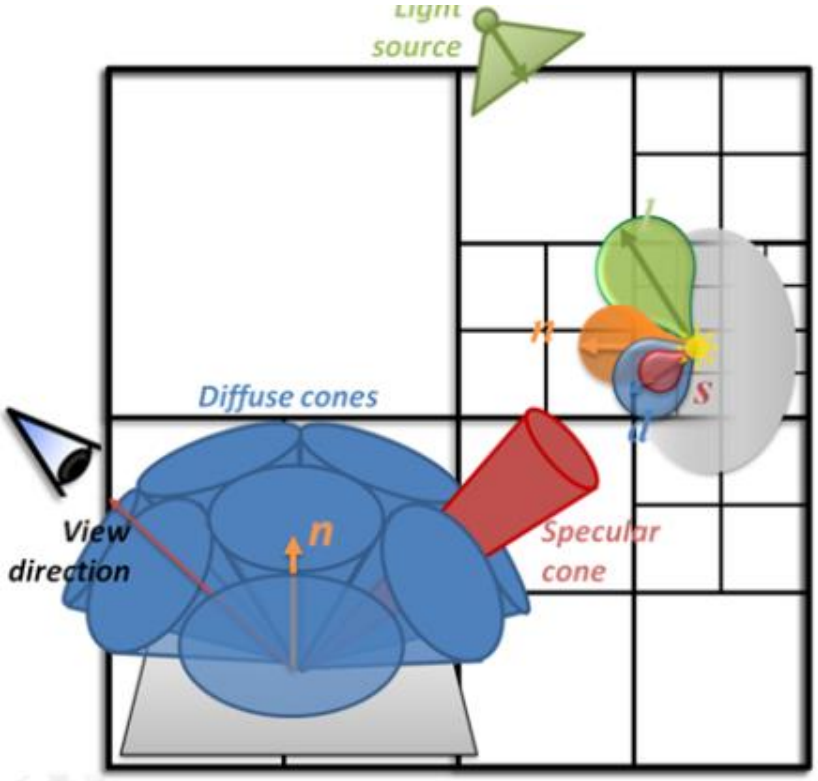
- Есть много памяти

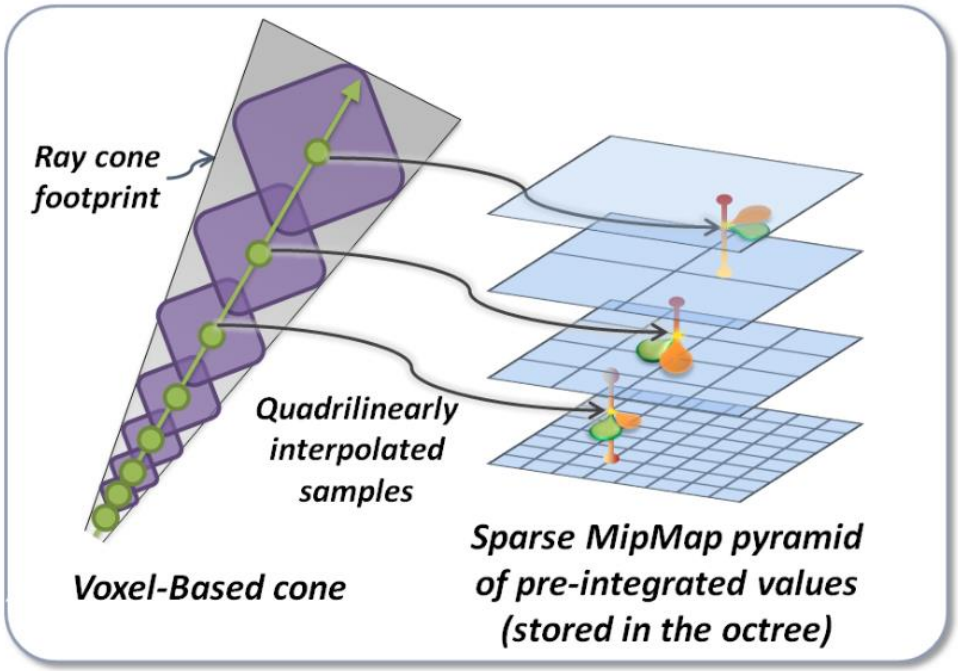
Voxel Cone Tracing

Voxel Cone Tracing

Voxel Cone Tracing

- Дорого
- Сложно





Глобальное освещение в играх

- Приближённые методы глобального:
- Precomputed Light Transport
  - Карты освещённости
  - Сферические гармоники
    - Они же “volumetric lightmaps”
    - Оно же “radiance cache”
    - Precomputed Radiance Transfer как развитие метода
  - Излучательность

- Динамические методы
  - Light Propagation Volumes (полностью динамический)
  - Voxel Cone Tracing (полностью динамический)