Universitat Pompeu Fabra
Barcelona

Final Project
Soft Shadows

Group 20

Gemma Alaix i Granell

Pol Valls Rué

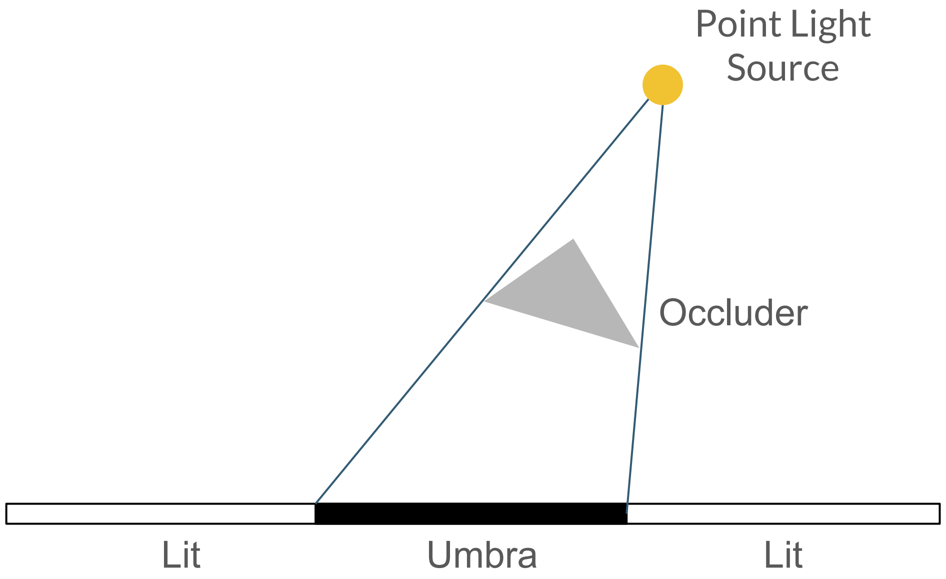
1. Introduction

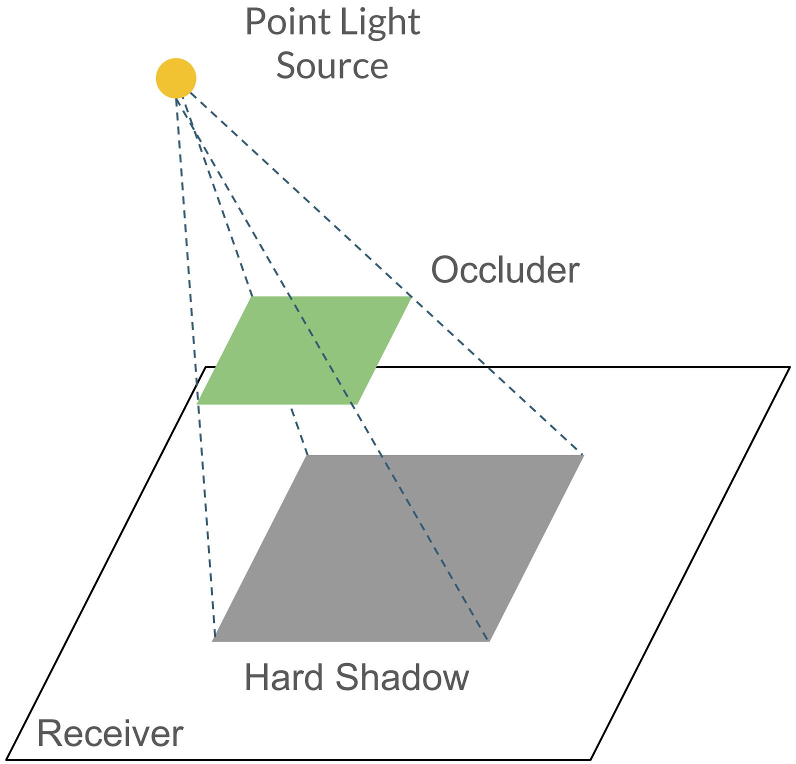
1.1. Hard Shadows

1.2. Soft Shadows

1.3. Motivation & Objectives

1.1 Hard shadows

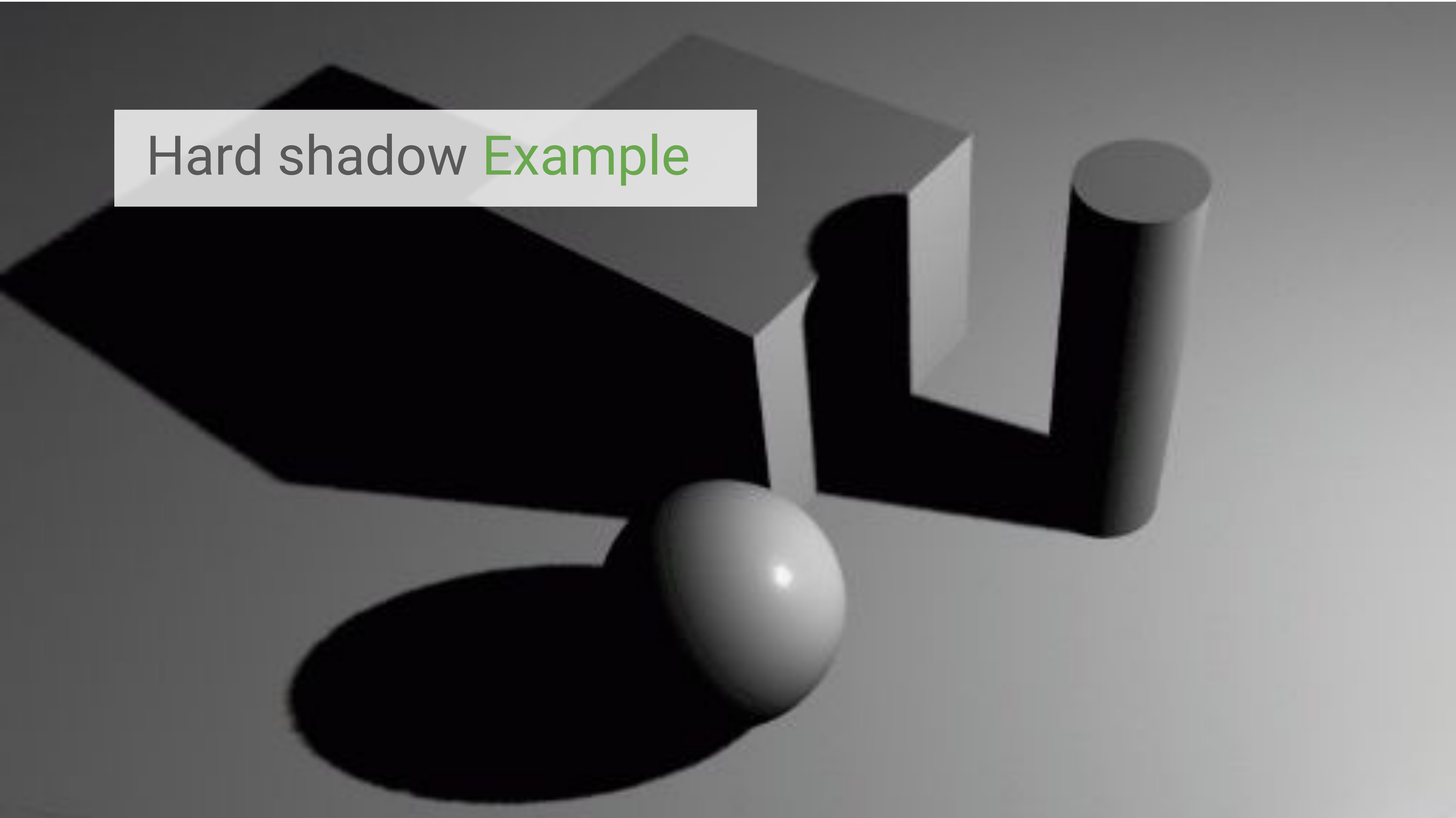




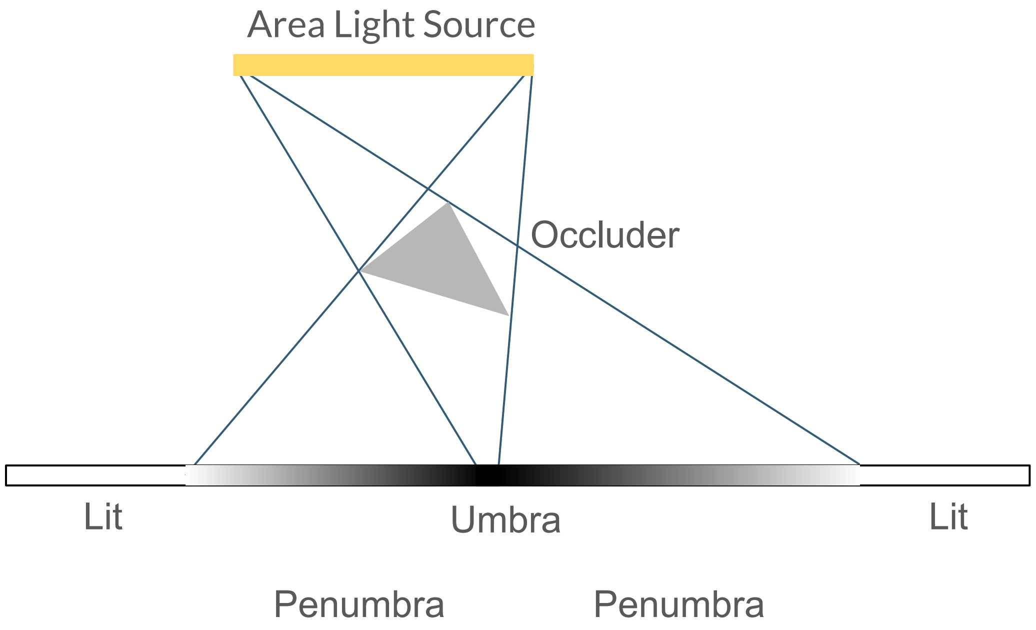
1.1 Hard shadows

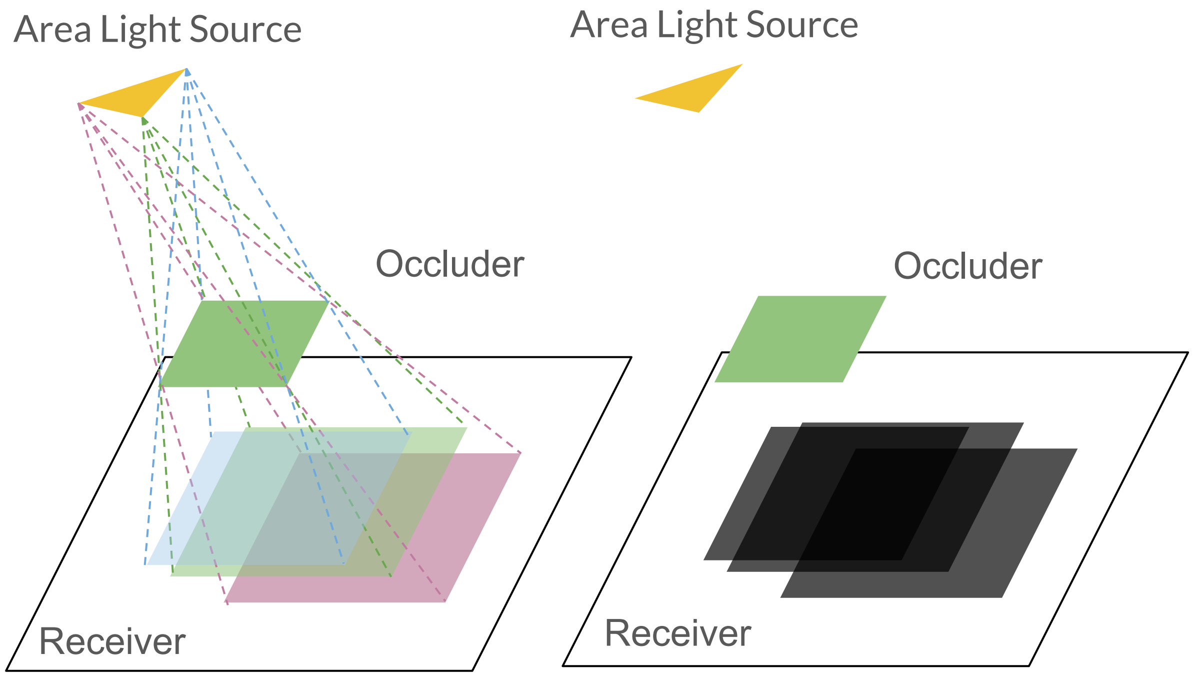
> *A point source of light casts only a simple hard shadow, called umbra*

Hard shadow Example

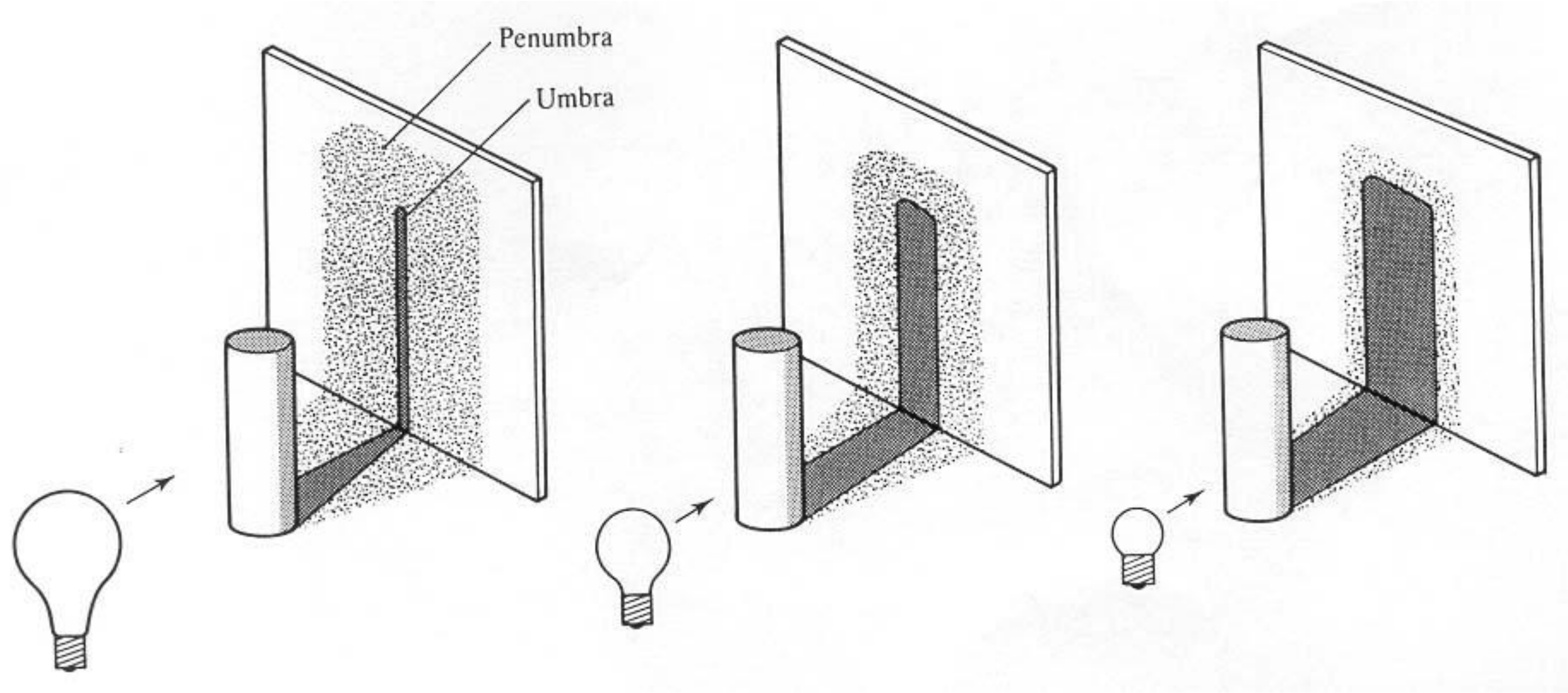


1.2 Soft shadows





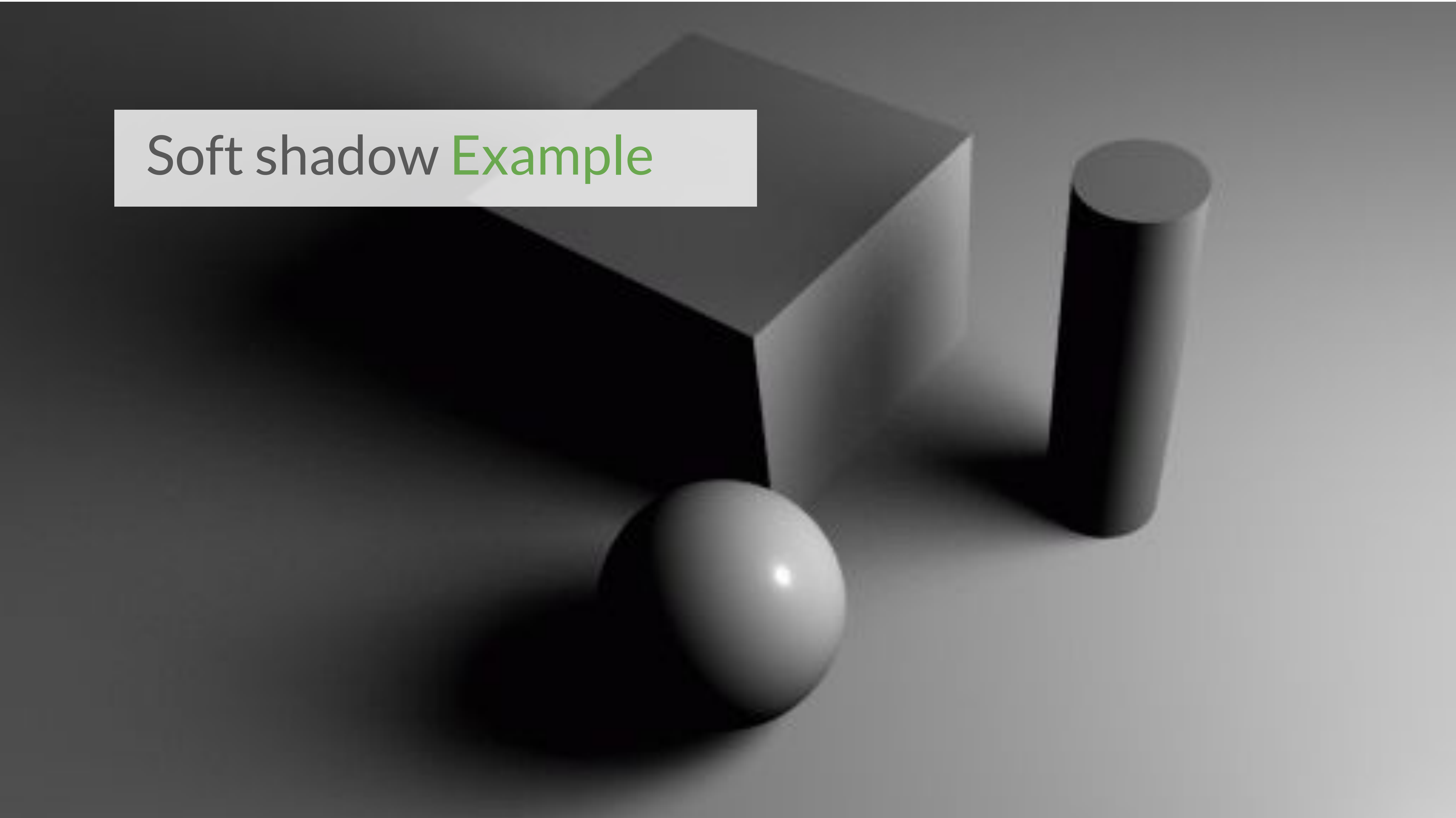
1.2 Soft shadows



1.2 Soft shadows

> *For an extended source of light, the shadow is divided into the umbra and penumbra*

Soft shadow Example



1.3 Motivation

- Obtain a more realistic and advanced scene

Objectives

- Implement area light sources
- Implement new shapes

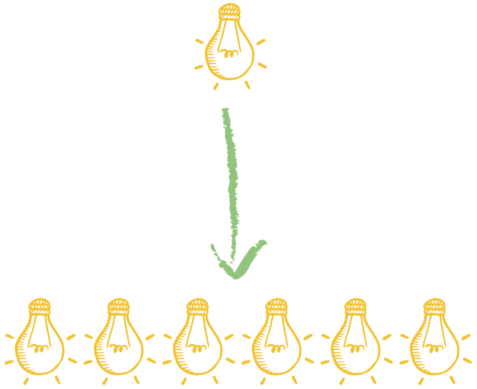
2. Main Challenges & Approach

2.1. Find appropriate description for Area Light Sources
2.2. Sampling visibility with efficiency
2.3. Ray object intersection for Disks & Cylinders

2.1 Find appropriate description

Main idea:

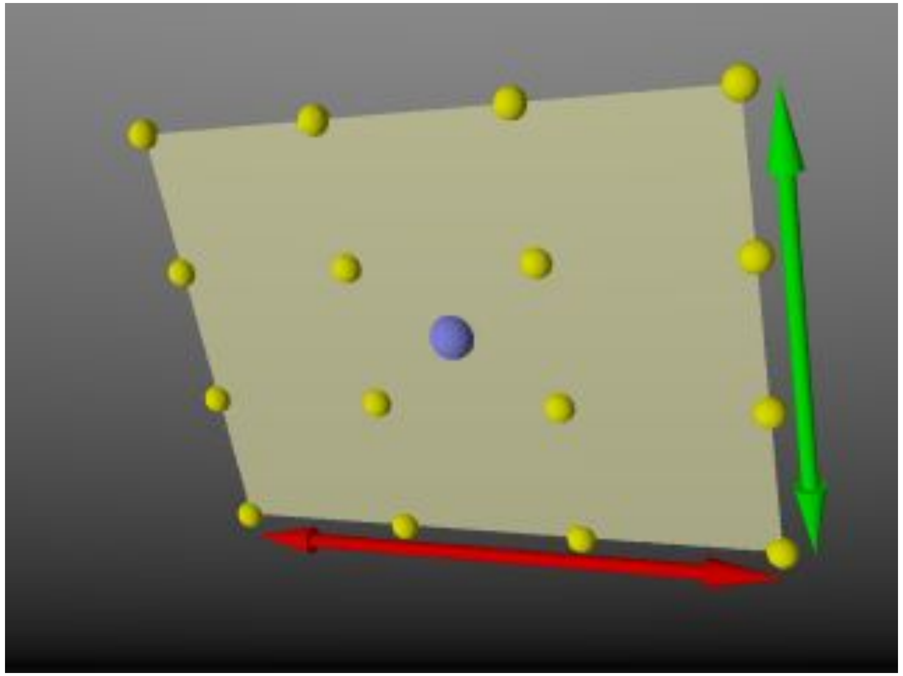
- Model area light sources as a set of Point Light Sources (PLS).



2.1 Find appropriate description

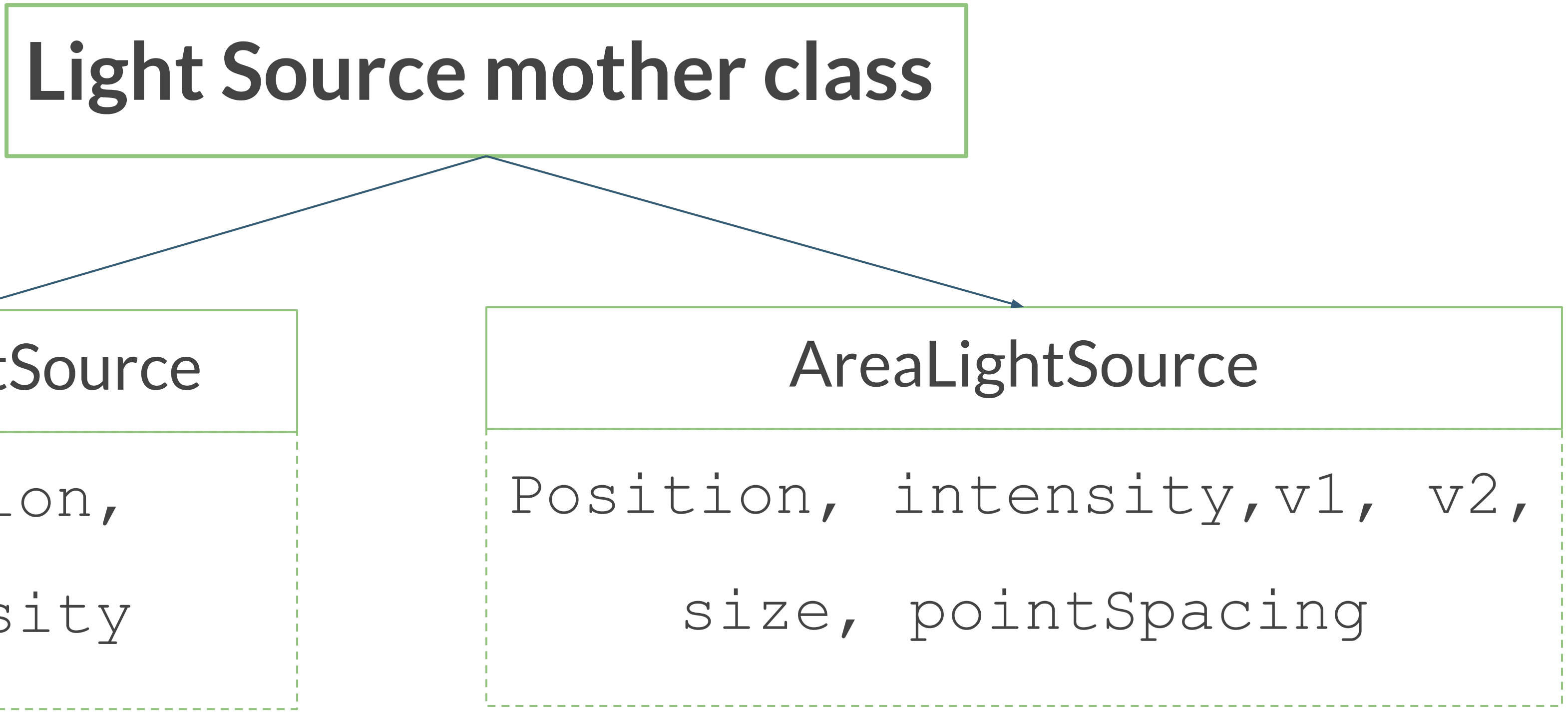
Our approach:

- Evenly distribute PLS in a finite plane defined by its center position and two vectors.
- As opposed to random points



2.1 Find appropriate description

Our implementation:



Light Source mother class

PointLightSource

```
Position,
intensity
```

AreaLightSource

```
Position, intensity,v1, v2,
size, pointSpacing
```

2.1 Find appropriate description

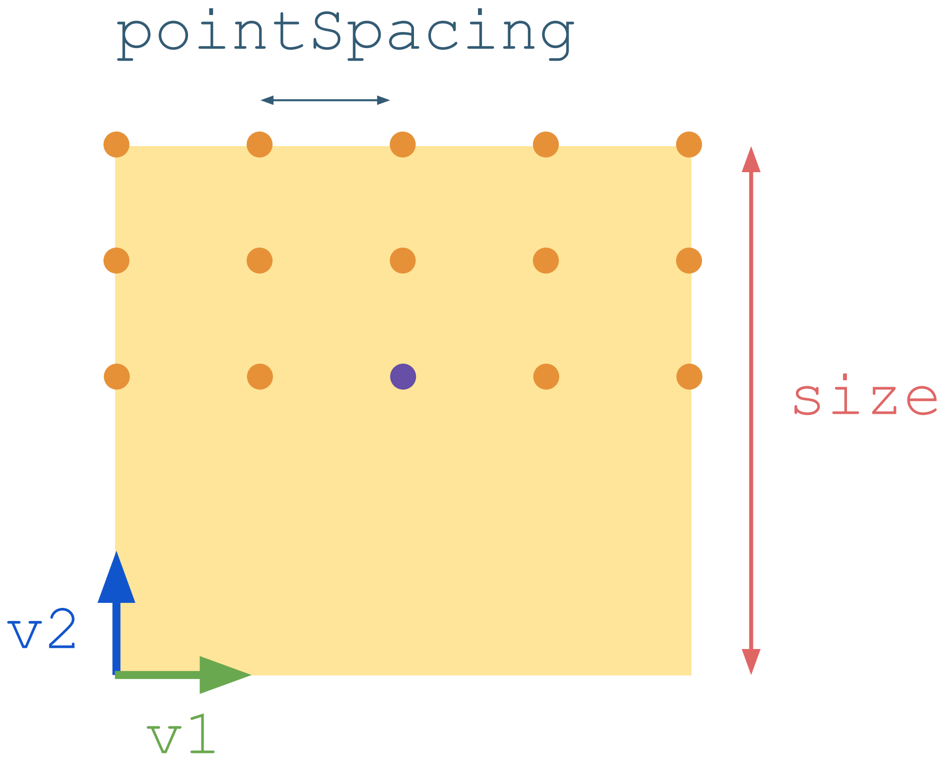
AreaLightSource

Position, intensity,v1, v2,

size, pointSpacing

lightSourceList ● ●

keyPointList



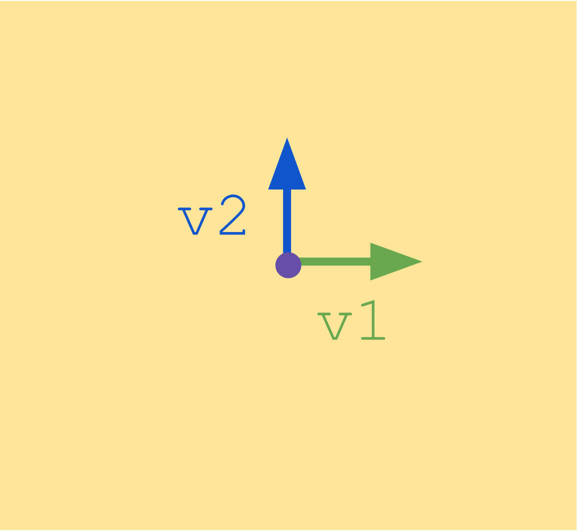
2.1 Find appropriate description

Our implementation:

- Sample points by vector equation of plane

```
Points = Position + A*v1 + B*v2

where A, B ∈ [-size/2 , size/2]
```

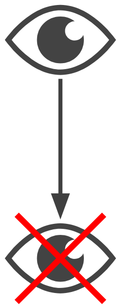


2.2 Sampling visibility with efficiency

Our main idea:

Precompute if the area lightsource is

- Totally visible (Points completely lit)
- Partially occluded (Some point in penumbra)
- Totally occluded (Points in umbra)



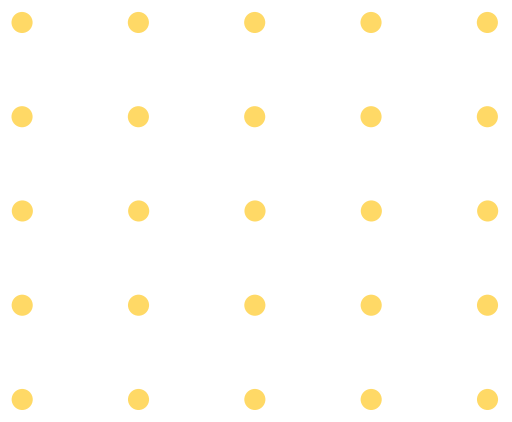
Without testing hasIntersection() for each pointlightSource

2.2 Sampling visibility with efficiency

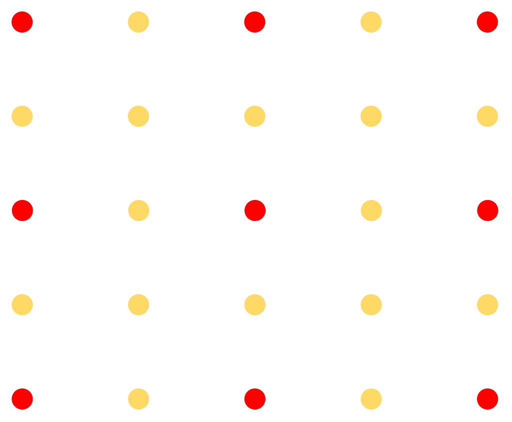
Our approach:

- Create a list with the keypoints of the arealightsource.

Area light source points



Area light source keypoints



2.2 Sampling visibility with efficiency

Our approach:

- Shader checks visibility of keypoints:

```
if(all keypoints are visible)
    Assume all points are visible
if (some keypoints are visible)
    Check visibility for all points
if (no keypoints are visible)
    Assume all points are not visible
```

2.2 Sampling visibility with efficiency

Outcome:

- Visibility for all the points in the area light source is **only checked** for the **points** corresponding to **penumbra** and **keypoints**.

2.3 Ray Object intersection

Cylinder

- Compute coefficients intersecting ray and Cylinder Equations

A. Cylinder Equation

$$x^2 + y^2 - r^2 = 0.$$

B. Cylinder Equation substituting by ray equation

$$(\mathrm{o}_x + t\mathbf{d}_x)^2 + (\mathrm{o}_y + t\mathbf{d}_y)^2 = r^2.$$

C. Obtained coefficients for t

$$a = \mathbf{d}_x^2 + \mathbf{d}_y^2$$

$$b = 2(\mathbf{d}_x \mathrm{o}_x + \mathbf{d}_y \mathrm{o}_y)$$

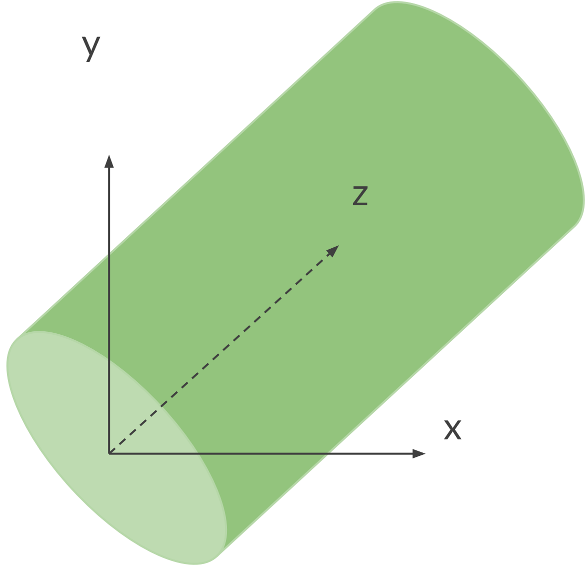
$$c = \mathrm{o}_x^2 + \mathrm{o}_y^2 - r^2.$$

2.3 Ray Object intersection

Cylinder

- Compute eq solver with coefficients a,b and c and check roots

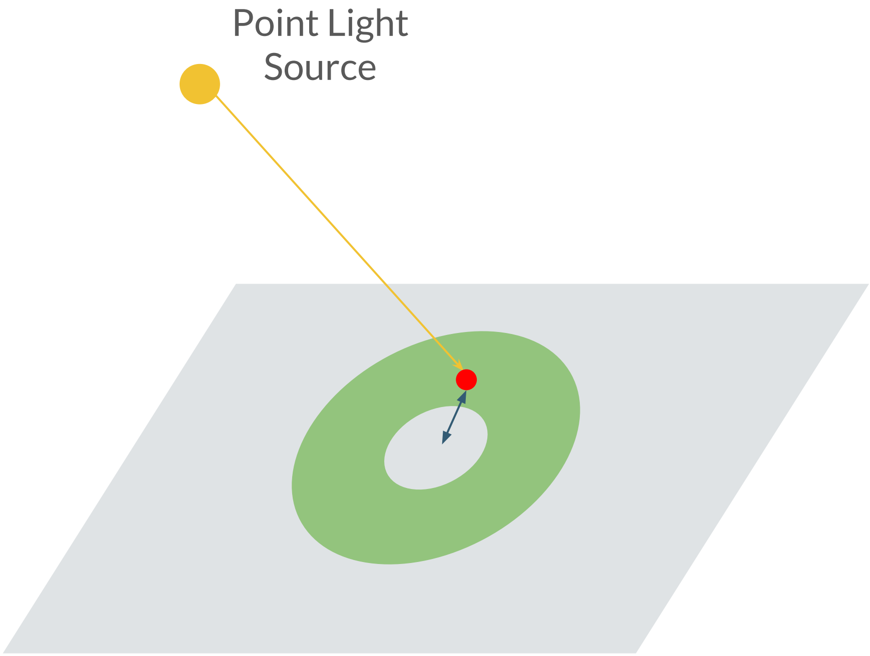
- Check if **z** value is between zMin and zMax



2.3 Ray Object intersection

Disk

- Modified infinite plane ray intersection.
- Compute distance from intersection to center.



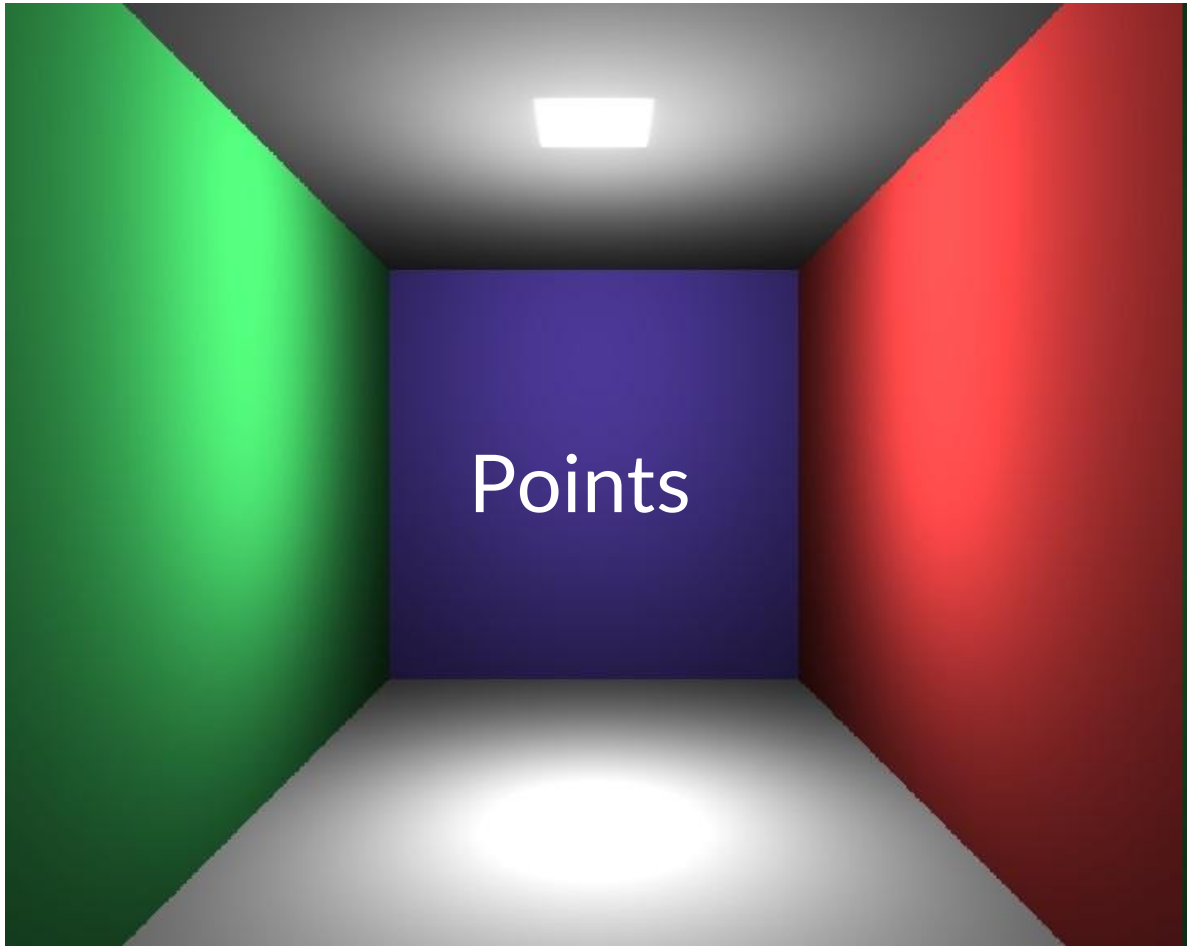
3. Results

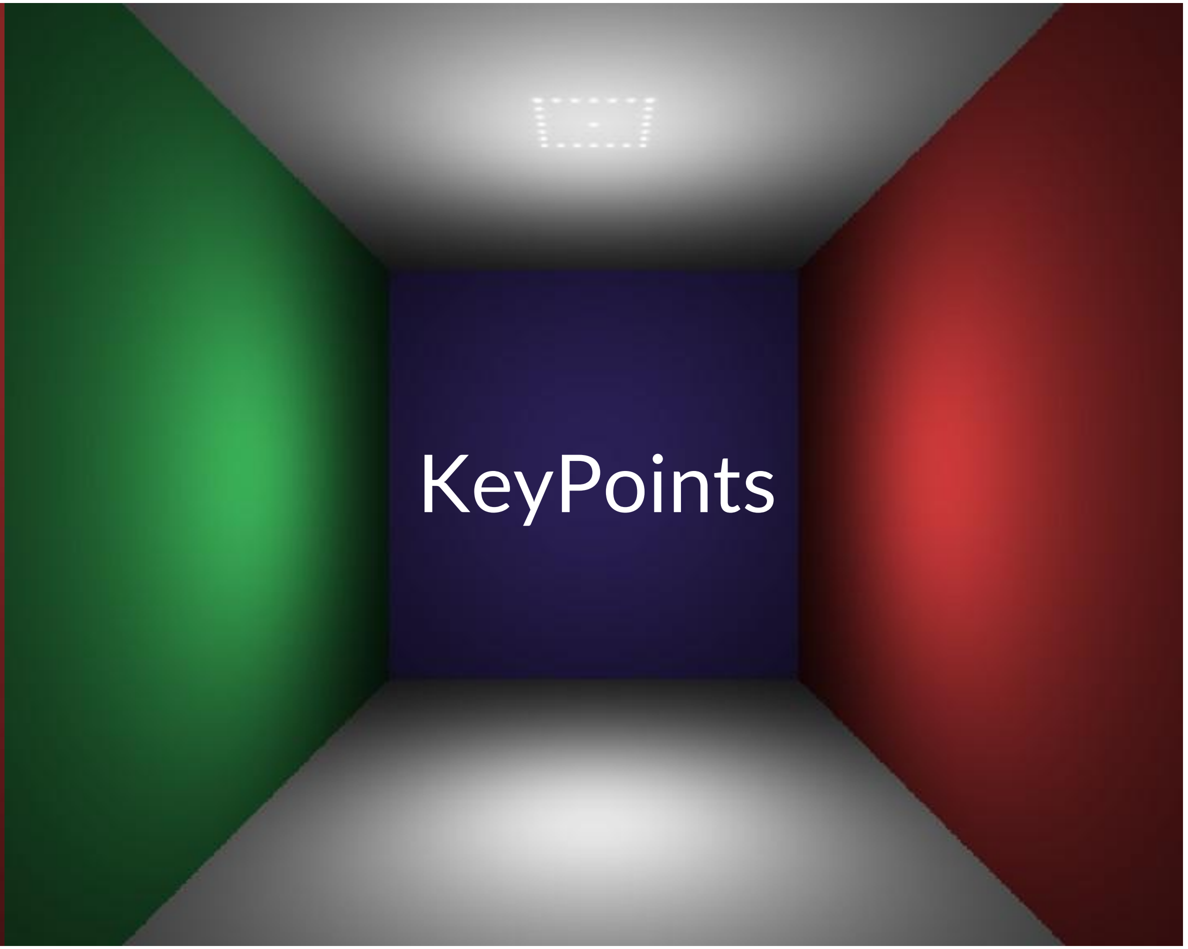
3.1. Visualization of areaLighSources

3.2. Shadow comparisons

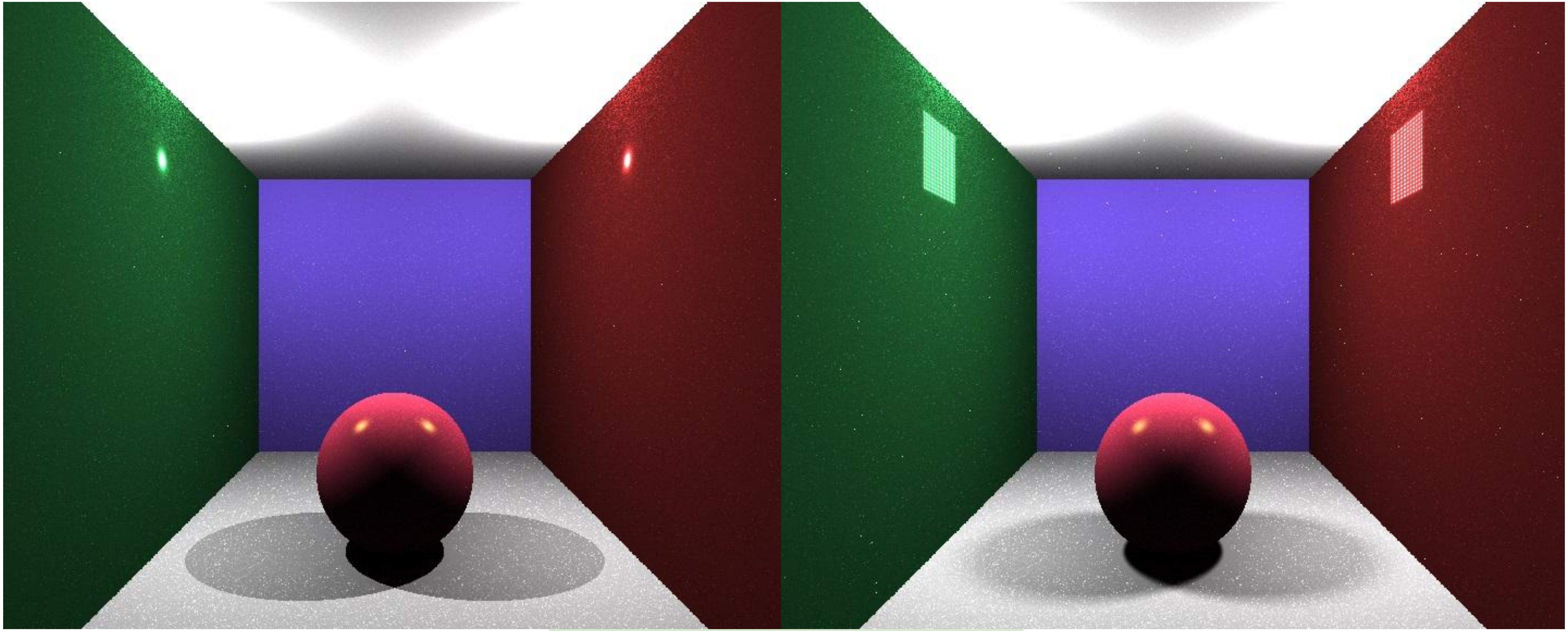
3.3. Scenes

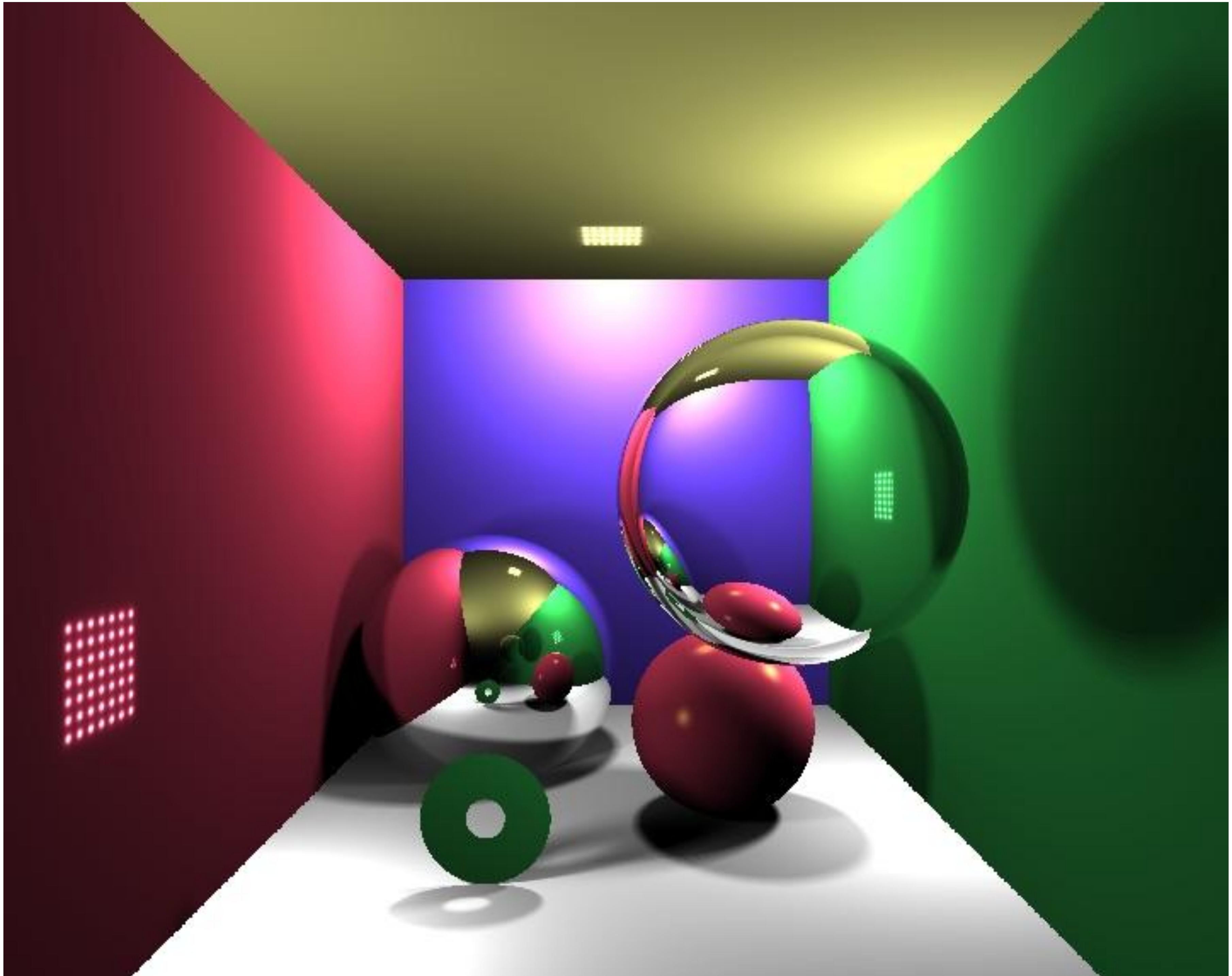
3.1 Visualization of our areaLightSources

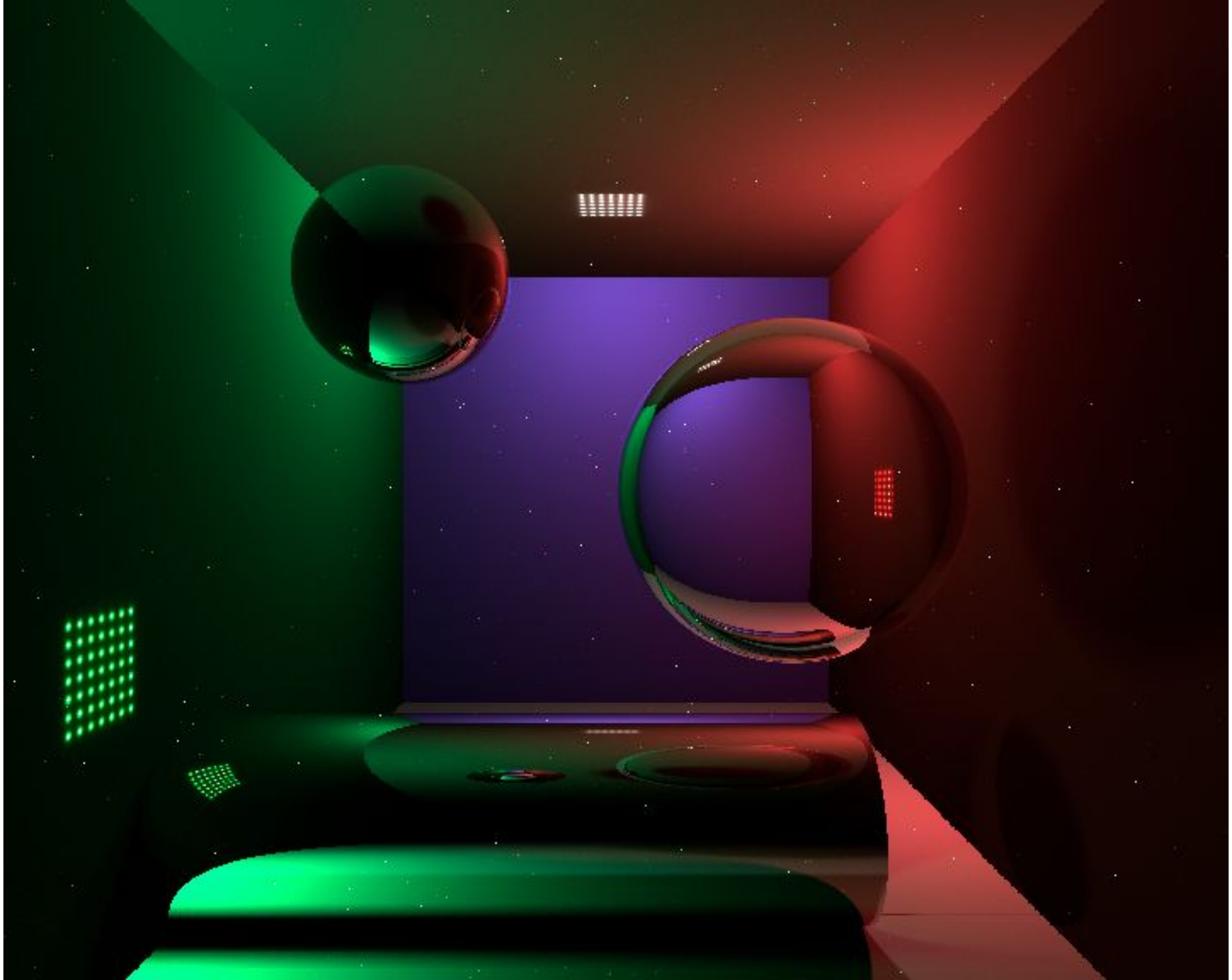




3.1 Shadow Comparison







4. Discussion

4.1. First evaluation

4.2. Improvements

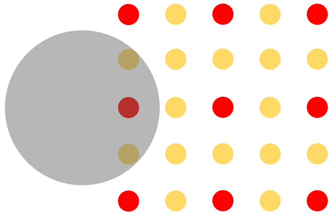
4.3. Final evaluation

4.1 First evaluation

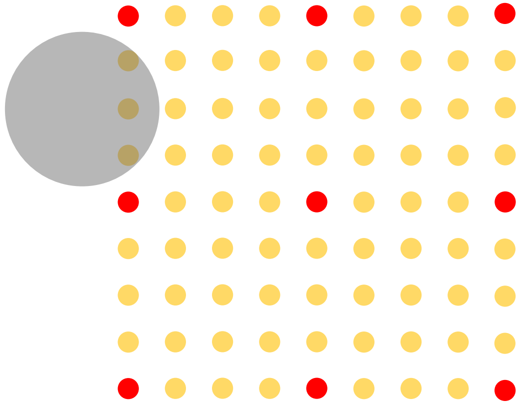
Outcome:

- Using only 9 keypoints causes artifacts for larger areaLightSource.

Smaller area light source



Larger area light source



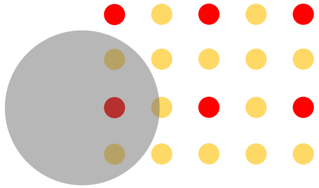
4.2 Improvements

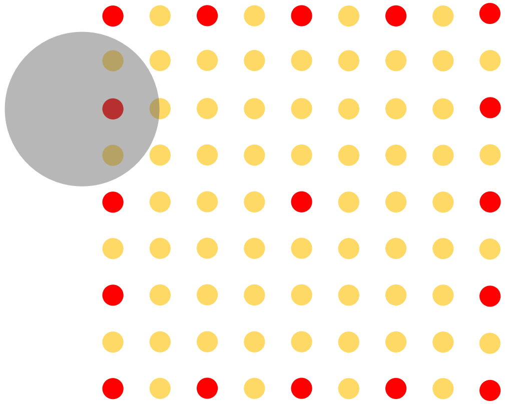
2nd approach:

- Instead of using only 9 points:

  → Step between keypoints = 2*Step between point light sources

Smaller area light source **Larger area light source**





4.3 Final evaluation

Improved outcome:

- Still have less computational cost when checking visibility

Keypoint
vs
Brute force

2.5x
FASTER

Thank you for your attention!