UNIVERSITY OF DERBY

5CM507 Graphics

Lecture 09 Shadow Maps

Dr Youbing Zhao

November 24, 2025

Last Week

- Normal mapping
- Cube Maps
- PBR Maps
- ...

Pre-session Readings

Shadows in painting

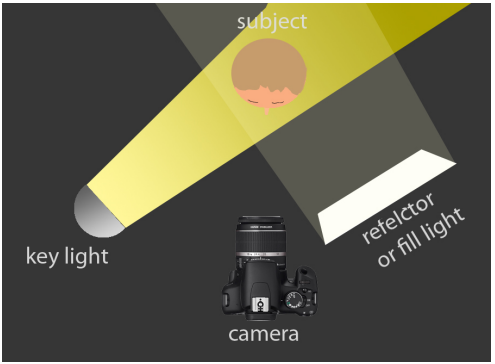
Light is connected with shadows
Shadows can

- Add depth and perspective
- Enhance realism
- Create contrast (chiaroscuro) in artworks and photography

Supper at Emmaus, Caravaggio, 1601, National Gallery, London

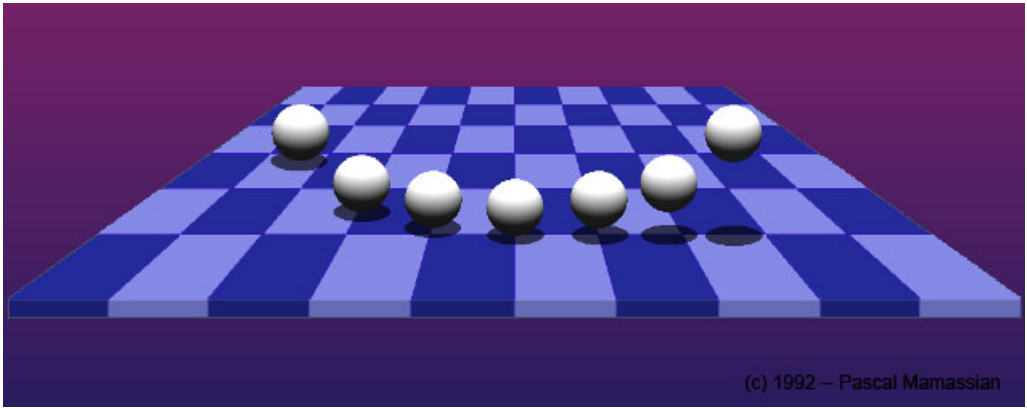
Rembrandt Lighting

The Rembrandt patch in Portrait of Johannes Wtenbogaert, 1633



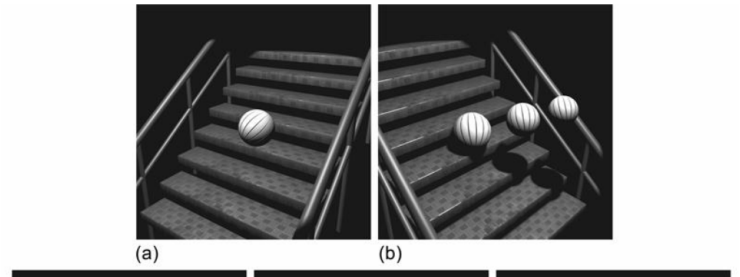
Rembrandt light setup

Shadows Provide Position and Depth Cues



Shadows provides depth cues

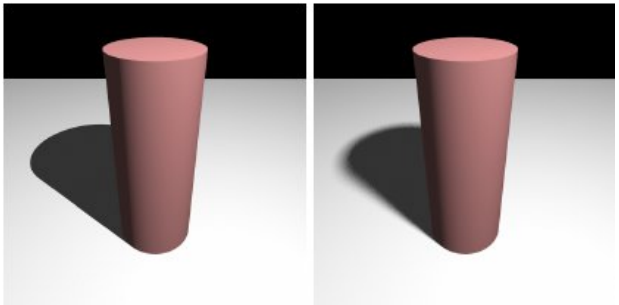
Shadows Provide Position and Depth Cues

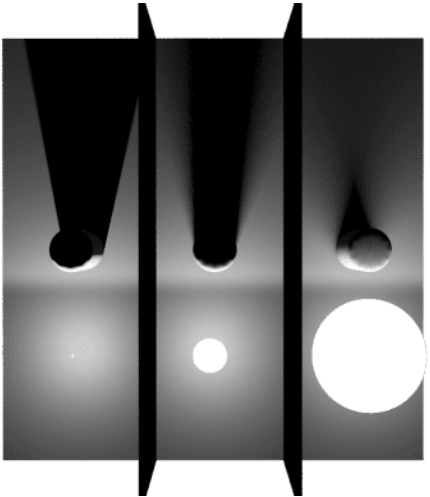


Shadows provides depth cues

Shadow Types

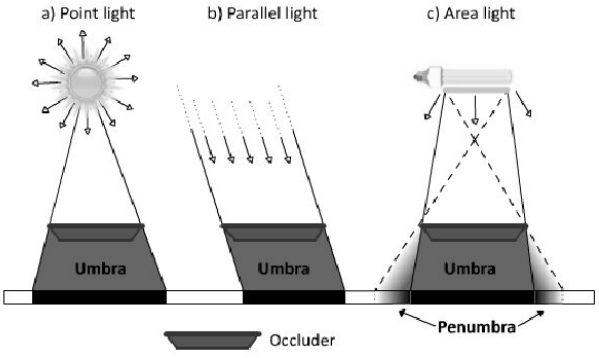
- Hard light example: direct sun light
- Soft light example: lamp with shade



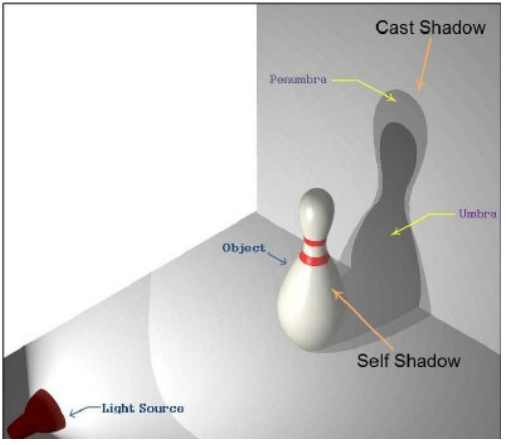


Hard and soft shadows

Shadow Types



Umbra and Penumbra

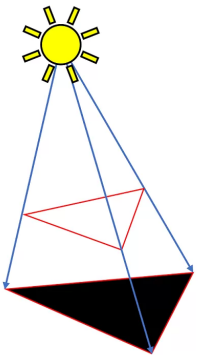


Shadow types

Projective Fake Shadows in Early Games

- Use a fake shadow polygon
- Shadows are totally opaque
- Project the polygon from the light source to the ground.

Projective shadows in Tomb Raider 1996



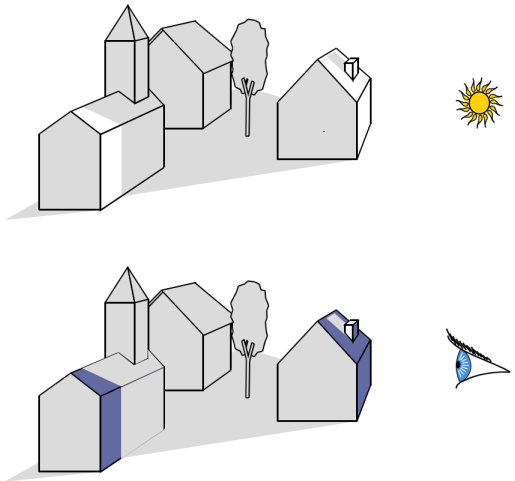
Lecture

Contents

Shadow Projection vs Perspective Projection

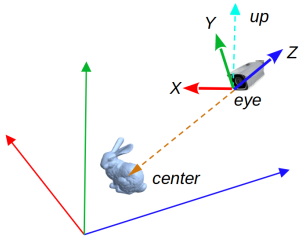
Analogy between lighting and viewing:

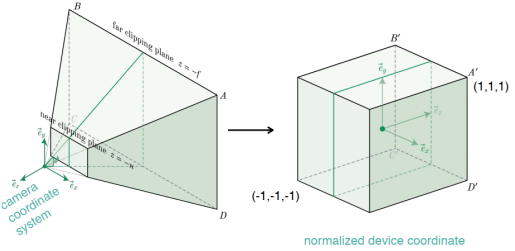
- A point light source : a camera
- Both cast “rays”
- Shadows: not “visible” from the light source
- Shadow computation: can make use of depth in perspective projection



Projection from a point light source vs perspective projection

Perspective Projection Revisited





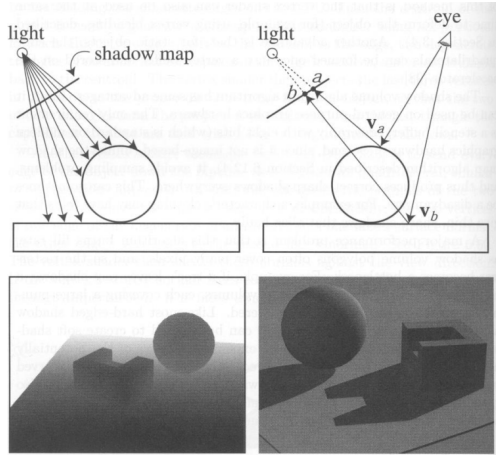
$$\mathbf{V} = \begin{bmatrix} X_c^x & X_c^y & X_c^z & -\vec{e}\cdot\vec{X_c} \\ Y_c^x & Y_c^y & Y_c^z & -\vec{e}\cdot\vec{Y_c} \\ Z_c^x & Z_c^y & Z_c^z & -\vec{e}\cdot\vec{Z_c} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{P} = \begin{bmatrix} \frac{2n}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2n}{t-b} & 0 & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix} \quad v_{clip} = \mathbf{PVM}v = \begin{bmatrix} x_{clip} \\ y_{clip} \\ z_{clip} \\ -z \end{bmatrix}$$

$$\mathrm{v}_{norm} = \begin{bmatrix} -\frac{x_{clip}}{z} & -\frac{y_{clip}}{z} & -\frac{z_{clip}}{z} \end{bmatrix}^T \quad v' = \frac{v_{norm}+1}{2}$$

$$\mathrm{z}_{norm} \in \left[\overbrace{-1}^{\text{near}}, \overbrace{1}^{\text{far}} \right] \quad depth = z' = \frac{z_{norm}+1}{2} \in [0,1]$$

Shadow Maps

- Limitations
  - Single light source
  - Hard shadows
- Shadow Map: a depth map texture $d(x', y')$
  - closest scene depth from light source
  - $(x', y') = (\frac{x_{norm}+1}{2}, \frac{y_{norm}+1}{2}) \in [0, 1]$
  - $d(x', y') = z' \in [\overbrace{0}^{\text{near}}, \overbrace{1}^{\text{far}}]$
- Shadow Check: “visible” to the light source ?
  - In shadow: $z > d_{map}$, behind lighted surfaces
  - Not in shadow: $z \leq d_{map}$

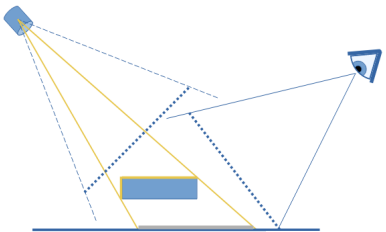


[Foley et al. “Computer Graphics Principles and Practice”]

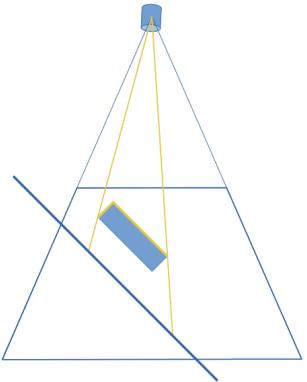
Shadow Map Generation: Pass I - Light-space Depth

Take the light source as the camera, perform offscreen rendering;

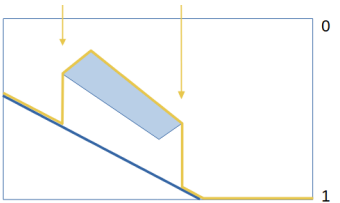
Save the depth buffer as the shadow map texure.



Top view of the scene



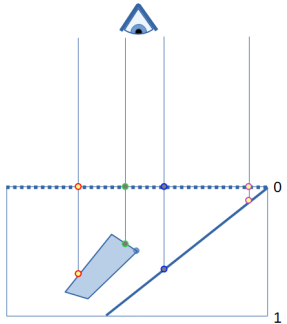
Projection from the light source



Depth map

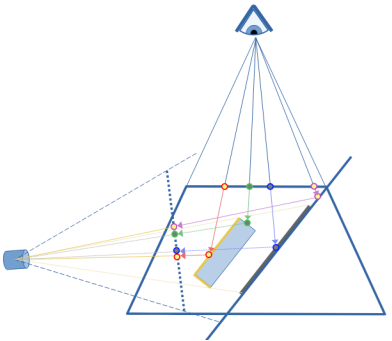
Shadow Generation: Pass II - Reproject to Light-space

- Project the vertex in both the viewer space and the light space in the vertex shader
- Compare the light-space fragment depth with the shadow map in the fragment shader

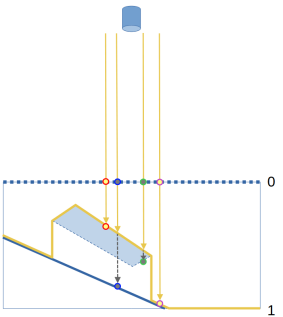


Render the scene

in viewer space



Reproject the 3D fragment to the light-space



Compare the fragment depth with the shadow map

Shadow Generation: Pass II - Depth Comparison

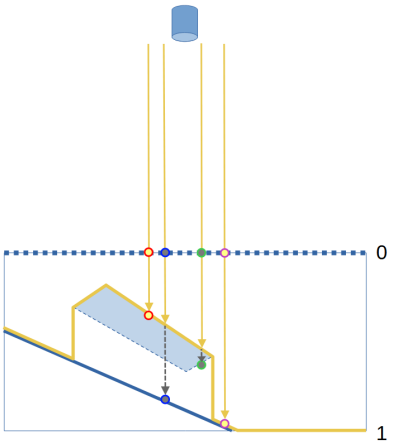
Light-space fragment coordinate after projection:

- Normalised : $(x_{norm}, y_{norm}, z_{norm}) \in [-1, 1]$
- Rescaled: $(x', y', z') \in [0, 1]$

Larger $z' \iff$ Further from the light source

Compare the depth z' with the depth stored in the shadow map at (x', y')

- $z' > ShadowMap(x', y')$
  - Further from lighted surfaces $\Rightarrow$ In shadow
- $z' \leq ShadowMap(x', y')$
  - On the lighted surface $\Rightarrow$ Not in shadow



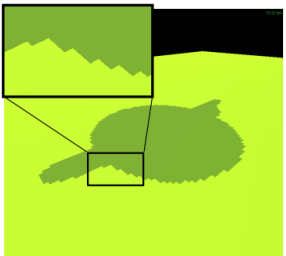
Shadow Map Artifacts

- Aliasing (Jagged shadow edges)
- Shadow map precision problem (Shadow Acne)
- Shadow detachment (Peter Panning)

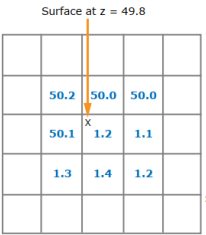
Shadow Aliasing and Solutions

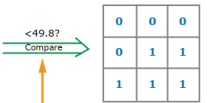
1. Increasing shadow map resolution;

2. Percentage-Closer Filtering (PCF): Filter the result of the shadow test (using weighted average of comparison results for shadow blending, can also be used for soft shadows)



Jagged shadow edges





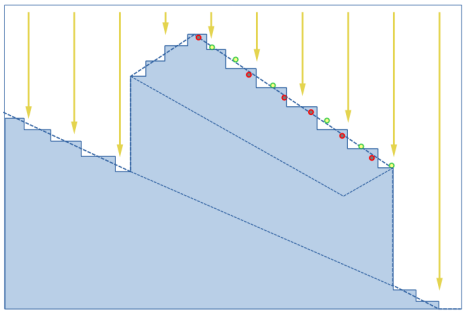
Filtering

Smoothed Shadow Boundary

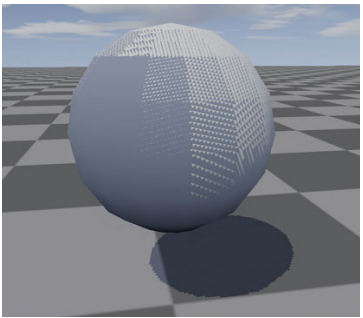
Shadow Map Precision Artefacts (Shadow Acne)

Visible points taken as shadows due to z-fighting, showing speckles and streaks.

Solutions: 1. Depth offsetting; 2. Front face culling.



Shadow map depth precision problem

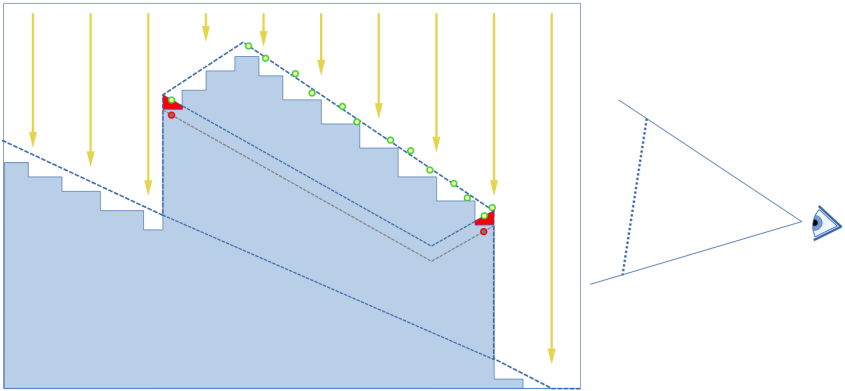


Shadow acne

Using Depth Bias

Subtract the depth of the point by a small bias.

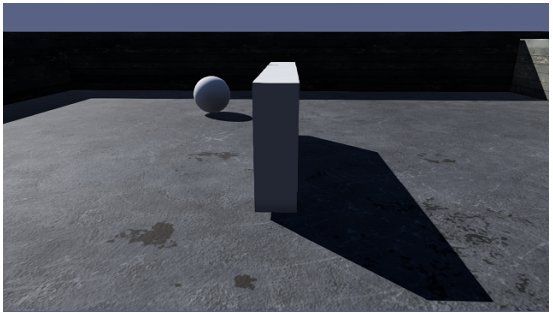
Equivalent to moving the geometry towards the light source.



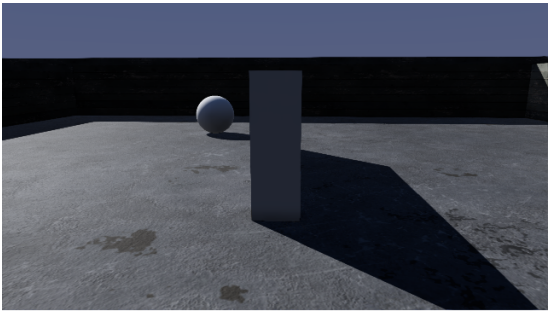
Depth bias (Incorrect shadow detachment shown in red)

Shadow Detachment (Peter Panning)

A large bias make the shadow detaches from the object.



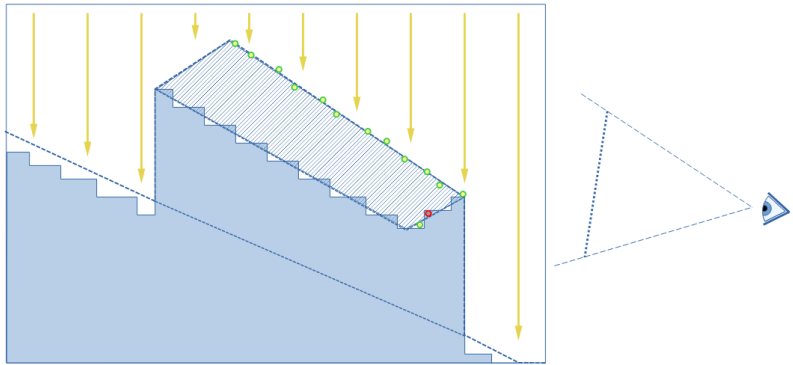
Peter Panning



Without Peter Panning

Using Front Face Culling

- Shadow acne happens on the visible/front surface of a solid object.
- Use the back faces (front face culling) of a solid object when rendering the depth map
- Can avoid using a bias.

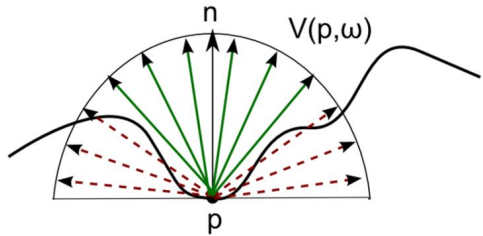


Using the depth of back faces in a shadow map

Ambient Occlusion

Ambient Occlusion

- Average amount of self-occlusion
- Darkening areas light hardly reaches
- Techniques
  - Ray tracing
  - Screen space ambient occlusion (SSAO)
  - Ambient occlusion map (static)



Ambient Occlusion

Interactive comparisons: Tomb Raider

Summary

- Shadow Maps
  - Two pass rendering
  - Pass 1: light source as camera, render the depth to texture as shadow map
  - Pass 2: project to both the viewer space and light space, compare light space depth to shadow map
- Shadow Map Artifacts
  - Front face culling
- Ambient Occlusion

Questions?

Chinese shadows