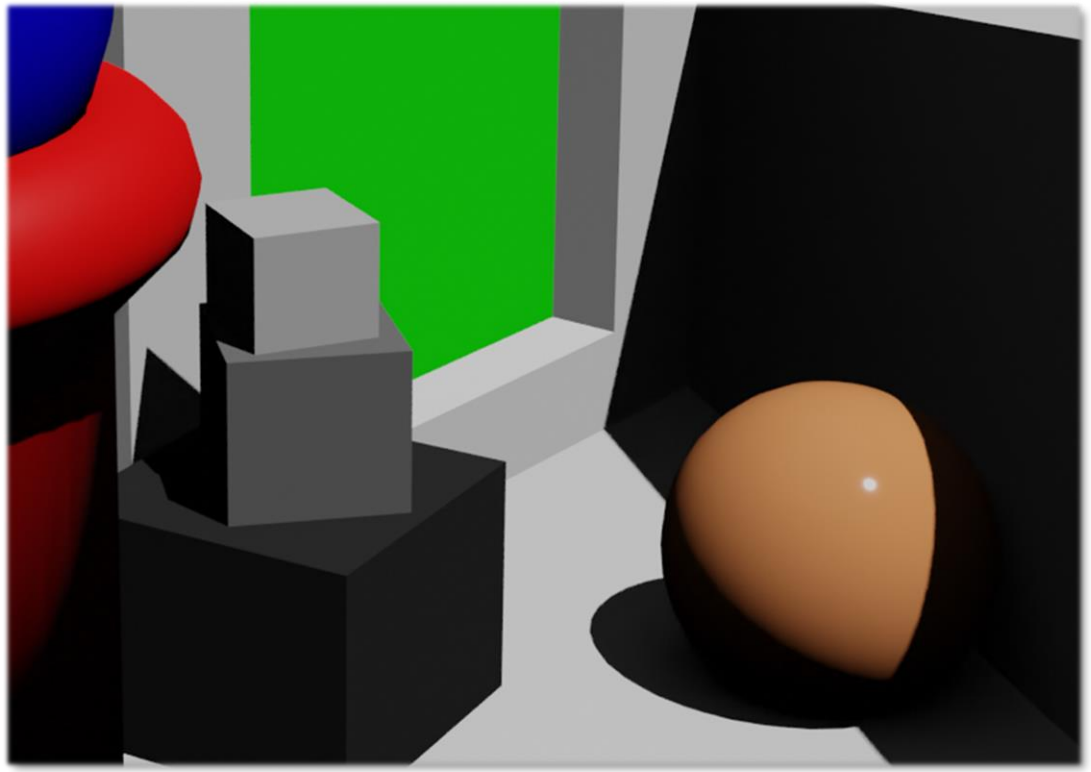
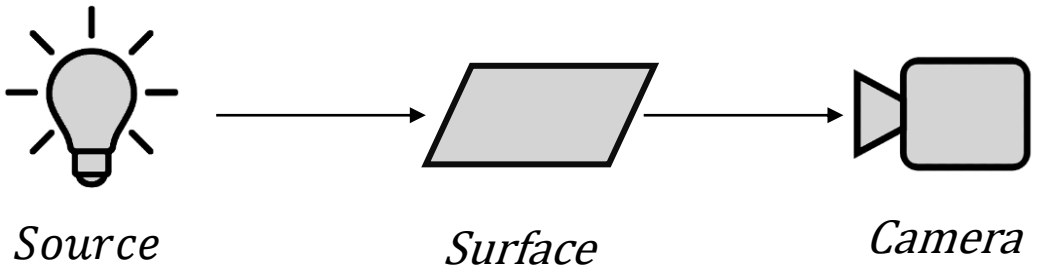
Hardware Acceleration of Progressive Refinement Radiosity using Nvidia RTX

Benjamin Kahl

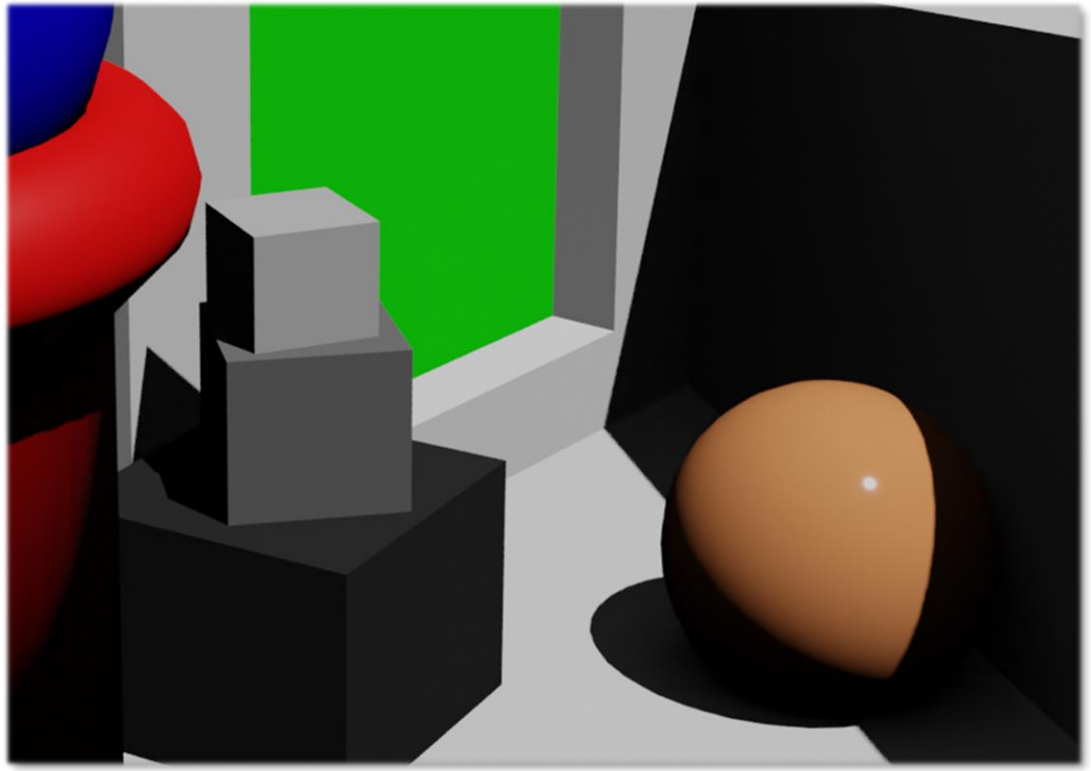
10.11.2022

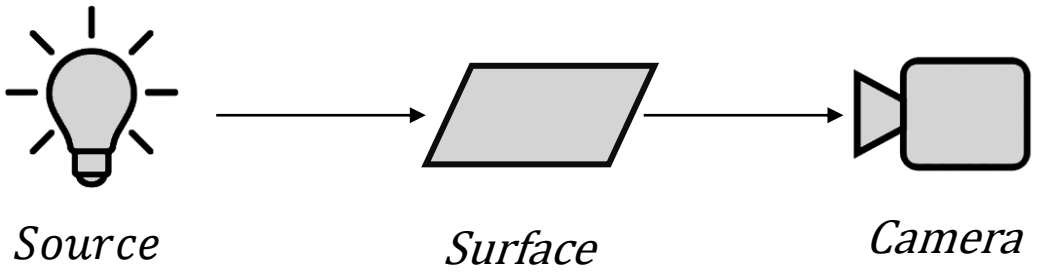
Local Illumination



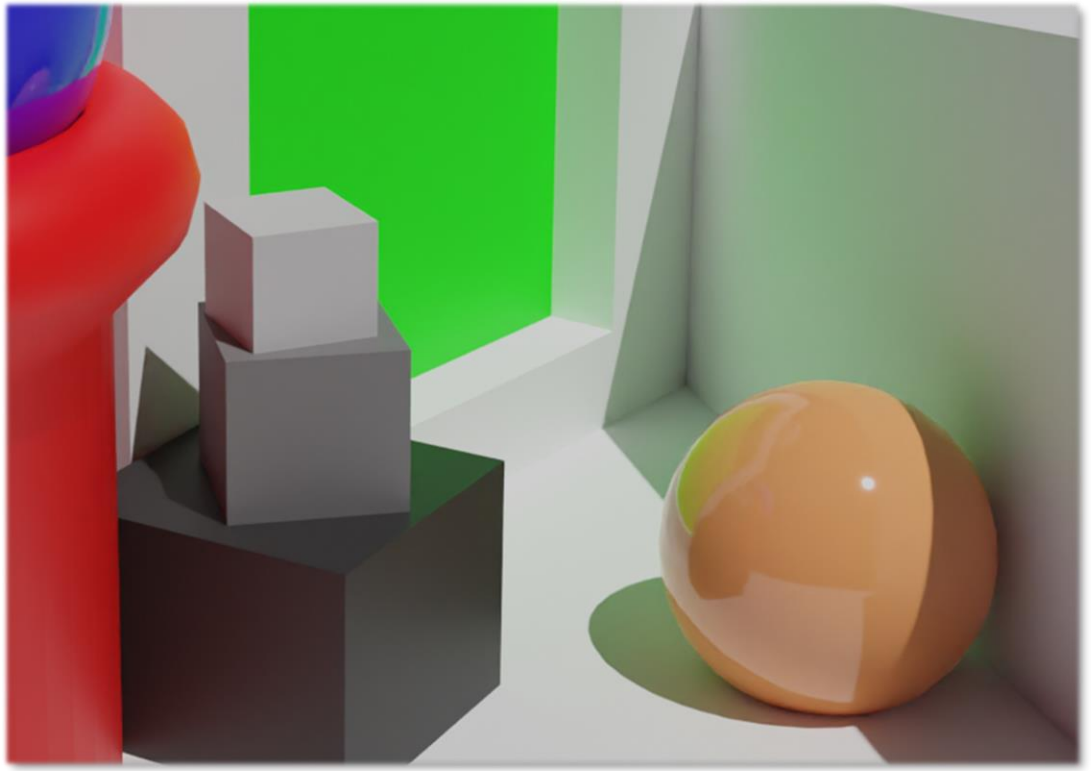


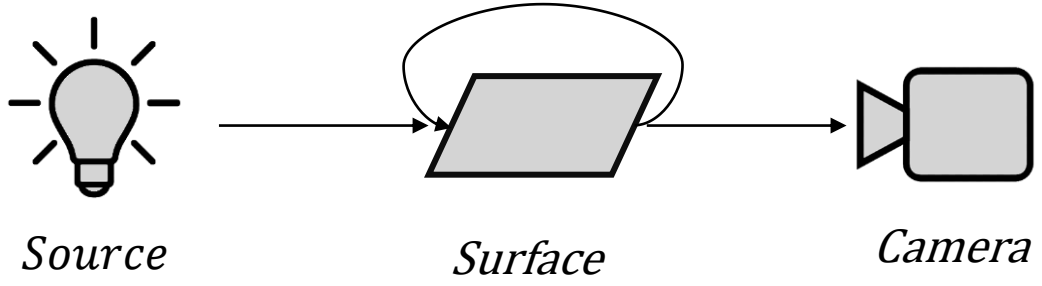
Local Illumination

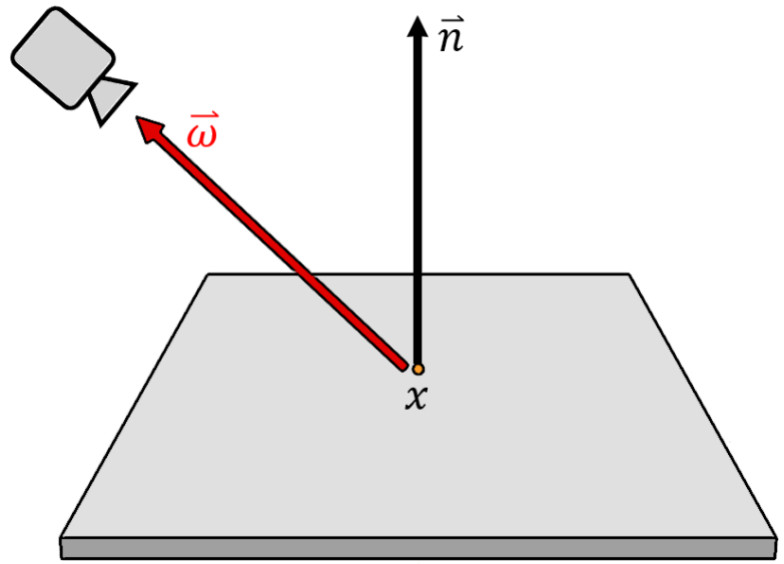


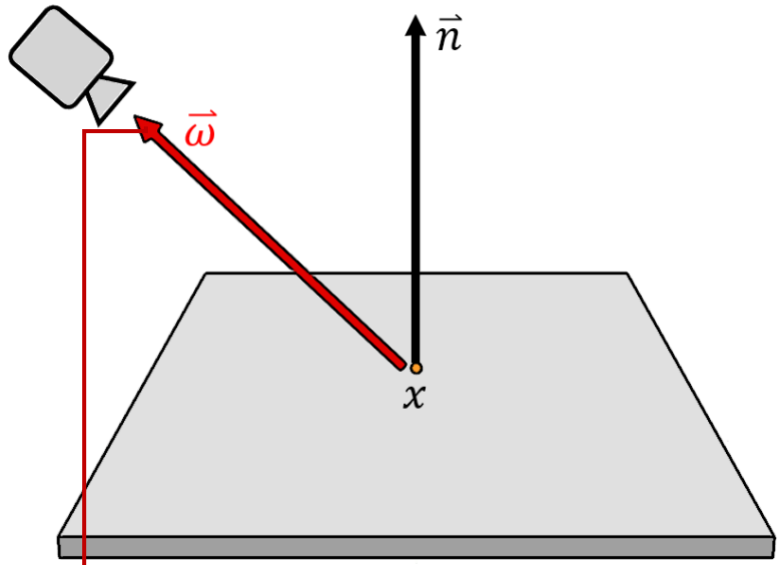


Global Illumination



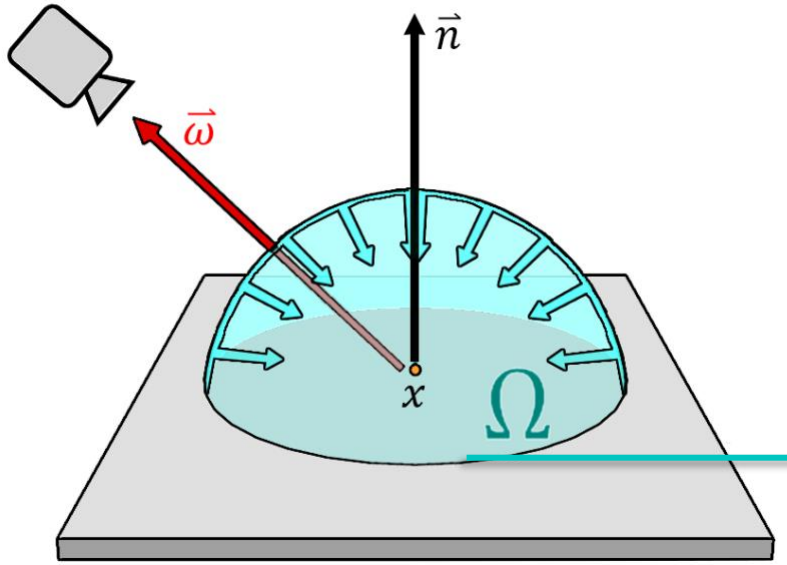



$\vec{n}$
$\vec{\omega}$
x



Rendering Equation:
(Kajiya et al. 1986)

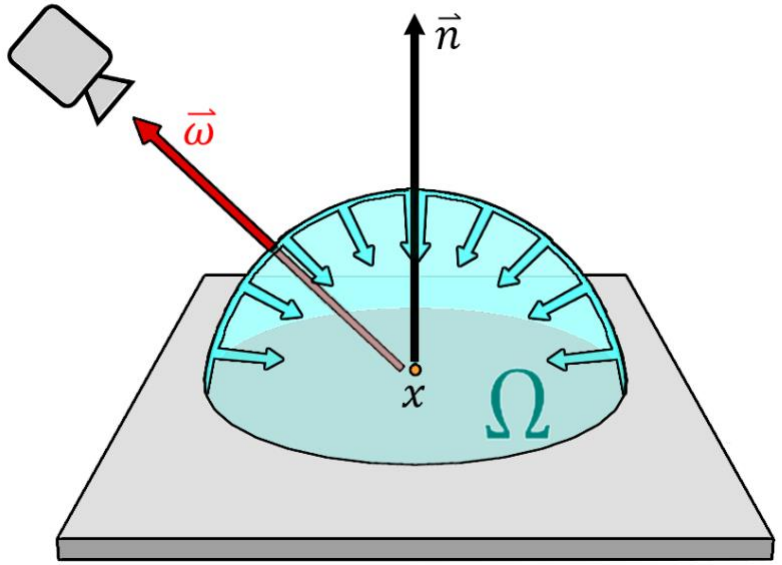
$$L(x,\vec{\omega}) = L_{e(x,\vec{\omega})} + L_{r(x,\vec{\omega})}$$



Rendering Equation:

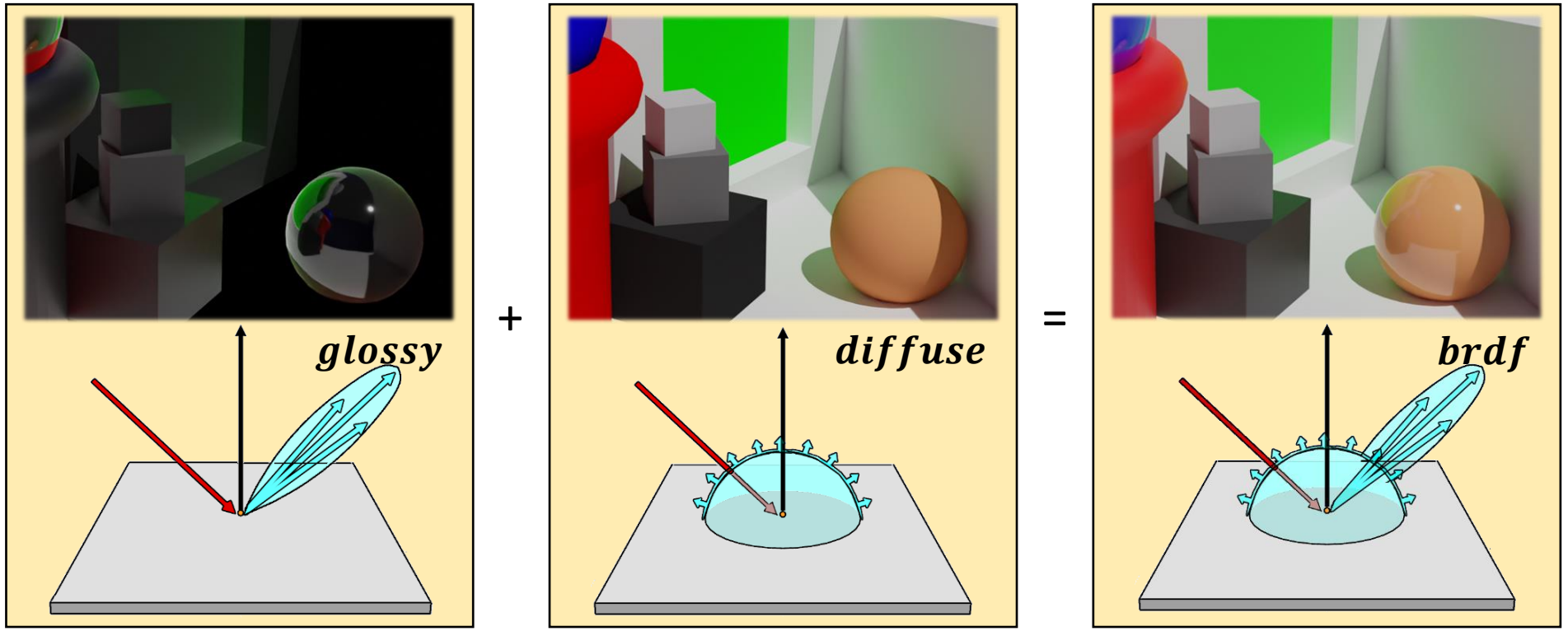
(Kajiya et al. 1986)

$$L(x,\vec{\omega}) = L_{e(x,\vec{\omega})} + \int_{\Omega} L_i(x,\vec{\omega'}) \dots$$

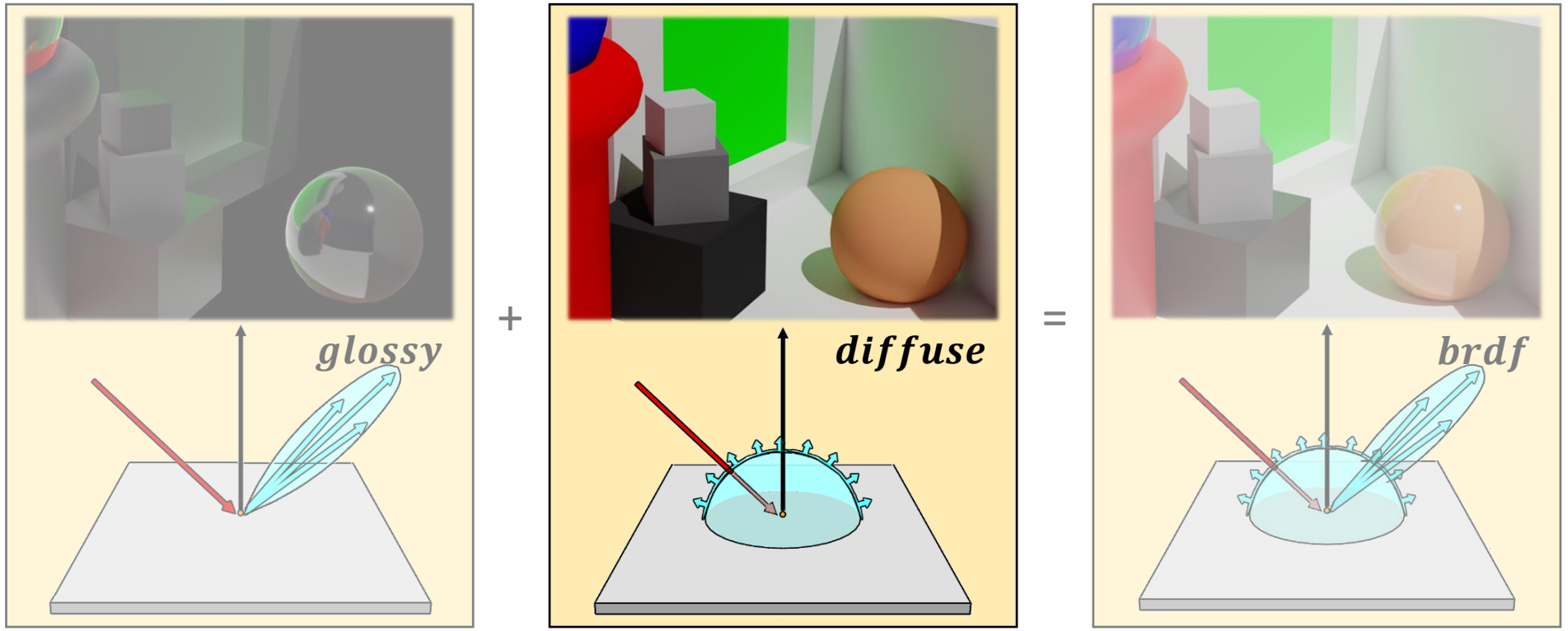


Rendering Equation:
(Kajiya et al. 1986)

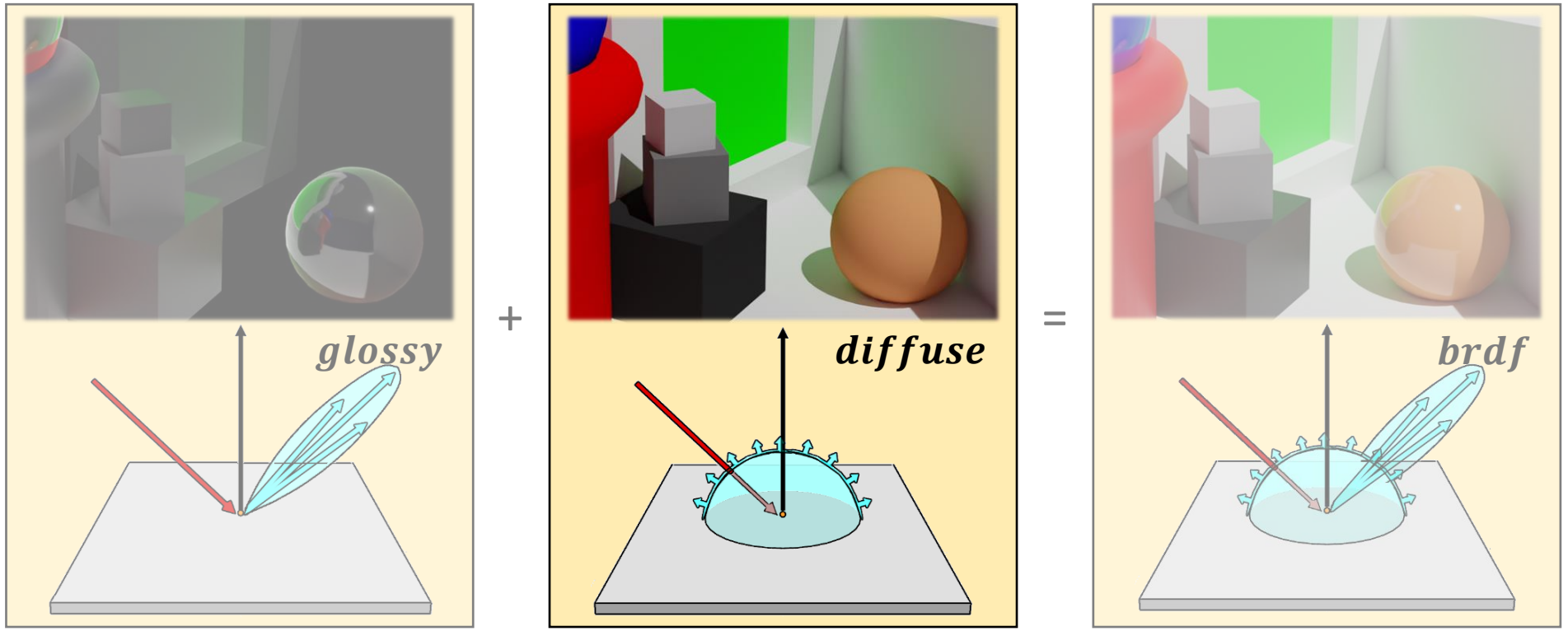
$$L(x,\vec{\omega}) = L_{e(x,\vec{\omega})} + \int_{\Omega} L_i(x,\vec{\omega'}) f_r(\vec{\omega'} \rightarrow \vec{\omega}) \cos\theta' \, d\omega'$$



$$L(x,\vec{\omega}) = L_{e(x,\vec{\omega})} + \int_{\Omega} L_i(x,\vec{\omega'}) f_r(\vec{\omega'} \rightarrow \vec{\omega}) \cos\theta' \, d\omega'$$



$$L(x,\vec{\omega}) = L_{e(x,\vec{\omega})} + \int_{\Omega} L_i(x,\vec{\omega'}) f_r(\vec{\omega'} \rightarrow \vec{\omega}) \cos\theta' \, d\omega'$$

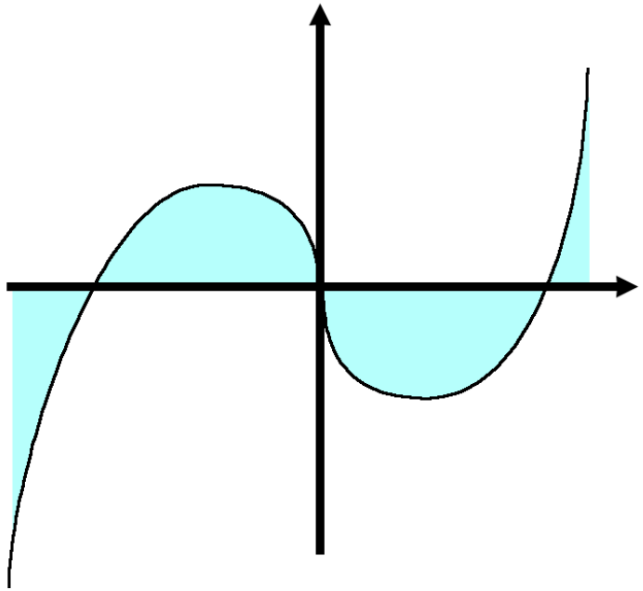


$$L(x,\vec{\omega}) = L_{e(x,\vec{\omega})} + \frac{\rho}{\pi}\int_{\Omega} L_i(x,\vec{\omega'})\cos\theta'\, d\omega'$$

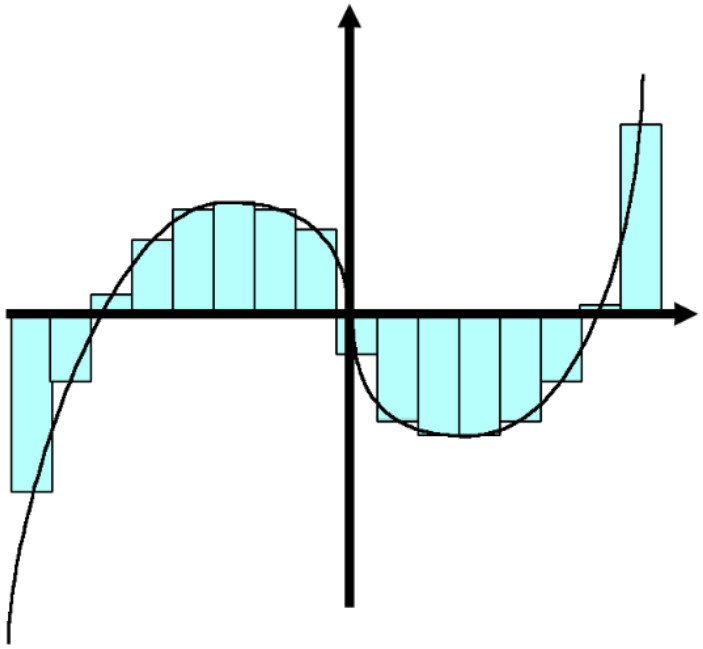
$$L(x, \vec{\omega}) = L_{e(x,\vec{\omega})} + \frac{\rho}{\pi} \int_{\Omega} L_i(x, \vec{\omega'}) \cos\theta' \, d\omega'$$

$$L(x,\vec{\omega}) = L_{e(x,\vec{\omega})} + \frac{\rho}{\pi}\int_S L_o\left(x',\vec{\omega'}\right) V(x,x') \frac{\cos\theta \cos\theta'}{|x-x'|^2} d\omega'$$

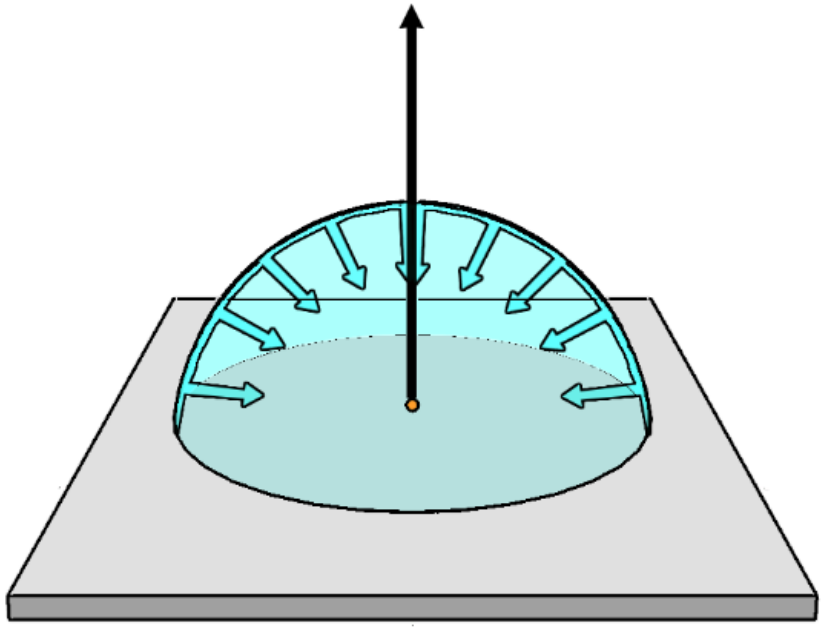
$$L(x,\vec{\omega}) = L_{e(x,\vec{\omega})} + \frac{\rho}{\pi}\int_\Omega L_i\left(x,\vec{\omega'}\right) \cos\theta' \, d\omega'$$



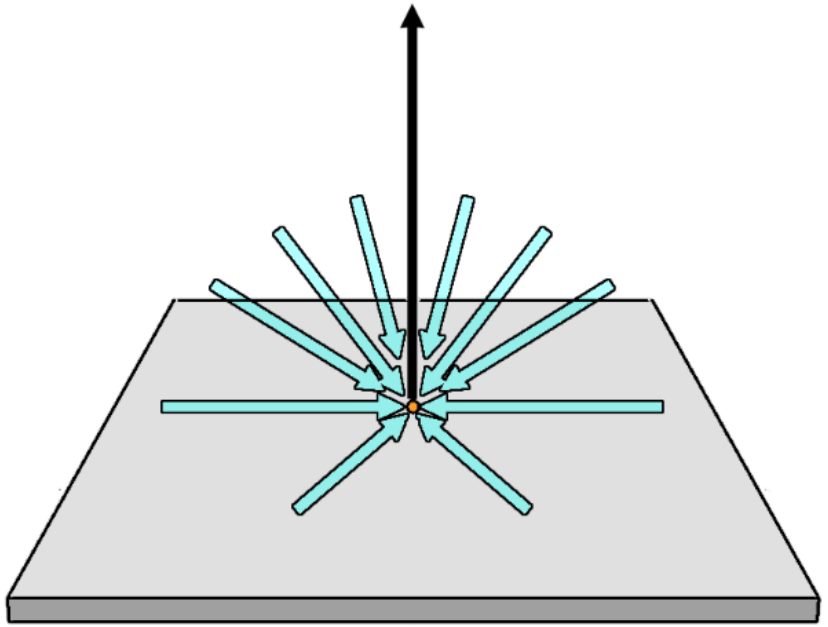
$$\int f(x)\,dx$$



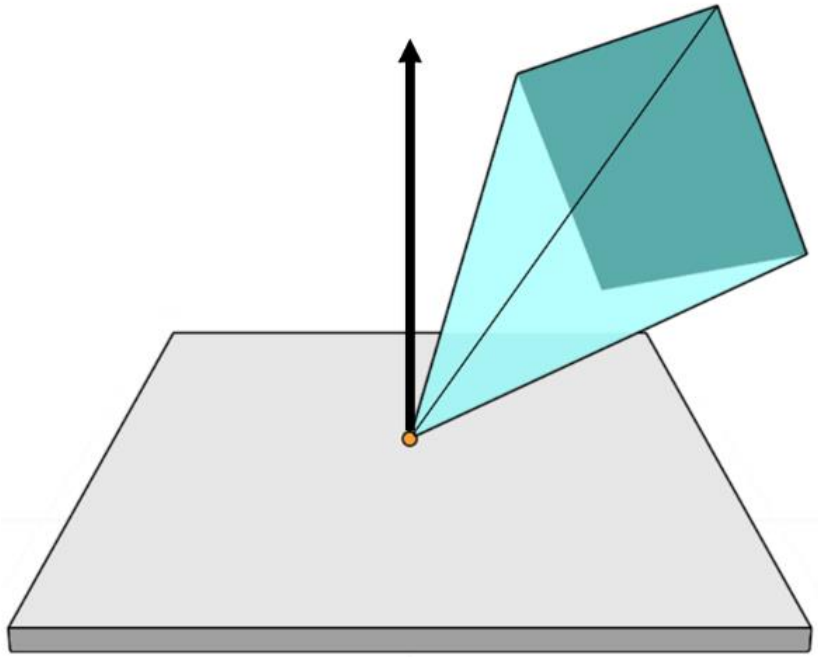
$$\sum f(x_i)\,\Delta x$$



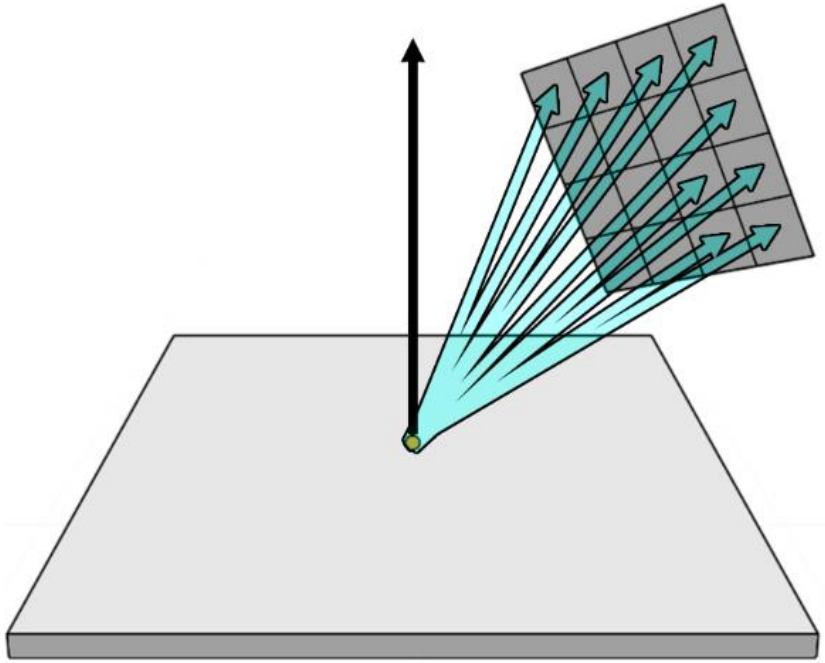
$$L_{e(x,\vec{\omega})} + \frac{\rho}{\pi} \int_{\Omega} L_i\left(x, \overrightarrow{\omega'}\right) \cos\theta' \, d\omega'$$



$$L_{e(x,\vec{\omega})} + \frac{1}{|\Omega|} \frac{\rho}{\pi} \sum L_i\left(x, \overrightarrow{\omega'}\right) \cos\theta'$$



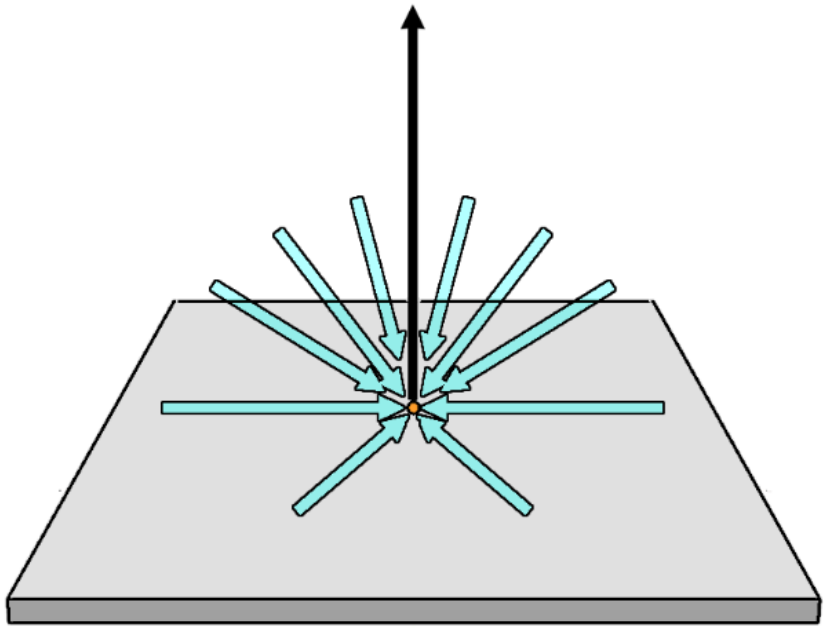
$$L_{e(x,\vec{\omega})} + \frac{\rho}{\pi}\int_S L_o\left(x', \overrightarrow{\omega'}\right) V(x,x') \frac{\cos\theta \cos\theta'}{|x-x'|^2} d\omega'$$



$$L_{e(x,\vec{\omega})} + \frac{1}{|S|}\frac{\rho}{\pi}\sum L_o\left(x', \overrightarrow{\omega'}\right) V(x,x') \frac{\cos\theta \cos\theta'}{|x-x'|^2}$$

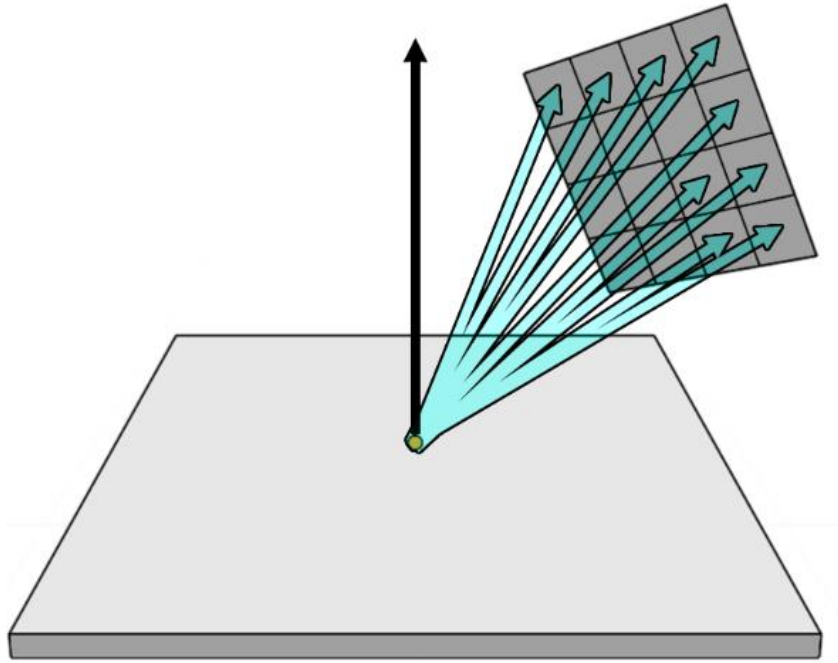
Raytracing

$$L_{e(x,\vec{\omega})} + \frac{1}{|\Omega|}\frac{\rho}{\pi}\sum L_i\left(x, \overrightarrow{\omega'}\right)\cos\theta'$$



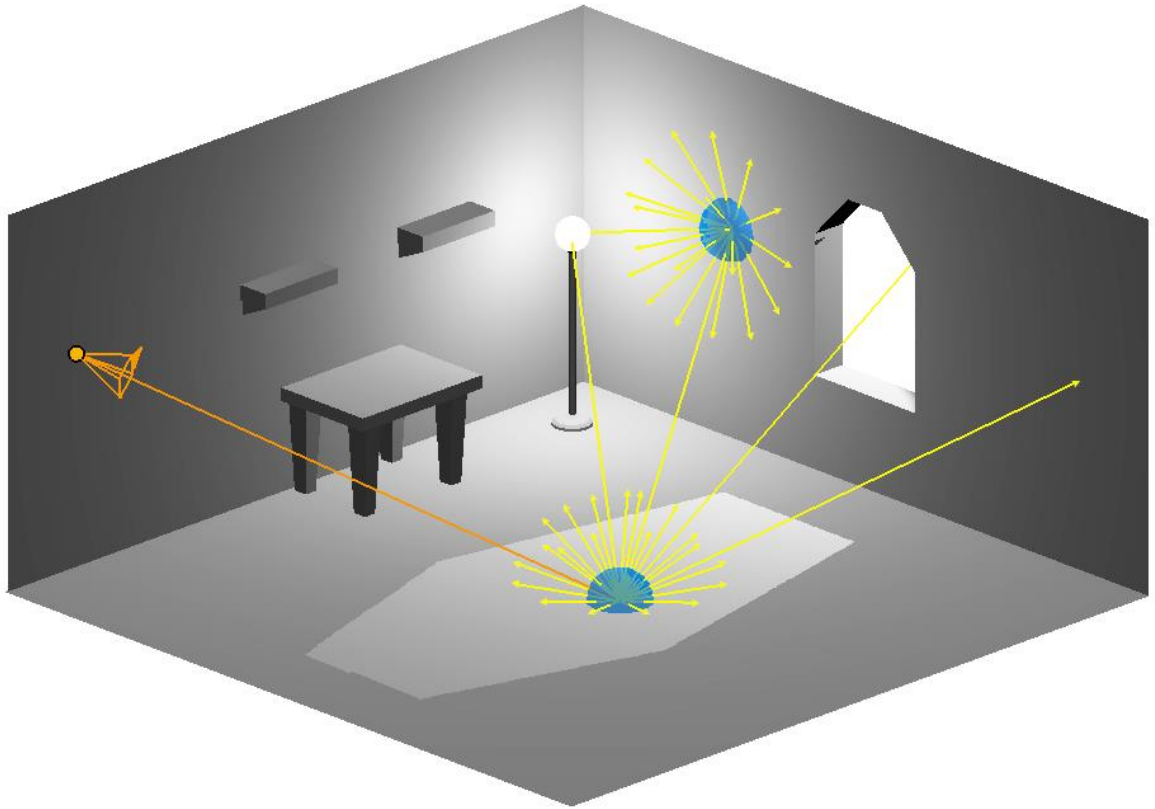
Radiosity

$$L_{e(x,\vec{\omega})} + \frac{1}{|S|}\frac{\rho}{\pi}\sum L_o\left(x', \overrightarrow{\omega'}\right)V(x,x')\frac{\cos\theta\cos\theta'}{|x-x'|^2}$$



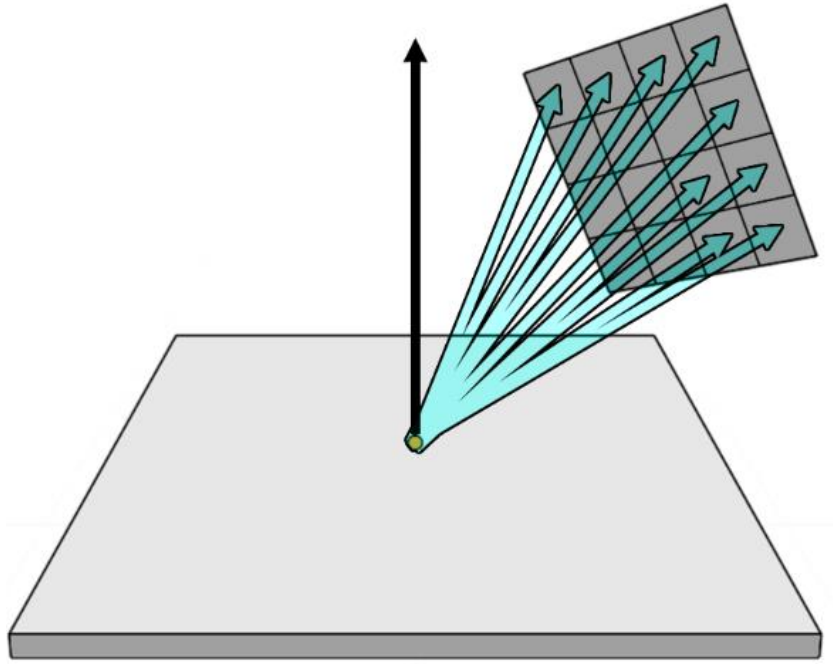
Raytracing

$$L_{e(x,\vec{\omega})} + \frac{1}{|\Omega|}\frac{\rho}{\pi}\sum L_i\left(x, \overrightarrow{\omega'}\right)\cos\theta'$$



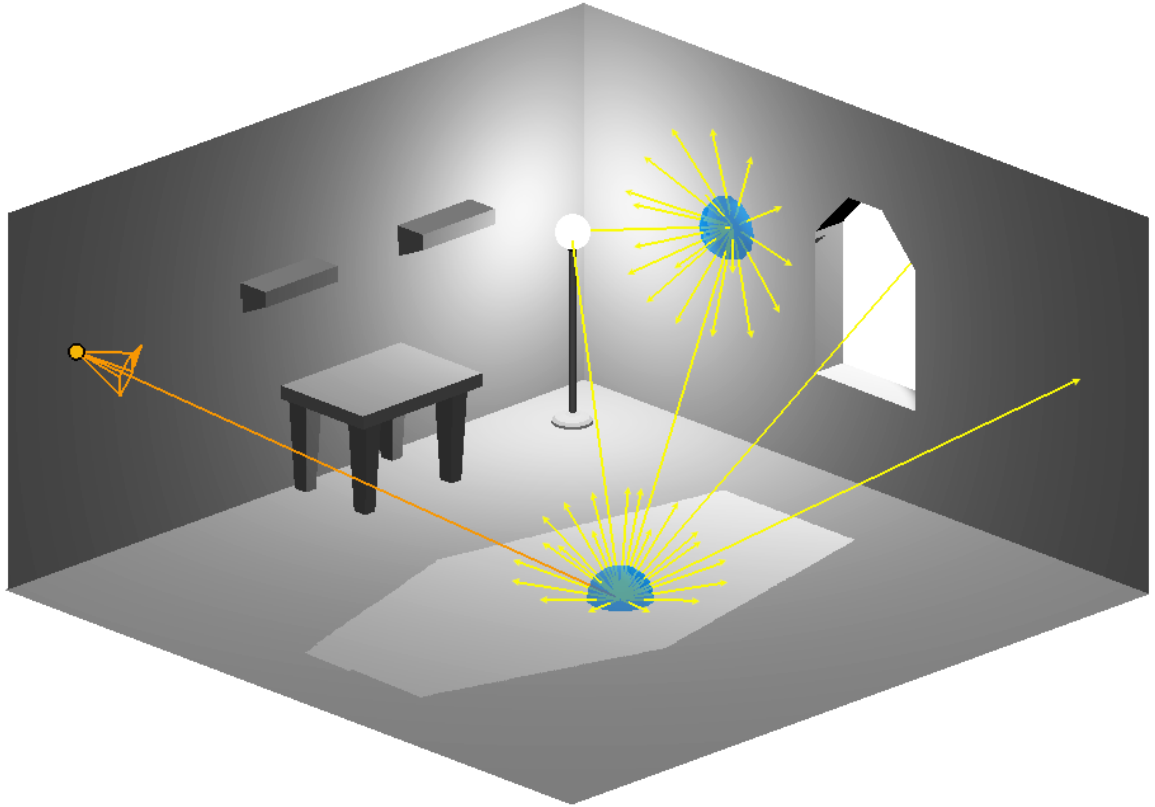
Radiosity

$$L_{e(x,\vec{\omega})} + \frac{1}{|S|}\frac{\rho}{\pi}\sum L_o\left(x', \overrightarrow{\omega'}\right)V(x,x')\frac{\cos\theta\cos\theta'}{|x-x'|^2}$$



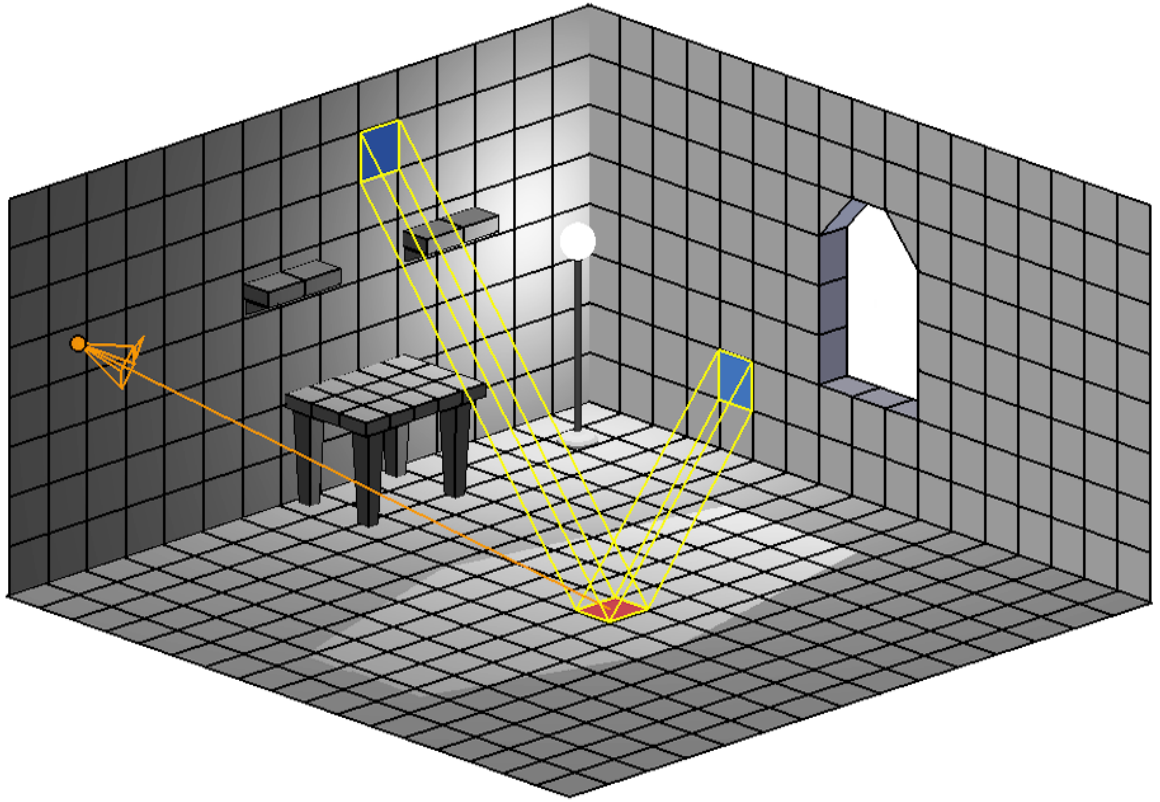
Raytracing

$$L_{e(x,\vec{\omega})} + \frac{1}{|\Omega|}\frac{\rho}{\pi}\sum L_i\left(x, \overrightarrow{\omega'}\right)\cos\theta'$$



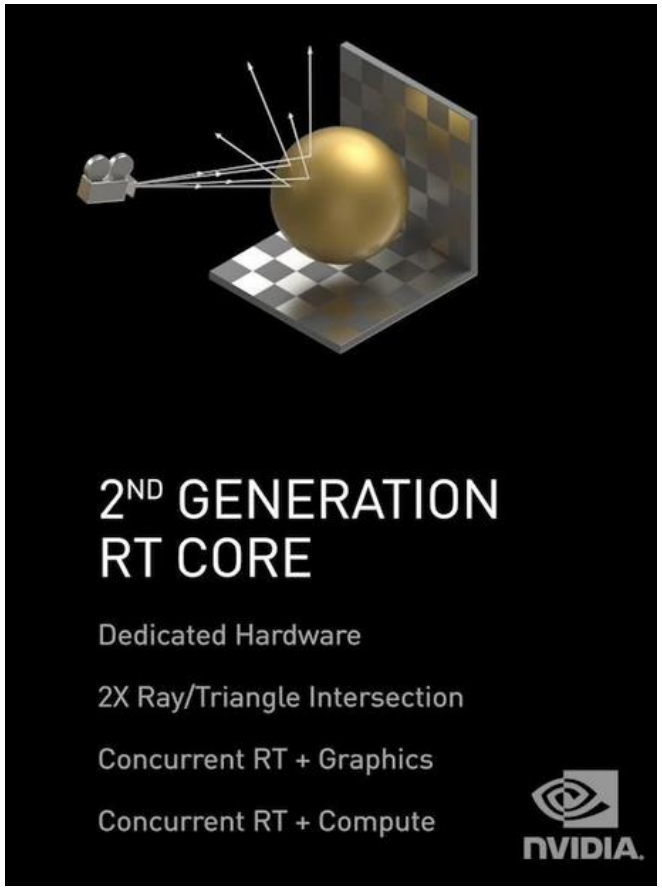
Radiosity

$$L_{e(x,\vec{\omega})} + \frac{1}{|S|}\frac{\rho}{\pi}\sum L_o\left(x', \overrightarrow{\omega'}\right)V(x,x')\frac{\cos\theta\cos\theta'}{|x-x'|^2}$$



Raytracing

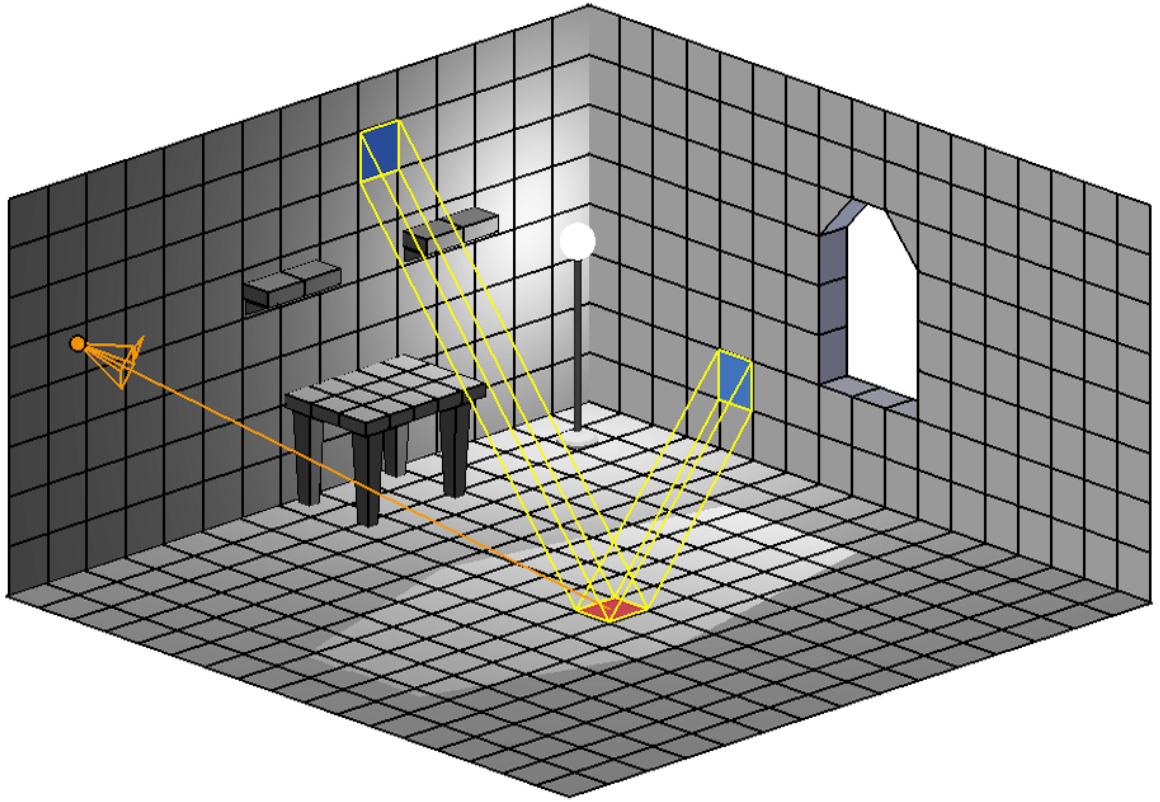
$$L_{e(x,\vec{\omega})} + \frac{1}{|\Omega|}\frac{\rho}{\pi}\sum L_i\left(x, \overrightarrow{\omega'}\right)\cos\theta'$$



From*: GeForce RTX 30 Series Launch Event*

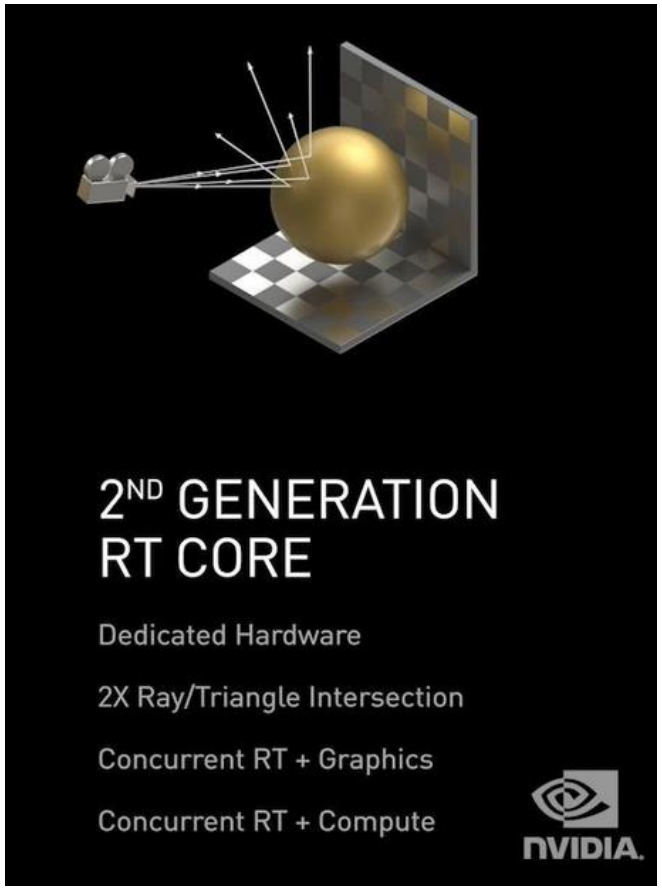
Radiosity

$$L_{e(x,\vec{\omega})} + \frac{1}{|S|}\frac{\rho}{\pi}\sum L_o\left(x', \overrightarrow{\omega'}\right)V(x,x')\frac{\cos\theta\cos\theta'}{|x-x'|^2}$$



Raytracing

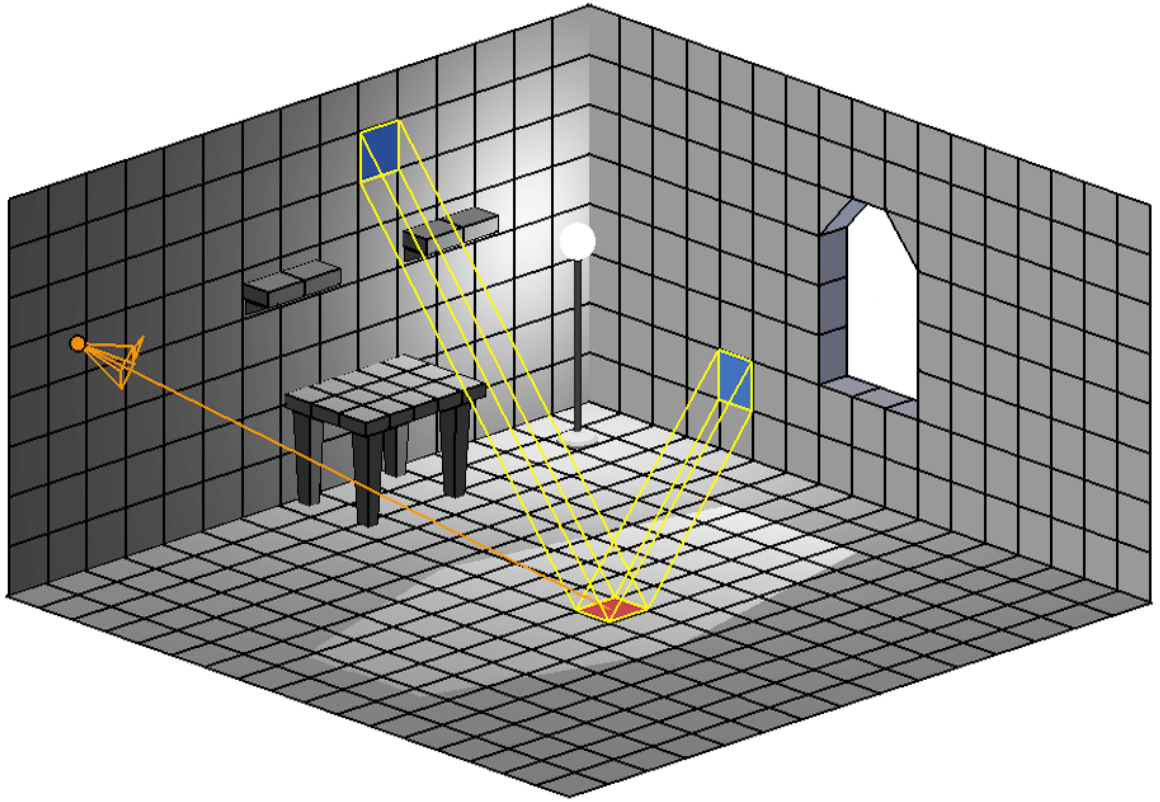
$$L_{e(x,\vec{\omega})} + \frac{1}{|\Omega|}\frac{\rho}{\pi}\sum L_i(x,\overrightarrow{\omega'})\cos\theta'$$



From*: GeForce RTX 30 Series Launch Event*

Radiosity

$$L_{e(x,\vec{\omega})} + \frac{1}{|S|}\frac{\rho}{\pi}\sum L_o(x',\overrightarrow{\omega'})V(x,x')\frac{\cos\theta\cos\theta'}{|x-x'|^2}$$



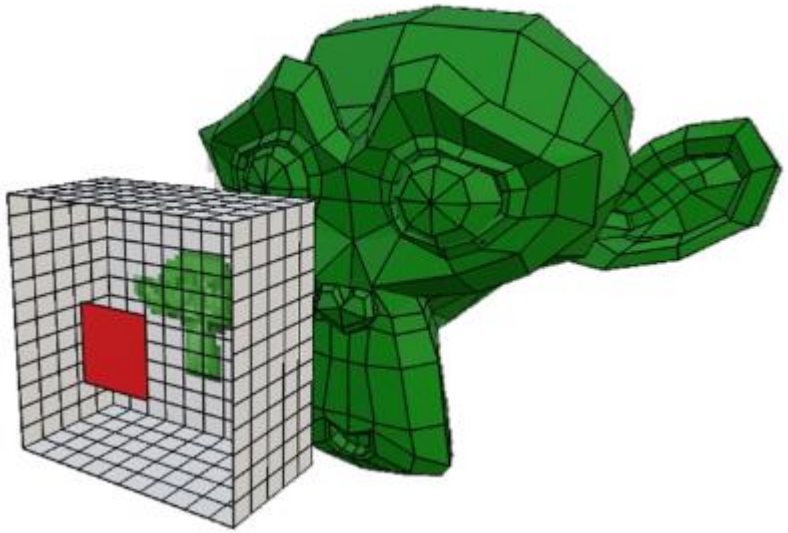
Raytracing

$$L_{e(x,\vec{\omega})} + \frac{1}{|\Omega|}\frac{\rho}{\pi}\sum L_i\left(x, \overrightarrow{\omega'}\right)\cos\theta'$$

Radiosity

$$L_{e(x,\vec{\omega})} + \frac{1}{|S|}\frac{\rho}{\pi}\sum L_o\left(x', \overrightarrow{\omega'}\right)V(x,x')\frac{\cos\theta\cos\theta'}{|x-x'|^2}$$

Z-Buffering



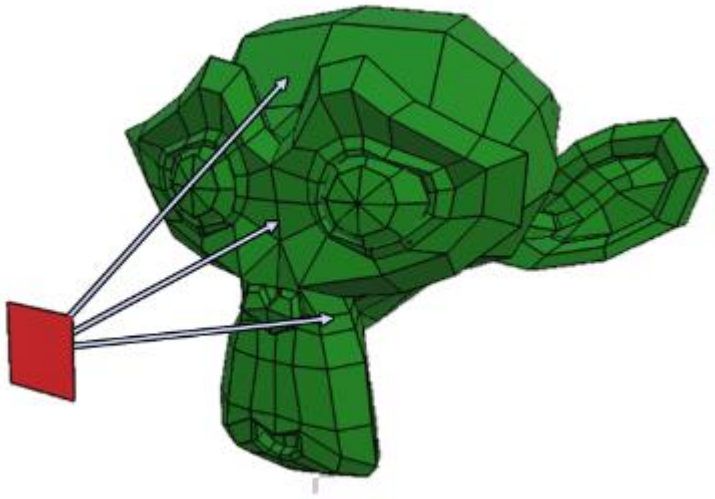
Raytracing

$$L_{e(x,\vec{\omega})} + \frac{1}{|\Omega|}\frac{\rho}{\pi}\sum L_i\left(x, \overrightarrow{\omega'}\right)\cos\theta'$$

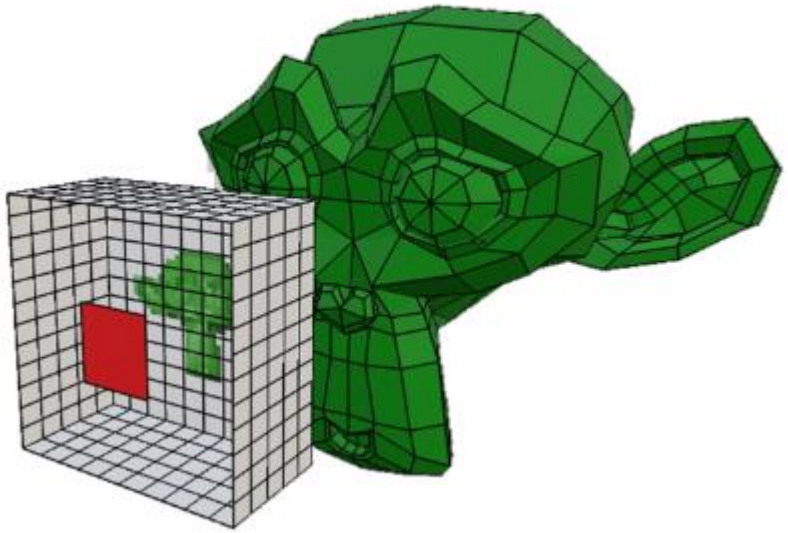
Radiosity

$$L_{e(x,\vec{\omega})} + \frac{1}{|S|}\frac{\rho}{\pi}\sum L_o\left(x', \overrightarrow{\omega'}\right)V(x,x')\frac{\cos\theta\cos\theta'}{|x-x'|^2}$$

RTX



Z-Buffering



Raytracing

$$L_{e(x,\vec{\omega})} + \frac{1}{|\Omega|}\frac{\rho}{\pi}\sum L_i\left(x, \overrightarrow{\omega'}\right)\cos\theta'$$

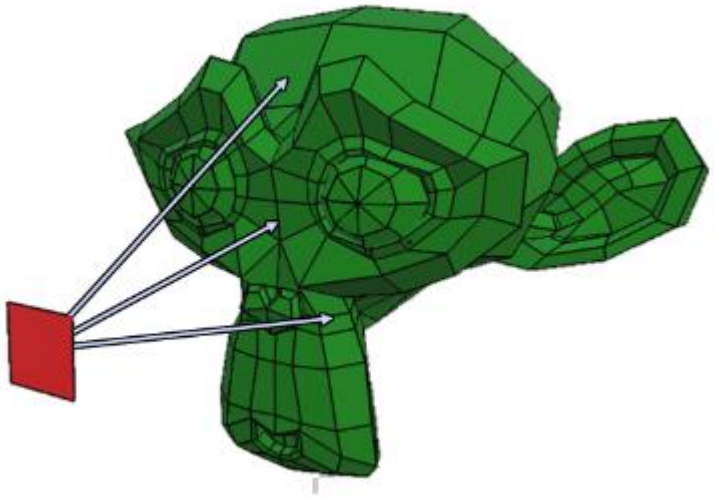
Radiosity

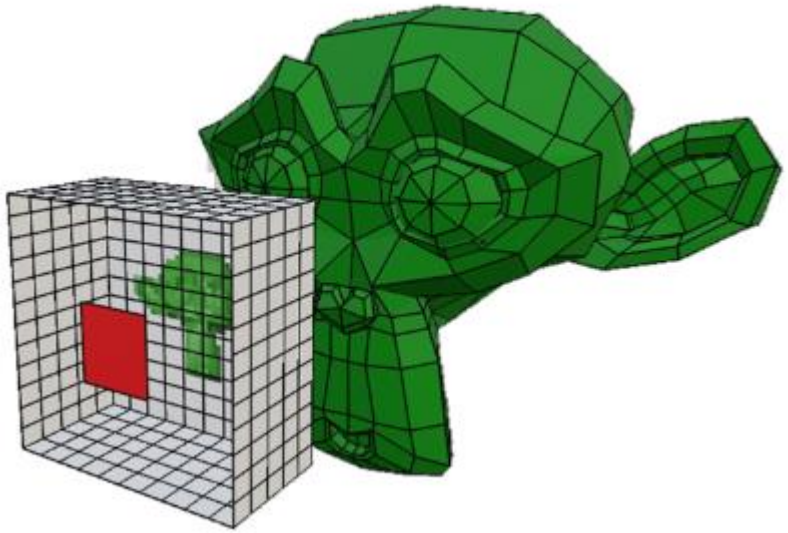
$$L_{e(x,\vec{\omega})} + \frac{1}{|S|}\frac{\rho}{\pi}\sum L_o\left(x', \overrightarrow{\omega'}\right)V(x,x')\frac{\cos\theta\cos\theta'}{|x-x'|^2}$$

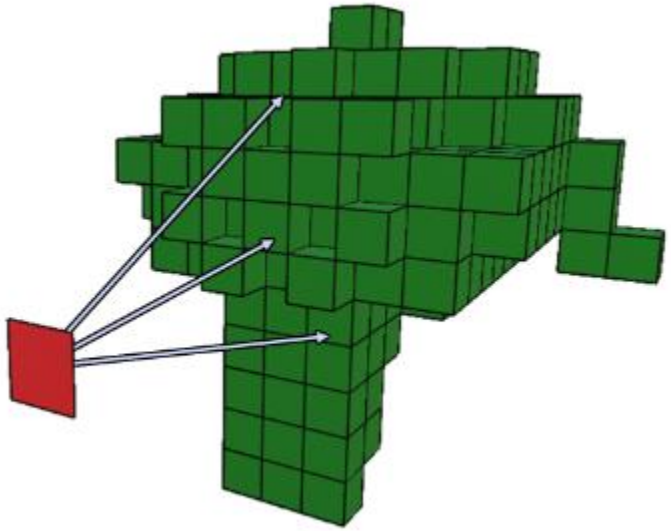
RTX

Z-Buffering

Voxel-Raymarching







RTX Radiosity?

Goals:

- Determine if RTX can be used to accelerate Radiosity.

RTX Radiosity?

Goals:

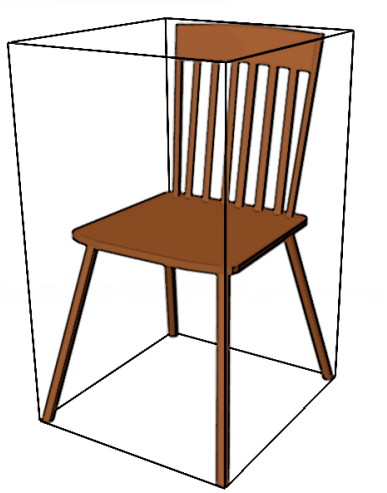
- Determine if RTX can be used to accelerate Radiosity.
- Demonstrate this through a competitively fast implementation.

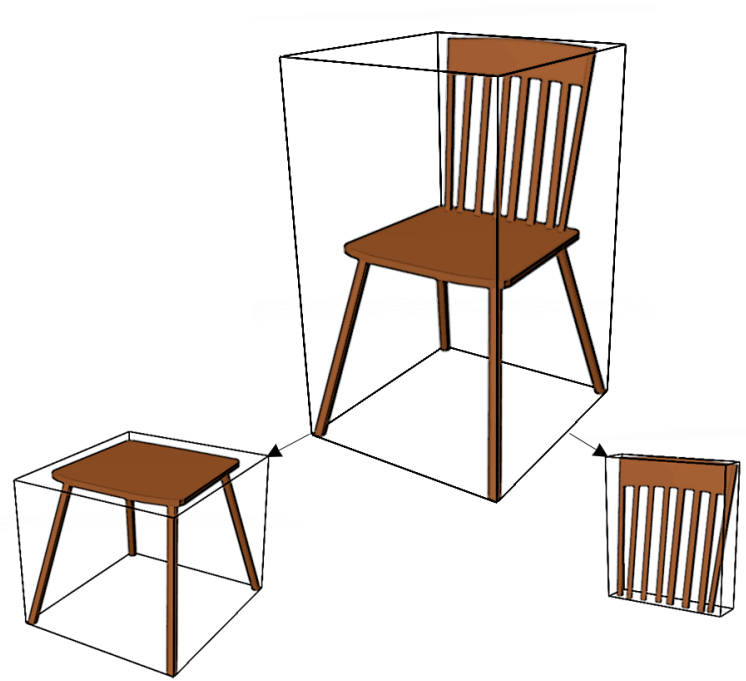
RTX Radiosity?

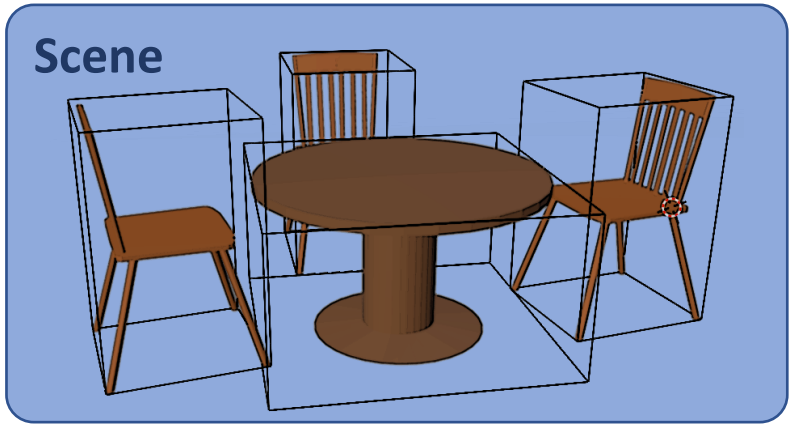
Goals:

- Determine if RTX can be used to accelerate Radiosity.
- Demonstrate this through a competitively fast implementation.
- Find the best practices for RTX radiosity.

Bounding Volume Hierarchies

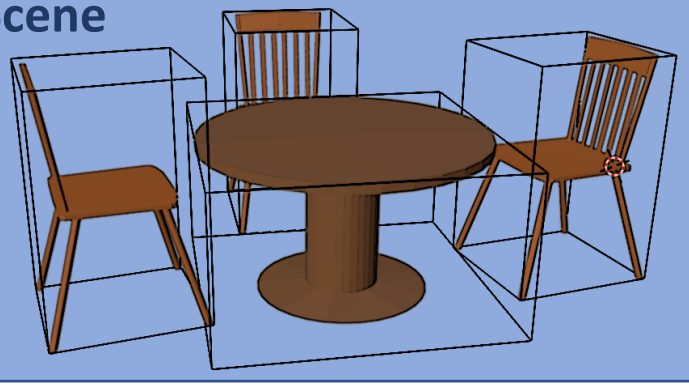





Scene

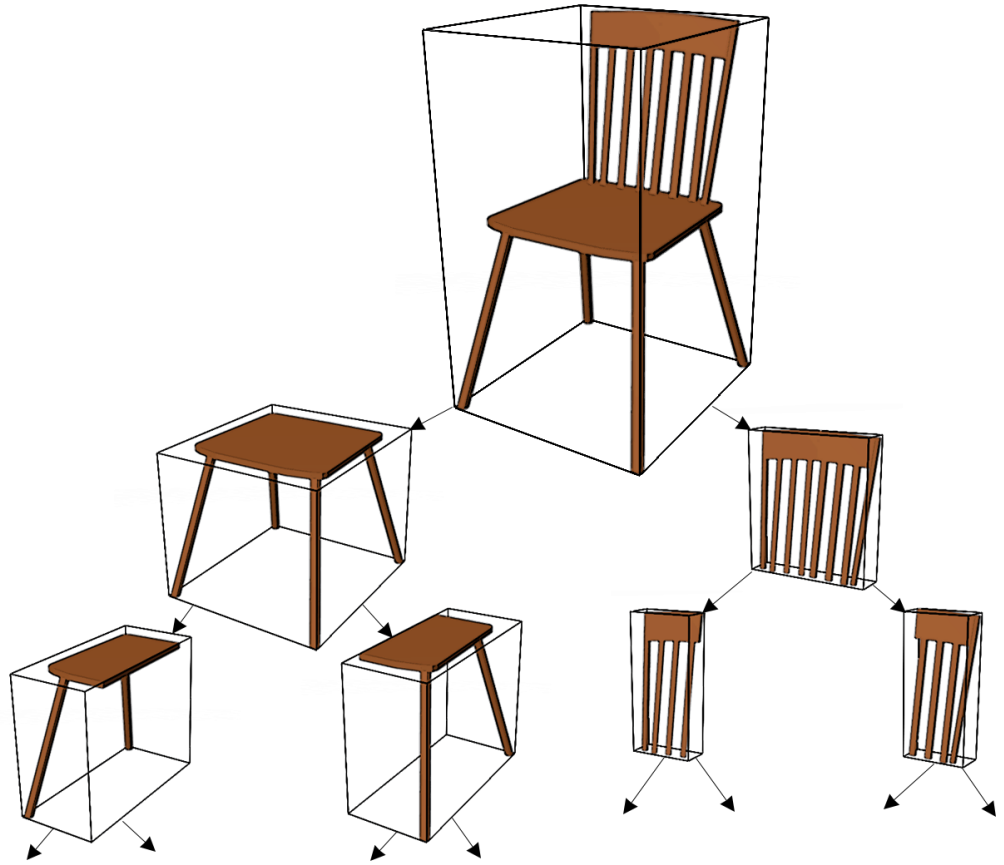

...
...

Scene



TLAS

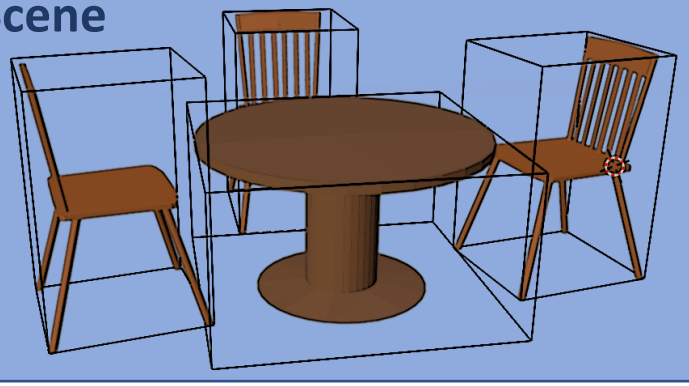
Chair 1	Chair 2	Chair 3	Table 1
$M_1 = [\dots]$	$M_2 = [\dots]$	$M_3 = [\dots]$	$M_4 = [\dots]$



...

...

Scene



TLAS

Chair 1

$M_1 = [\dots]$

Chair 2

$M_2 = [\dots]$

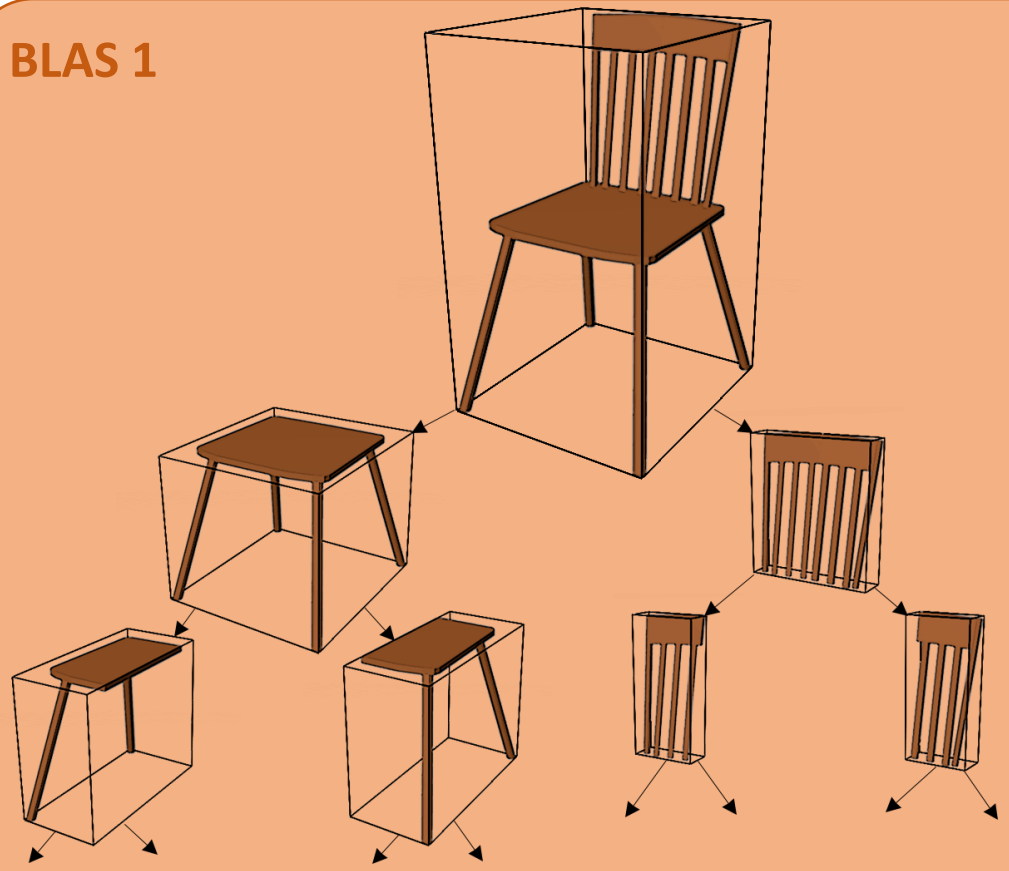
Chair 3

$M_3 = [\dots]$

Table 1

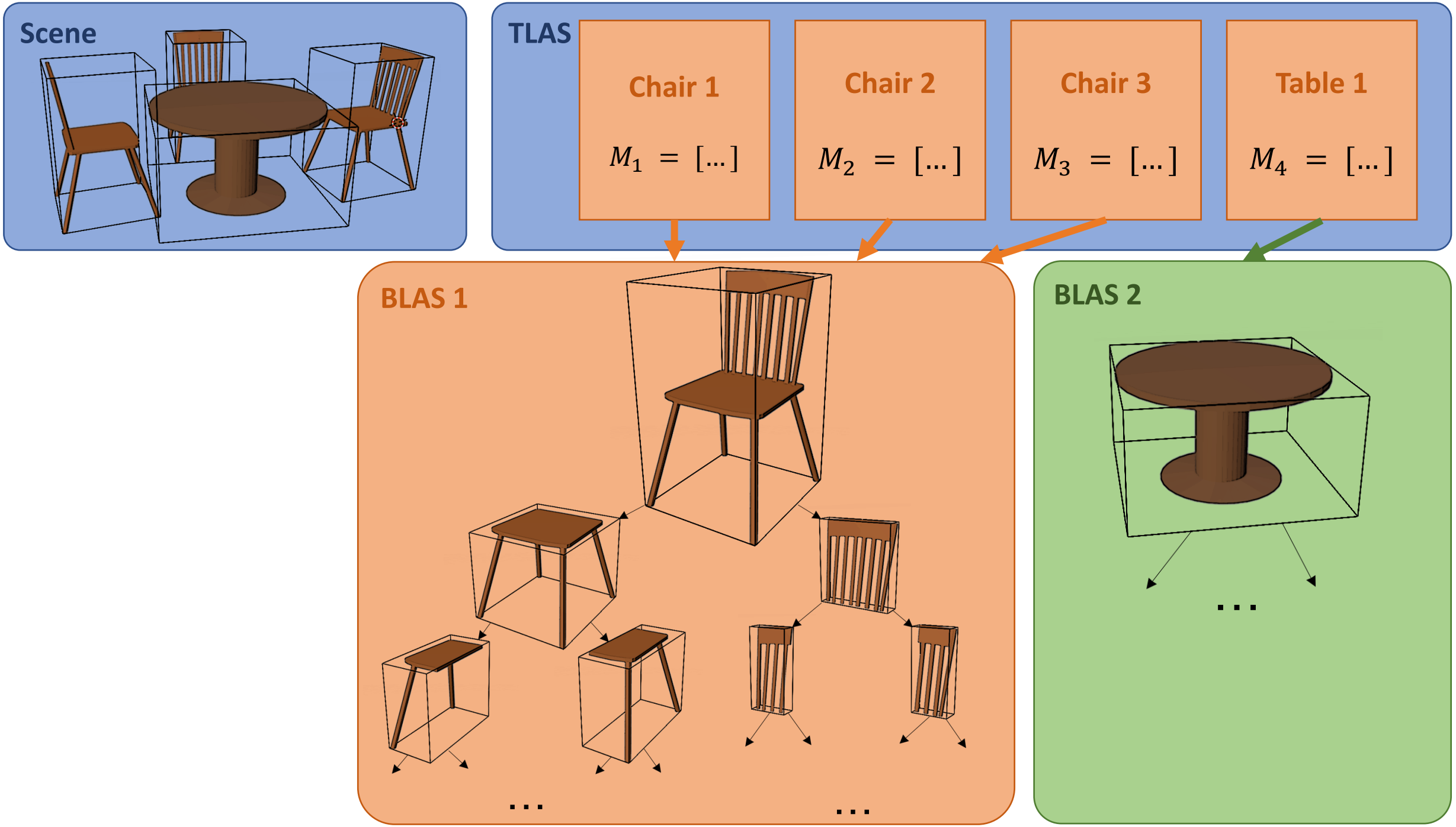
$M_4 = [\dots]$

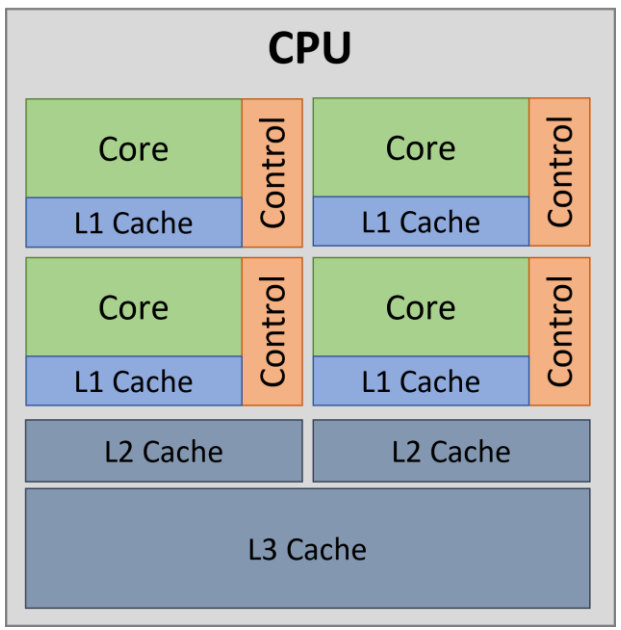
BLAS 1

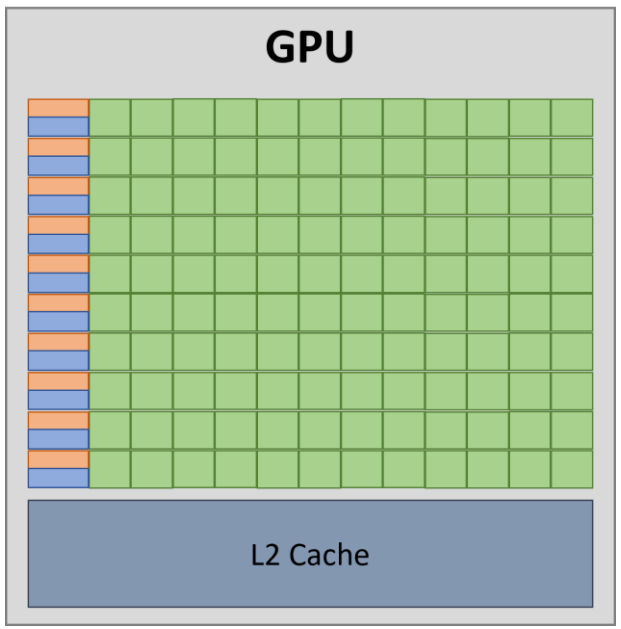


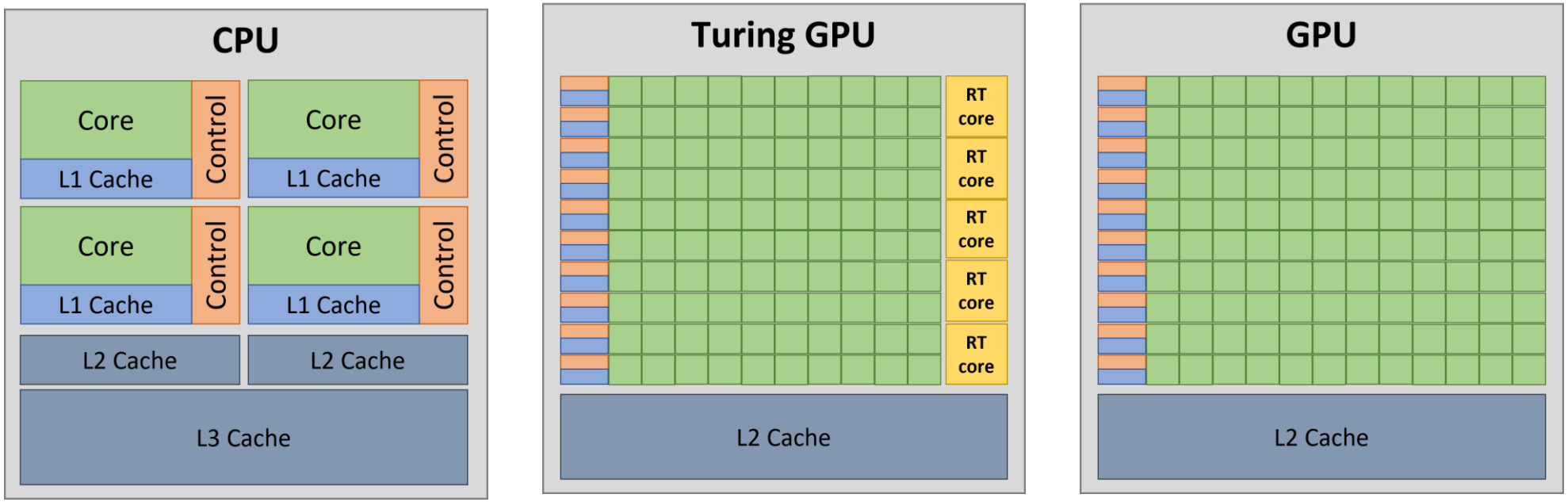
...

...


Scene
TLAS
Chair 1
$M_1 = [\ldots]$
Chair 2
$M_2 = [\ldots]$
Chair 3
$M_3 = [\ldots]$
Table 1
$M_4 = [\ldots]$
BLAS 1
...
...
BLAS 2
...


CPU
Core
Control
L1 Cache
Core
Control
L1 Cache
Core
Control
L1 Cache
Core
Control
L1 Cache
L2 Cache
L2 Cache
L3 Cache

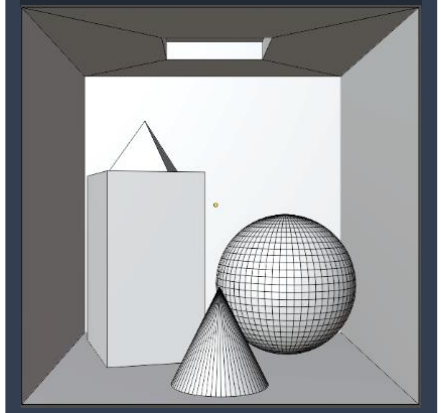

GPU
L2 Cache


CPU
Core
Control
L1 Cache
Core
Control
L1 Cache
Core
Control
L1 Cache
Core
Control
L1 Cache
L2 Cache
L2 Cache
L3 Cache
Turing GPU
RT core
RT core
RT core
RT core
RT core
L2 Cache
GPU
L2 Cache

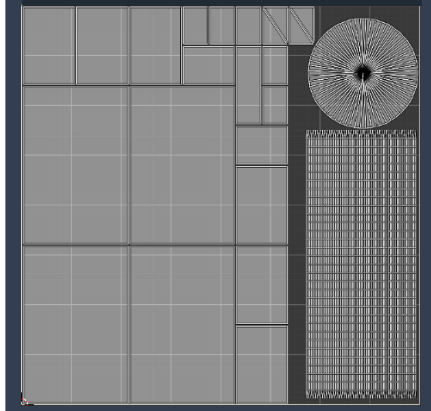
RTRad

Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

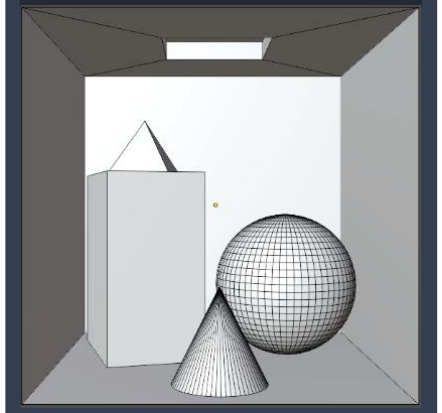
Material2 (color, L_e) …

…

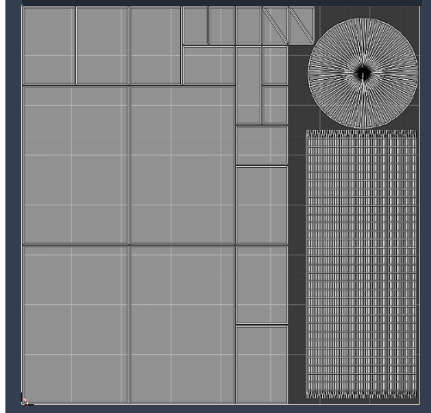
RTRad

Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

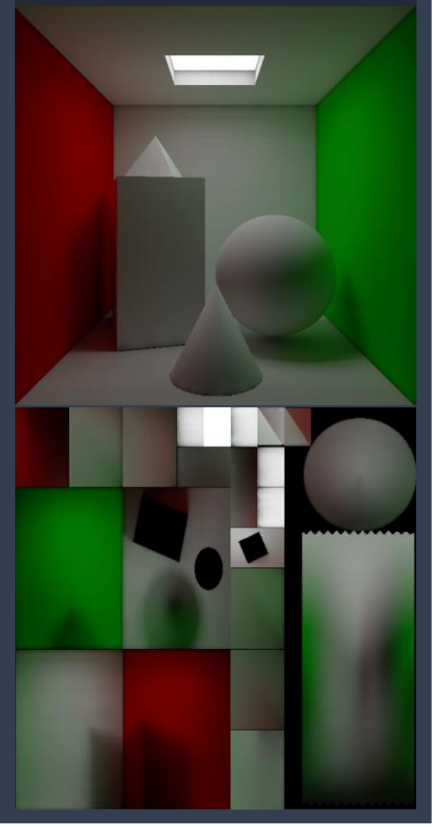
Material2 (color, L_e) …

…

RTRad

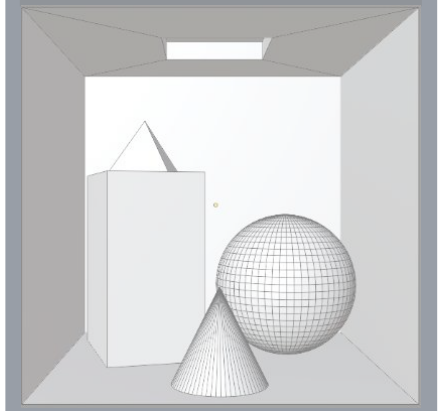
Output

Lightmap

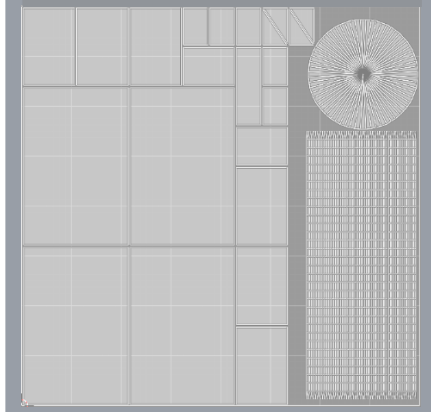


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

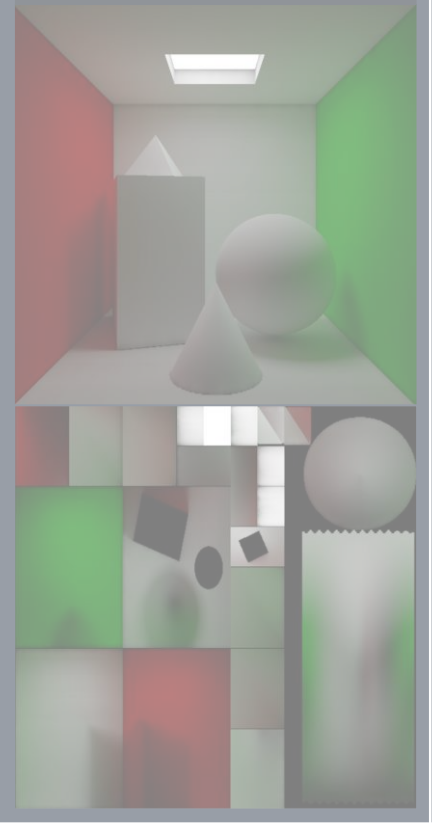
RTRad

RTLPass

$For\ i\ \in S$

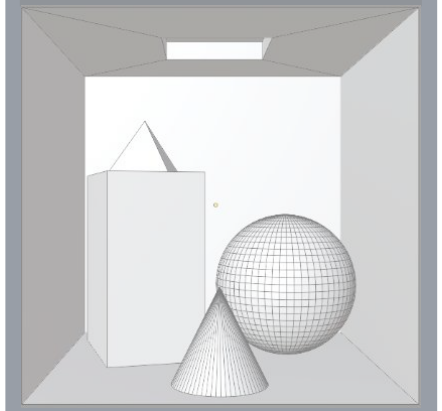
Output

Lightmap

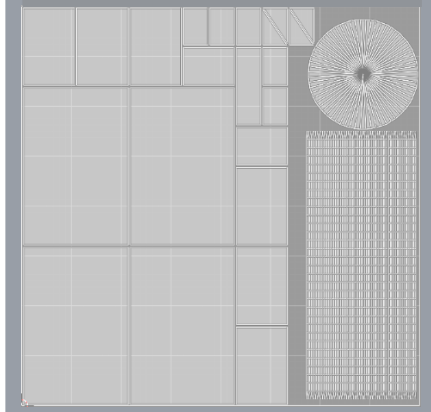


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

RTRad

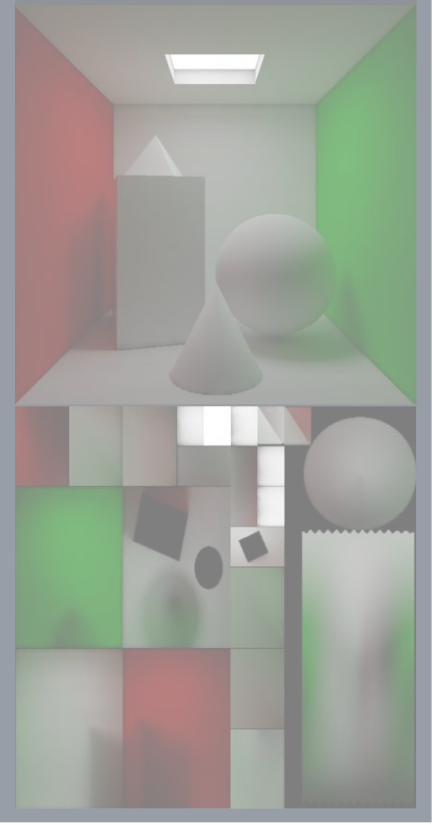
RTLPass

$For\ i \in S$

$For\ j \in S \backslash i$

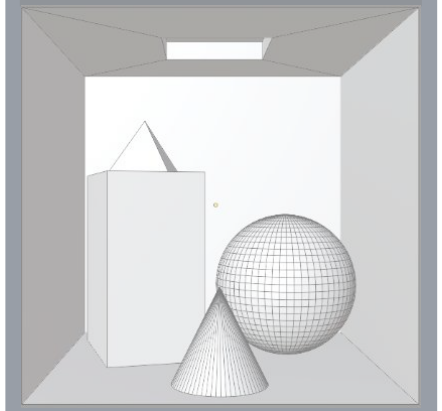
Output

Lightmap

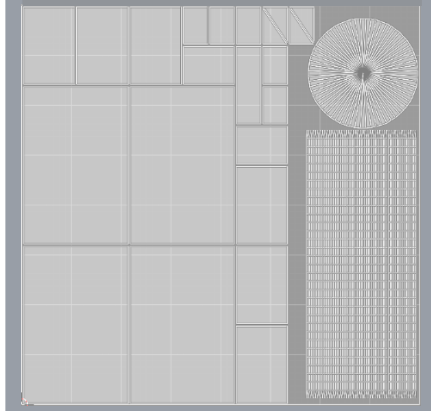


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

RTRad

RTLPass

For $i \in S$

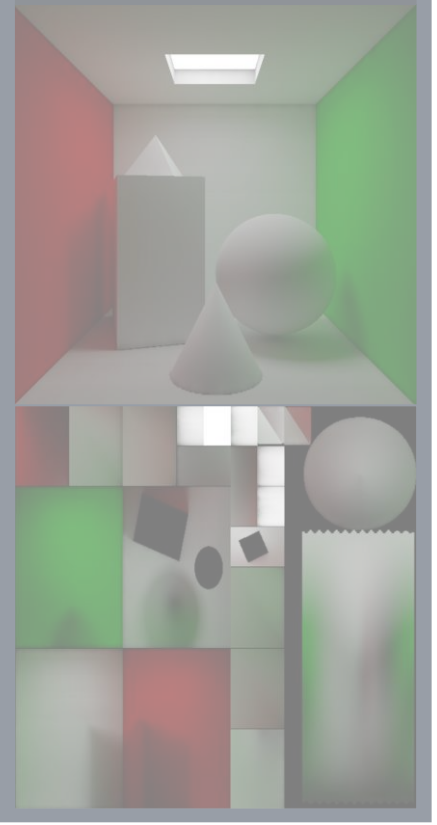
For $j \in S \backslash i$

Trace ray from i to j

Miss shader

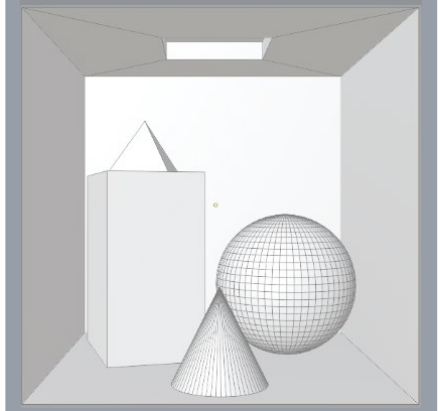
Output

Lightmap

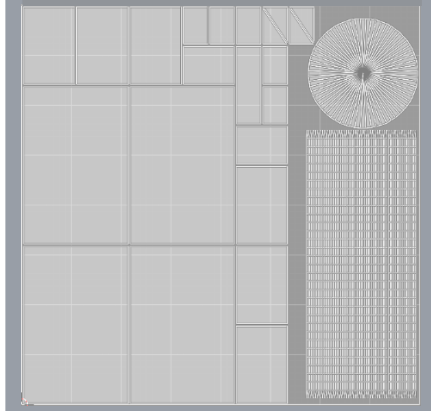


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

RTRad

RTLPass

For $i \in S$

For $j \in S \backslash i$

Trace ray from i to j

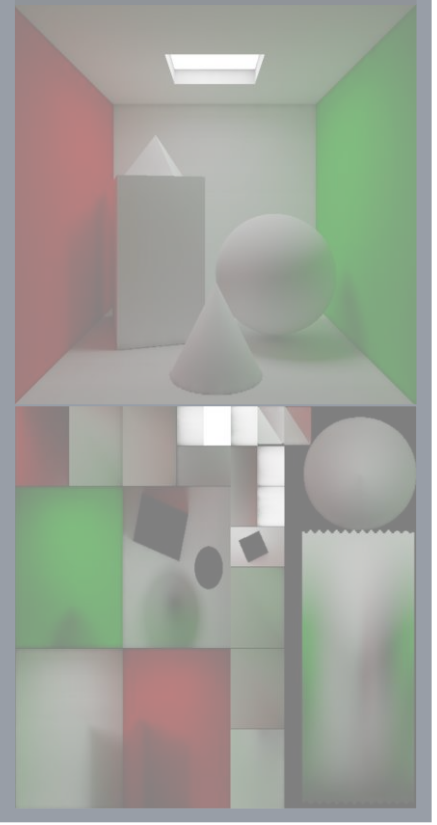
Miss shader

$$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$$

$$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$$

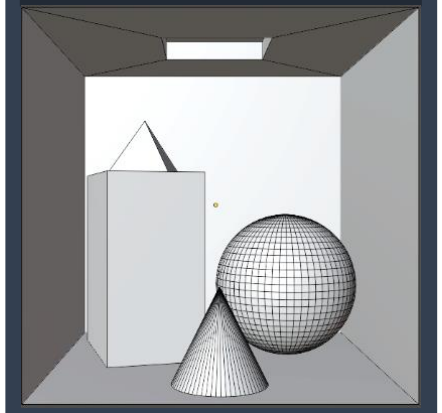
Output

Lightmap

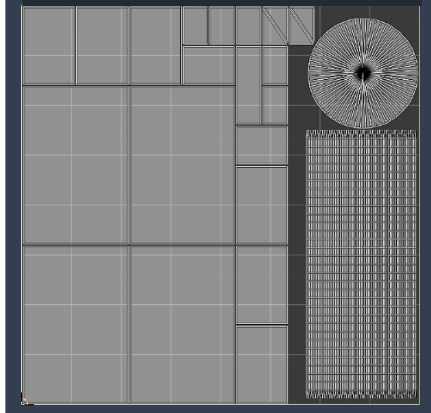


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

RTRad

RTLPass

$For\ i\ \in S$

$For\ j\ \in S \backslash \mathrm{i}$

$Trace\ ray\ from\ i\ to\ j$

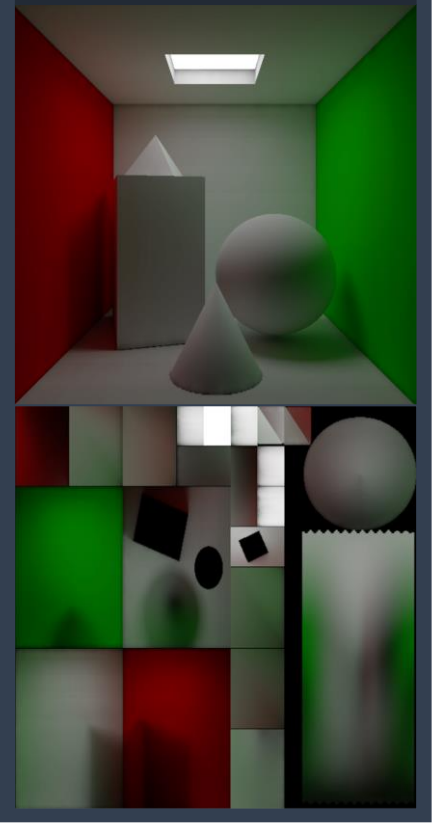
$Miss\ shader$

$$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$$

$$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$$

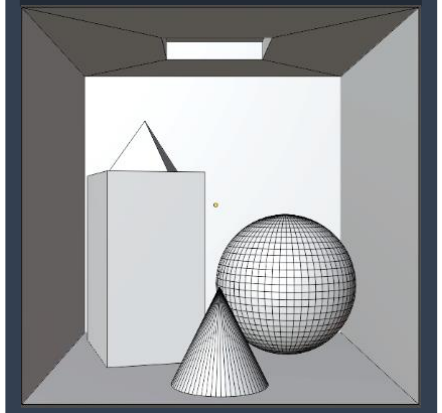
Output

Lightmap

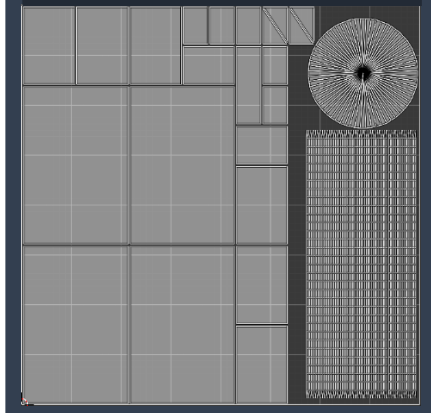


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

RTRad

RTLPass

$For\ i\ \in S$

$For\ j\ \in S \backslash i$

$Trace\ ray\ from\ i\ to\ j$

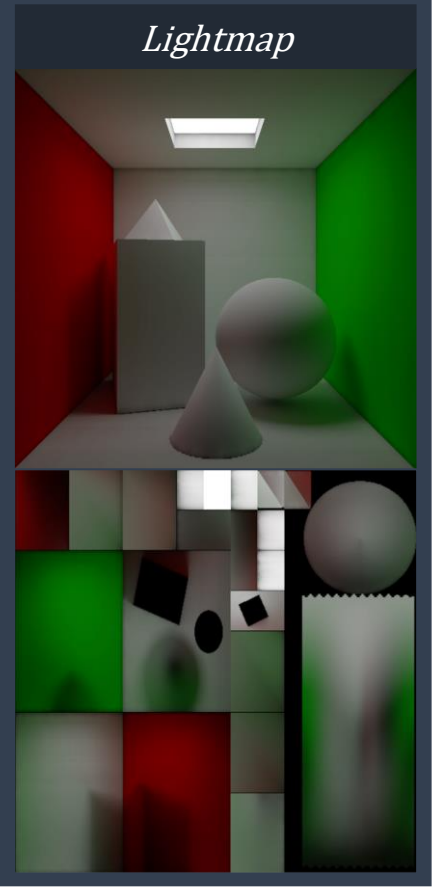
$Miss\ shader$

$$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$$

$$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$$

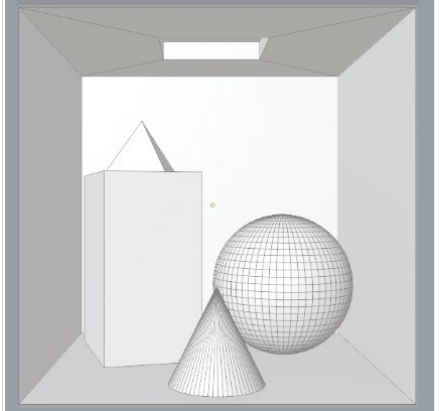
Output

Lightmap

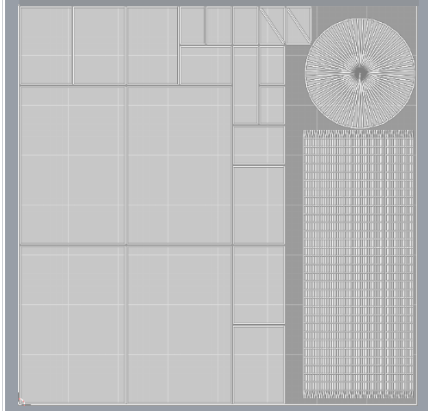


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

RTRad

CITPass

RTLPass

For $i \in S$

For $j \in S \backslash i$

Trace ray from i to j

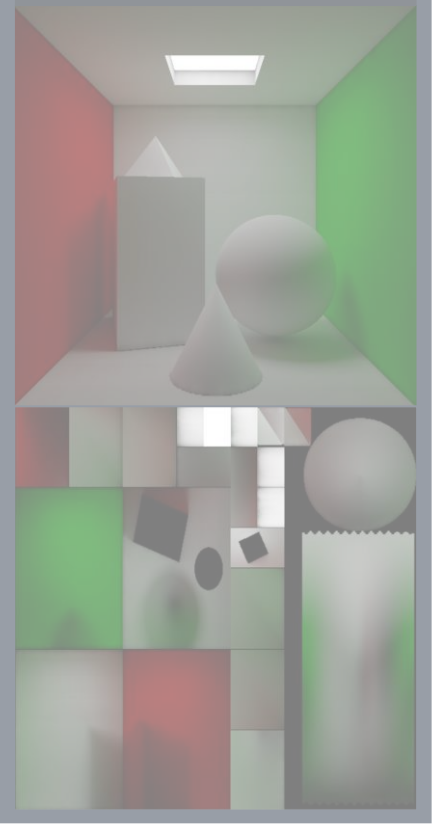
Miss shader

$$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$$

$$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$$

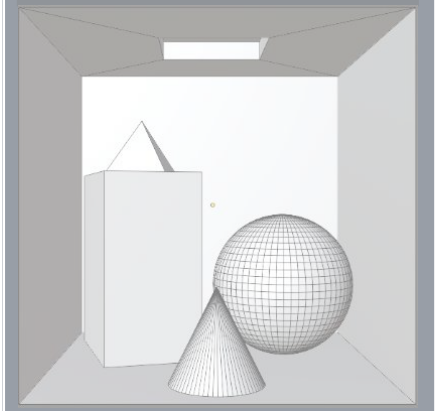
Output

Lightmap

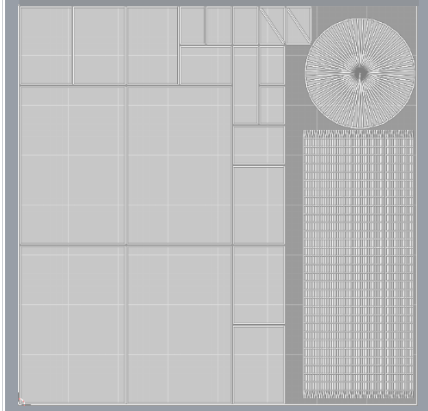


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

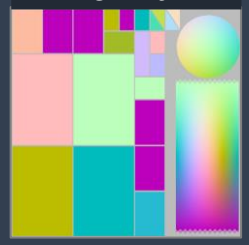
RTRad

CITPass

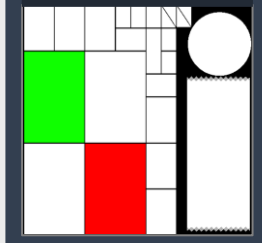
Texture Group

position

normal



material



surface area

RTLPass

For $i \in S$

For $j \in S \backslash i$

Trace ray from i to j

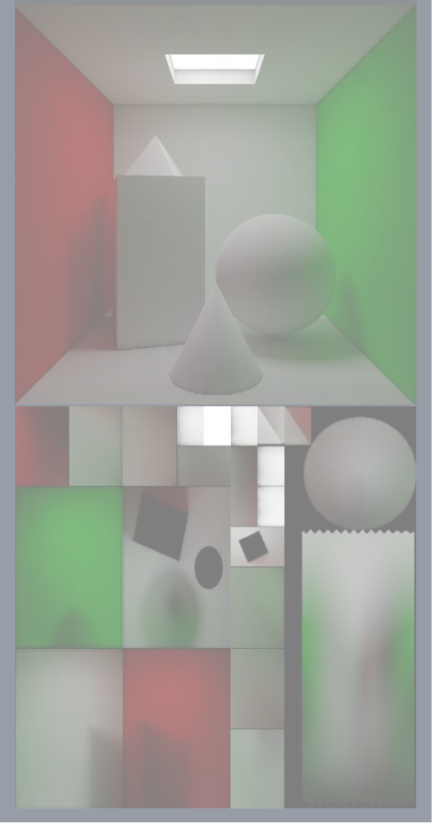
Miss shader

$$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$$

$$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$$

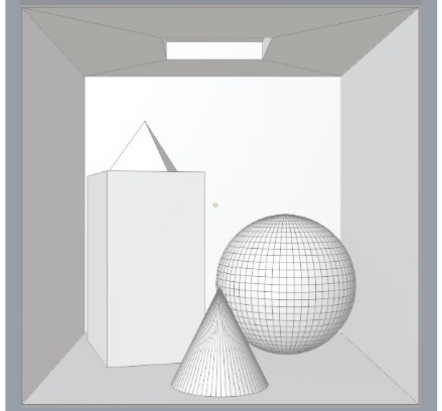
Output

Lightmap

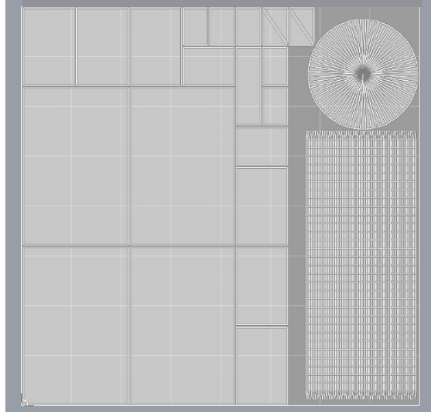


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

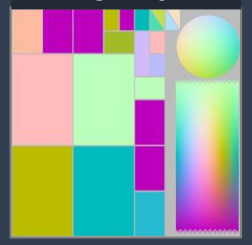
RTRad

CITPass

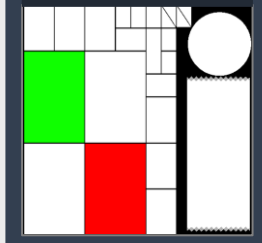
Texture Group

position

normal



material



surface area

L_{out}

L_{in}

RTLPass

$For\ i\ \in S$

$For\ j\ \in S \backslash i$

$Trace\ ray\ from\ i\ to\ j$

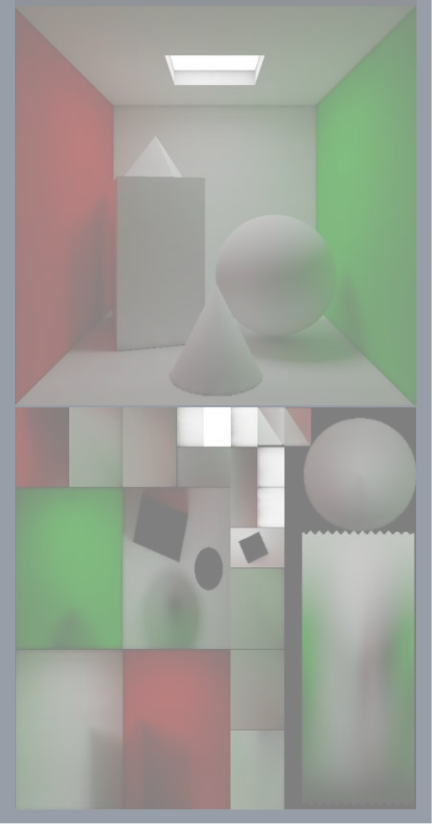
$Miss\ shader$

$$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$$

$$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$$

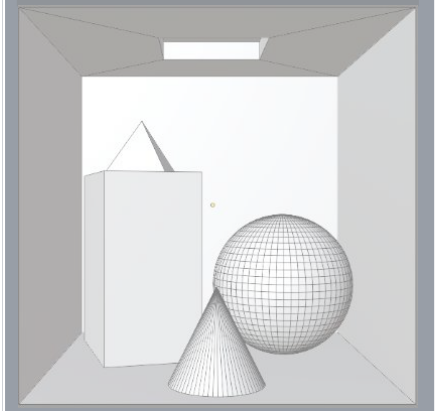
Output

Lightmap

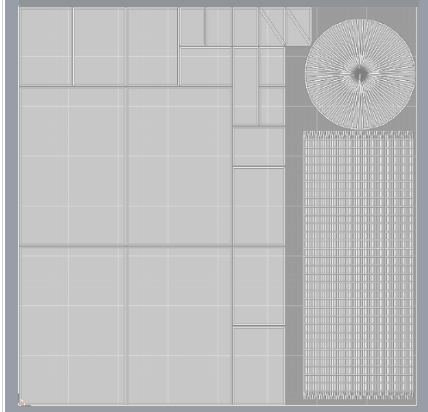


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

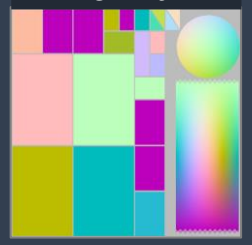
RTRad

CITPass

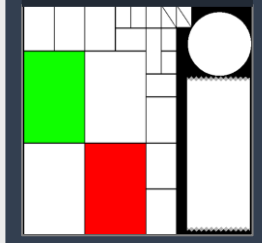
Texture Group

position

normal



material



surface area

L_{out}

L_{in}

RTLPass

$For\ i\ \in S$

$For\ j\ \in S \backslash i$

$Trace\ ray\ from\ i\ to\ j$

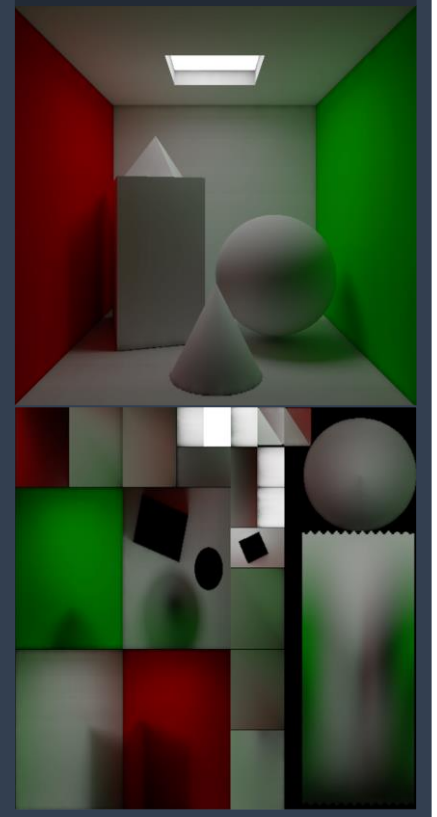
$Miss\ shader$

$$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$$

$$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$$

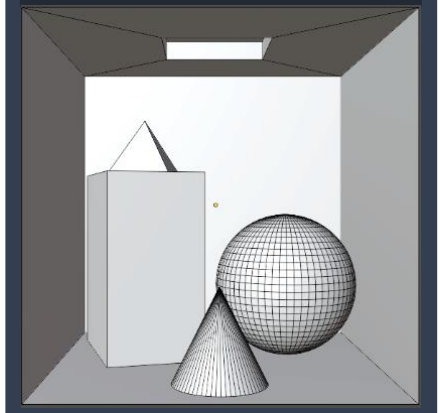
Output

Lightmap

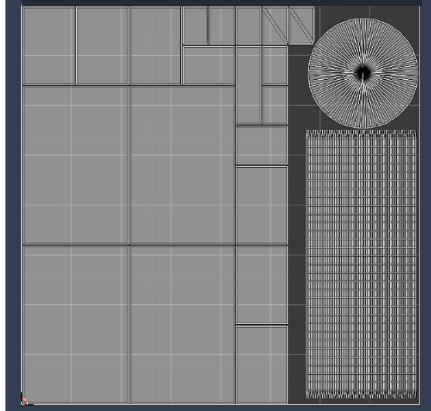


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

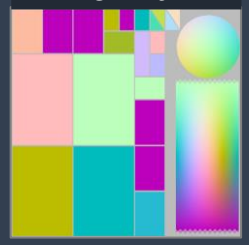
RTRad

CITPass

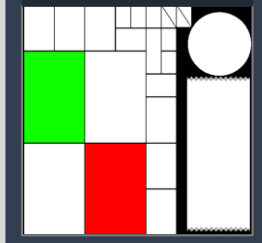
Texture Group

position

normal



material



surface area

L_{out}

L_{in}

RTLPass

$For\ i\ \in S$

$For\ j\ \in S \backslash i$

$Trace\ ray\ from\ i\ to\ j$

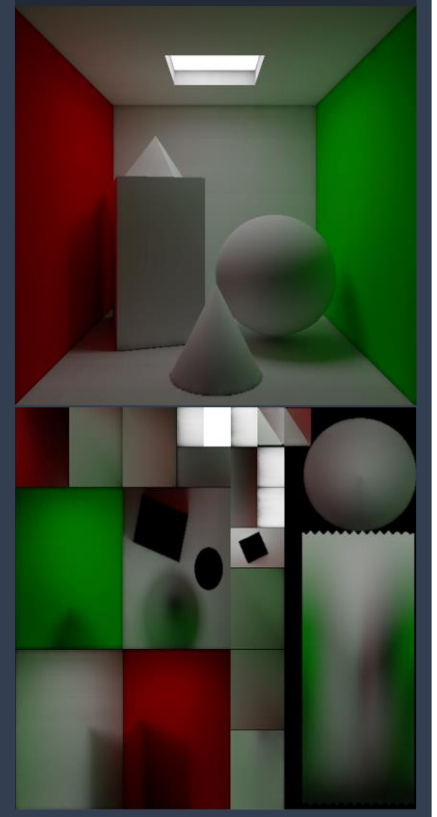
$Miss\ shader$

$$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$$

$$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$$

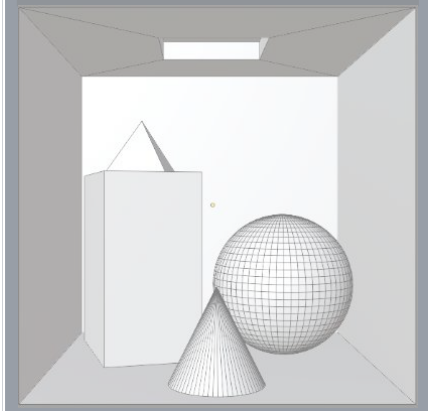
Output

Lightmap

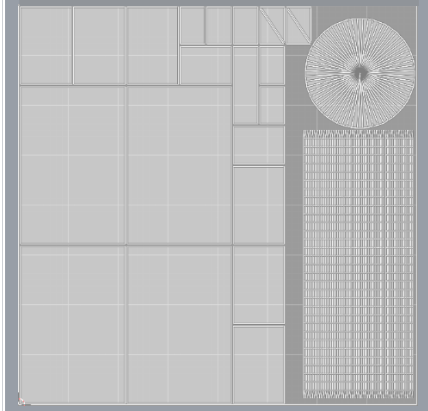


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

RTRad

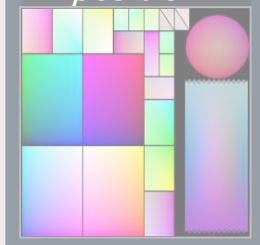
Acceleration Structure

TLAS

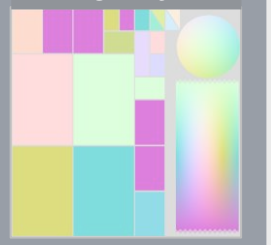
BLAS 0 *BLAS 1* …

Texture Group

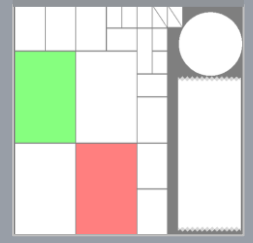
position



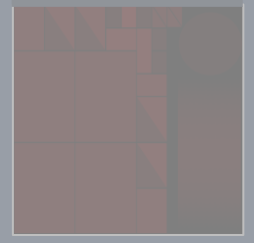
normal



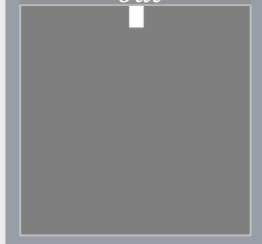
material



surface area



L_{out}



L_{in}

CITPass

RTLPass

For $i \in S$

For $j \in S \backslash i$

Trace ray from i to j

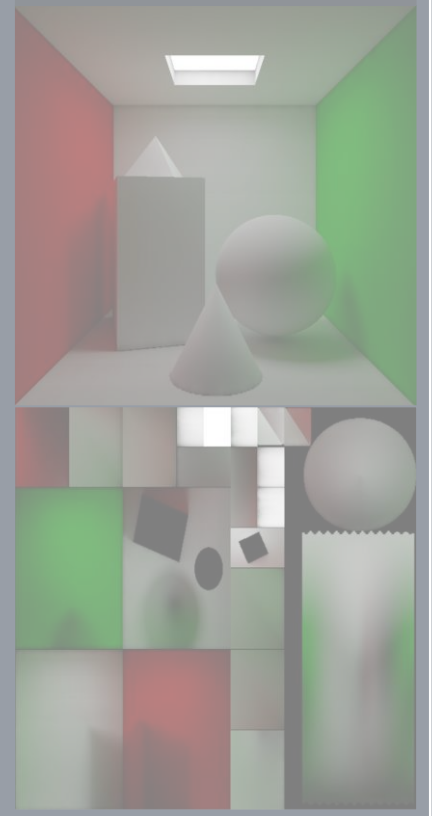
Miss shader

$$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$$

$$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$$

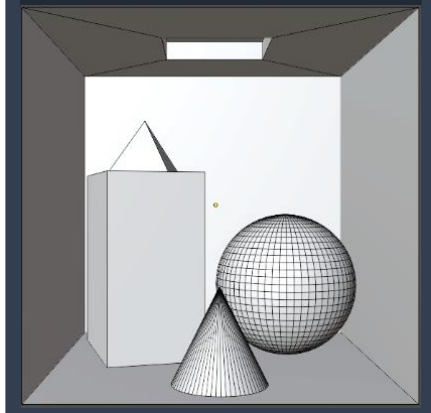
Output

Lightmap

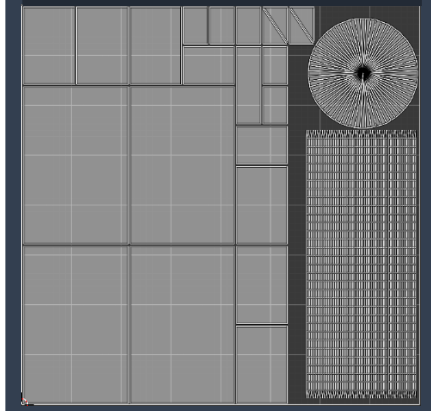


Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

RTRad

Acceleration Structure

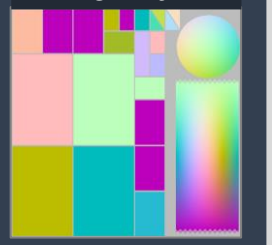
TLAS

BLAS 0 *BLAS 1* …

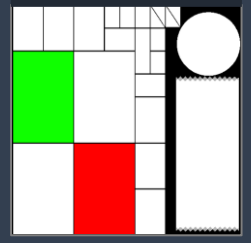
Texture Group

position

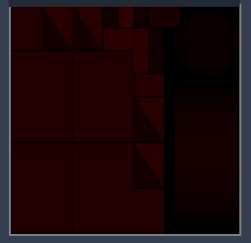
normal



material



surface area



L_{out}

L_{in}

CITPass

RTLPass

$For\ i \in S$

$For\ j \in S \backslash i$

$Trace\ ray\ from\ i\ to\ j$

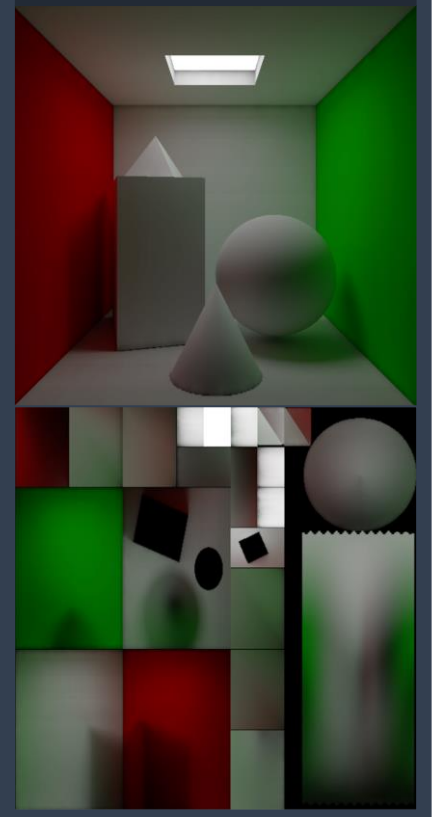
$Miss\ shader$

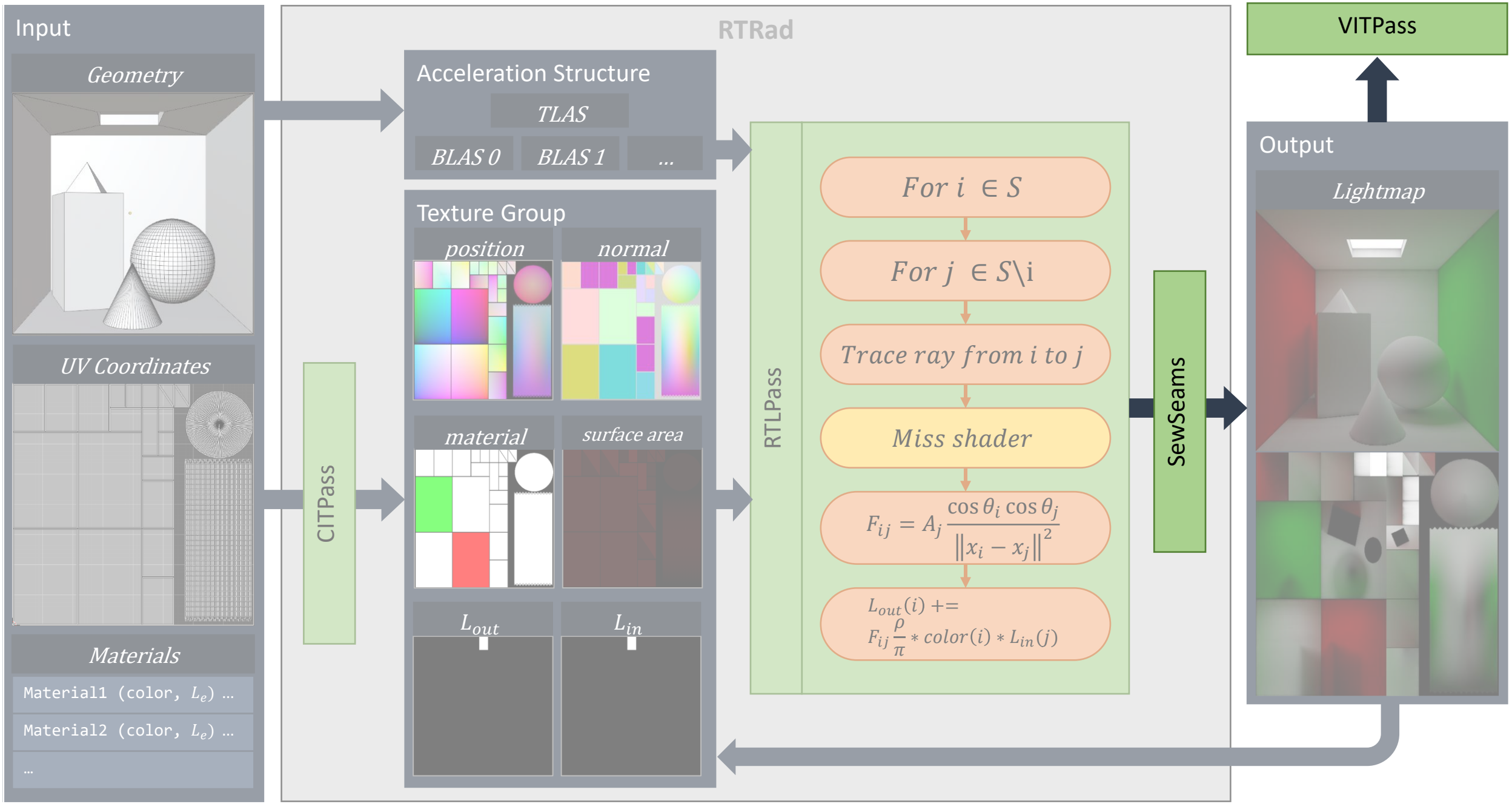
$$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$$

$$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$$

Output

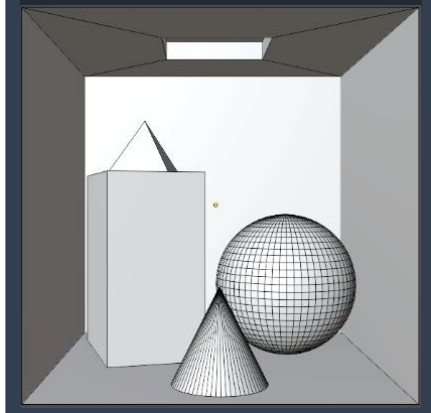
Lightmap



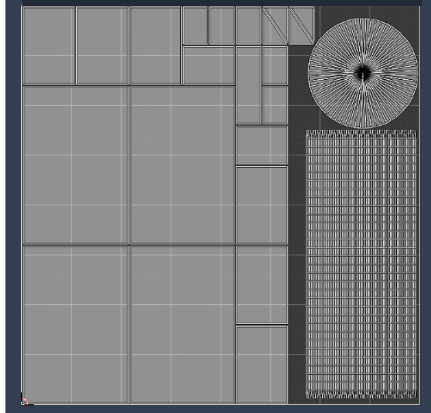

Input
Geometry
UV Coordinates
Materials
Material1 (color, L_e) …
Material2 (color, L_e) …
…
CITPass
RTRad
Acceleration Structure
TLAS
BLAS 0
BLAS 1
…
Texture Group
position
normal
material
surface area
L_{out}
L_{in}
RTLPass
For $i \in S$
For $j \in S \backslash i$
Trace ray from i to j
Miss shader
$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$
$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$
SewSeams
VITPass
Output
Lightmap

Input

Geometry



UV Coordinates



Materials

Material1 (color, L_e) …

Material2 (color, L_e) …

…

CITPass

RTRad

Acceleration Structure

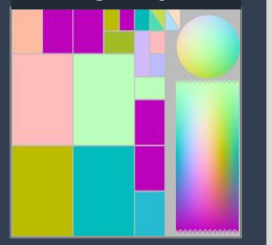
TLAS

BLAS 0 *BLAS 1* …

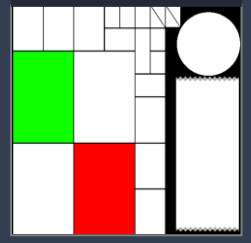
Texture Group

position

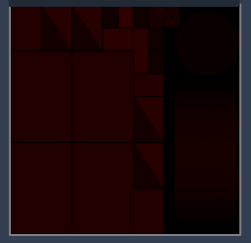
normal



material



surface area



L_{out}

L_{in}

RTLPass

For $i \in S$

For $j \in S \backslash i$

Trace ray from i *to* j

Miss shader

$$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$$

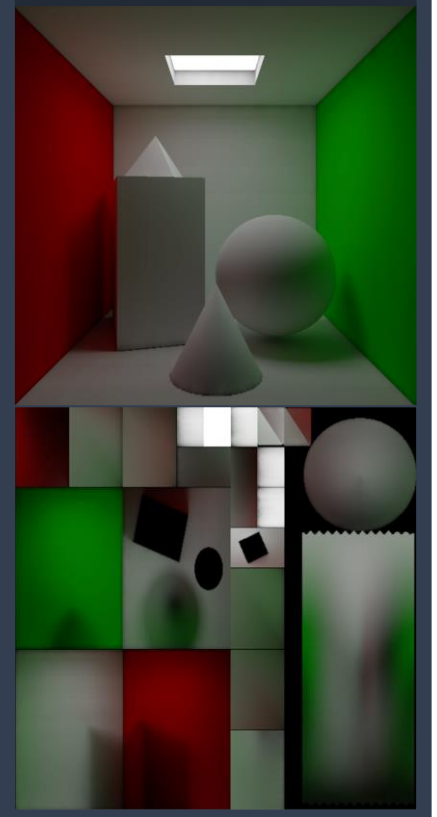
$$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$$

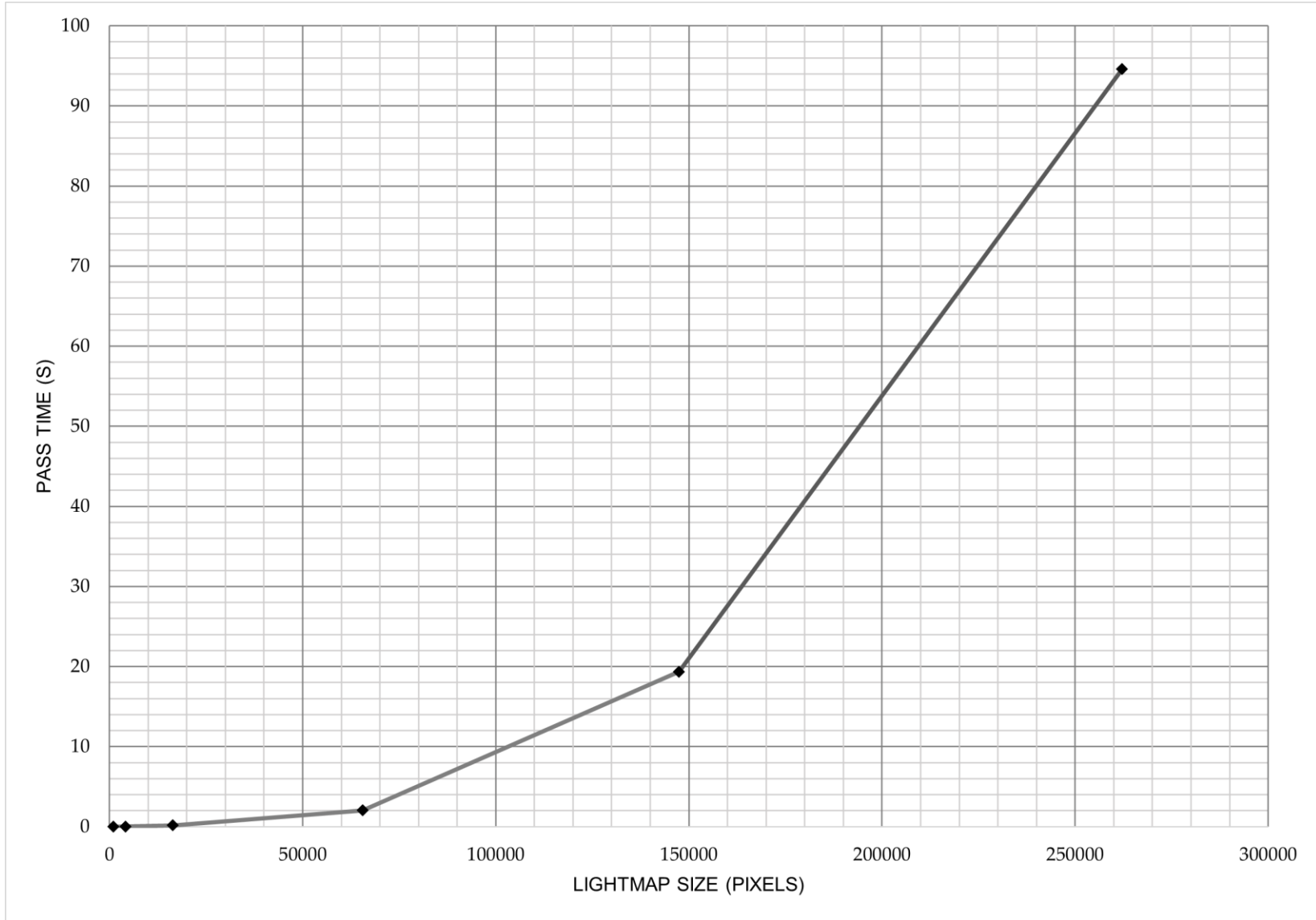
SewSeams

VITPass

Output

Lightmap

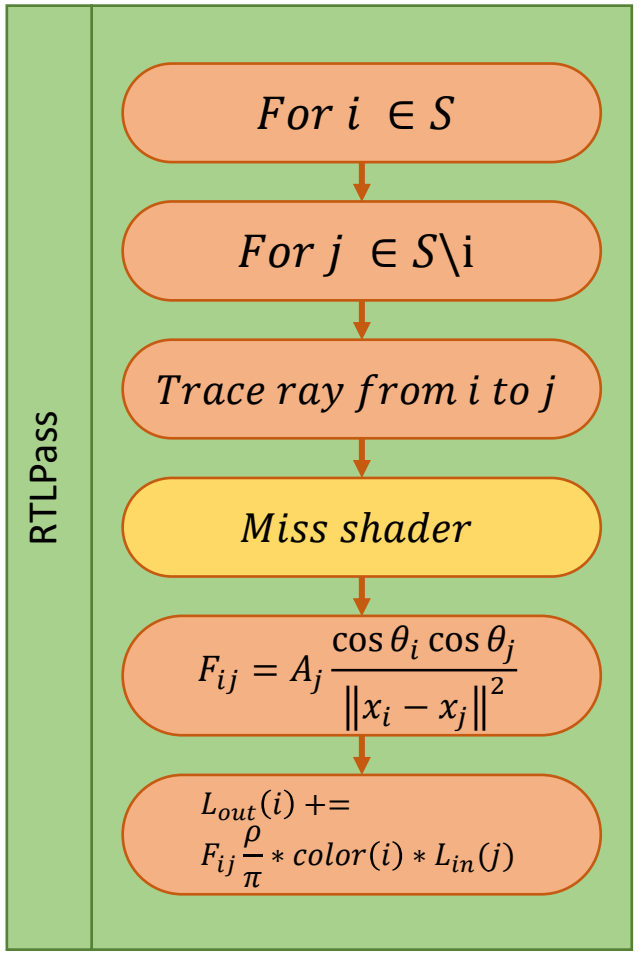



100
90
80
70
60
50
40
30
20
10
0
PASS TIME (S)
0
50000
100000
150000
200000
250000
300000
LIGHTMAP SIZE (PIXELS)

Reciprocity Rule:

$$V(i,j) = V(j,i)$$

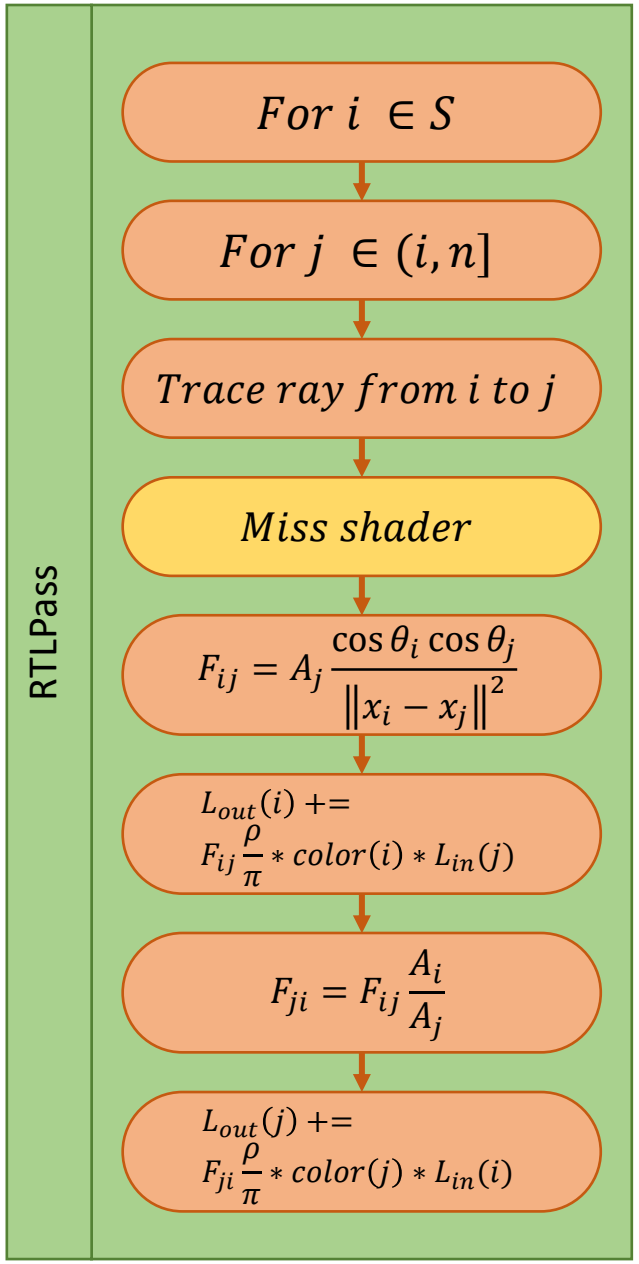
$$F_{ij} = F_{ji}\frac{A_j}{A_i}$$



Reciprocity Rule:

$$V(i,j) = V(j,i)$$

$$F_{ij} = F_{ji} \frac{A_j}{A_i}$$



Reciprocity Rule:

$$V(i,j) = V(j,i)$$

$$F_{ij} = F_{ji} \frac{A_j}{A_i}$$

RTLPass

$For\ i\ \in S$

$For\ j\ \in (i, n]$

$Trace\ ray\ from\ i\ to\ j$

$Miss\ shader$

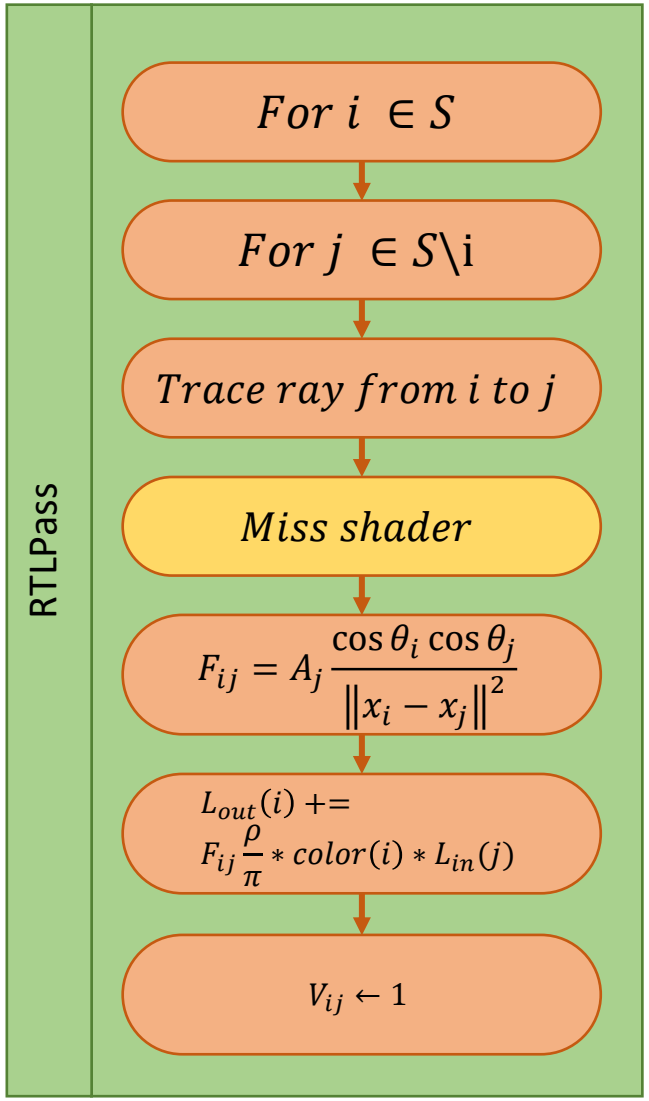
$$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$$

$$L_{out}(i) \mathrel{+}= F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$$

$$F_{ji} = F_{ij} \frac{A_i}{A_j}$$

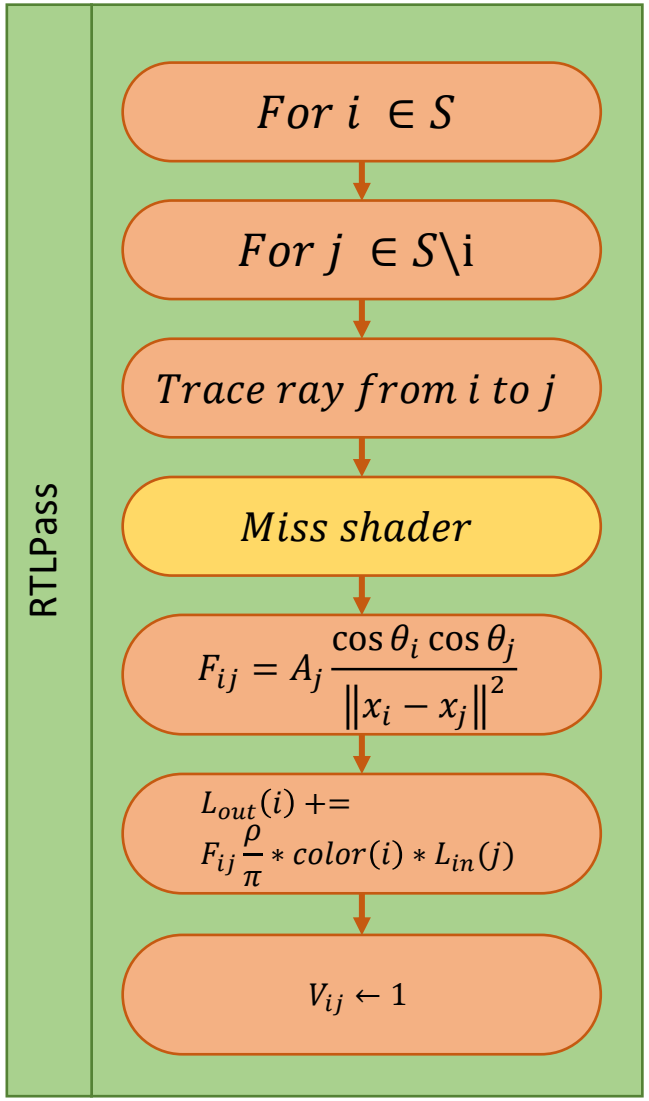
$$L_{out}(j) \mathrel{+}= F_{ji} \frac{\rho}{\pi} * color(j) * L_{in}(i)$$

Visibility Caching:

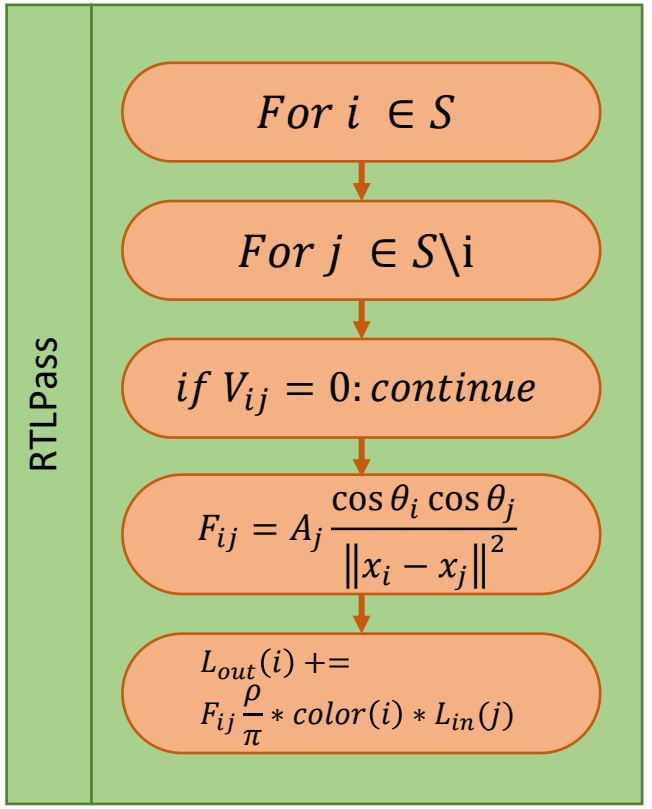


First Iteration

Visibility Caching:

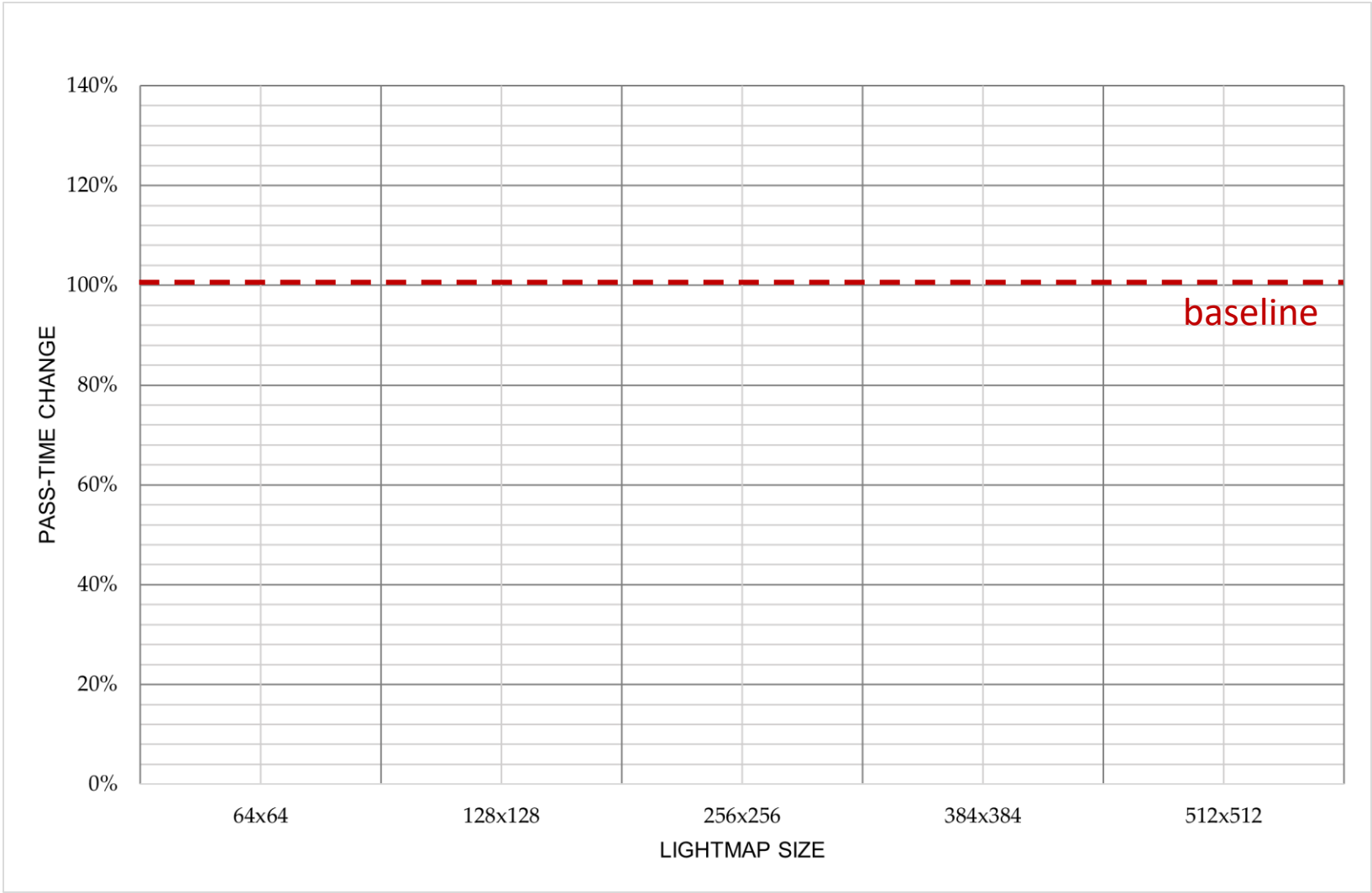


First Iteration

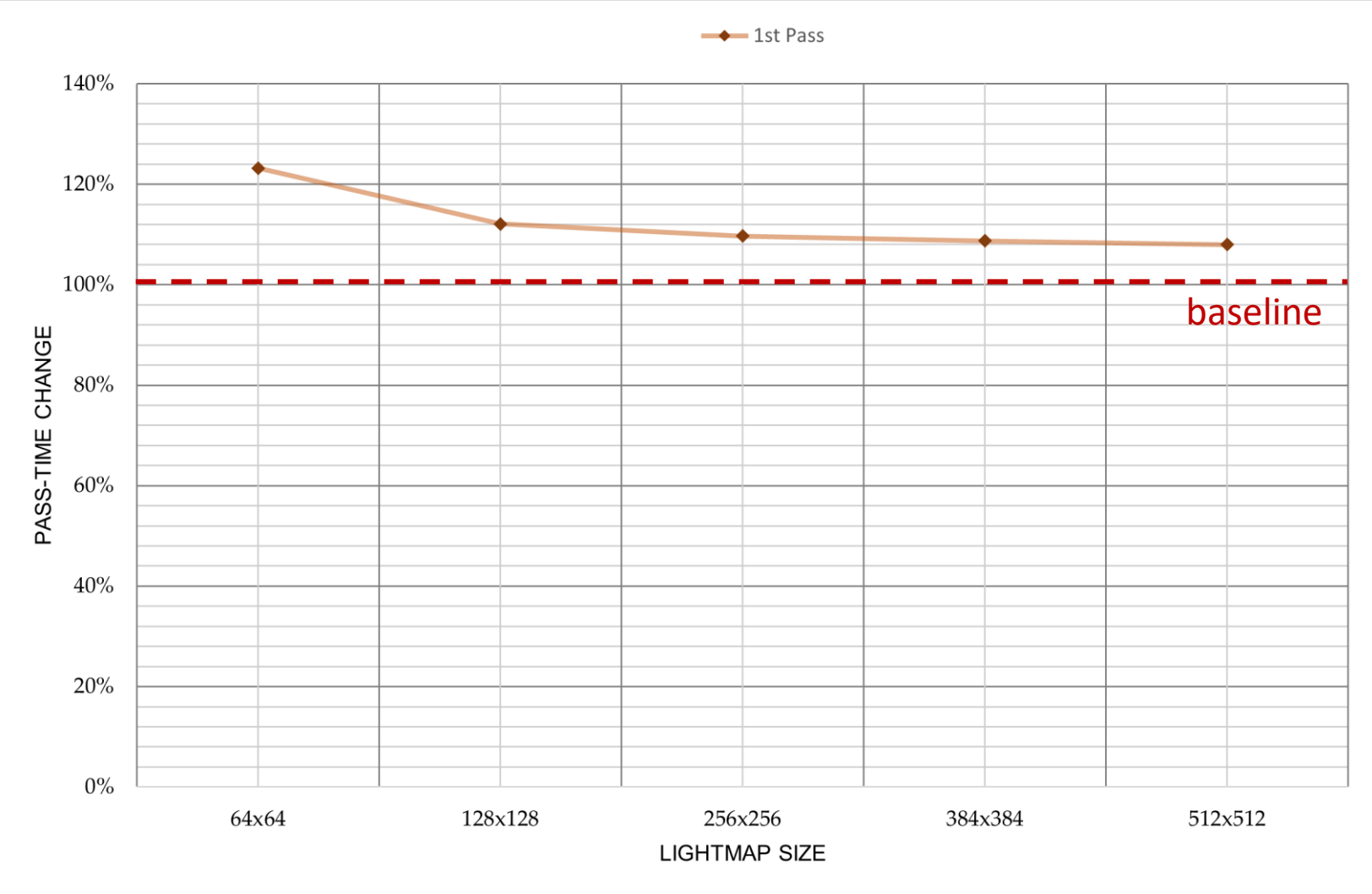


Subsequent Iterations

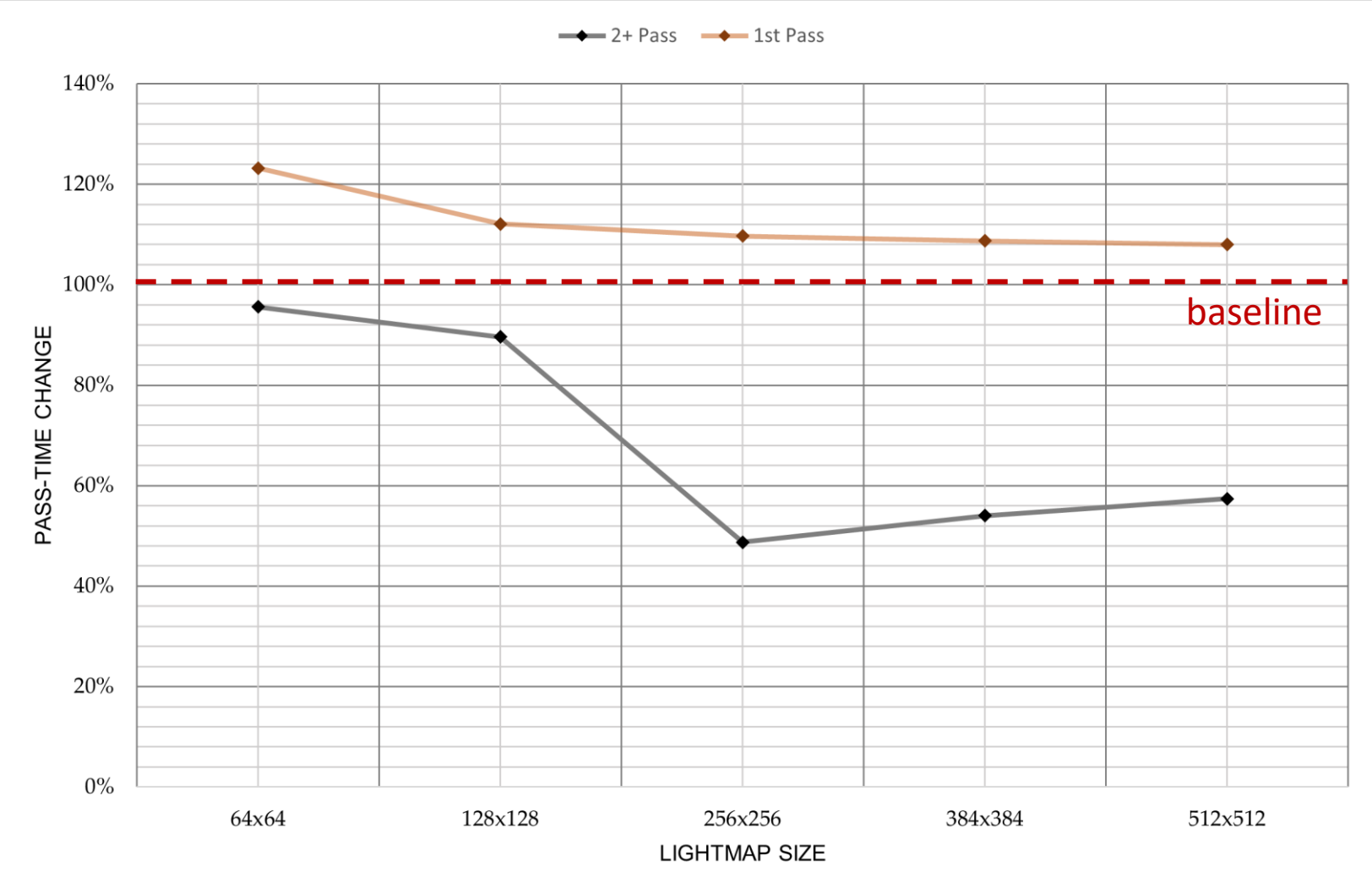
Visibility Caching:



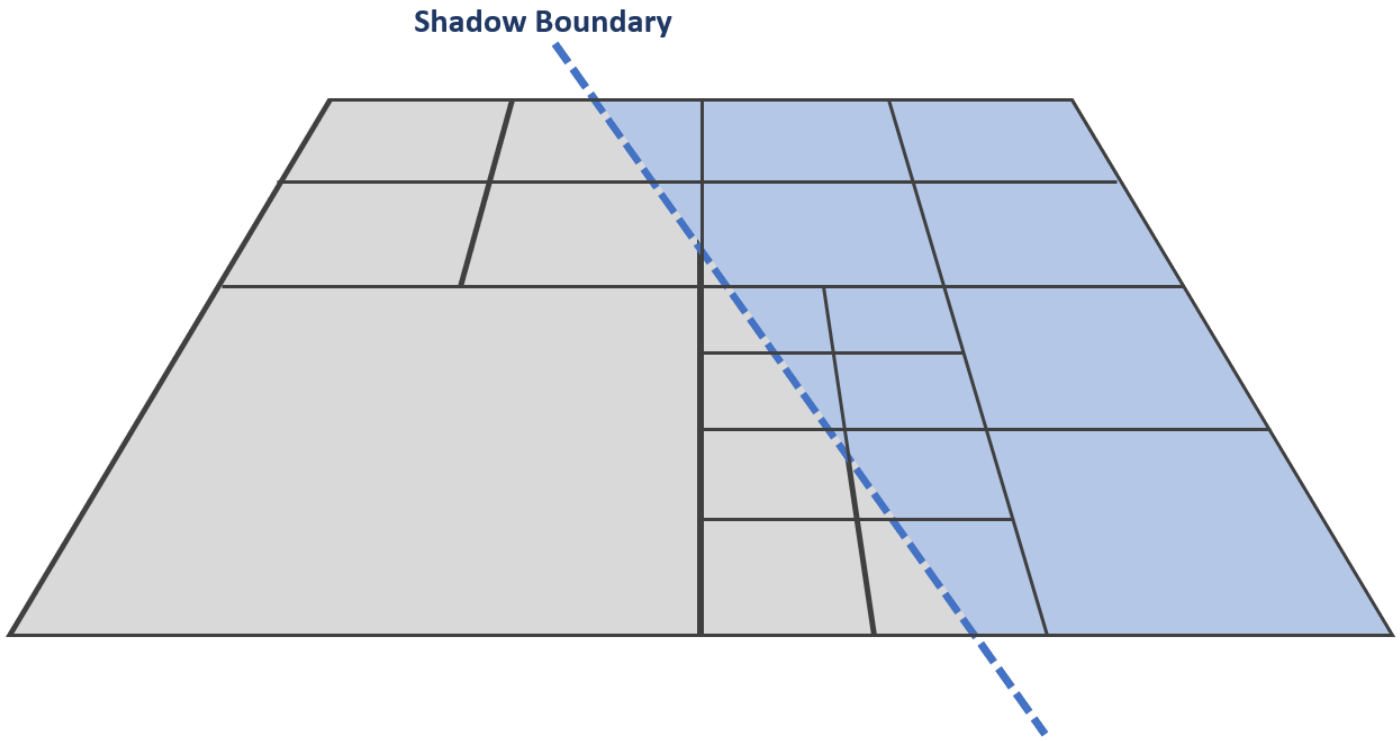
Visibility Caching:



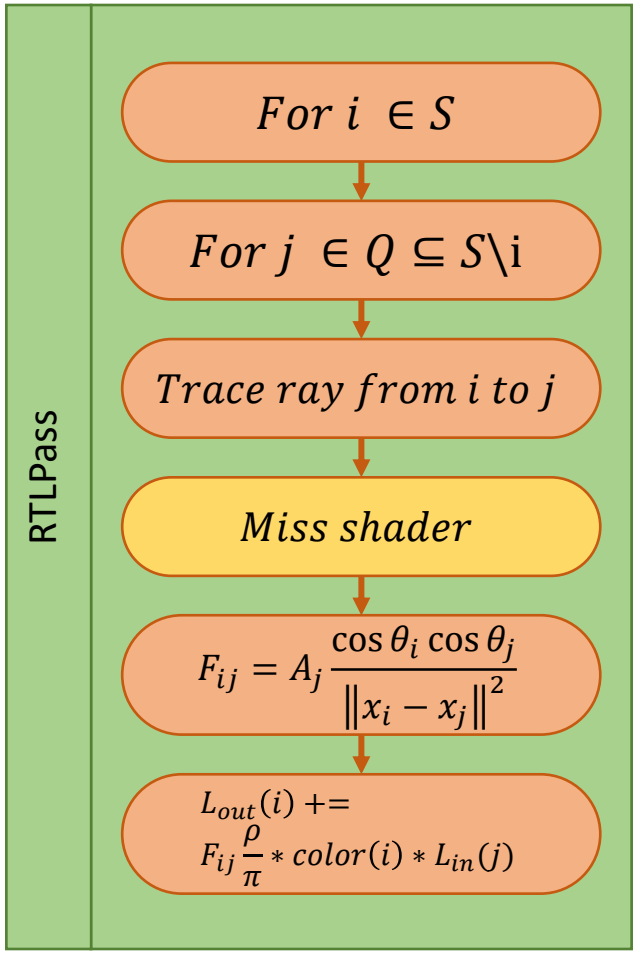
Visibility Caching:



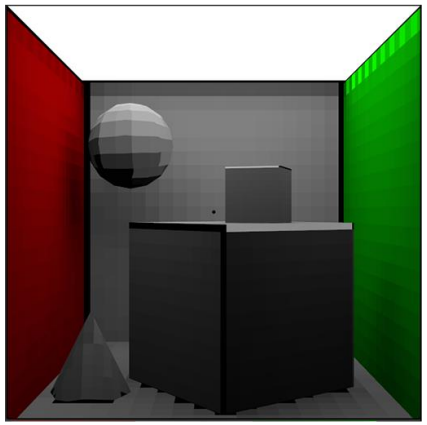
Refinement

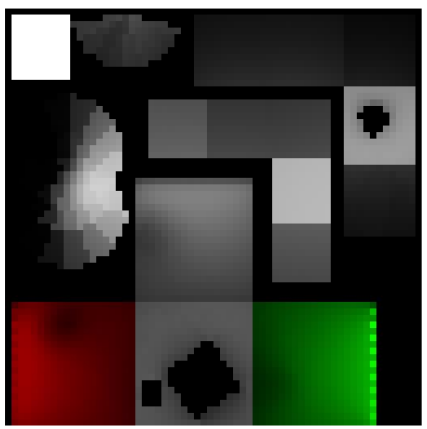


Refinement

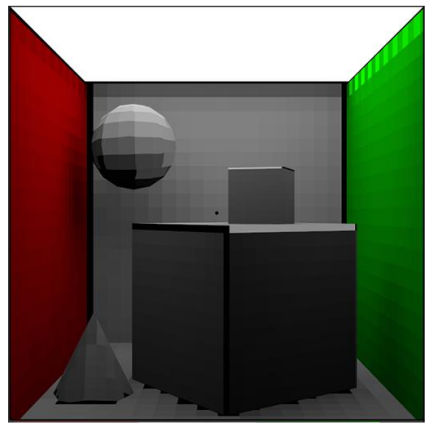

RTLPass
For $i \in S$
For $j \in Q \subseteq S \backslash i$
Trace ray from i to j
Miss shader
$F_{ij} = A_j \frac{\cos\theta_i \cos\theta_j}{\|x_i - x_j\|^2}$
$L_{out}(i) +=$
$F_{ij} \frac{\rho}{\pi} * color(i) * L_{in}(j)$

Scene

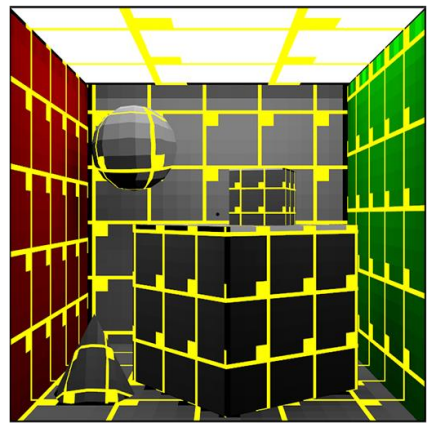


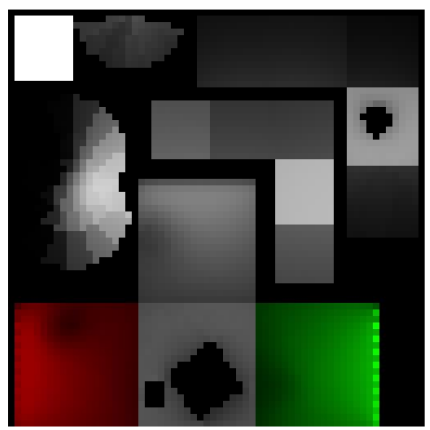


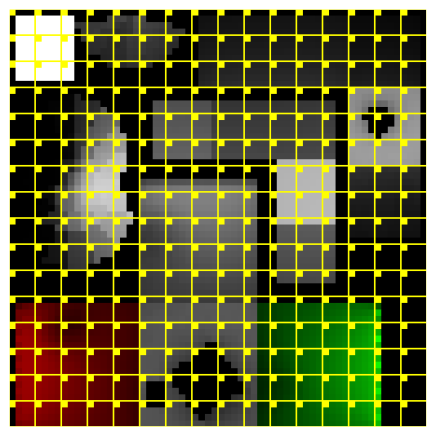
Scene



Static Undersampling



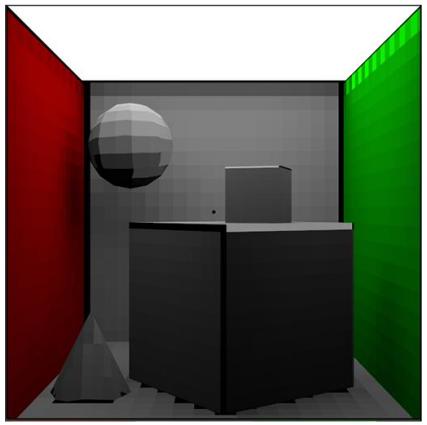


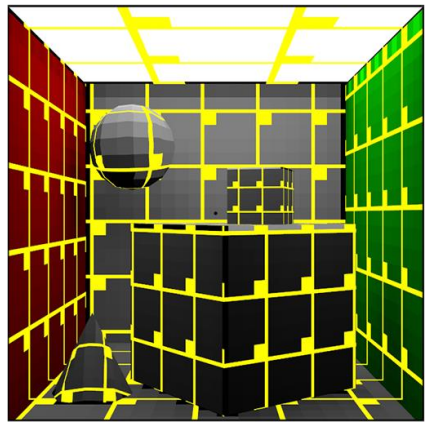


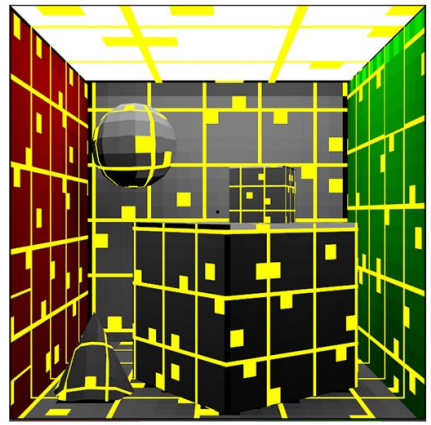
Scene

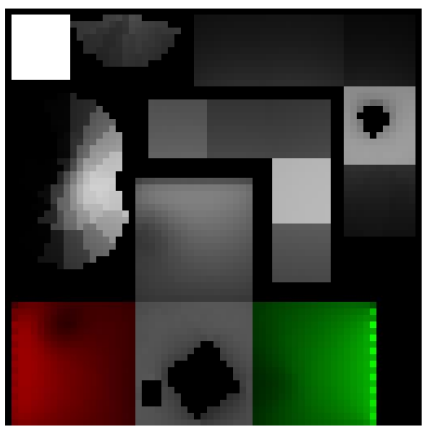
Static Undersampling

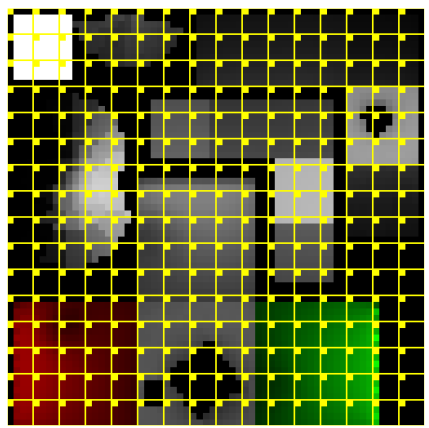
Monte Carlo Undersampling

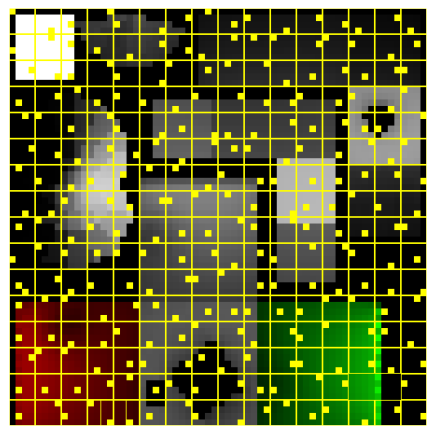


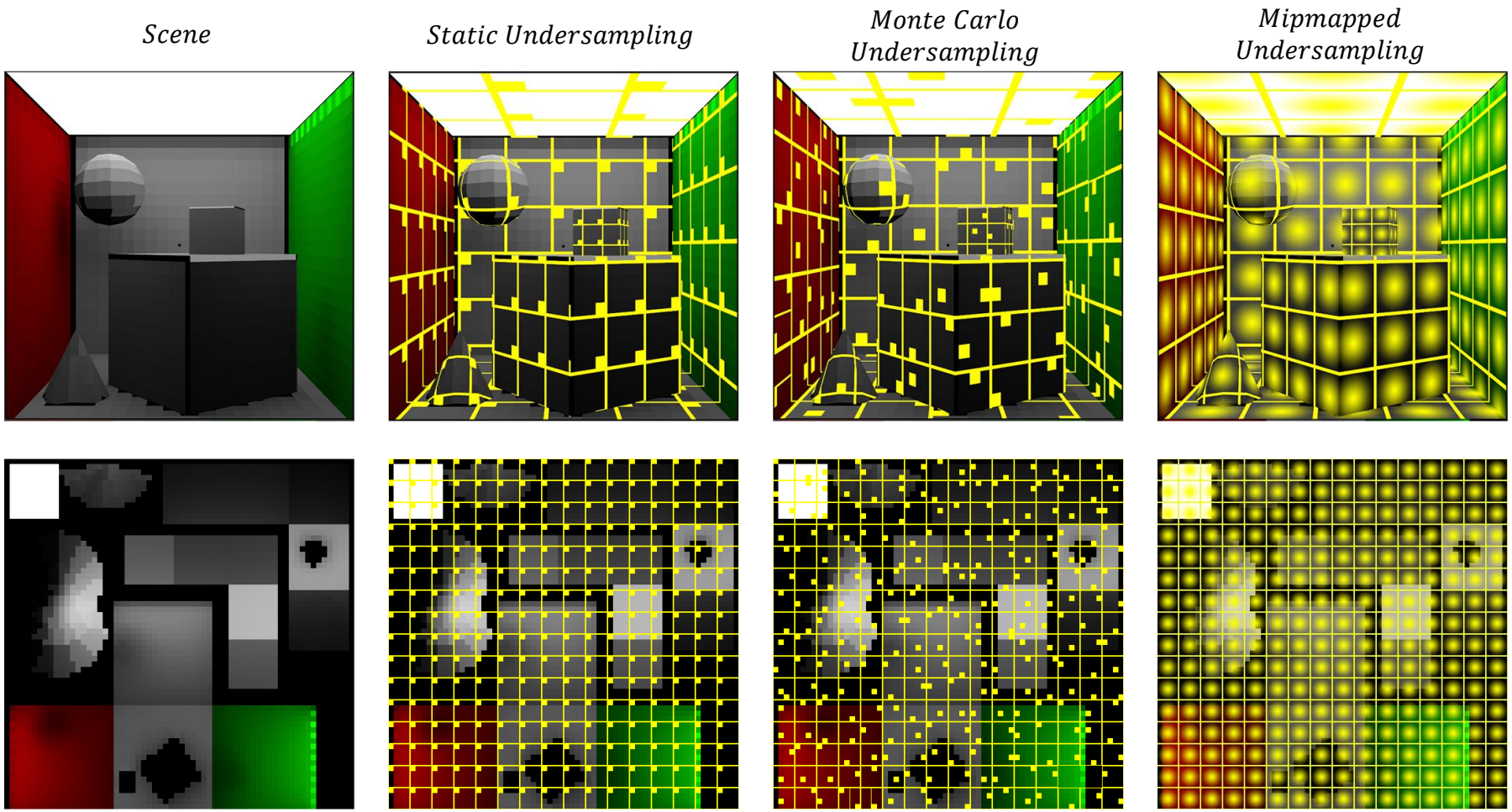


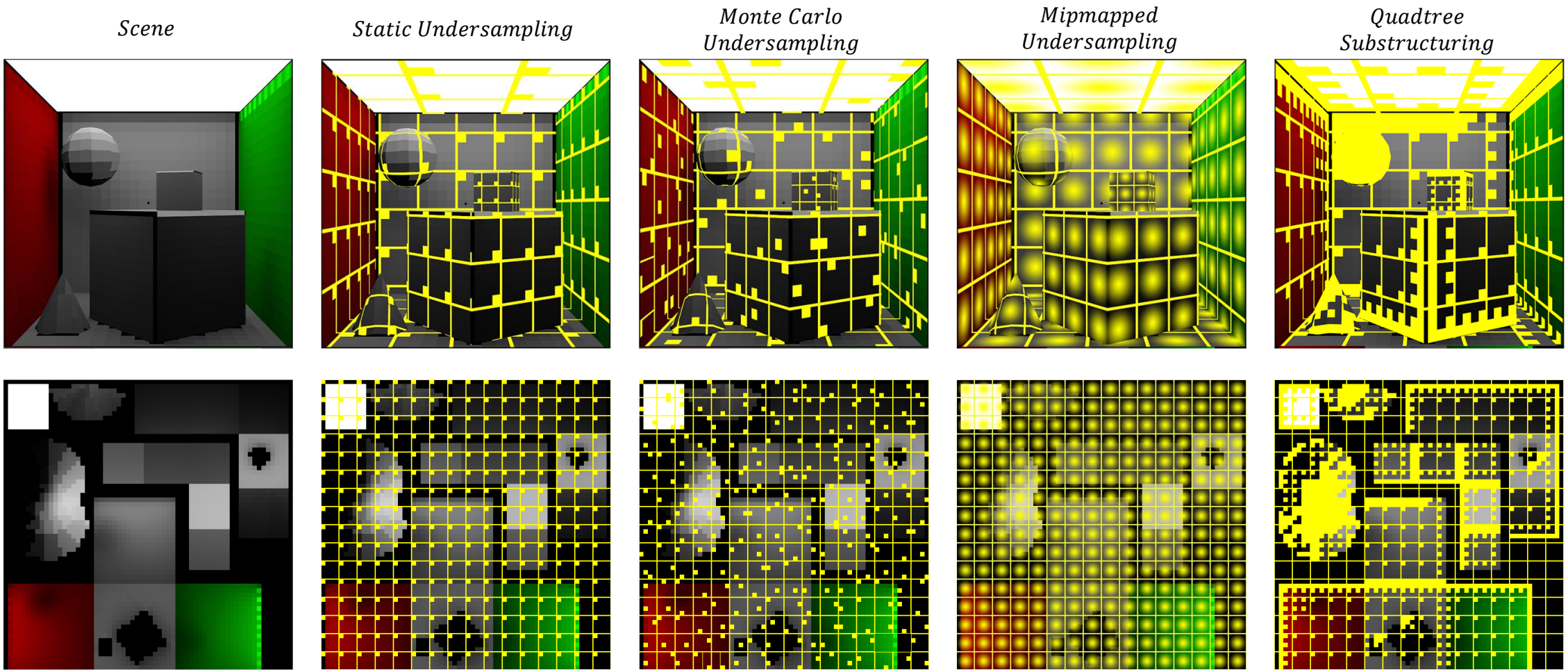


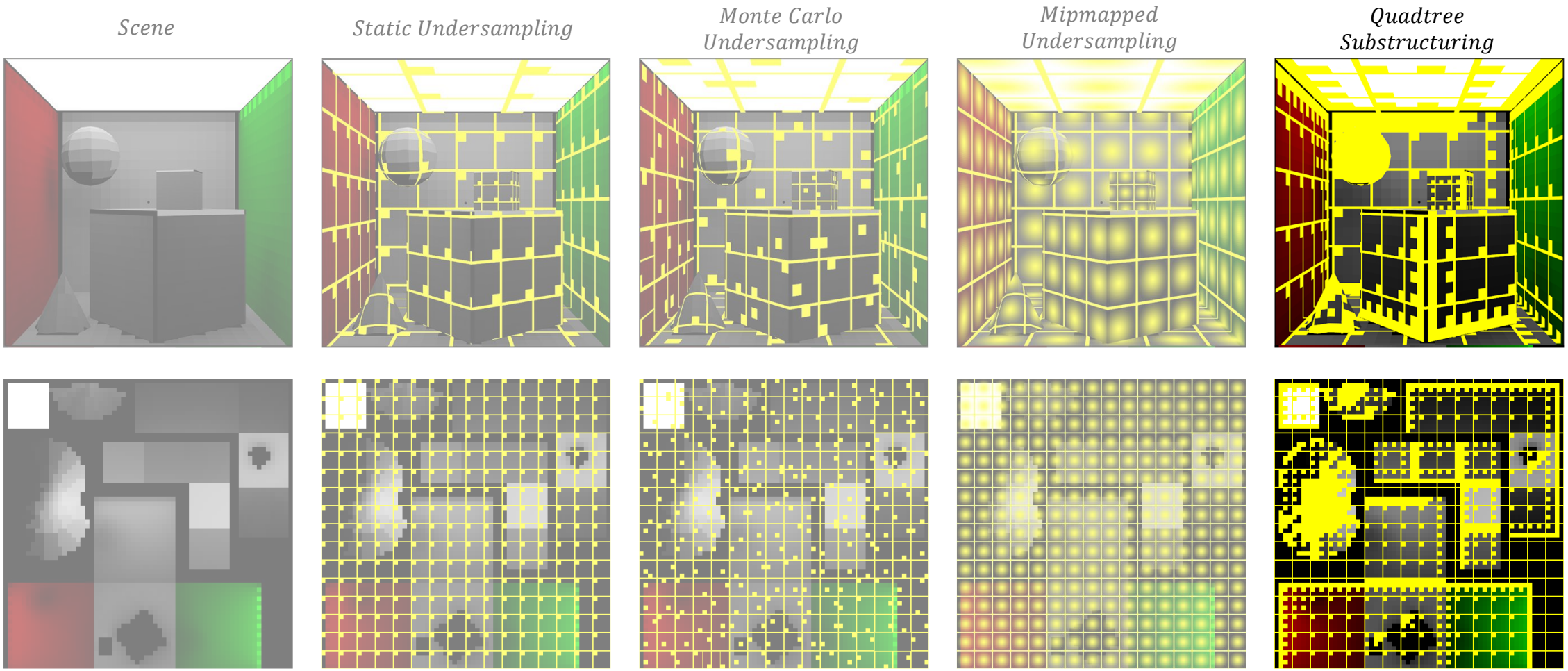




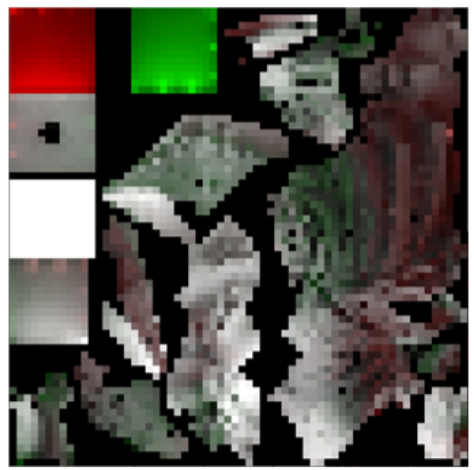



Scene
Static Undersampling
Monte Carlo Undersampling
Mipmapped Undersampling

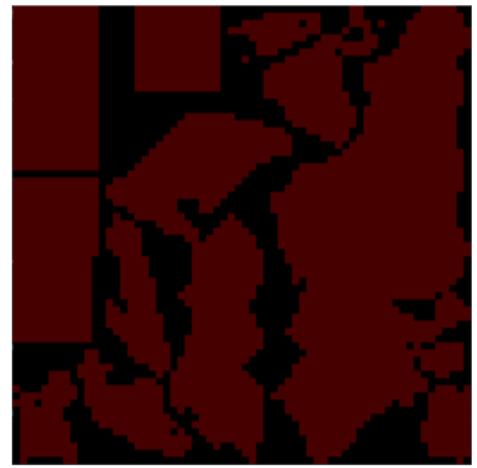

Scene
Static Undersampling
Monte Carlo Undersampling
Mipmapped Undersampling
Quadtree Substructuring


Scene
Static Undersampling
Monte Carlo Undersampling
Mipmapped Undersampling
Quadtree Substructuring

lightmap

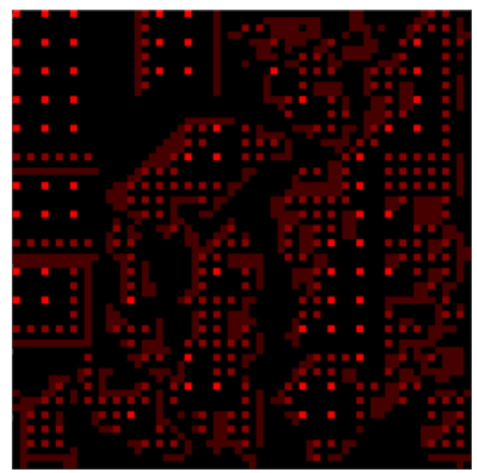


1st iteration

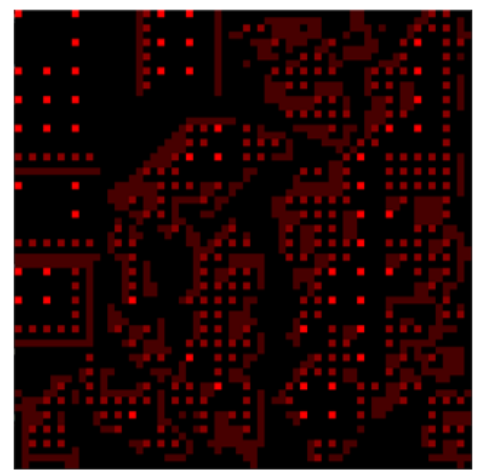


2nd iteration

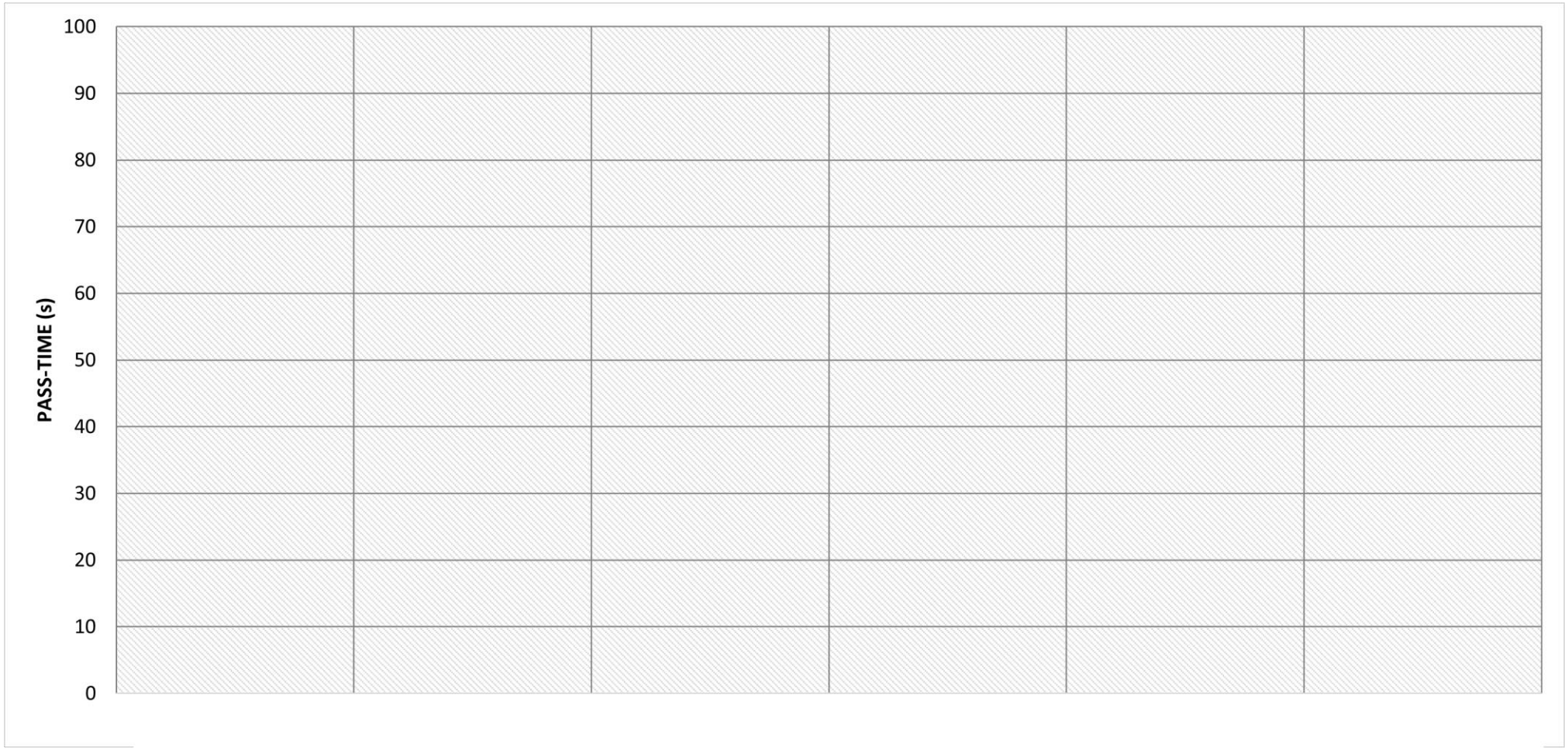
3rd iteration



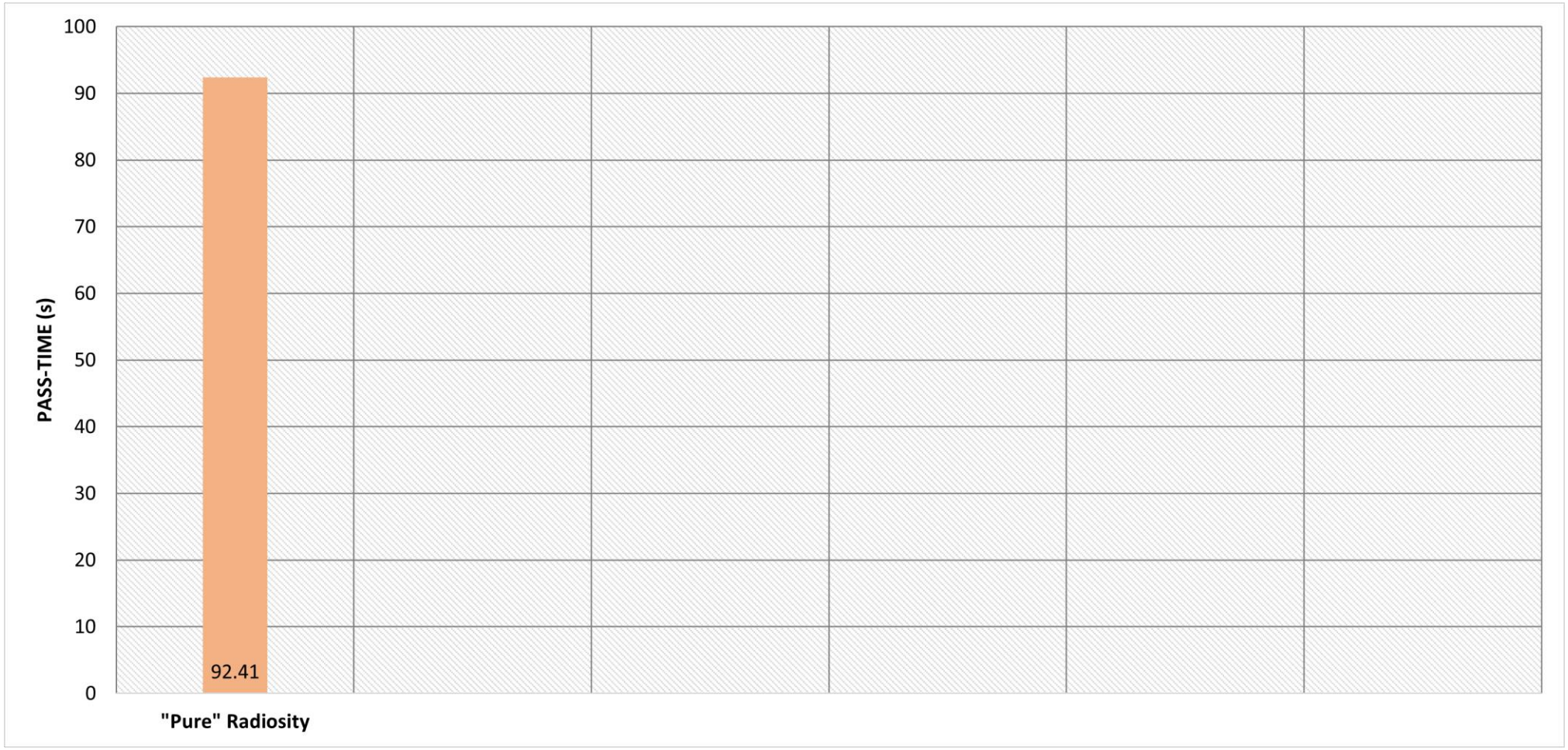
4th iteration



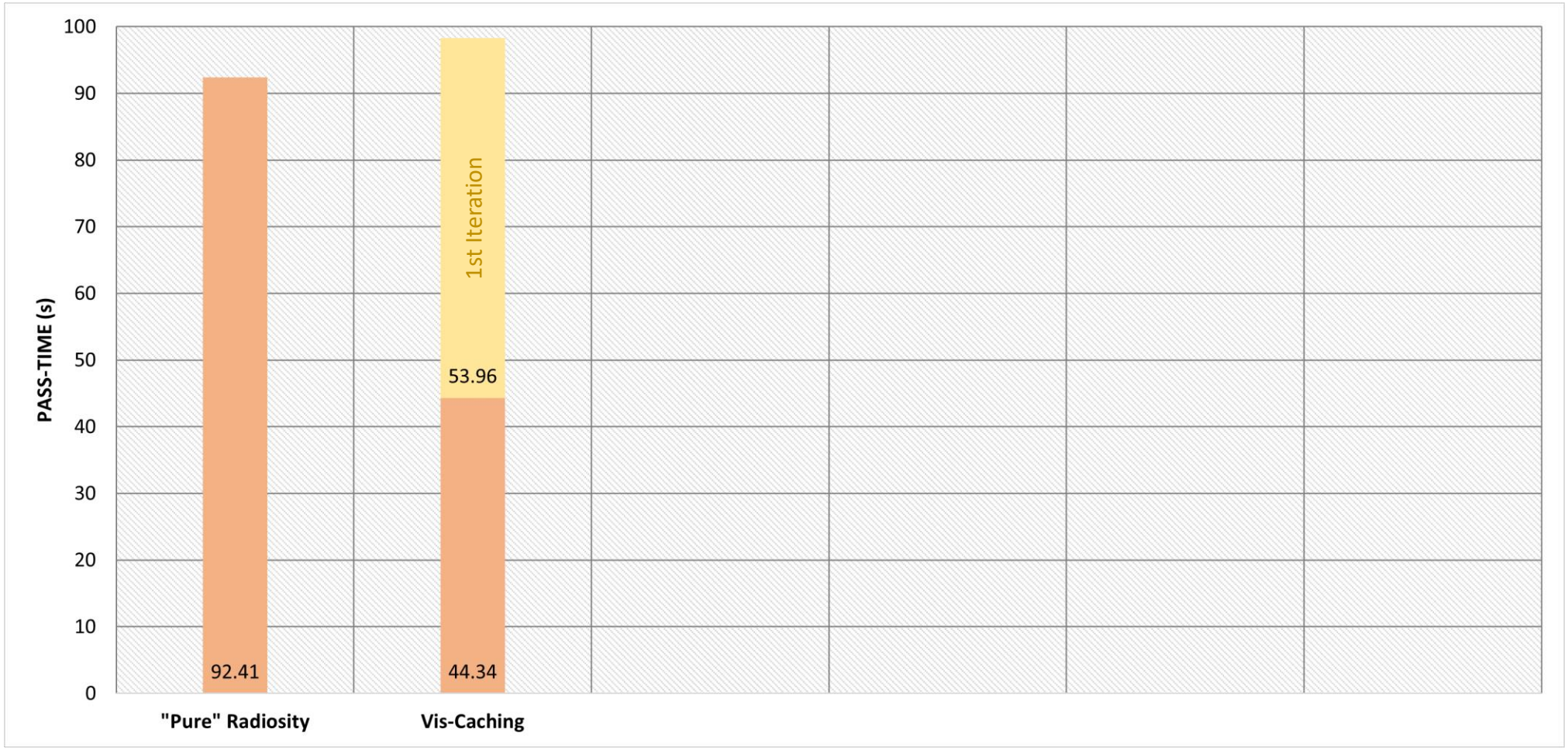
Lightmap size: 512x512



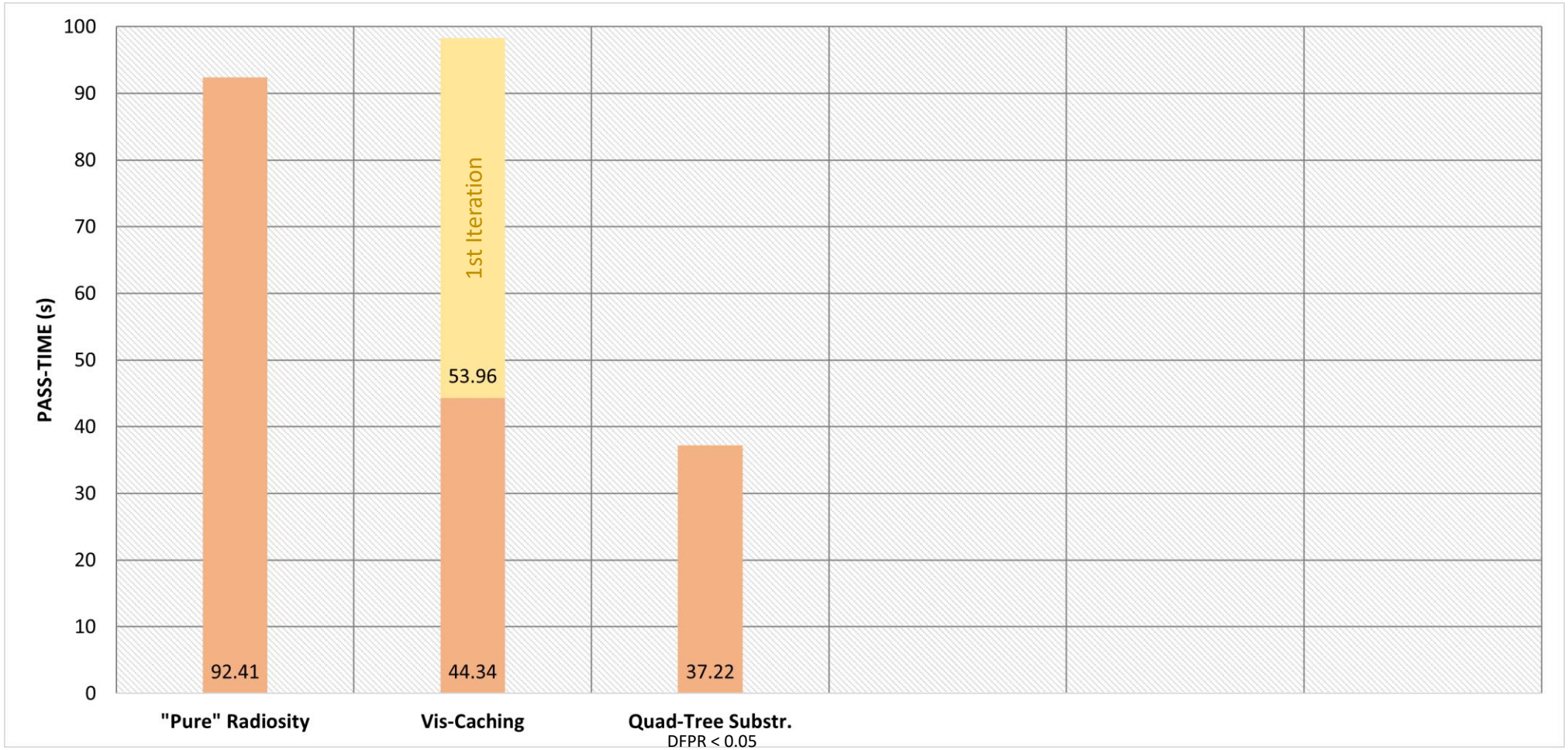
Lightmap size: 512x512



Lightmap size: 512x512



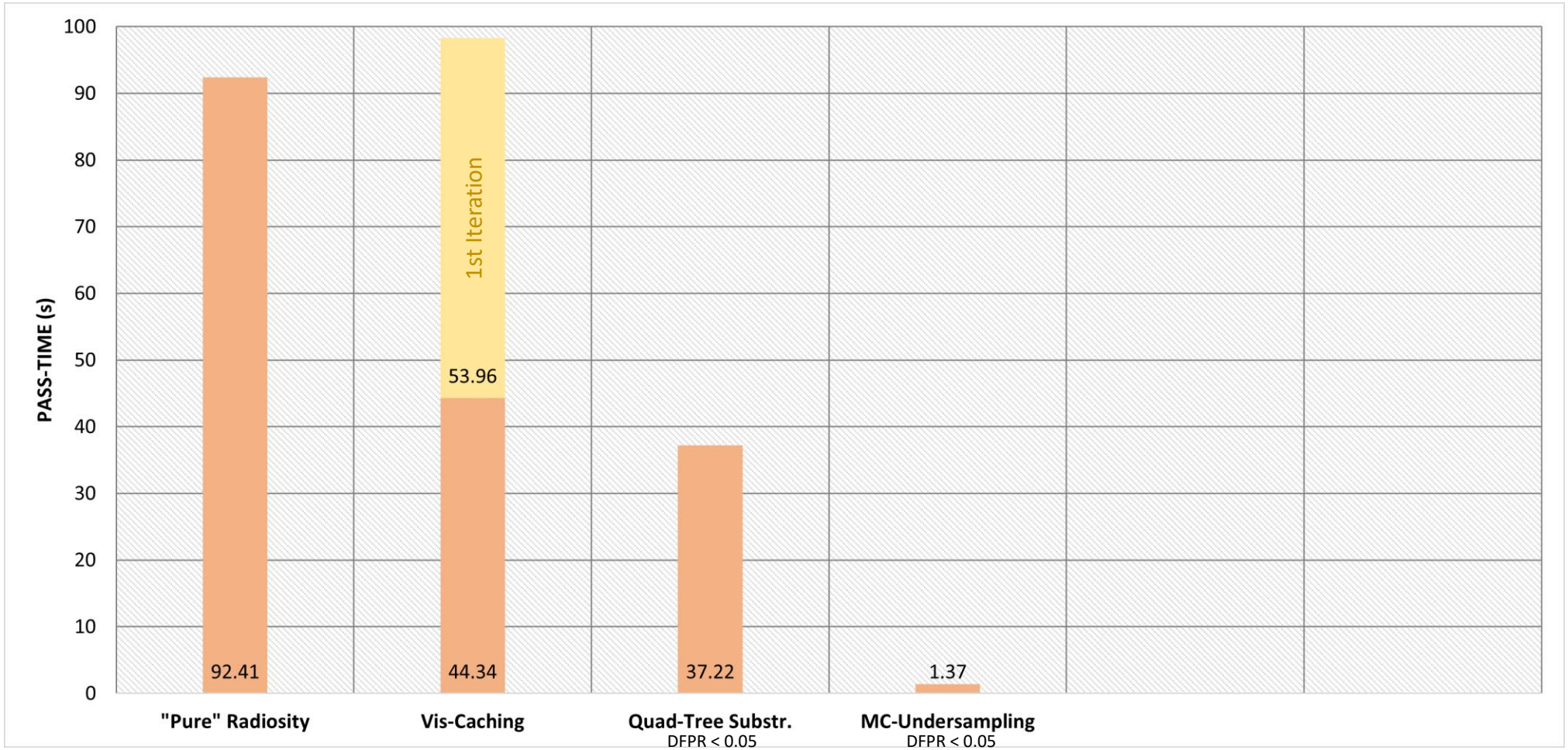
Lightmap size: 512x512



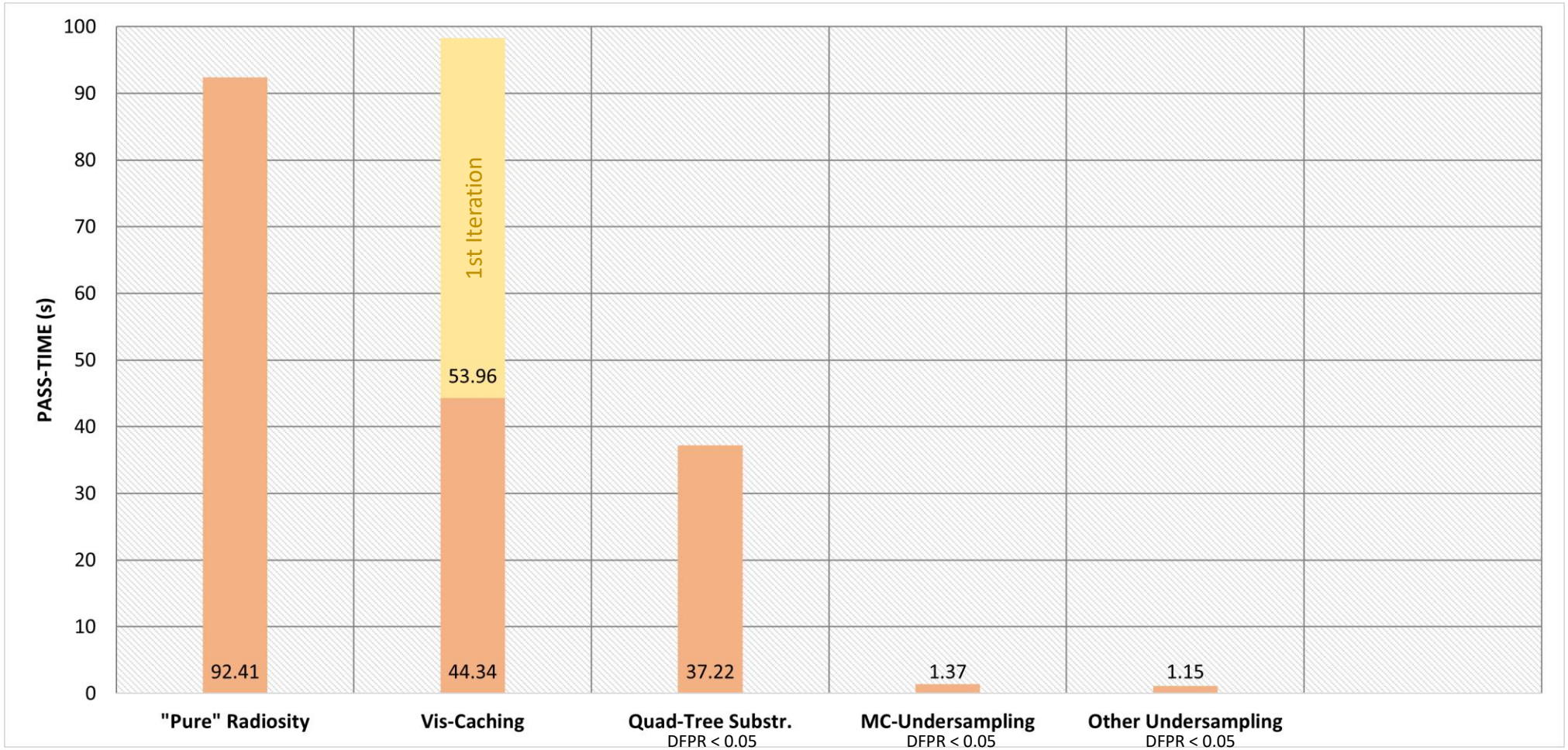
DFPR < 0.05

Node size 4x4

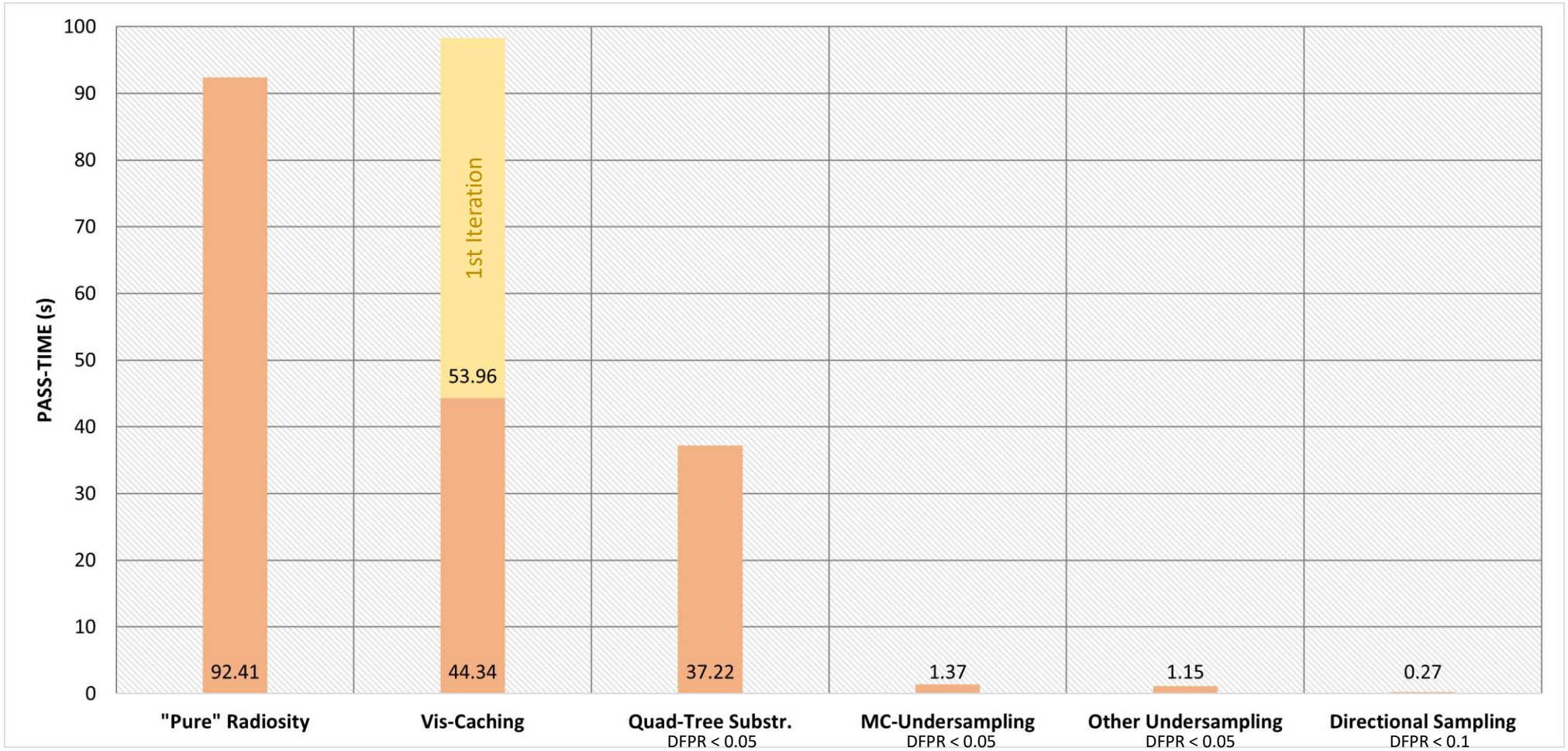
Lightmap size: 512x512



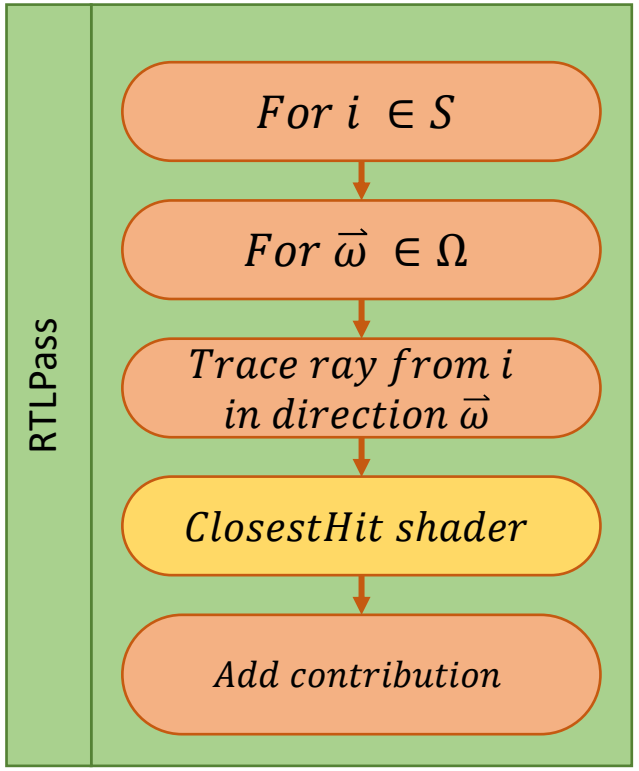
Lightmap size: 512x512



Lightmap size: 512x512

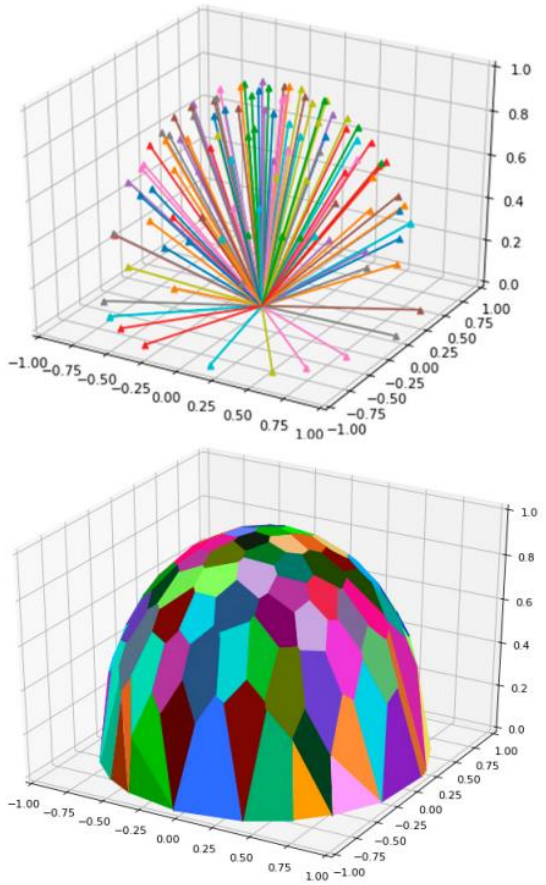


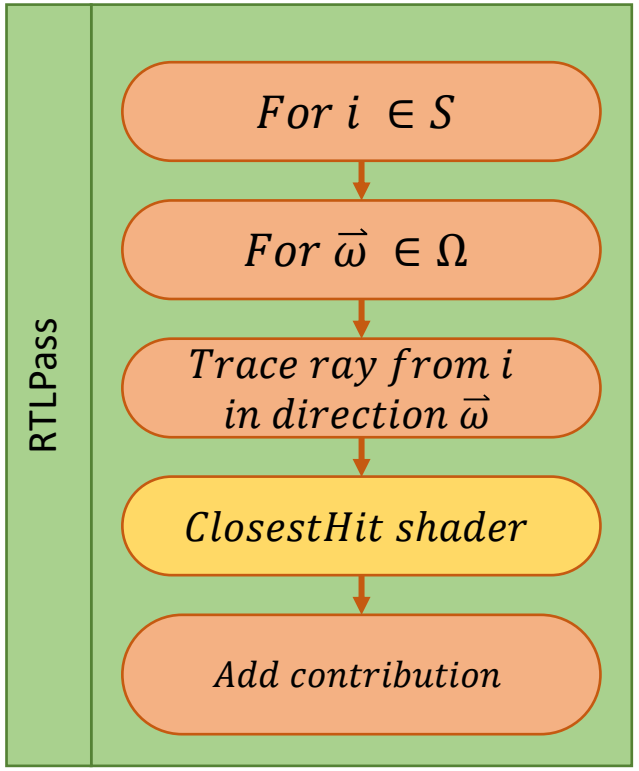
Directional Sampling

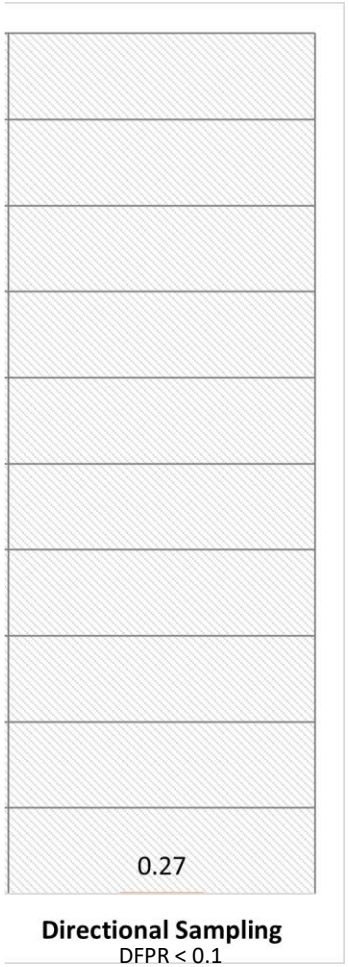


Directional Sampling
DFPR < 0.1
1024 samples

Directional Sampling

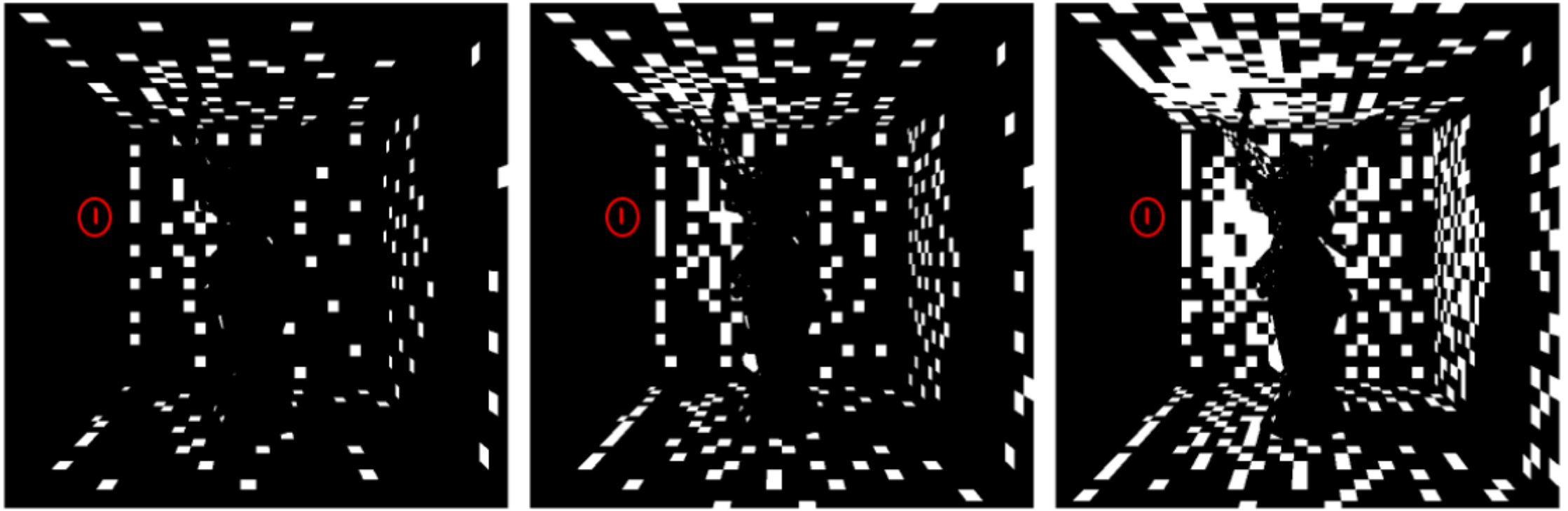






Directional Sampling
DFPR < 0.1
1024 samples

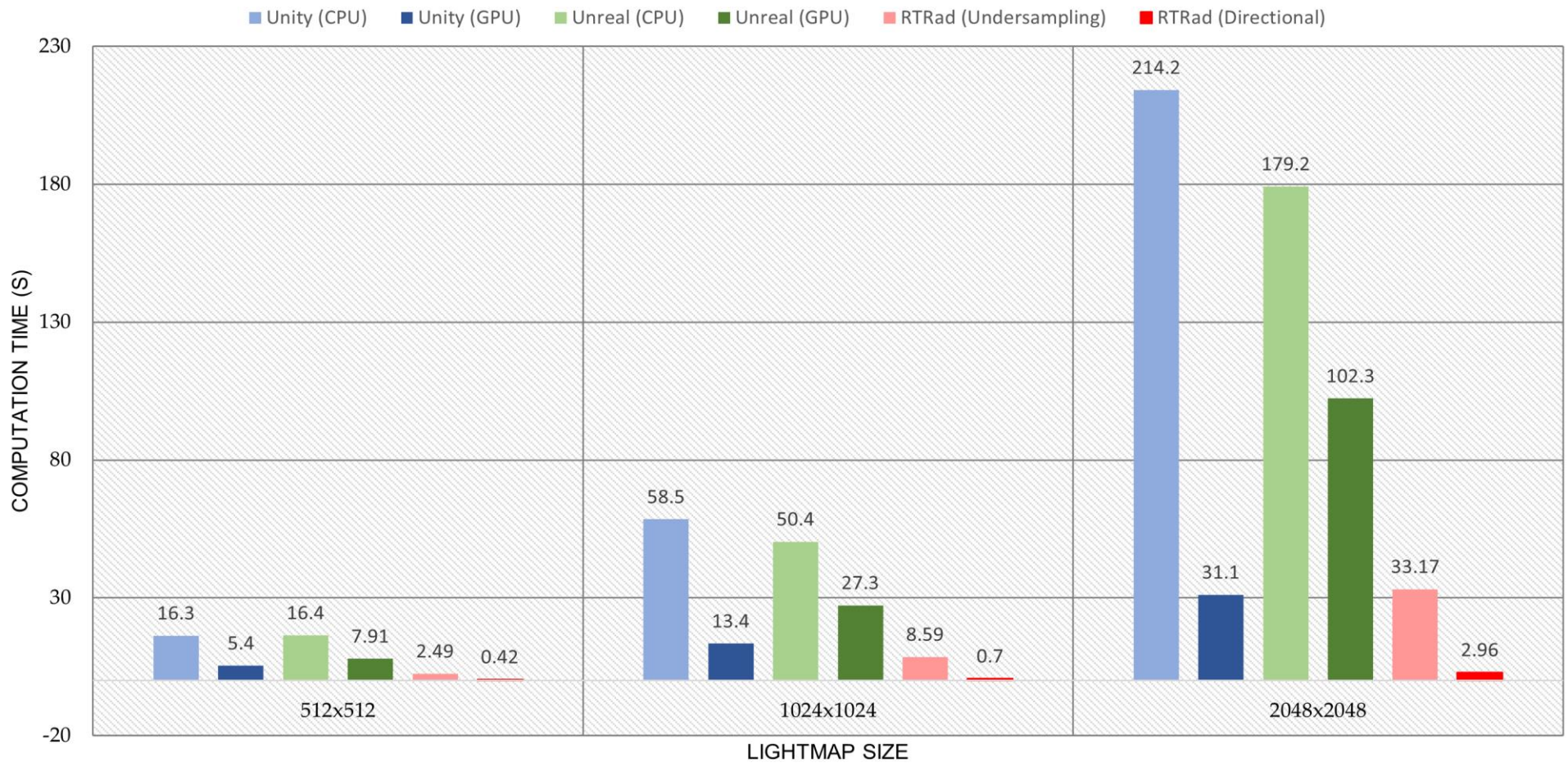
Directional Sampling



256 samples

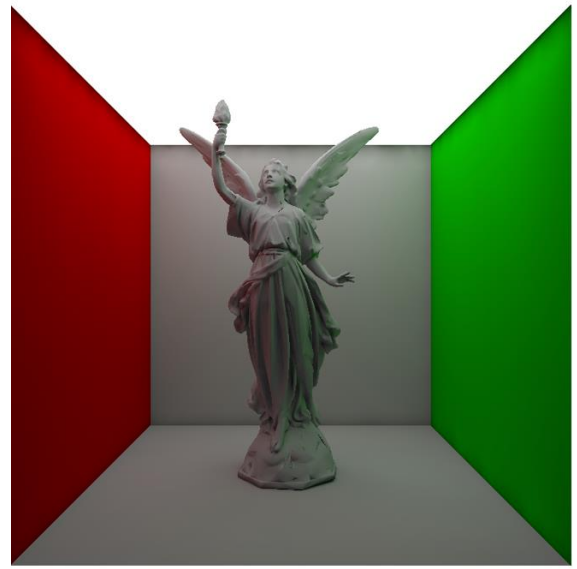
512 samples

1024 samples


Unity (CPU)
Unity (GPU)
Unreal (CPU)
Unreal (GPU)
RTRad (Undersampling)
RTRad (Directional)
COMPUTATION TIME (S)
230
180
130
80
30
-20
16.3
5.4
16.4
7.91
2.49
0.42
58.5
13.4
50.4
27.3
8.59
0.7
214.2
31.1
179.2
102.3
33.17
2.96
512x512
1024x1024
2048x2048
LIGHTMAP SIZE

Unreal Engine 5.0.3

Unity 2020.3.21



RTRad (Undersampling)

RTRad (Directional)

Conclusions

Conclusions

- RTX can speed up radiosity on the GPU, as well as visibility calculations at large.

Conclusions

- RTX can speed up radiosity on the GPU, as well as visibility calculations at large.
- RTX can not negate radiosity‘s quadratic complexity.

Conclusions

- RTX can speed up radiosity on the GPU, as well as visibility calculations at large.
- RTX can not negate radiosity‘s quadratic complexity.
- Specifically for RTRad:
  - Voxel-raymarching and visibility-caching are of limited usability.
  - A ratio of $\frac{1}{128}$ between sampling window and lightmap provides balanced results in undersampling.
  - Monte-Carlo undersampling tends to produce the least artifacts.
  - ‚Alpha-embedding‘ quadtrees in a lightmap is convenient, but not as fast.
  - Directional sampling can be tremendously fast, but does not provide the same quality and only works with larger light-sources.

The Future for *RTRad*

The Future for *RTRad*

- Direct light calculated separately.

The Future for *RTRad*

- Direct light calculated separately.
- Rely mostly on directional sampling with directions generated from a hemicube.

The Future for *RTRad*

- Direct light calculated separately.
- Rely mostly on directional sampling with directions generated from a hemicube.
- Hybrid inclusion of manually sampled patches deemed as ‚important‘ through a quad-tree or similar.