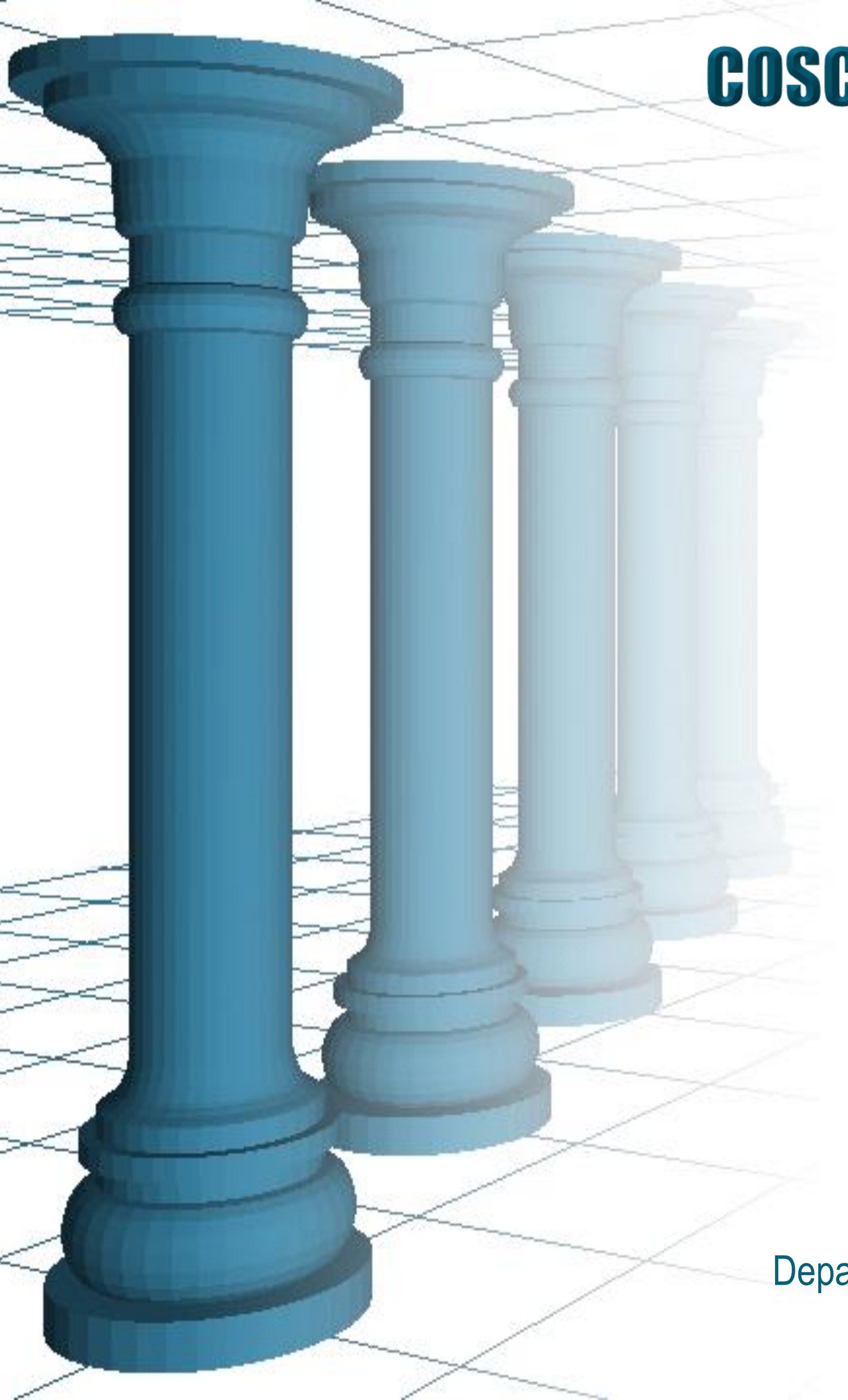
COSC363 Computer Graphics



Assignment-2

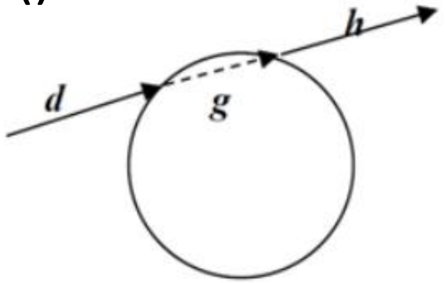
R. Mukundan (mukundan@canterbury.ac.nz)
Department of Computer Science and Software Engineering
University of Canterbury, New Zealand.

Assignment-2

- Due: 11:55pm, **2 June 2023**
  - Late penalty:
  - -1.5 marks for submissions received from 3 June to 6 June
  - -3 marks for submissions received from 7 June to 9 June
- Drop-dead date: 9 June 2023.
- Your submission must be based on Lab 6,7 code. Implementations using shaders, path tracing, photon mapping etc., not allowed.
- Not a group project. Your submission must represent your own individual work
- Students are encouraged to discuss assignment related problems using course forum. However, code segments or any part of your assignment submission should not be posted on Learn.

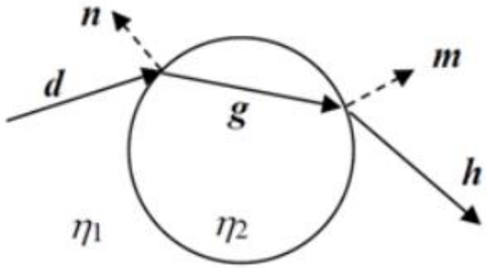
Assignment Specs

- Minimum Reqs (Max. 10 marks)
  - A good spatial arrangement of objects inside a box
  - Shadows
    - lighter shadows for transparent and refractive objects
  - One planar mirror-like object
  - Chequered pattern on a planar surface
  - A transparent object. Even though transparency may be treated as a special case of refraction where $\eta_1 = \eta_2$, the implementation of transparency effect does not require the refract() function.



Transparency

d = g = h



Refraction

g = refract(d, …)

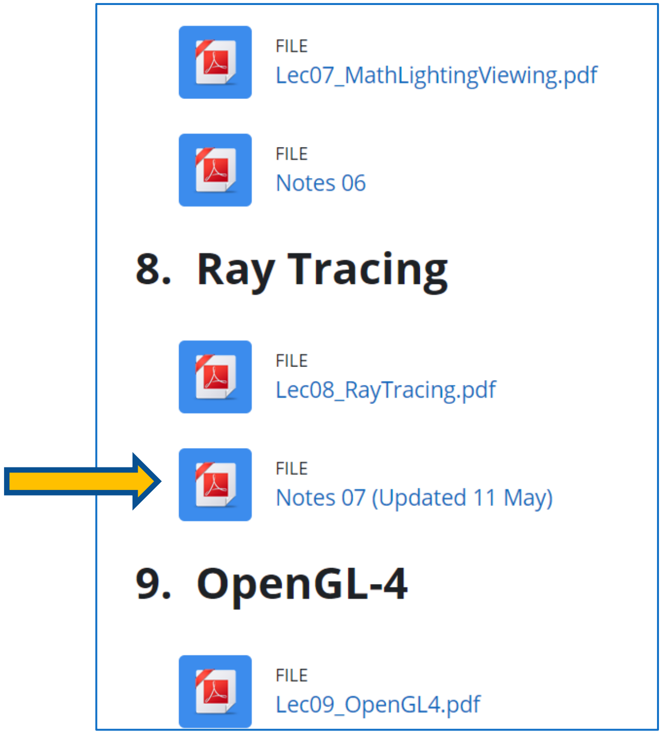
h = refract(g, …)

Assignment Specs

- Extensions (Max. 7 marks)
  - Cone/Double cone, Cylinder, Torus (?), + Cap
  - Refraction
  - Multiple light sources: multiple shadows, specular highlights
  - Multiple reflections on parallel surfaces
  - Spotlight
  - Anti-aliasing
  - Non-planar object textured using an image
    - E.g., textured sphere, textured cylinder.
  - Procedural patterns
  - Fog
  - Depth of field
  - Soft shadows

Supplementary Notes

- Information on modelling transparency, multiple light sources and shadows, spotlights, and fog can be found in "**Notes 07**" (Note07_RayTracing.pdf) in lecture material section.



FILE
Lec07_MathLightingViewing.pdf

FILE
Notes 06

8. Ray Tracing

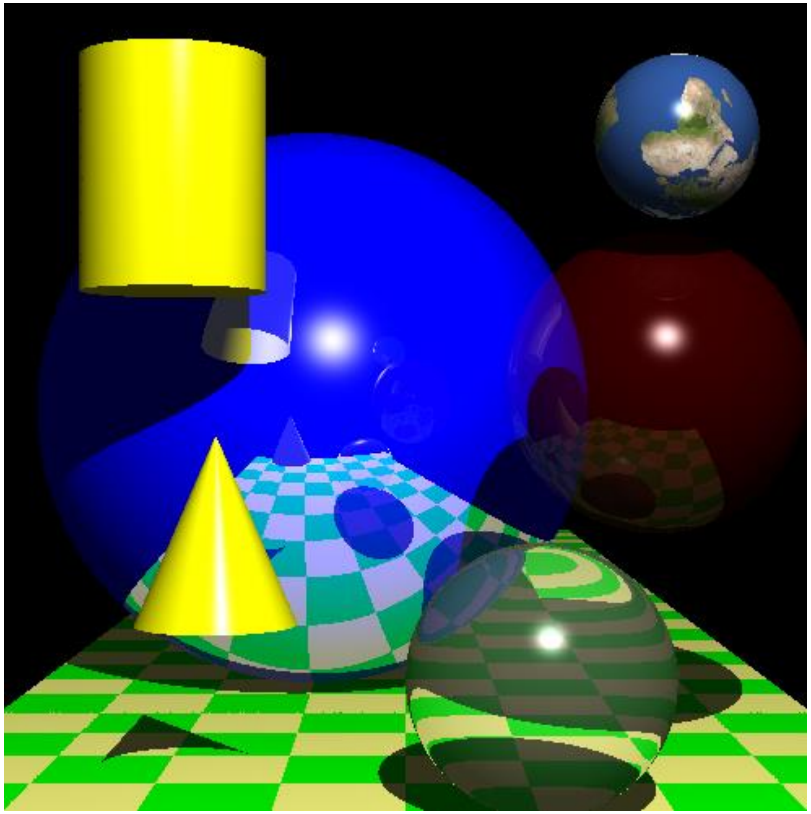
FILE
Lec08_RayTracing.pdf

FILE
Notes 07 (Updated 11 May)

9. OpenGL-4

FILE
Lec09_OpenGL4.pdf

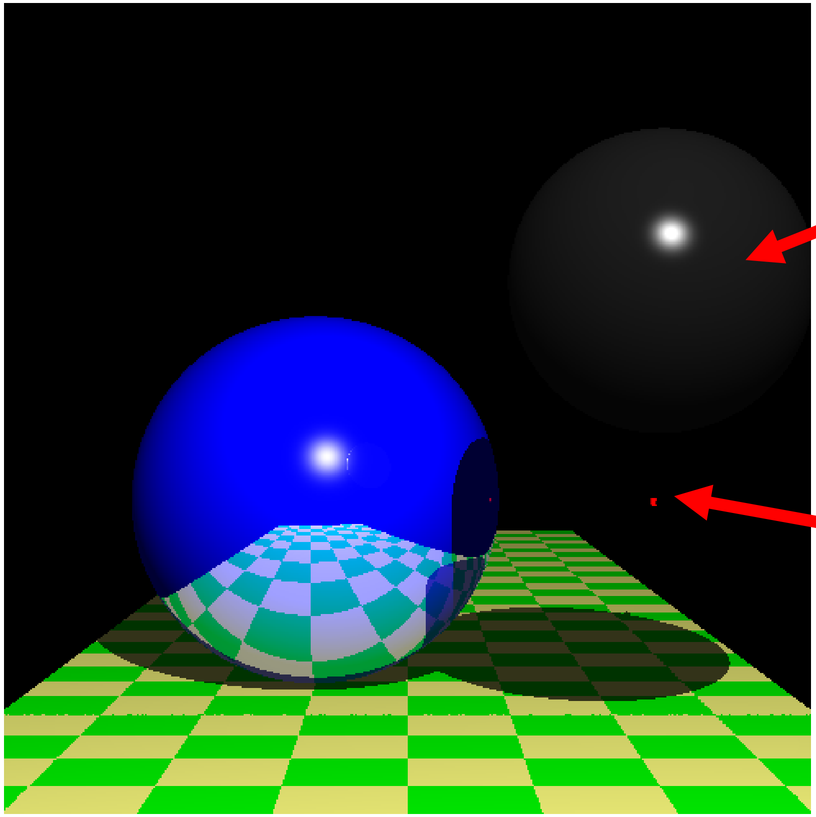
Bad Design



- Random placement of objects
- Objects and features not clearly visible
- Scene clutter
- Incorrect mapping of textures

Bad Design

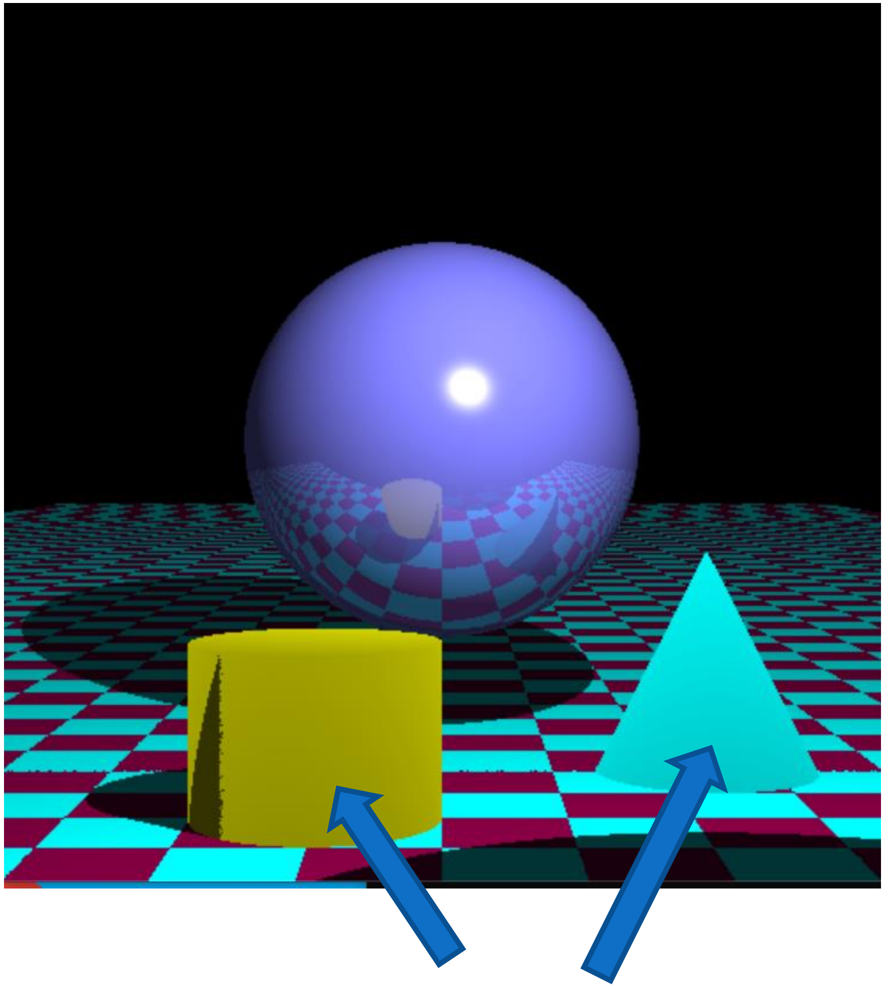
Marks will not be given to features not clearly visible in the output.



Refractive sphere ?

Cylinder ?   Cone ?

Bad Design



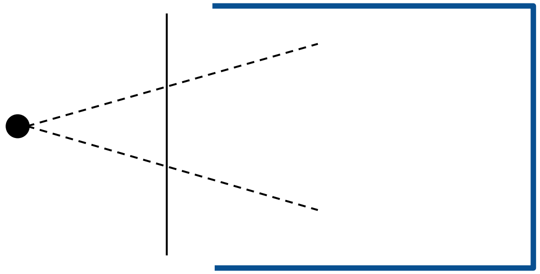
- Improper lighting

Examples of some of the Minimum Requirements

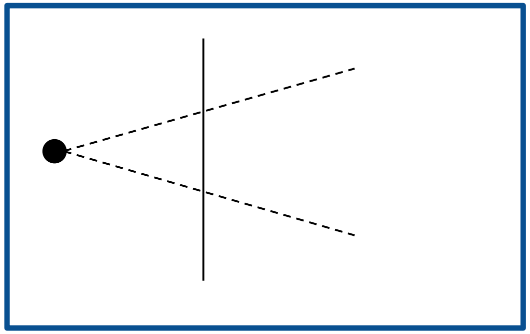
Box

- A box environment is commonly used for testing global illumination algorithms
  - E.g. Cornell Box (Wikipedia)

- 5 or 6 axis-aligned planes, each having a different colour or pattern.

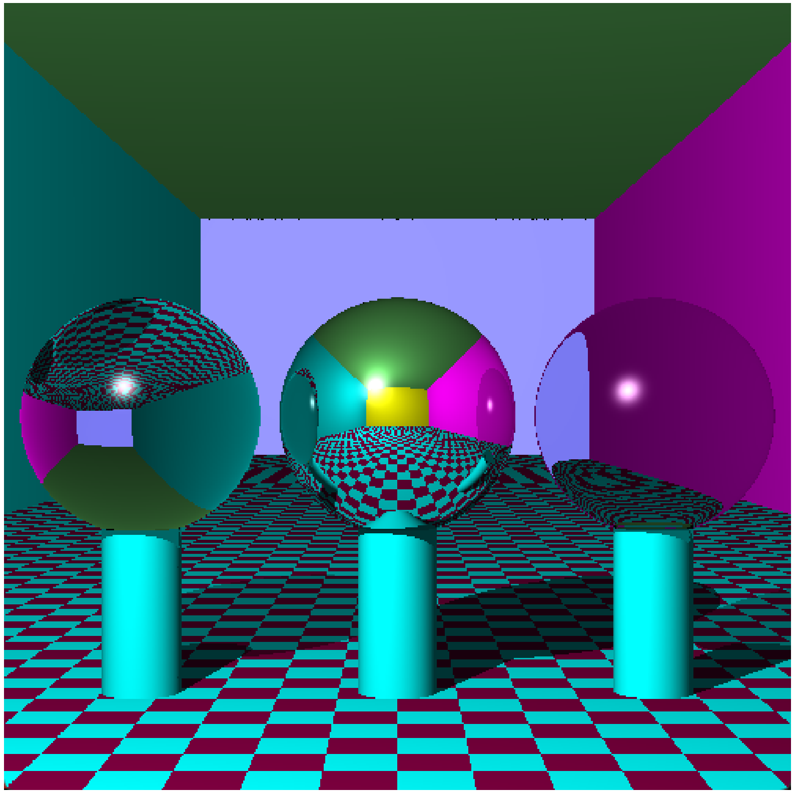


5 Planes



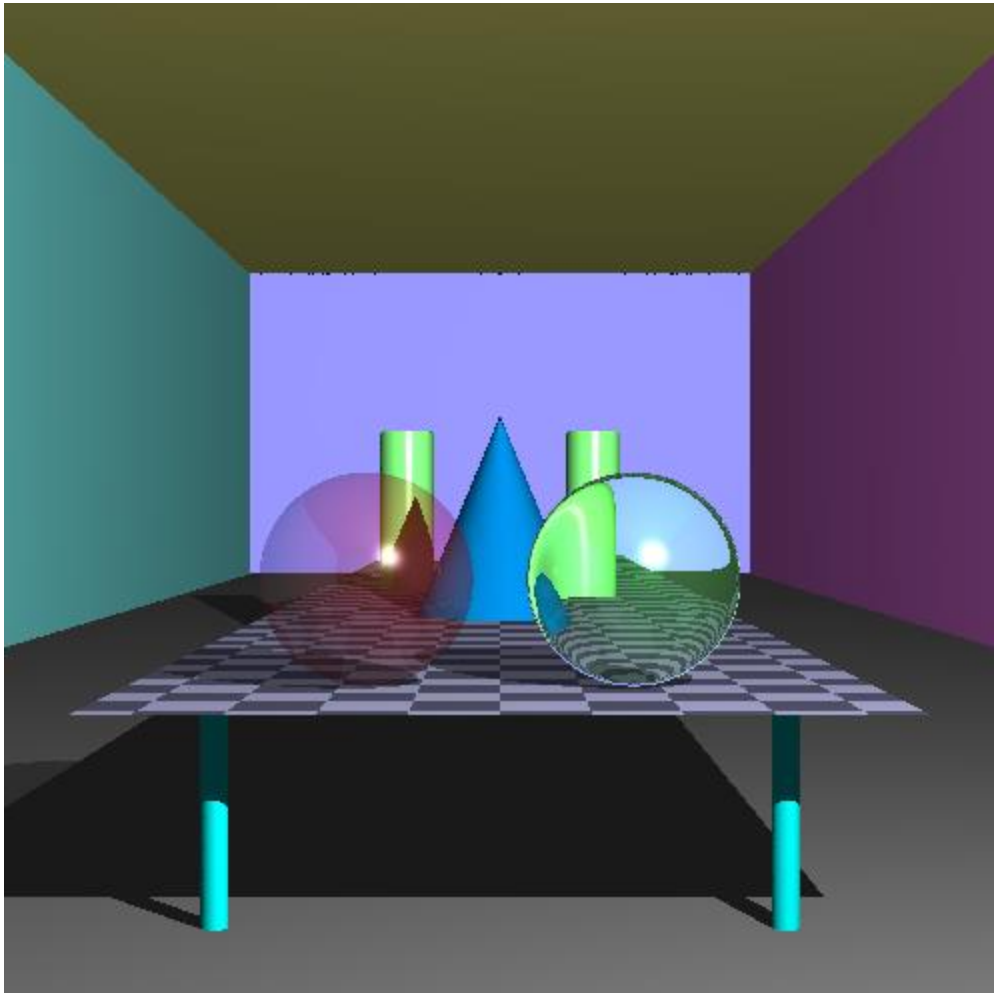
6 Planes

Box: Example



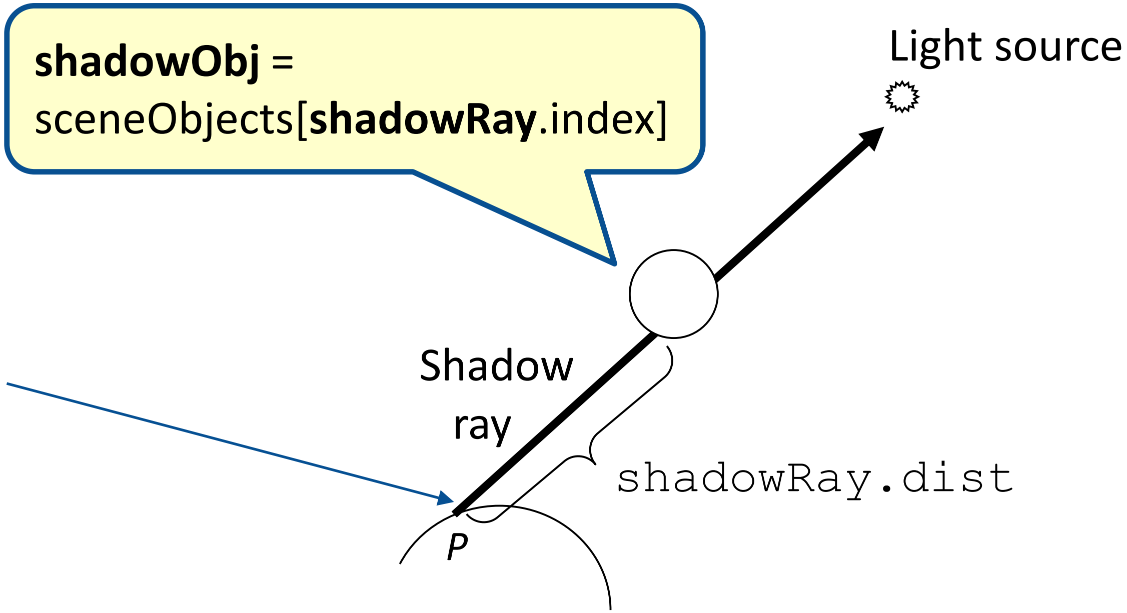
- The above example uses 6 planes for the box. Note the reflection of the back plane in yellow colour on the middle sphere
- Spheres: Refractive (η =1.5), Reflective, Refractive (η = 1.005)
- Refractive spheres cast lighter shadows

Transparent Object



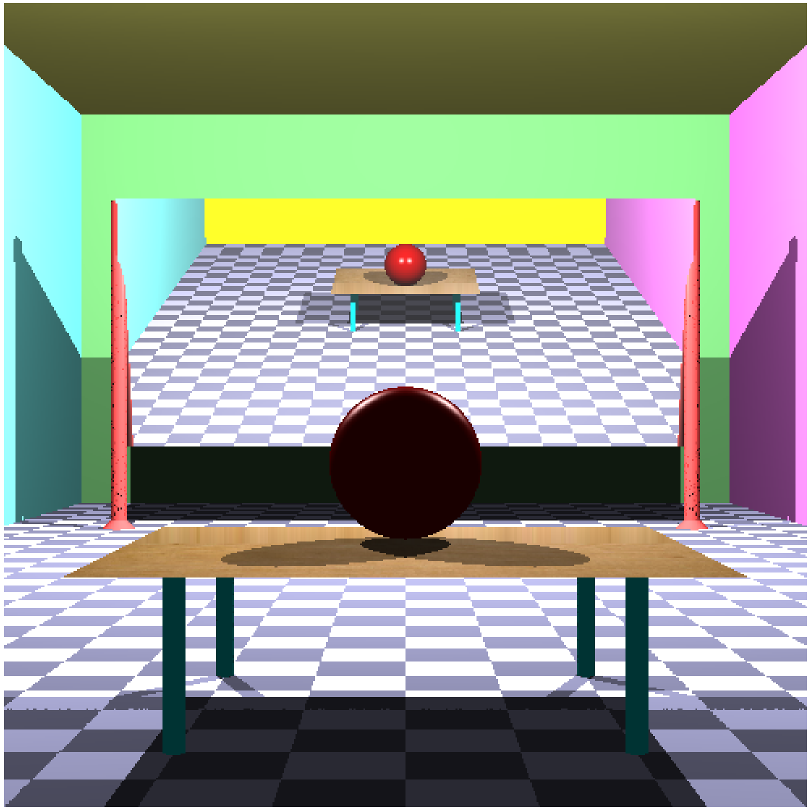
- Spheres: Transparent, Refractive (η = 1.01)
- Transparent and refractive spheres cast lighter shadows

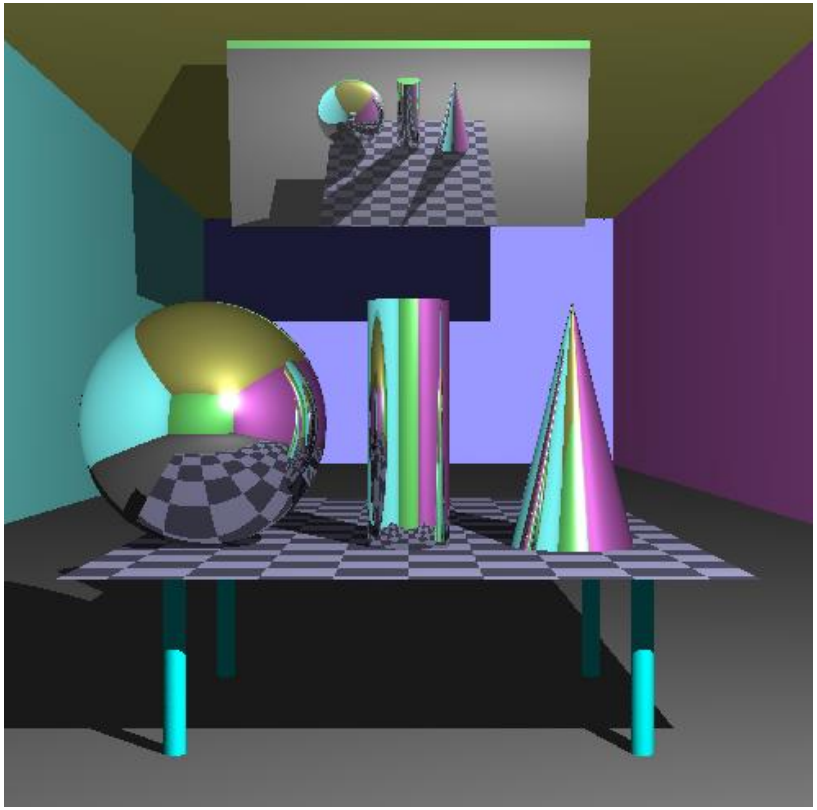
Lighter Shadows



- If `shadowObj` is reflective or refractive, assign a lighter ambient color (higher value for ambient term)

Mirror: Examples

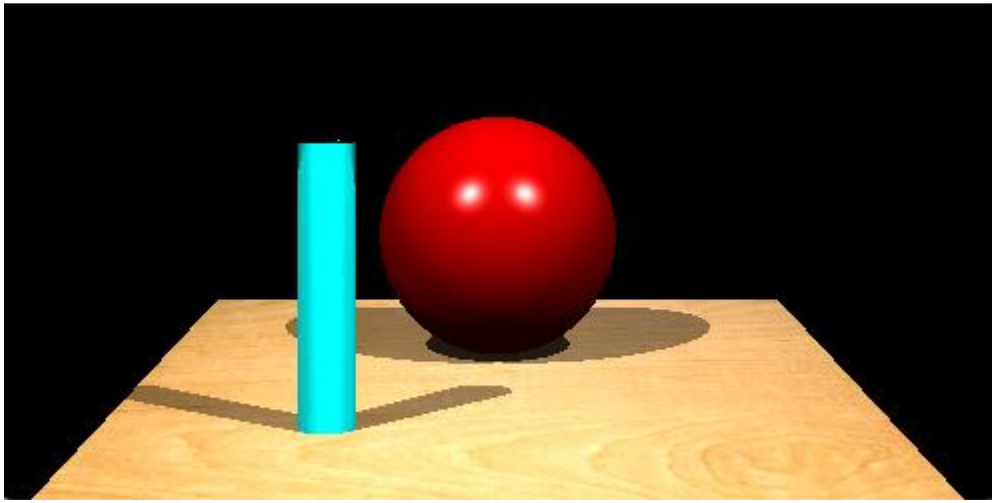




- The first example above contains two light sources

Extra Features

Multiple Lights



- Please trace shadow rays to each of the light sources to generate multiple shadows of objects in the scene.
- Multiple specular highlights must be visible on at least one object.

Multiple Light Sources

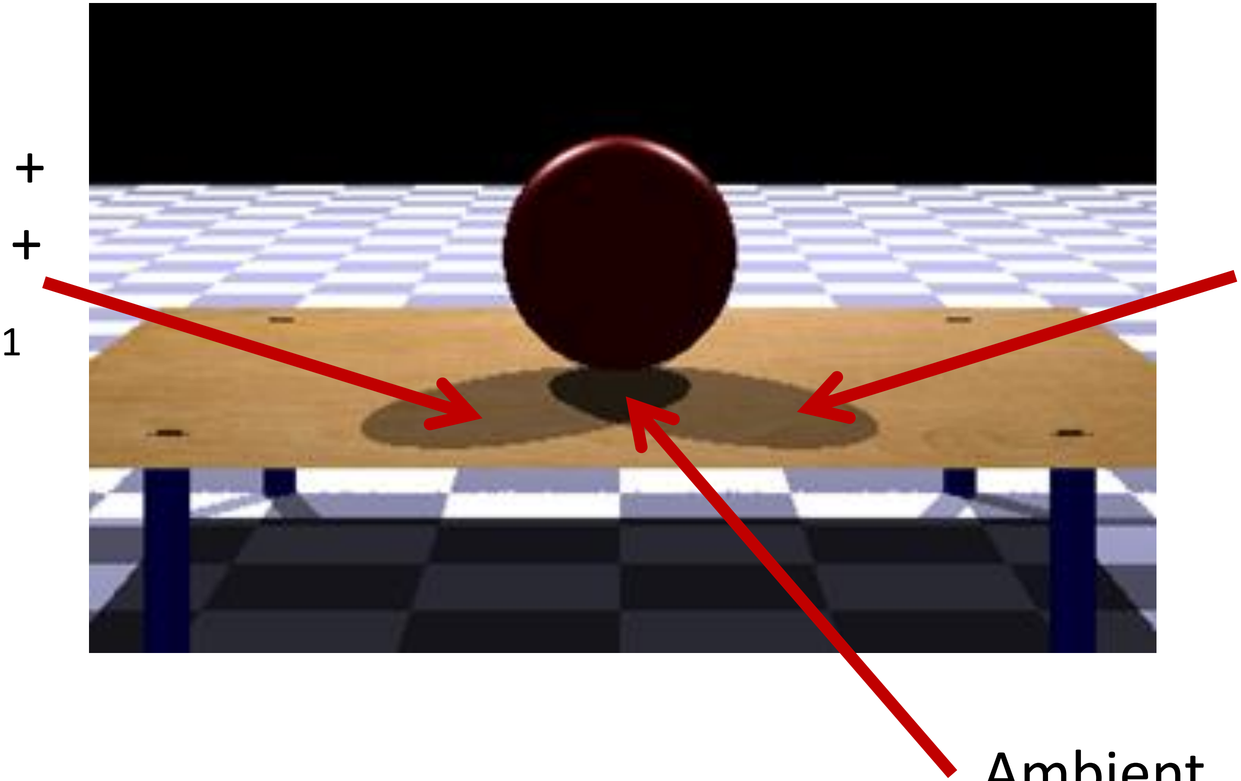
$$\text{colour} = (\text{ambient term}) + (\text{diffuse term})_{L1} + (\text{specular term})_{L1}$$
$$+ \; (\text{diffuse term})_{L2} + (\text{specular term})_{L2}$$

- Use only one ambient term.
  - You may have to modify the function “lighting()” in the SceneObject class.
- Reduce intensity of light sources if required.

Shadows Under Multiple Light Sources

Light source 1

Light source 2

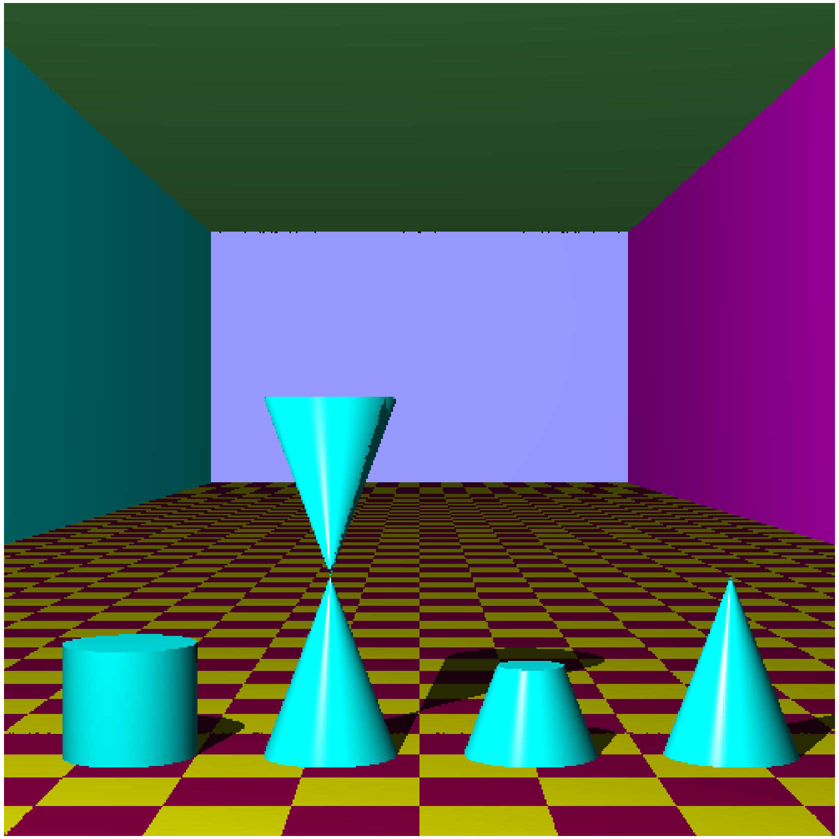


Ambient + Diffuse$_1$ + Specular$_1$

Ambient + Diffuse$_2$ + Specular$_2$

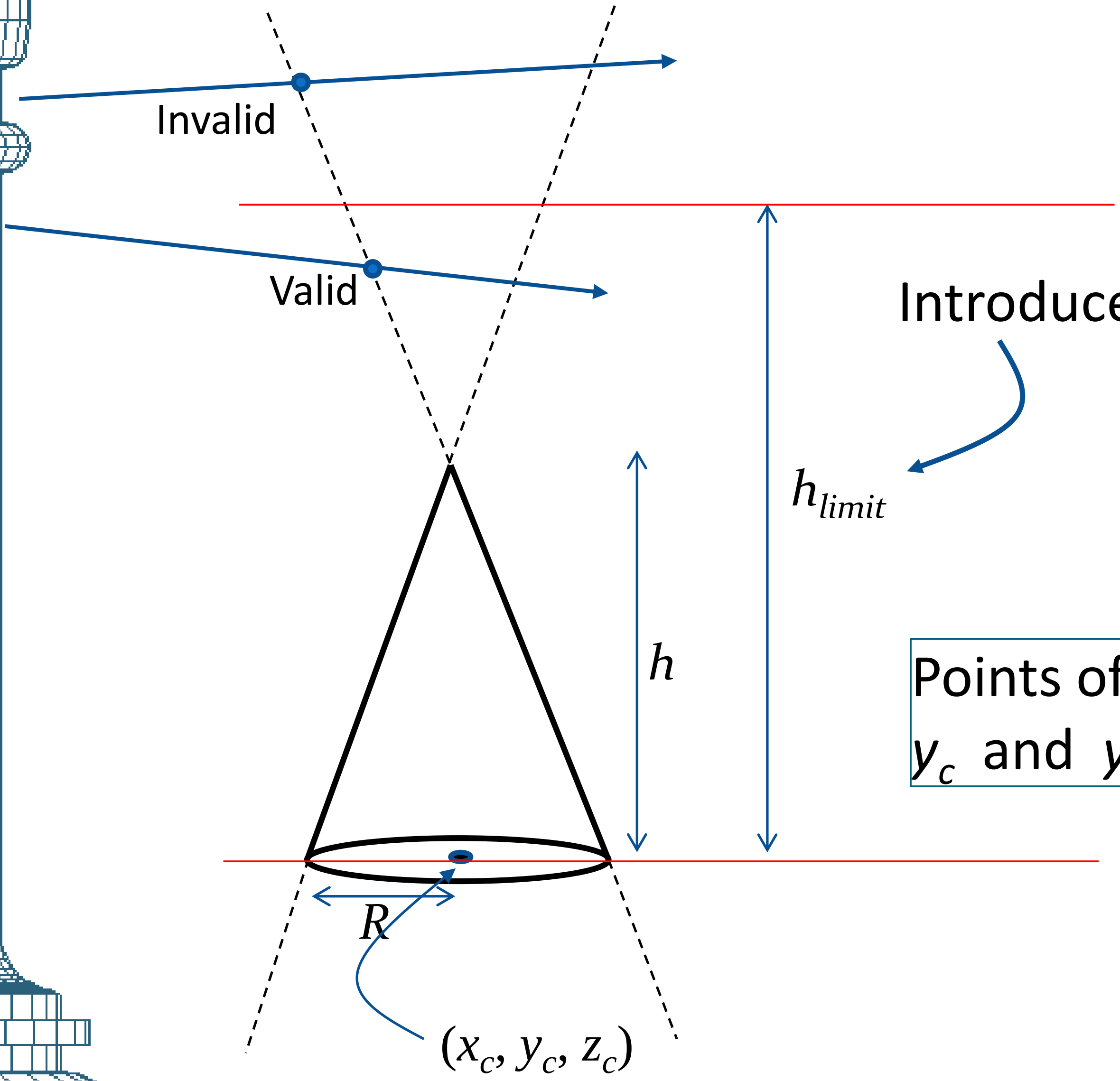
Ambient

Cones



- Double and truncated cones can be easily generated.
- In the above example, the cylinder and truncated cone have caps

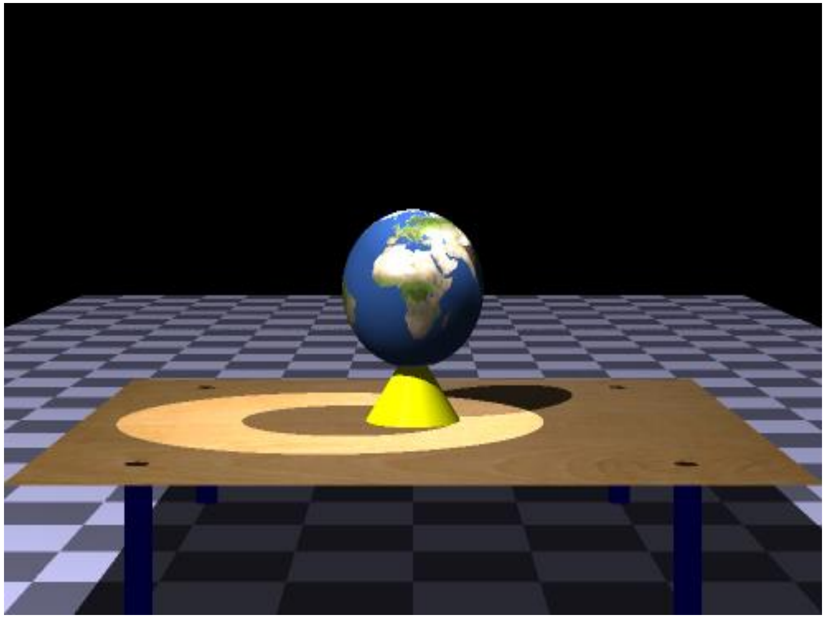
Cones

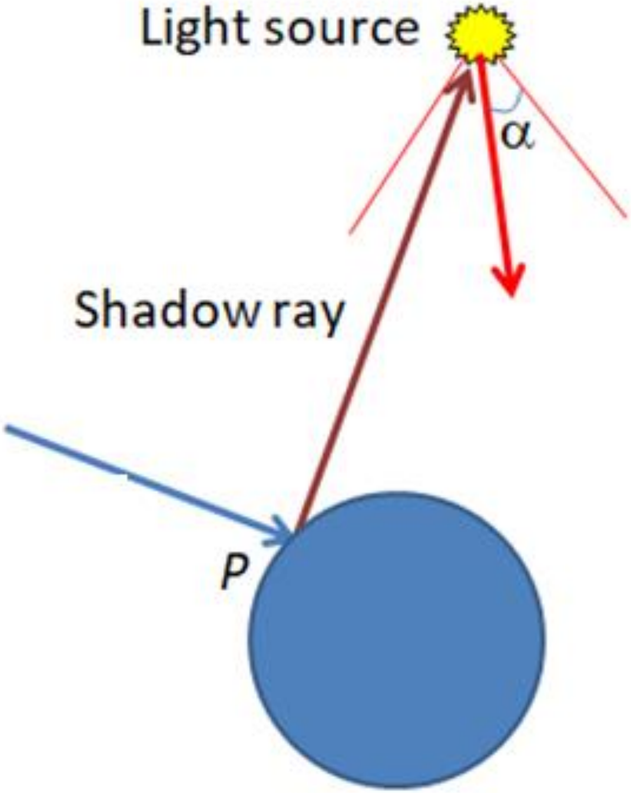


Introduce a new parameter h_{limit}

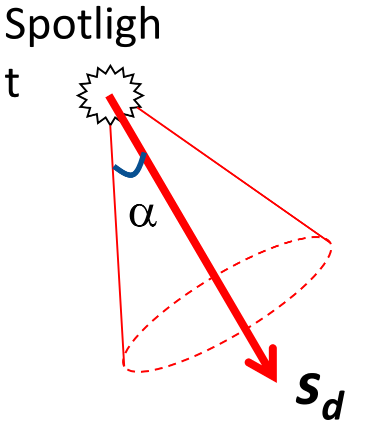
Points of intersection between y_c and y_c+h_{limit} are valid

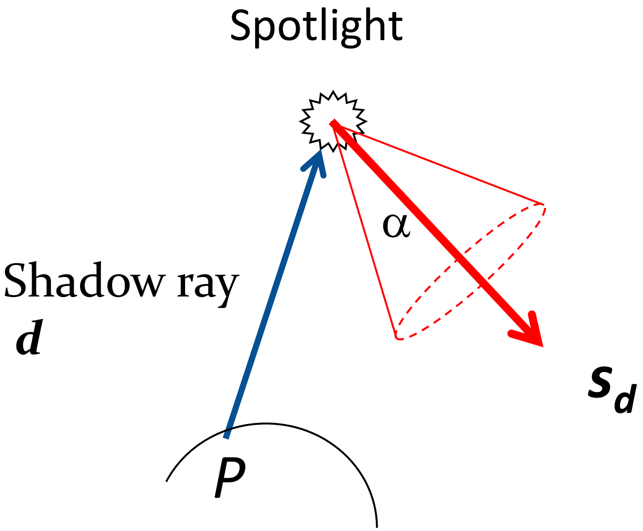
Spotlight





Spotlight

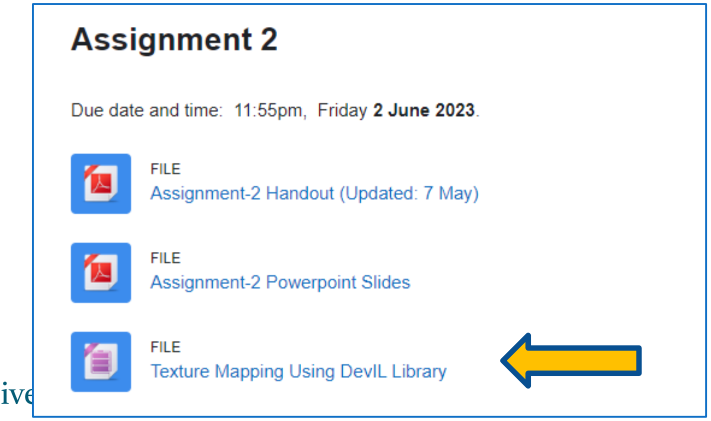




- Requires two additional parameters: a spot direction $\boldsymbol{s_d}$ and a cut-off angle α
- If the point *P* on an object is not in shadow, perform the following additional test:
  - If the angle between the shadow ray *d* and the spot direction s_d is greater than α, the object is in shadow.
  - Remember to use $-d$ in angle calculation!

Texture Mapping

- Images not loaded by OpenGL functions, and therefore not required to meet “power of 2” requirement for width and height.
- Bitmap Files:
  - 24 bits per pixel (not indexed color)
  - Uncompressed
  - Use files TextureBMP.h, TextureBMP.cpp (Lab07)
- Images in other formats (jpg, png etc.)
  - Requires DevIL/OpenIL library
  - Use files TextureOpenIL.h, TextureOpenIL.cpp (Assignment section on Learn)



Assignment 2

Due date and time: 11:55pm, Friday **2 June 2023**.

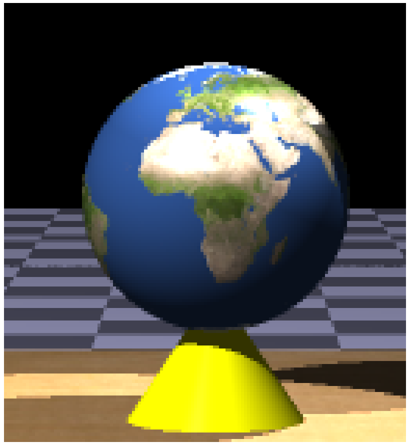
FILE
Assignment-2 Handout (Updated: 7 May)

FILE
Assignment-2 Powerpoint Slides

FILE
Texture Mapping Using DevIL Library

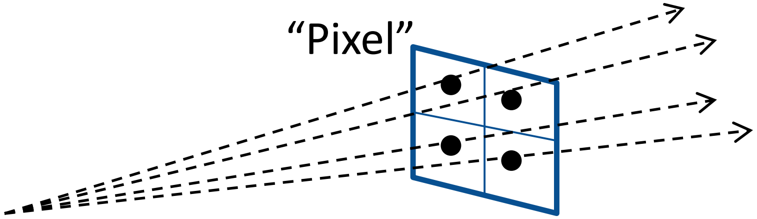
Texturing a Non-Planar Object

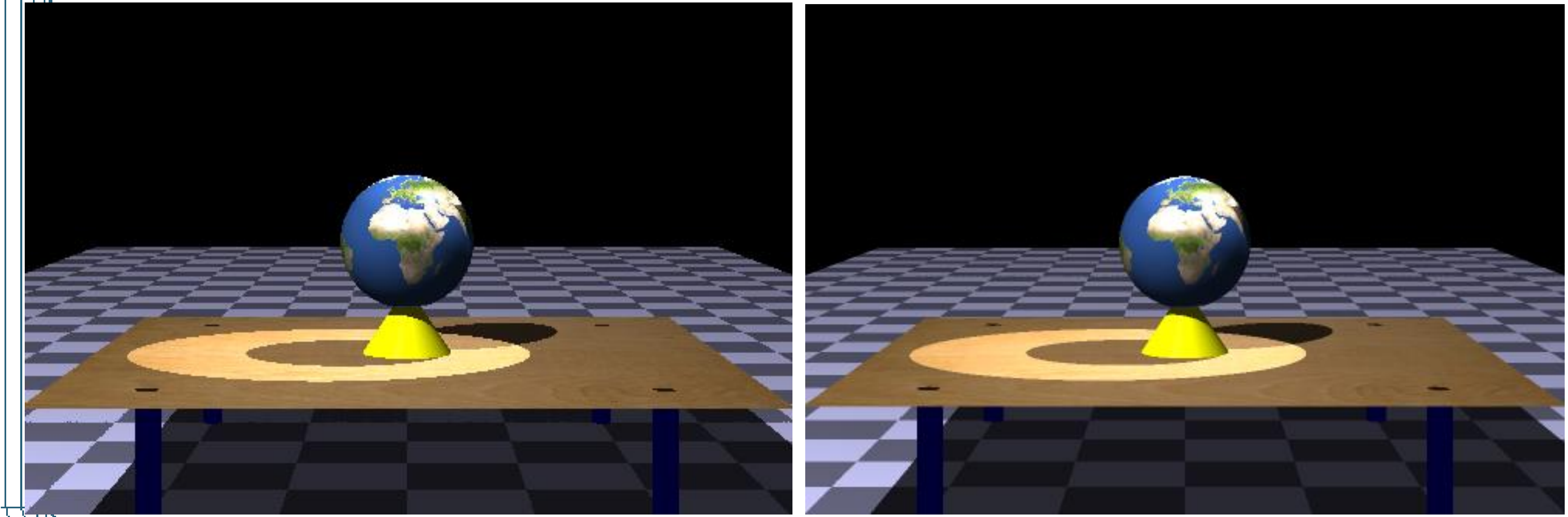
- Sphere: Compute spherical angles α, δ
  - Convert α to texture coordinate *s*
  - Convert δ to texture coordinate *t*
  - ⇨ Ref: Wikipedia: UV Mapping



- Cylinder: Computer cylindrical angle α
  - Convert α to texture coordinate *s*
  - Convert *y* to texture coordinate *t*

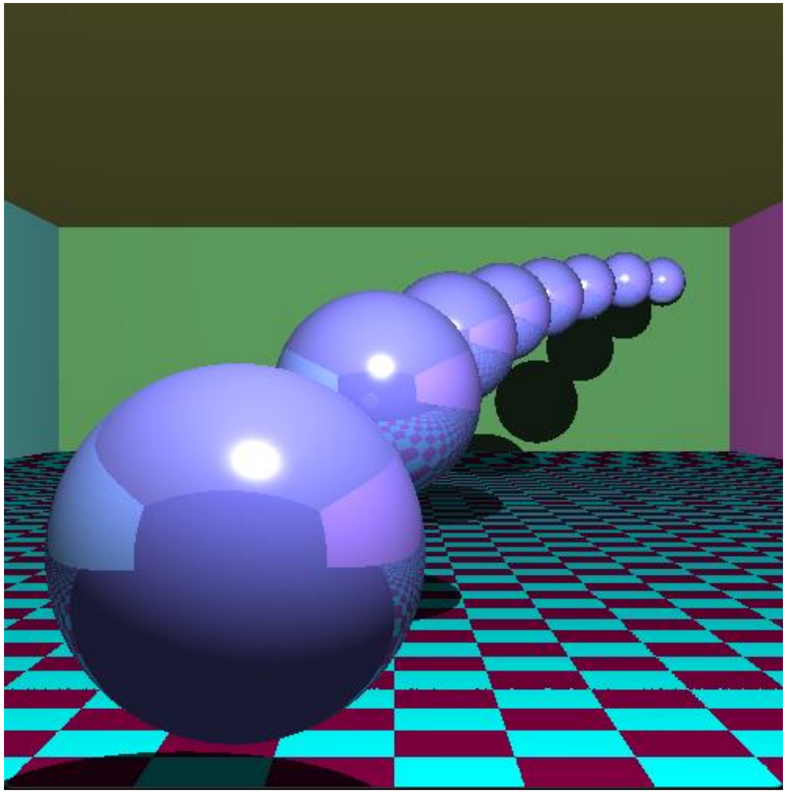
Anti-Aliasing

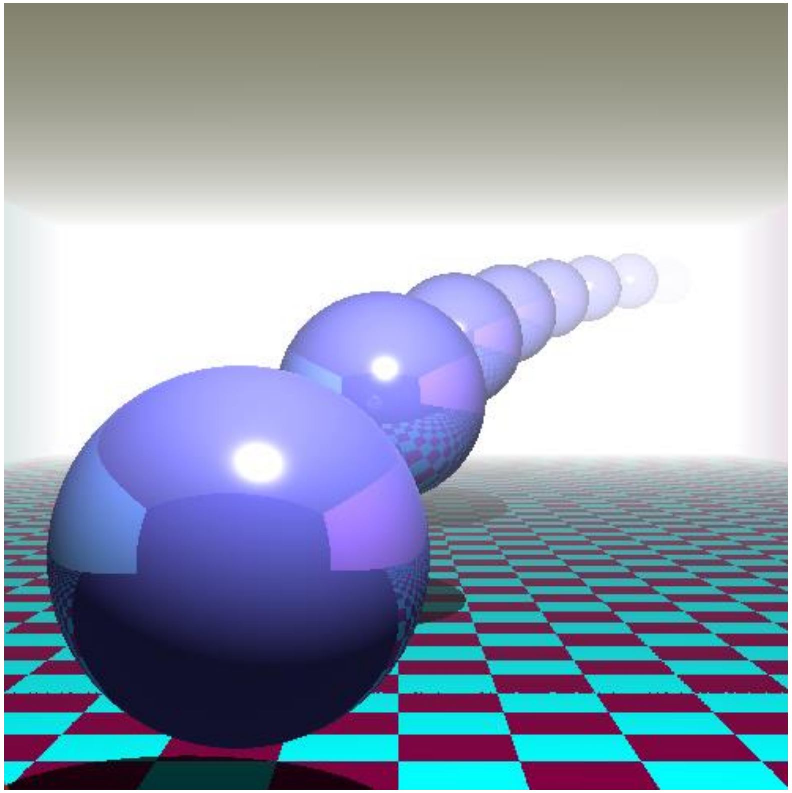




Please include screenshots of outputs with and without anti-aliasing.

Fog



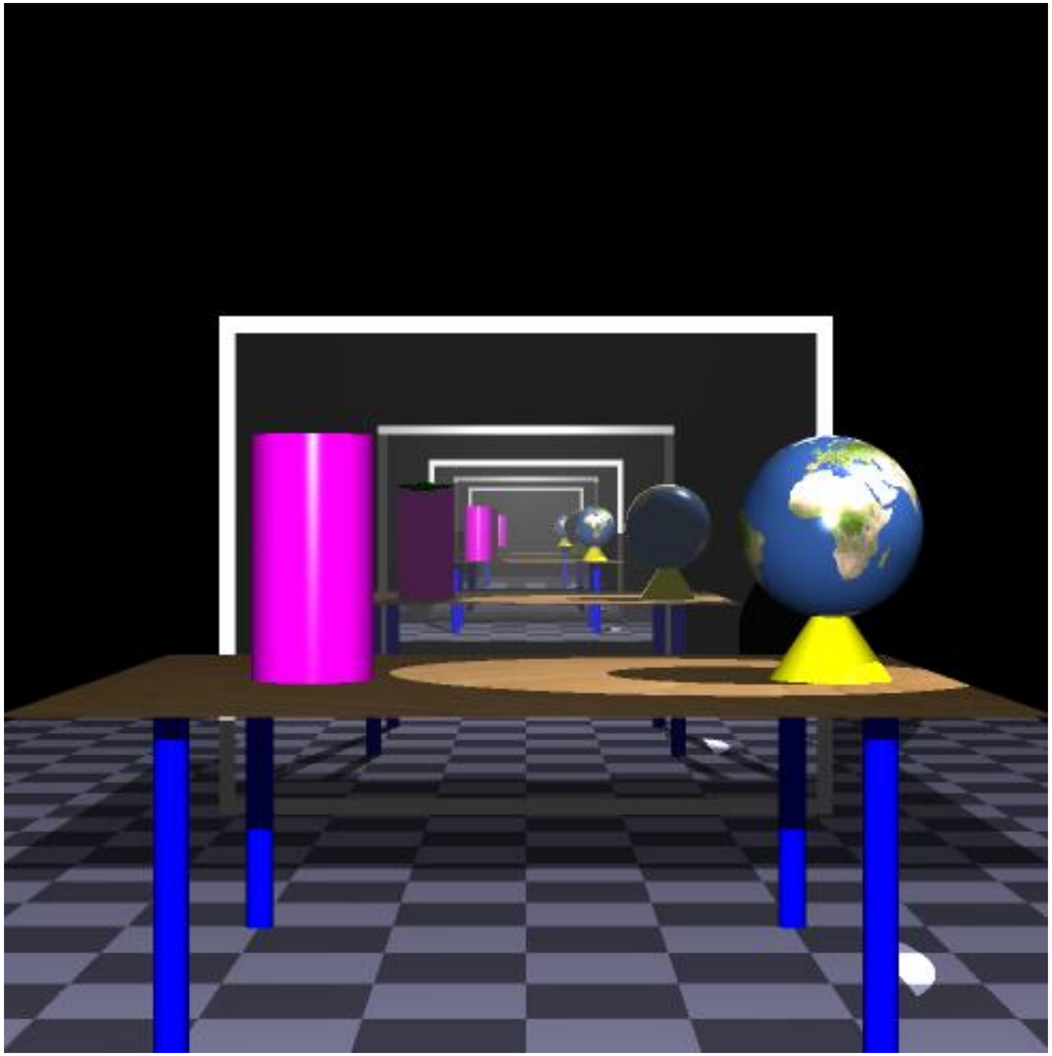


$$\lambda = \frac{(ray.hit.z) - z_1}{z_2 - z_1}$$

color = (1- λ) color + λ white

Please include screenshots of outputs with and without fog.

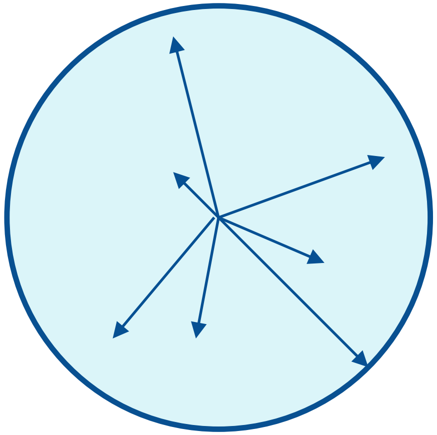
Multiple Reflections



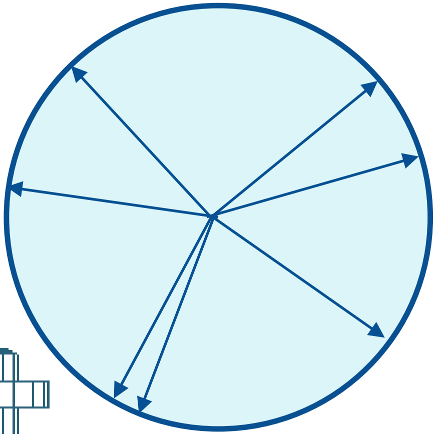
The camera must be placed between the two reflecting surfaces

Generating Random Vectors Using GLM

```
#include <glm/gtc/random.hpp>
```

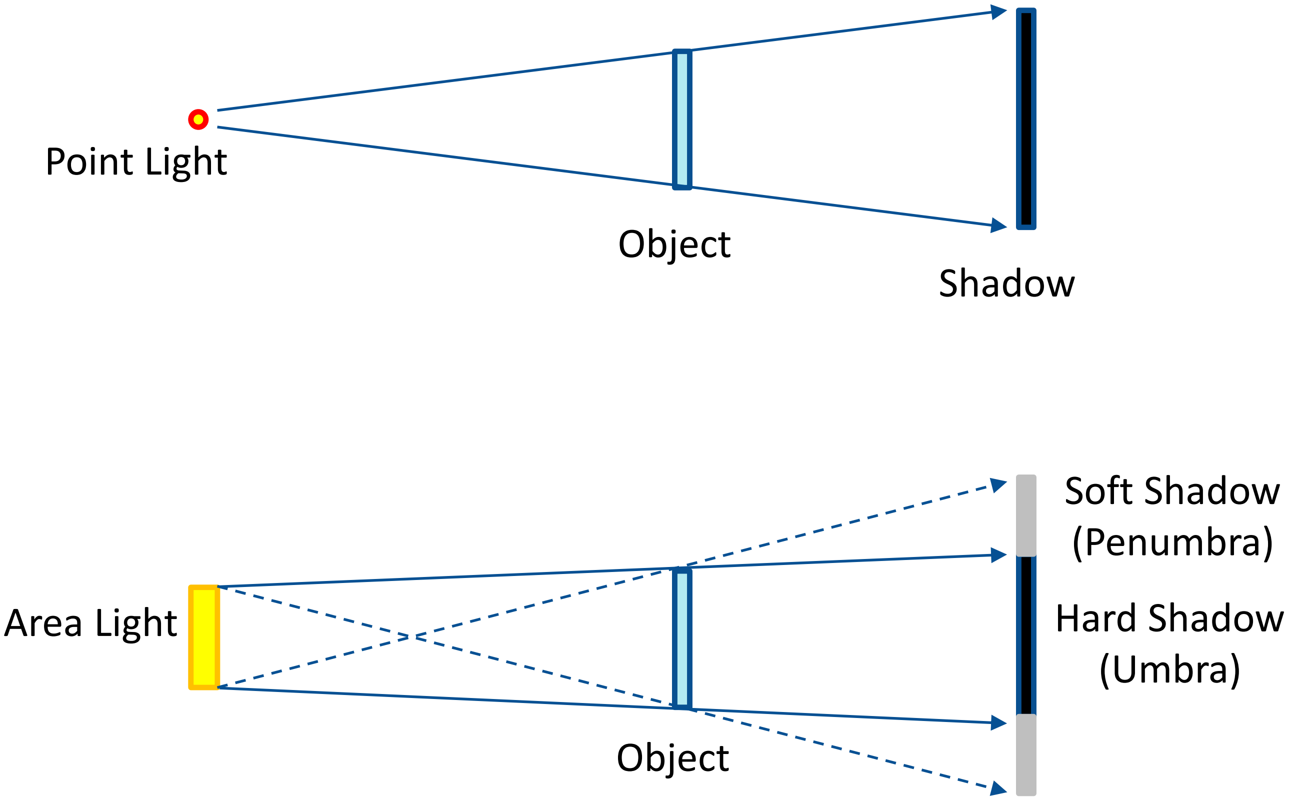


```
glm::vec2 rvec = glm::diskRand(radius);   //2D

glm::vec3 rvec = glm::ballRand(radius);   //3D
```

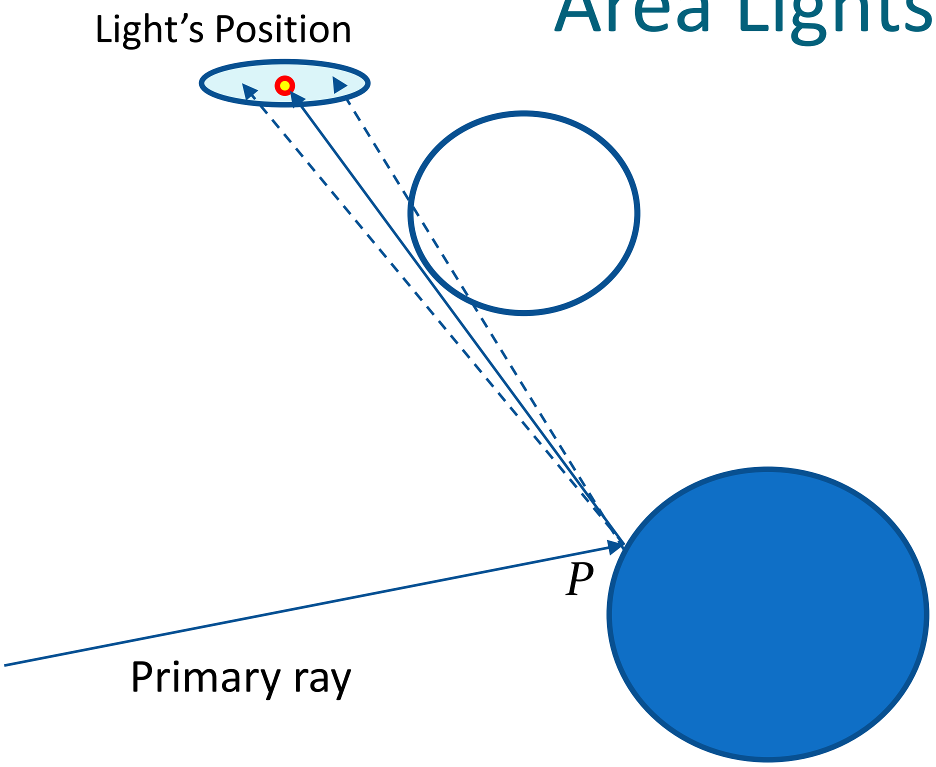


```
glm::vec2 rvec = glm::circularRand(radius);   //2D

glm::vec3 rvec = glm::sphericalRand(radius);   //3D
```

Area Lights and Soft Shadows

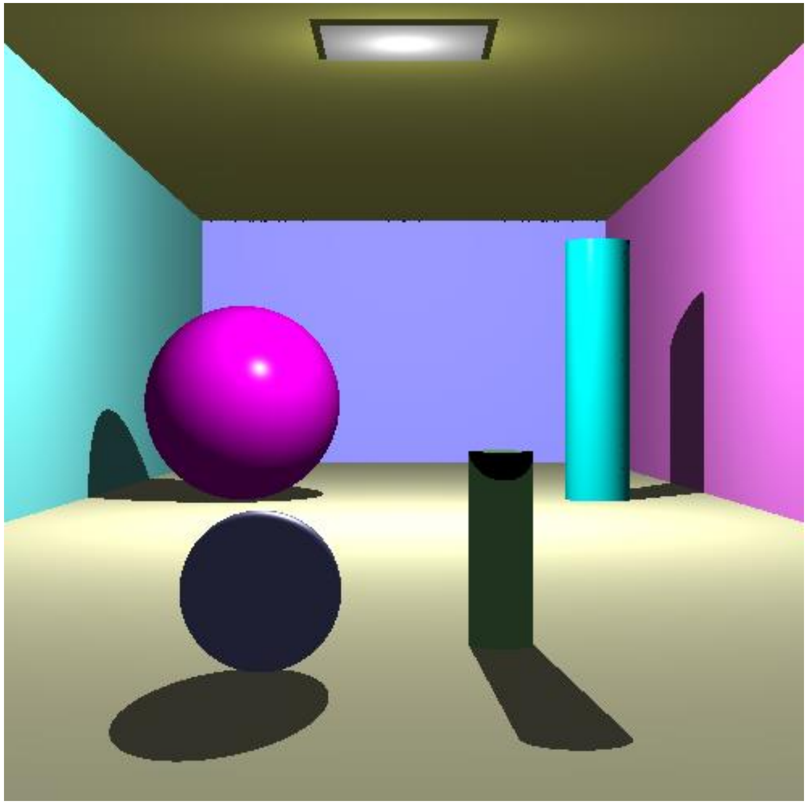


Area Lights

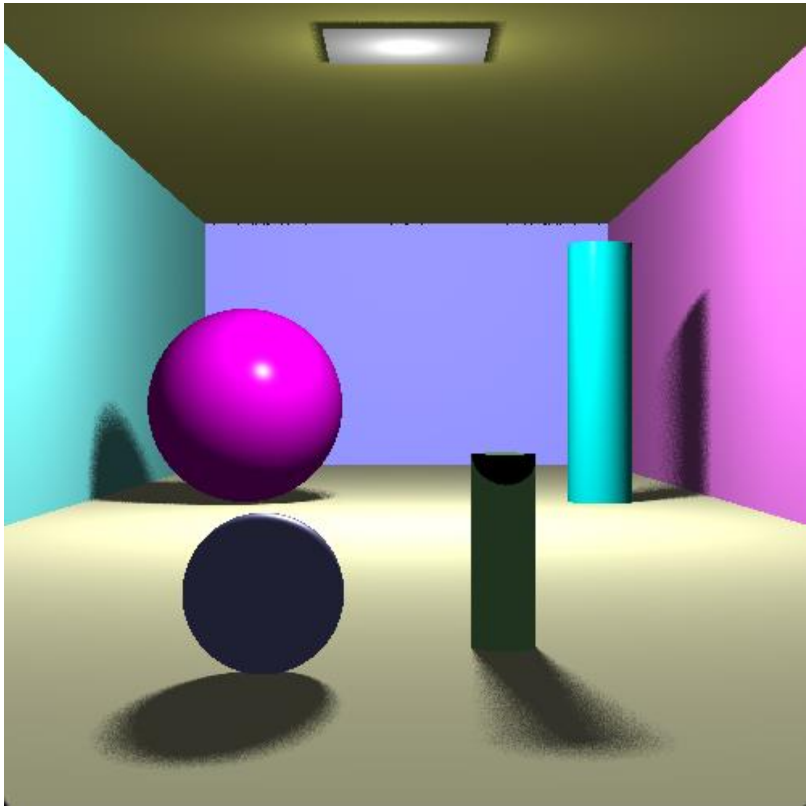


- Define a small circular region around a light source.
- Sample this area using a set of randomly distributed points, and generate multiple shadow rays to this region.
- Take the average of colour values obtained from each shadow ray.

Soft Shadows



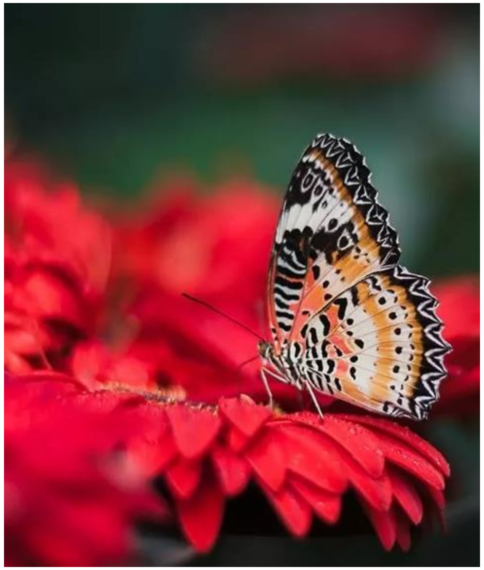
Shadows Using Point Light Source

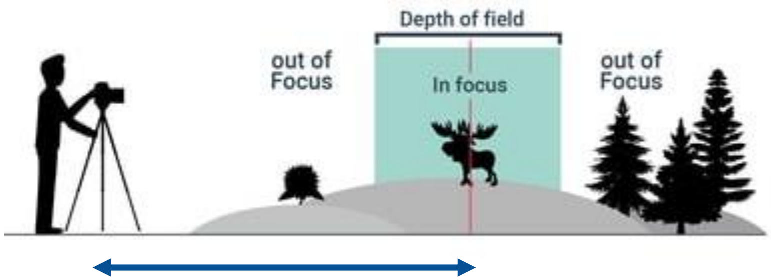


Shadows Using Area Light Source

Depth of Field

- DoF in Photography





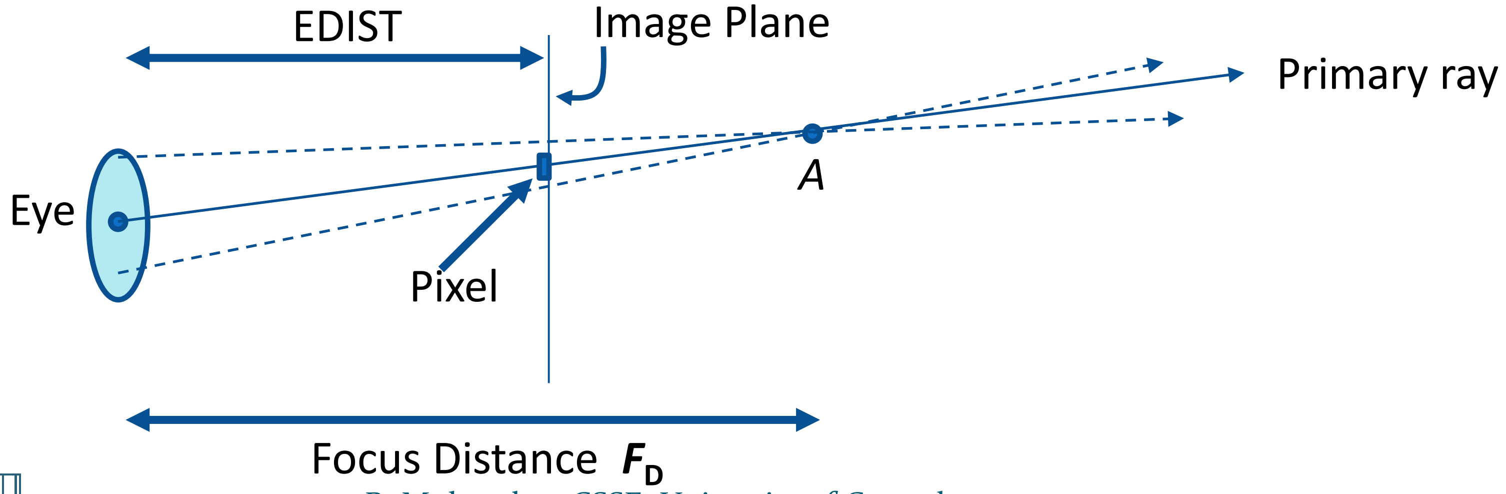
Focus Distance

F_D

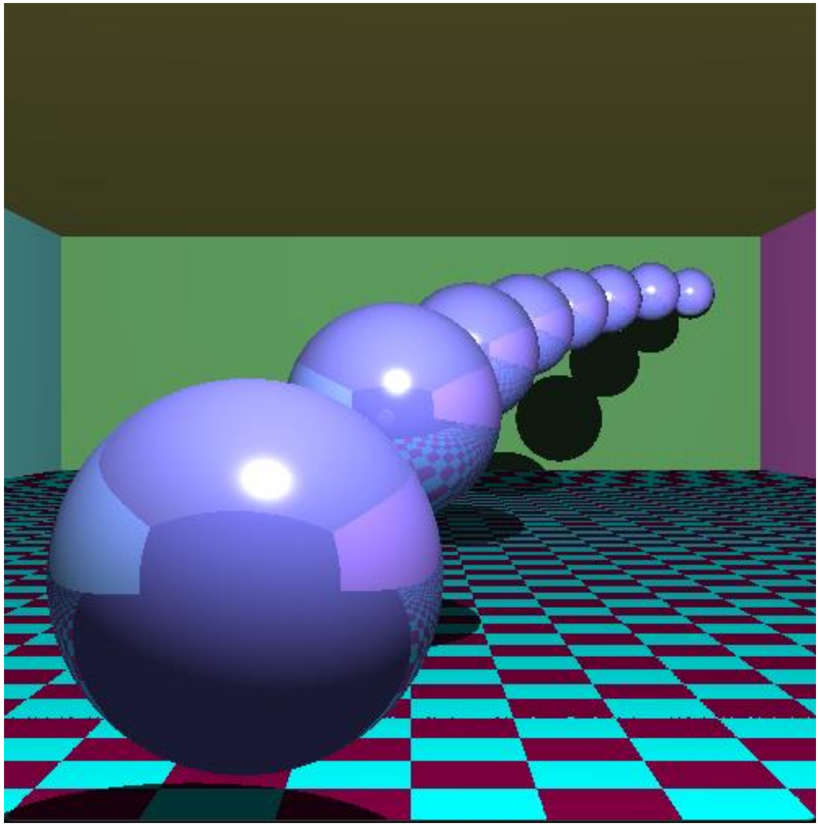
https://capturetheatlas.com/depth-of-field-photography/

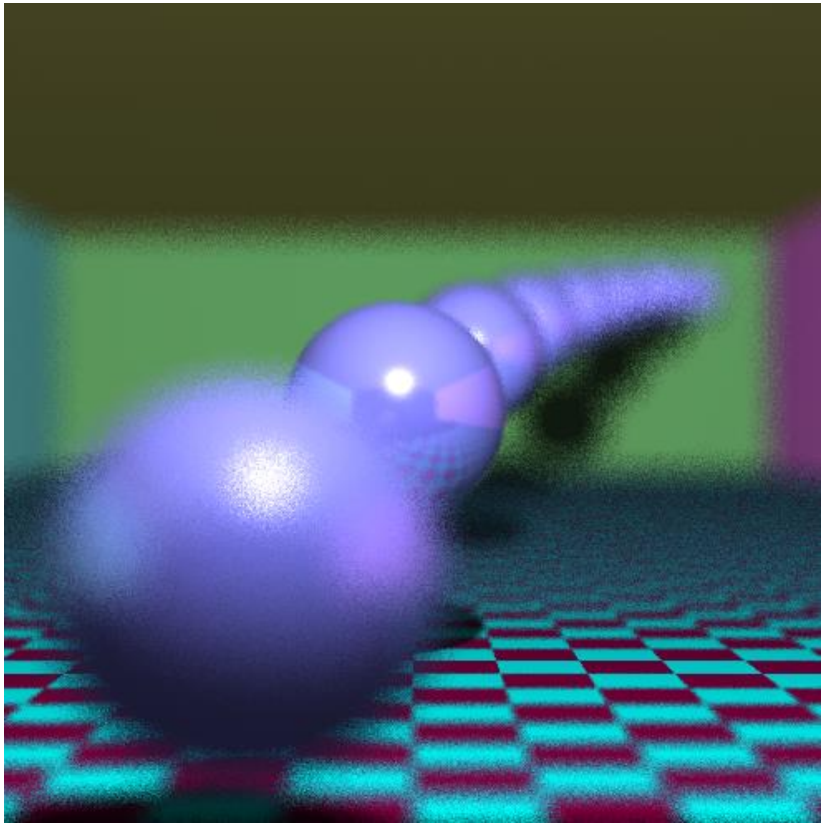
Depth of Field

- DoF in ray tracing is modelled using a “lens” of finite area as the origin of primary rays.
- Instead of using a pin-hole camera model where only a single primary ray is traced through a pixel, we trace multiple rays (randomly generated) from a small area surrounding the eye position, with all rays converging at a point *A* on the primary ray at distance F_D from the eye position. The pixel is assigned the average colour value.



Depth of Field





Focus Distance = 200

Assignment Submission

- Provide build details/command in the report
- Please submit report in PDF format only
- Please package all files in a zip folder.