Computer Graphics

Shadow

Teacher: A. Prof Chengying Gao(高成英)

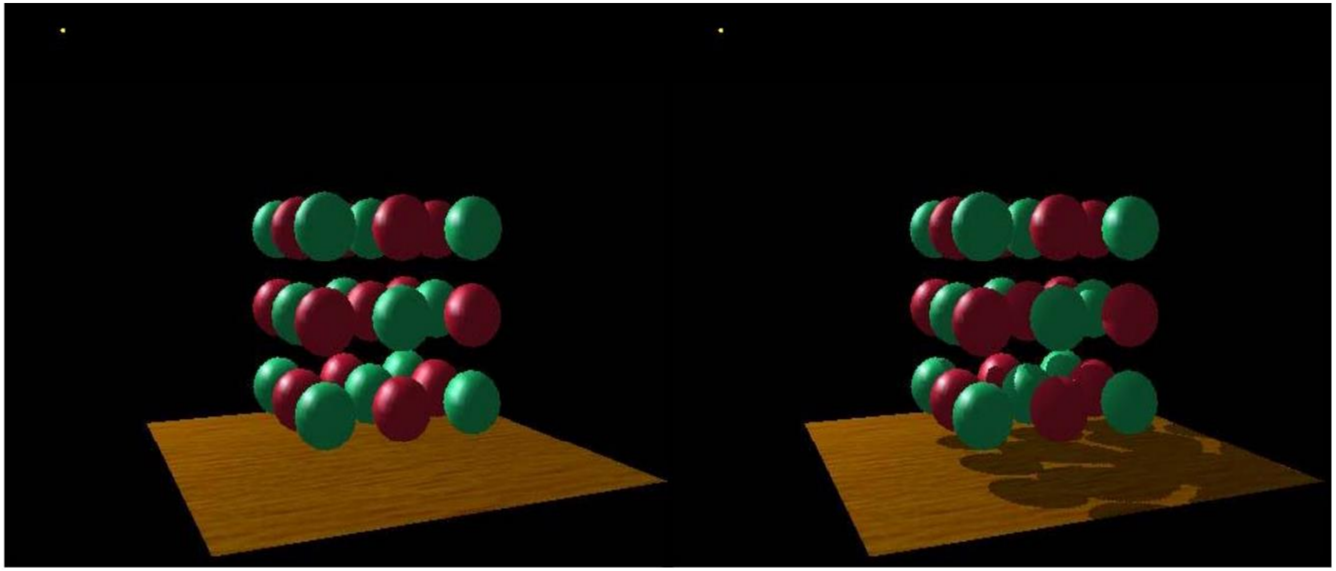
E-mail: mcsgcy@mail.sysu.edu.cn

School of Data and Computer Science

Most of slides are from Graphics & Geometry Computing Group of Tsinghua University

Shadow

- Shadows are important elements in creating realistic images and in providing the user with visual cues(暗示)about object placement.



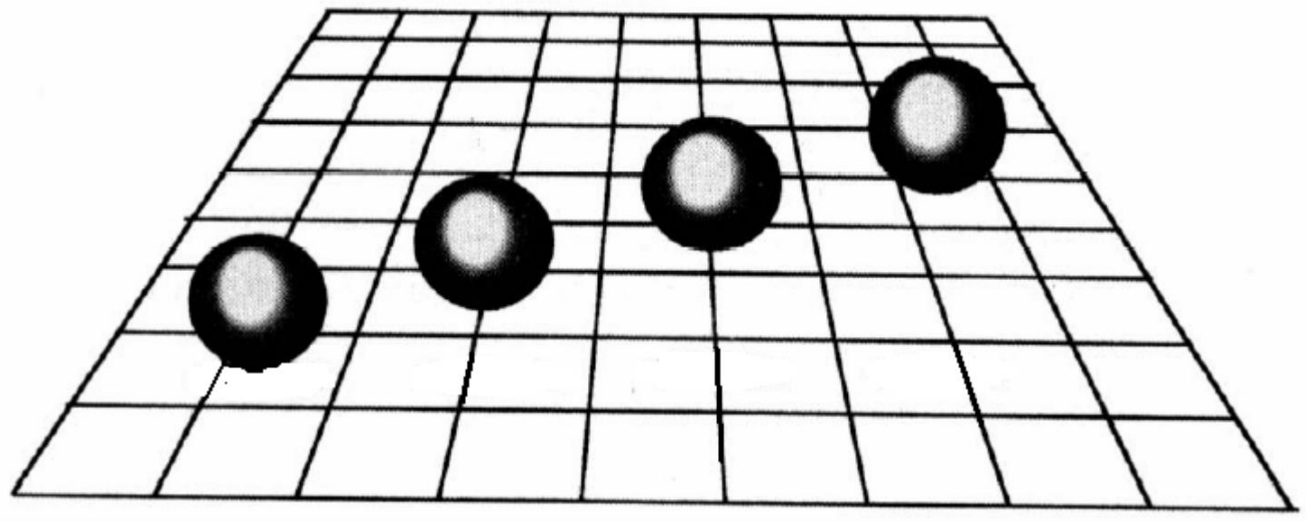
Shadow

- Shadow generation is a fundamental problem in Computer Graphics.

- Why shadow is so important?
  - Lets see an example.

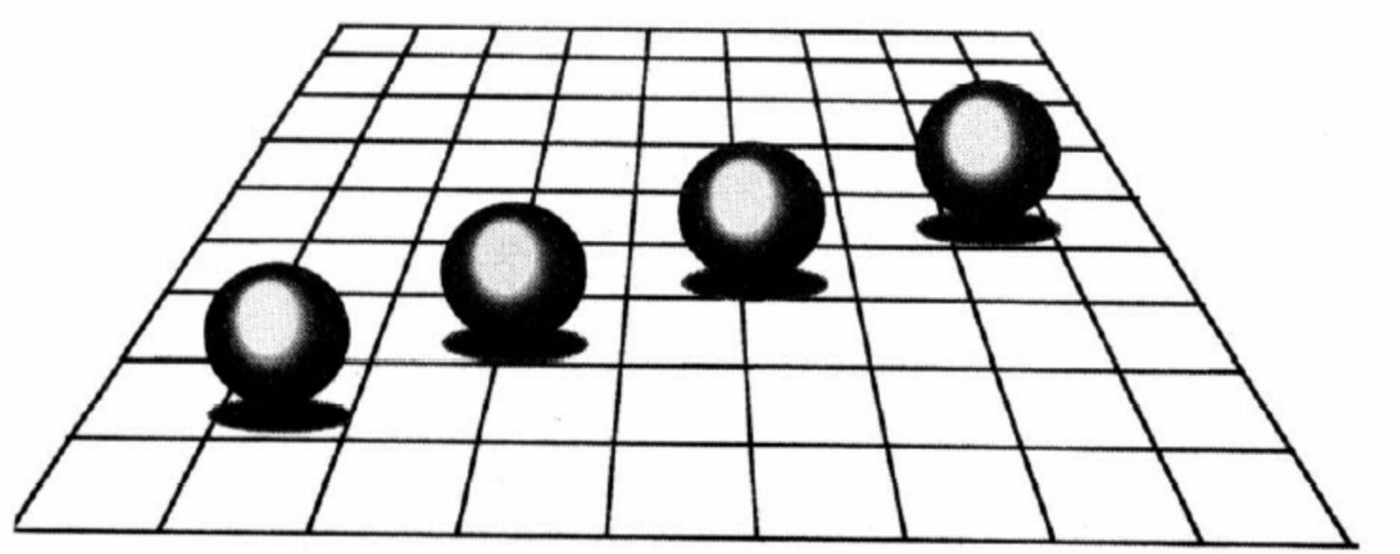
Shadow

- **Why is shadow important?**
  - **Do you know where are the balls located? (Without shadows)**
  - **It is hard to determine the actual position**



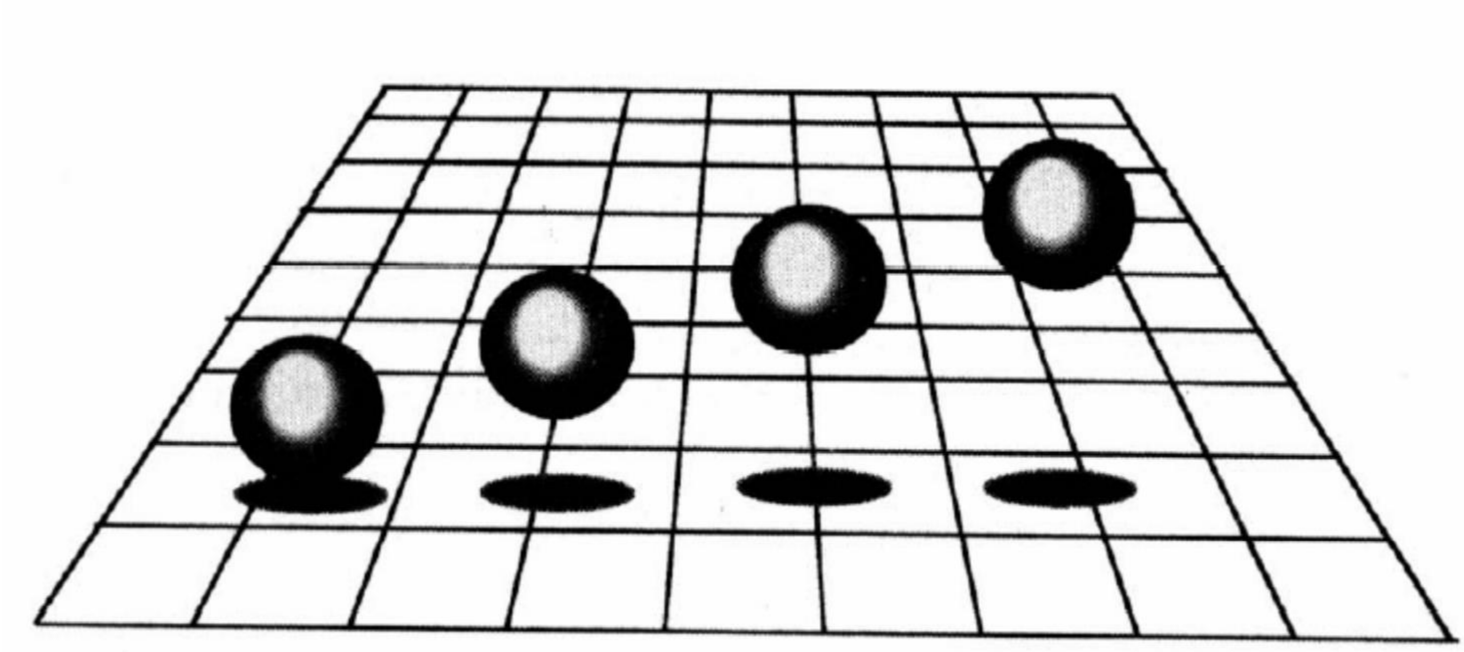
Shadow

- **Why is shadow important?**
  - **We know where the balls located are (With shadows)**

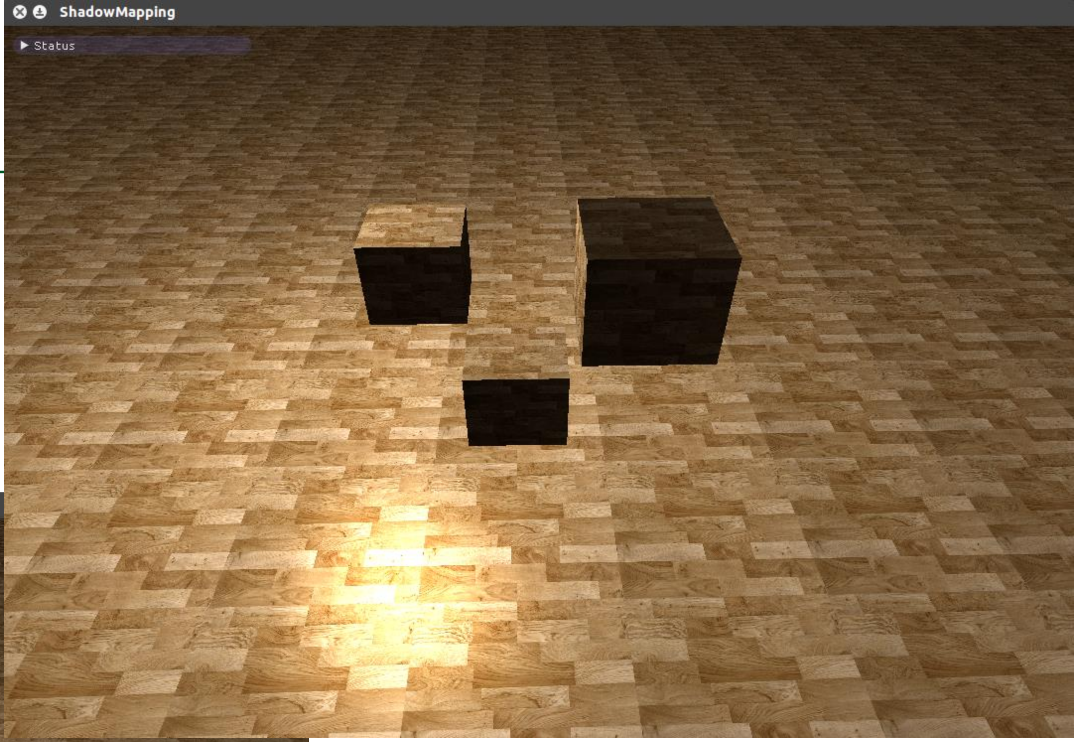


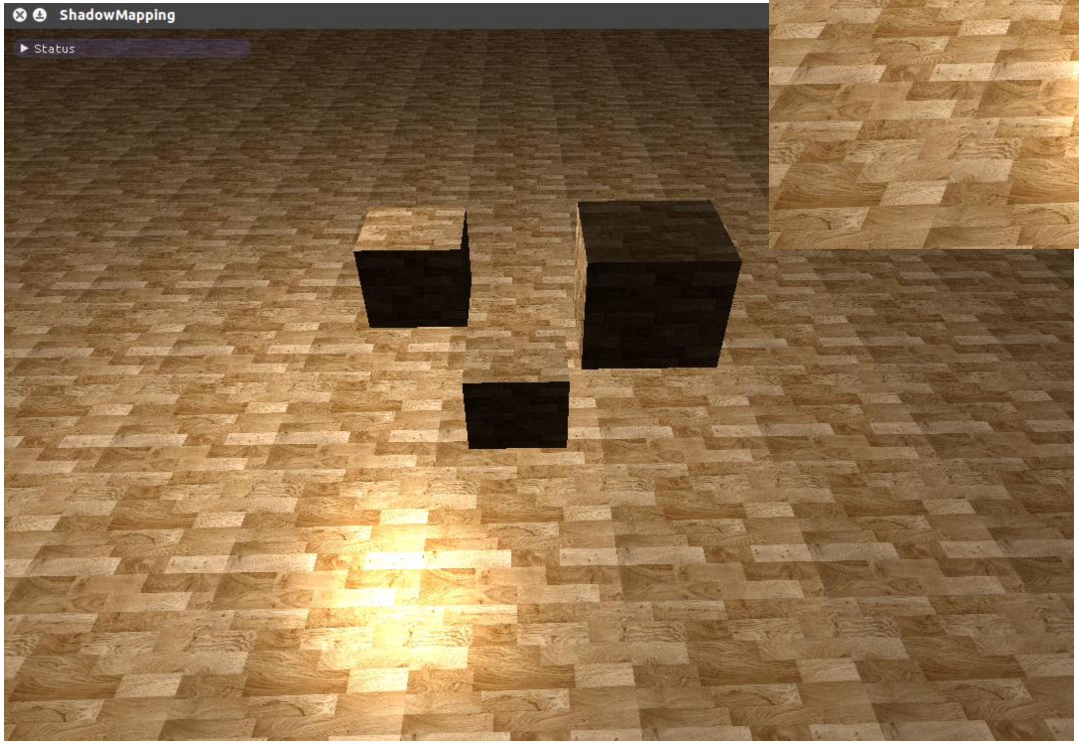
Shadow

- **Why is shadow important?**
  - **Different shadows implies different balls location**



Shadow

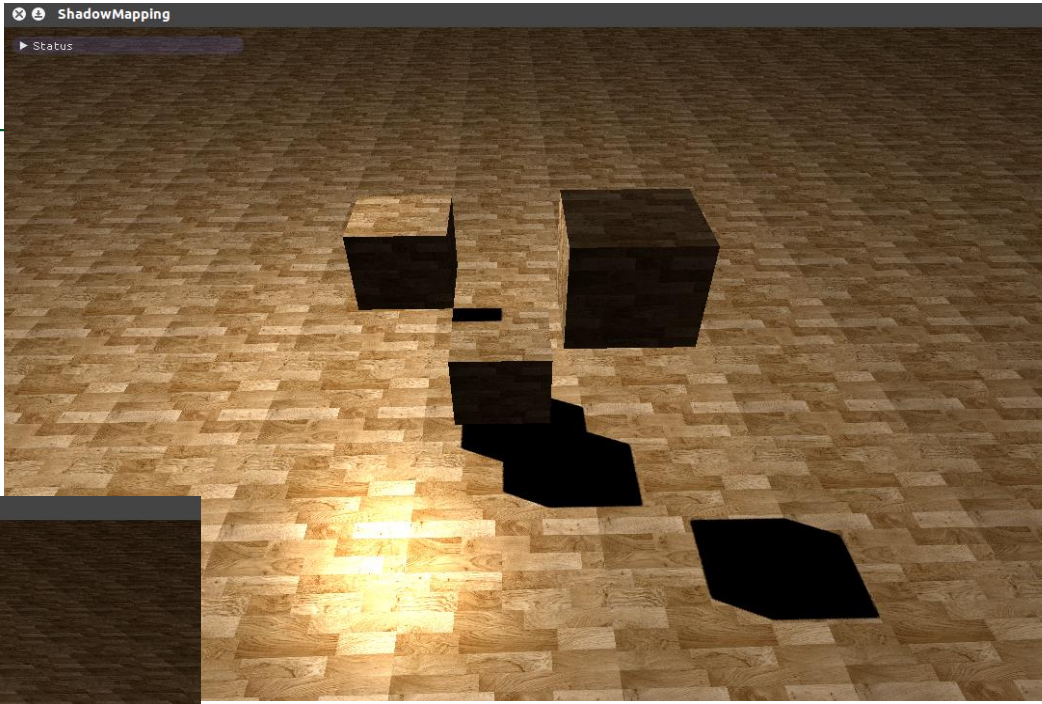




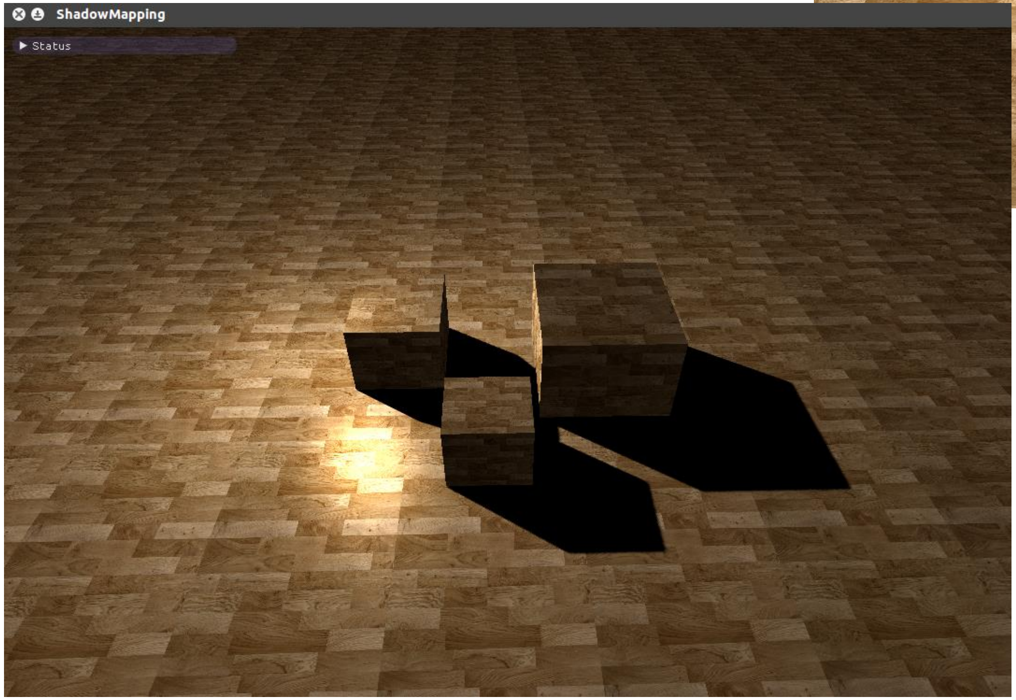
Scene A (without shadow)

Scene B (without shadow)

Shadow



Scene A (with shadow)



Scene B (with shadow)

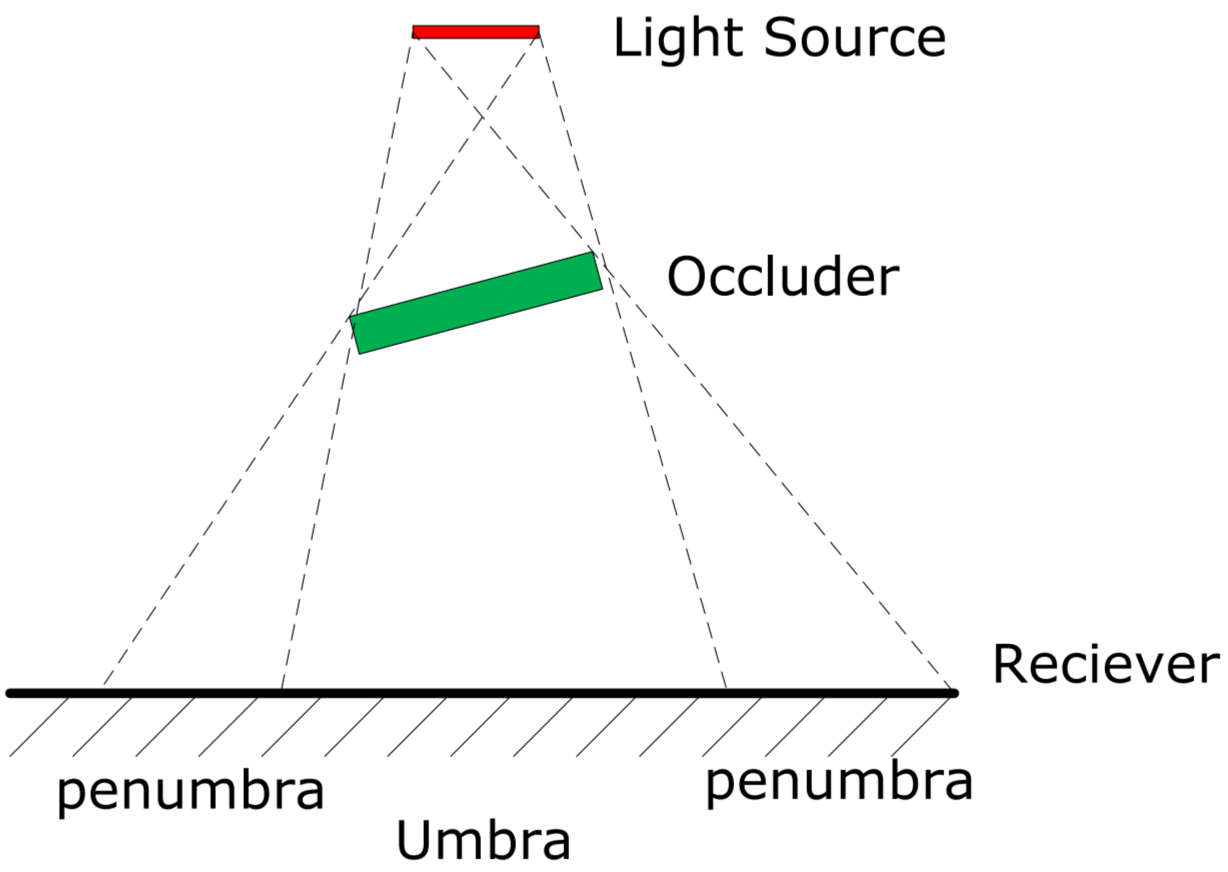
Why is shadow important？

- From this example, we could conclude that:
  - Shadows give an important visual cue of object position.
  - Same images with different shadows implies different object positions.
- So, shadow is important.

What is shadow?

- Definition and Terminations
  - Consider a light source L illuminating a scene:
  - Receivers are objects of the scene that are potentially illuminated by L.
  - A point P of the scene is considered to be in the umbra(本影） if it can not see any part of light source L.
  - If P can see a part of the light source , it is in the penumbra(半影）.
  - Shadow is the union of the umbra and penumbra, is the region of space for which at least one point of the light source is occluded.
  - Objects that hide a point from the light source are called occluders(遮挡物).

Shadow



Types of Shadows

- Attached shadows: occurring when the normal of the receiver is facing away from the light source;
- Cast shadows: occurring when a shadow falls on an object whose normal is facing toward the light source.
- Self-shadows: are a specific case of cast shadows that occur when the shadow of an object is projected onto itself, i.e. the occlude and the receiver are the same.

Importance of Shadow

- Shadows help to understand relative object position and size in a scene.
- Shadows can also help us understanding the geometry of a complex receiver.
- Shadows provide useful visual cues that help in understanding the geometry of a complex occluder.

- **Importance of Shadow**

Shadow helps to determine the geometry of receiver

- **Importance of Shadow**

Shadow helps to determine the geometry of occluder

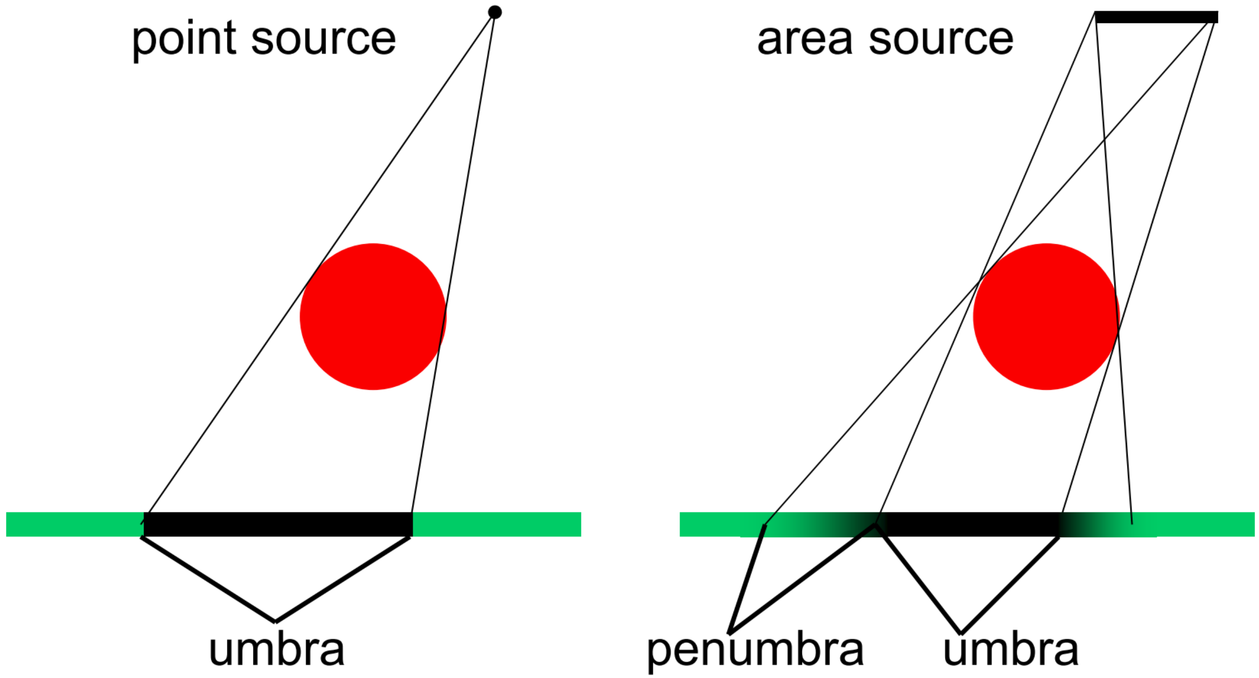
Hard Shadow vs Soft Shadow

- The common-sense notion of shadow is a binary status, i.e. a point is either "in shadow" or not. This corresponds to hard shadows, as produced by point light sources.
- However, point light sources do not exist in practice and hard shadows give a rather unrealistic feeling to images. Note that even the sun, the most common light source in our daily life, has a significant angular extent and does not create hard shadows.

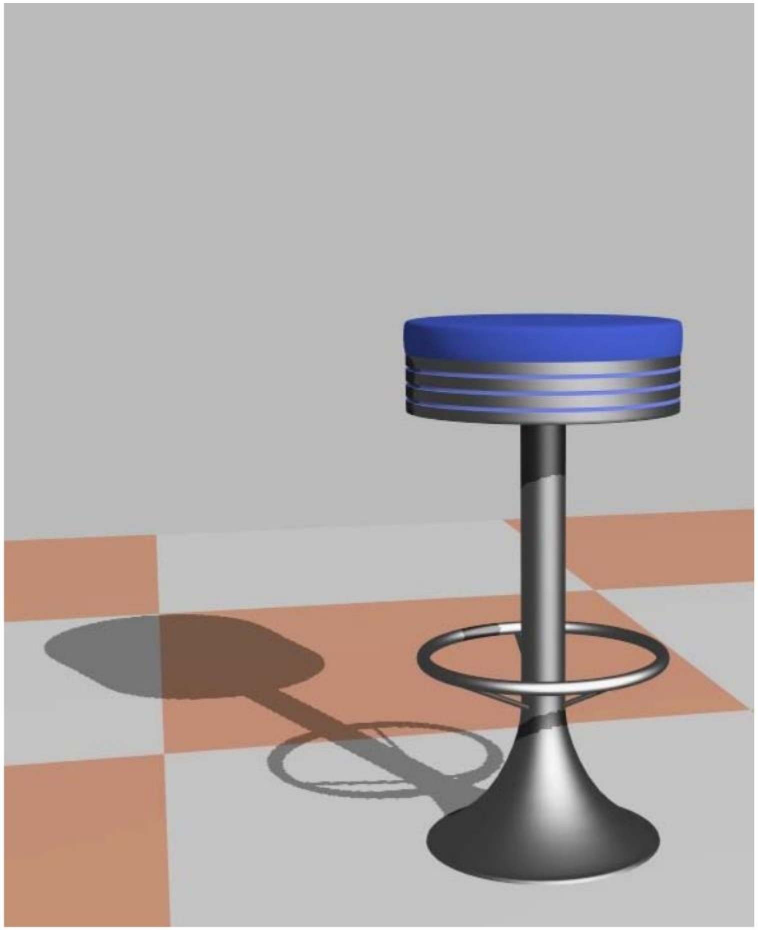
Hard Shadow vs Soft Shadow

- Still, point light sources are easy to model in computer graphics and we shall see that several algorithms let us compute hard shadows in real time.
- However, for a light source with finite extent (actually an area source), the determination of the umbra and penumbra is a difficult task in general, as it amounts to solving visibility relationships in 3D, a notoriously hard problem.

Hard Shadow vs Soft Shadow



Hard Shadow vs Soft Shadow

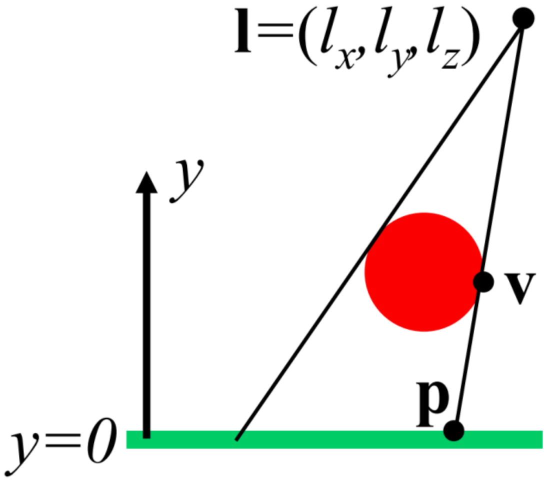


Planar shadow

- What is planar shadow?
  - A simple case of shadow when objects cast shadows on planar surface.

Planar shadow

- Projection Shadow
  - In this case, the three-dimensional objects is rendered second times. A matrix could be derived that projects the vertices of an object onto a plane. Consider the situation in the figure, where the light source is located at l, the vertex to be projected is at v, and the projected vertex is at p.
  - Further suppose the receiver plane is y=0 (this could also be generalized to work with any planes)

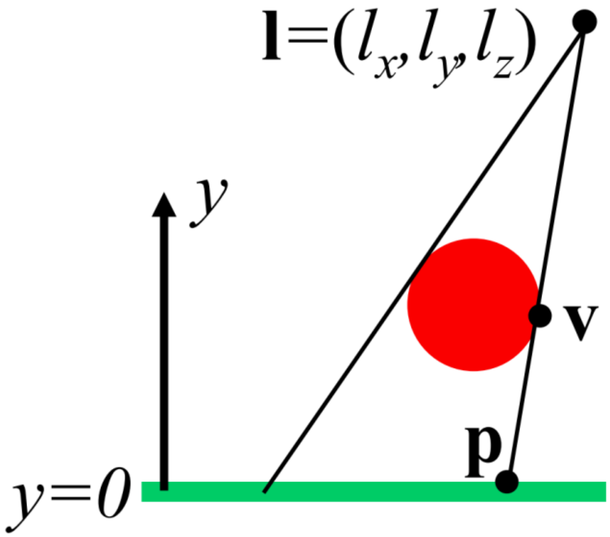


Planar shadow

- Projection Shadow
  - By deriving the projection of x-coordinated. From similar triangles, the following equation could be obtained

$$\frac{p_x - l_x}{v_x - l_x} = \frac{l_y}{l_y - v_y}$$

$$\Rightarrow p_x = \frac{l_y v_x - l_x v_y}{l_y - v_y}$$



Planar shadow

- Projection Shadow
  - The z-coordinate could be obtained in the same way.

$$p_z = \frac{l_y v_z - l_z v_y}{l_y - v_y}$$

  - These equations can be converted into a projection matrix M.

Planar shadow

- Projection Shadow
  - Projection Matrix

$$\mathbf{M} = \begin{pmatrix} l_y & -l_x & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -l_z & l_y & 0 \\ 0 & -1 & 0 & l_y \end{pmatrix}$$

  - It is easy to verity that: Mv = P.

Planar shadow

- Projection Shadow
  - In the general case, the plane onto which the shadows should be cast is not the plane y = 0. But instead a plane n*x +d = 0.

  - Similar to the y = 0 plane case, the projected point p could be described as:

$$p = l - \frac{\mathrm{d} + n \cdot l}{n \cdot (v - l)}(v - l)$$

Planar shadow

- Projection Shadow
  - The equation can also be converted into a projection matrix, which satisfy Mv = p.

$$\mathbf{M} = \begin{pmatrix} \mathbf{n}\cdot\mathbf{l} + d - l_x n_x & -l_x n_y & -l_x n_z & -l_x d \\ -l_y n_x & \mathbf{n}\cdot\mathbf{l} + d - l_y n_y & -l_y n_z & -l_y d \\ -l_z n_x & -l_z n_y & \mathbf{n}\cdot\mathbf{l} + d - l_z n_z & -l_z d \\ -n_x & -n_y & -n_z & \mathbf{n}\cdot\mathbf{l} \end{pmatrix}$$

  - As expected, this matrix turns into the matrix in previous page if the plane is y = 0(n = (0,1,0) and d =0)

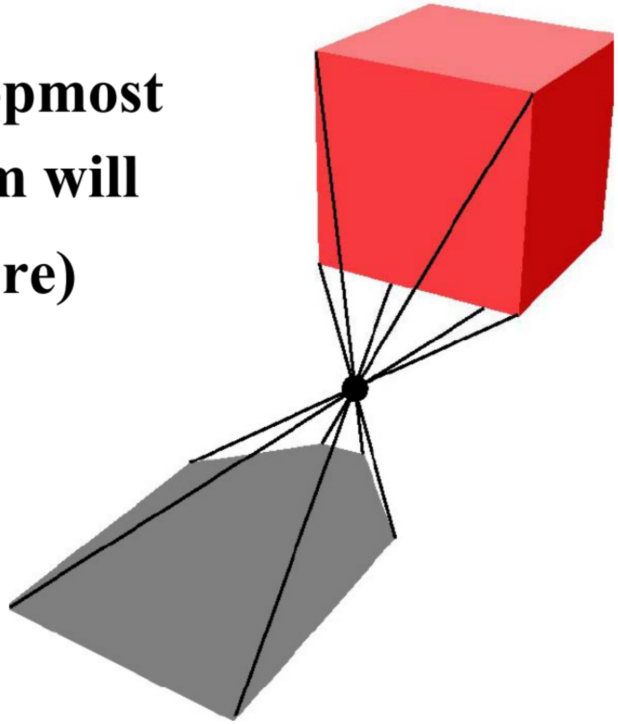
Planar shadow

- Projection Shadow
  - To render the shadow, simply apply this matrix to the objects that should cast shadows on the plane. And render this projected object with a dark color and no illumination（只绘制环境光，不绘制漫反射和镜面反射）.
  - Limitation of the projection shadow method:
    - The receiver must be planar
    - The shadow has to be rendered for each frame, even though the shadow may not change.

Planar shadow

- **Failure cases**
  - ***false shadow***

    If the light source is below the topmost point on the object, the algorithm will produce false shadow (right figure)



Shadows on Curved Surfaces

- One way to extend the idea of planar shadows to curved surfaces is to use a generated shadow image as projective texture.
- Think of shadows from the light's point of view. Whatever the light sees is illuminated, what it does not see is in shadow.

Shadows on Curved Surfaces

- The occlude is rendered in black from the light's viewpoint into a texture with white background.
- This texture can then be projected onto the surfaces that are to receive the shadow. Each vertex of the receiver has a(u,v) texture coordinate, which could be computed explicitly by the application.
- This method is referred to as "shadow texture" technique.

Shadows on Curved Surfaces

- **Disadvantage**
  - **Object cannot cast shadow to itself**

from light　　shadow texture　　shadows on stairs

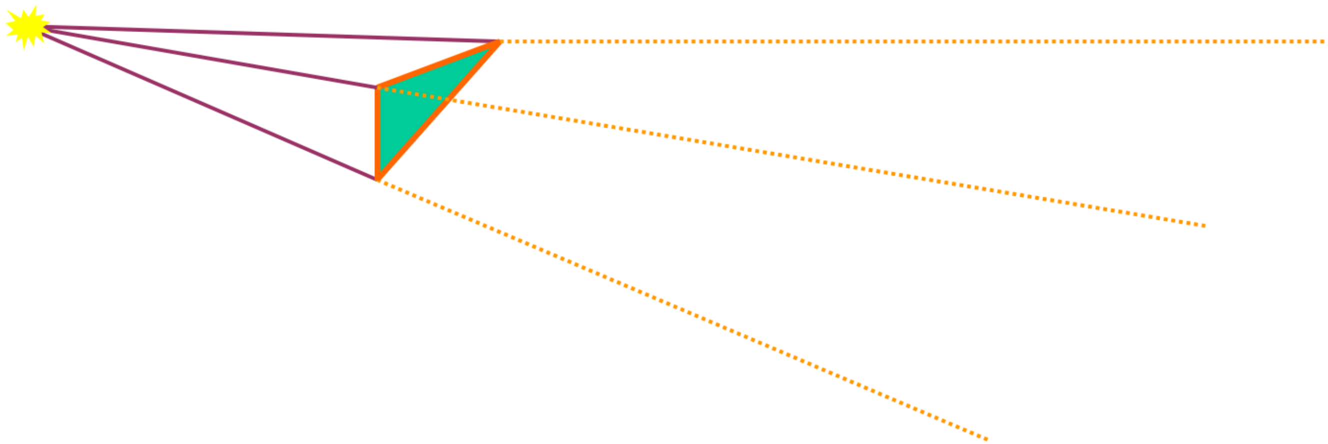
Shadow Volume

- A method proposed by Crow, which can cast shadows onto arbitrary objects.
- This technique is also sometimes called volumetric shadows.

Shadow Volume

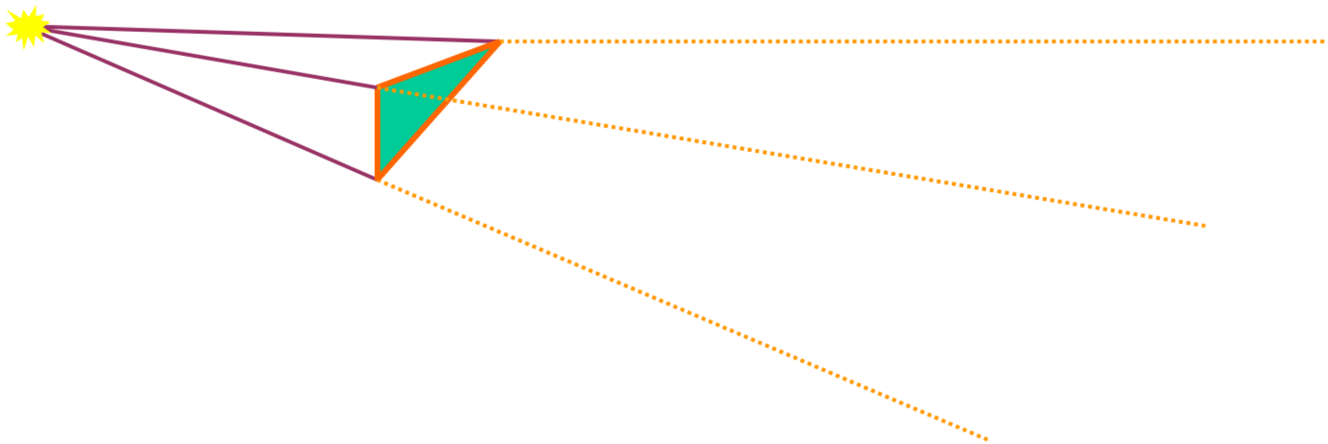
- To begin, imagine a point and a triangle.

  Extending the lines from the point through the vertices of the triangle to infinity yields an infinite pyramid.



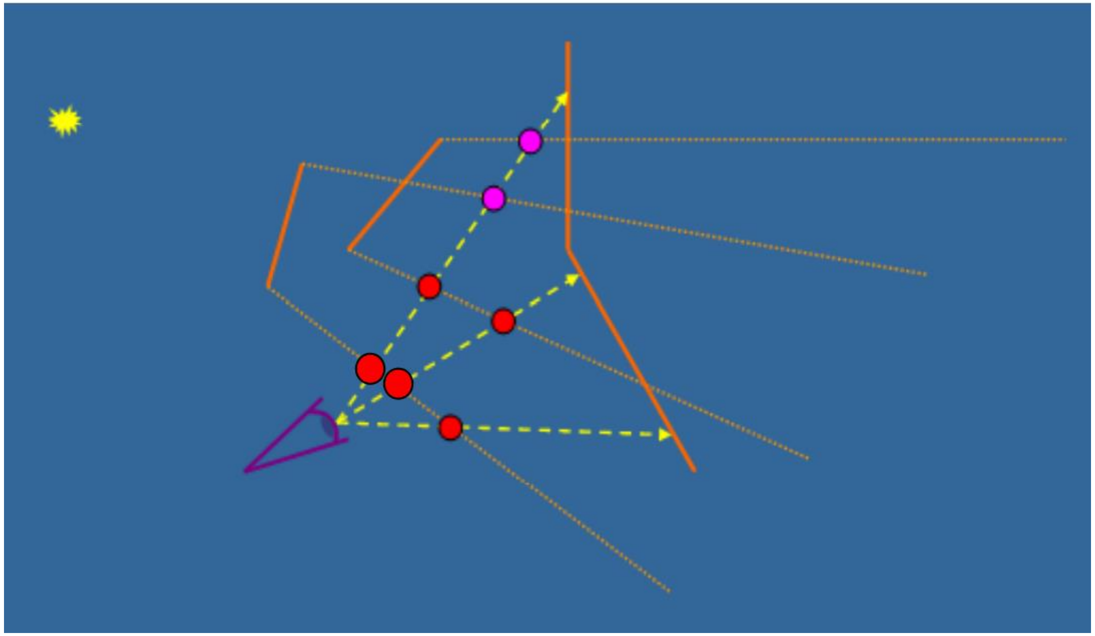
Shadow Volume

- Now, imagine the point is actually a point light source. Then any part of an object which is inside the volume of the truncated pyramid(under the triangle) is in shadow. This volume is called shadow volume.



Shadow Volume

- Suppose we cast a ray through a pixel until the ray hits an object in the scene. Now we need to determine whether or not the pixel is in shadow.
- What we need to do is to determine whether the point is in a shadow volume.



Shadow Volume

- While the ray is on its way to the object, we increment a counter each time when it crosses a face of the adow volume that is front-facing, thus the counter is incremented each time the ray goes into shadow.
- In the same manner, we decrement the same counter each time the ray crosses a back-facing face.
- Thus, finally, if the counter is greater than zero, then that pixel is in shadow; otherwise it is not.

Shadow Volume

- Using Stencil Buffer
  - Certainly, doing this geometrically is tedious and time-consuming. But there is a much smarter solution: using hardware's stencil buffer do the counting
  - A stencil buffer is a buffer which stores integer numbers in each pixel while Z-buffer stores depth value in real number.

Shadow Volume

- Using Stencil Buffer
  - **First,** clear the stencil buffer
  - **Second,** the whole scene is drawn into the frame buffer with only ambient component, in order to get these lighting components in the color buffer and the depth information into z-buffer.
  - **Third,** z-buffer updates and writing to the color buffer are turned off, and then the front faces of shadow volumes are drawn.
    - In this process, a stencil operation is set to increase the values in the stencil buffer whereever a poly gon is drawn(+1 each time).
  - **Fourth,** another pass is done by drawing the back-facing polygons. For this pass, the stencil operation is set to decrements(-1 each time)
  - **Finally,** the whole scene is rendered again, with diffuse and specular components, where the value in the stencil buffer is 0.

Shadow Volume

- Advantages
  - First, it can be used on general-purpose graphics hardware. The only requirementis a stencil buffer.
  - Second, since it is not image based method(unlike the shadow map method described later), it does not have sampling problems, and thus produces correct sharp shadows everywhere.
- Disadvantages
  - The performance problem.
  - This algorithm burns frame rate, as the number of shadow volume polygons is often large, and shadow volume polygons often cover many pixels, and so the rasterizer becomes a bottleneck

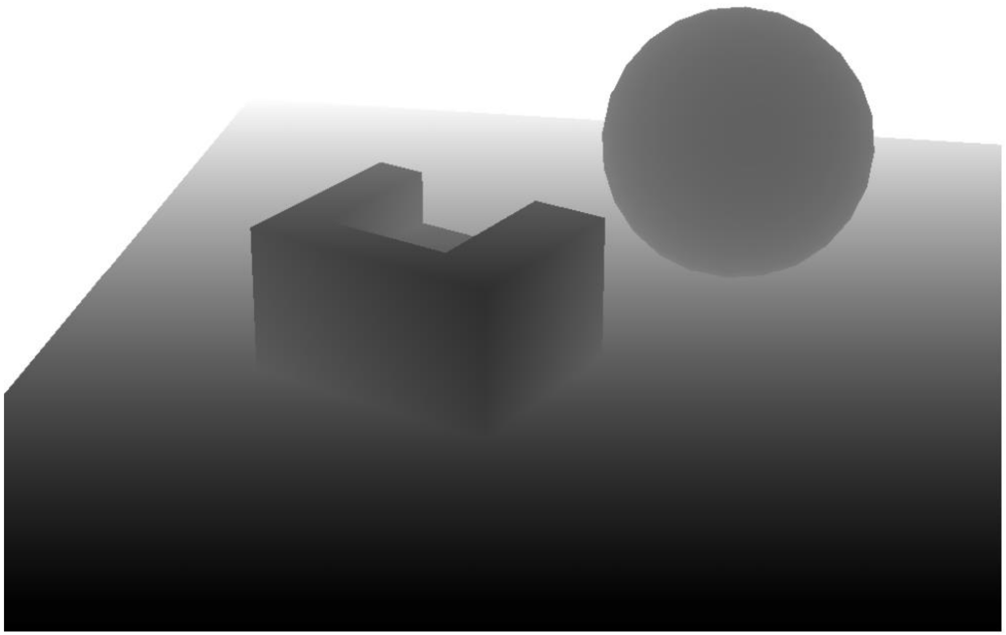
Shadow Volume

Shadow map

- In 1978, Williams proposed a common z-buffer based algorithm to generate shadows quickly on arbitrary objects.
- The idea is to render the scene, using the Z-buffer algorithm, from the position of the light source.

Shadow map

- By using z-buffer, the captured image from light's view records the distance to the object closest to the light.
- We call this entire content of the depth image “shadow map”.

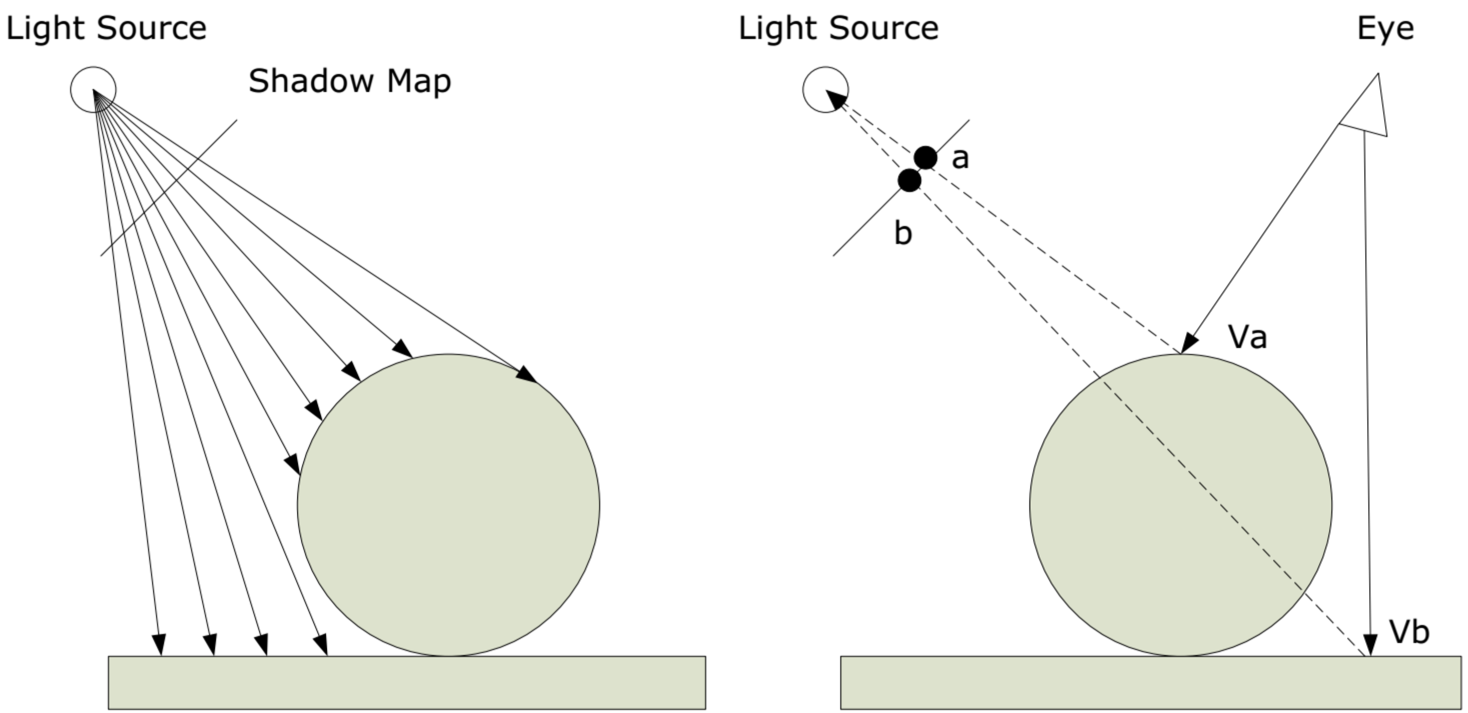


Shadow map

- To use the shadow map, the scene is rendered a second time, but this time from the viewer's view.
- Now, when each primitive is being rendered, its location is compared to the shadow map:
  - If a rendered point from the light source is farther away than the value in the shadow map, then that point is in shadow;
  - Otherwise it is not.

Shadow map

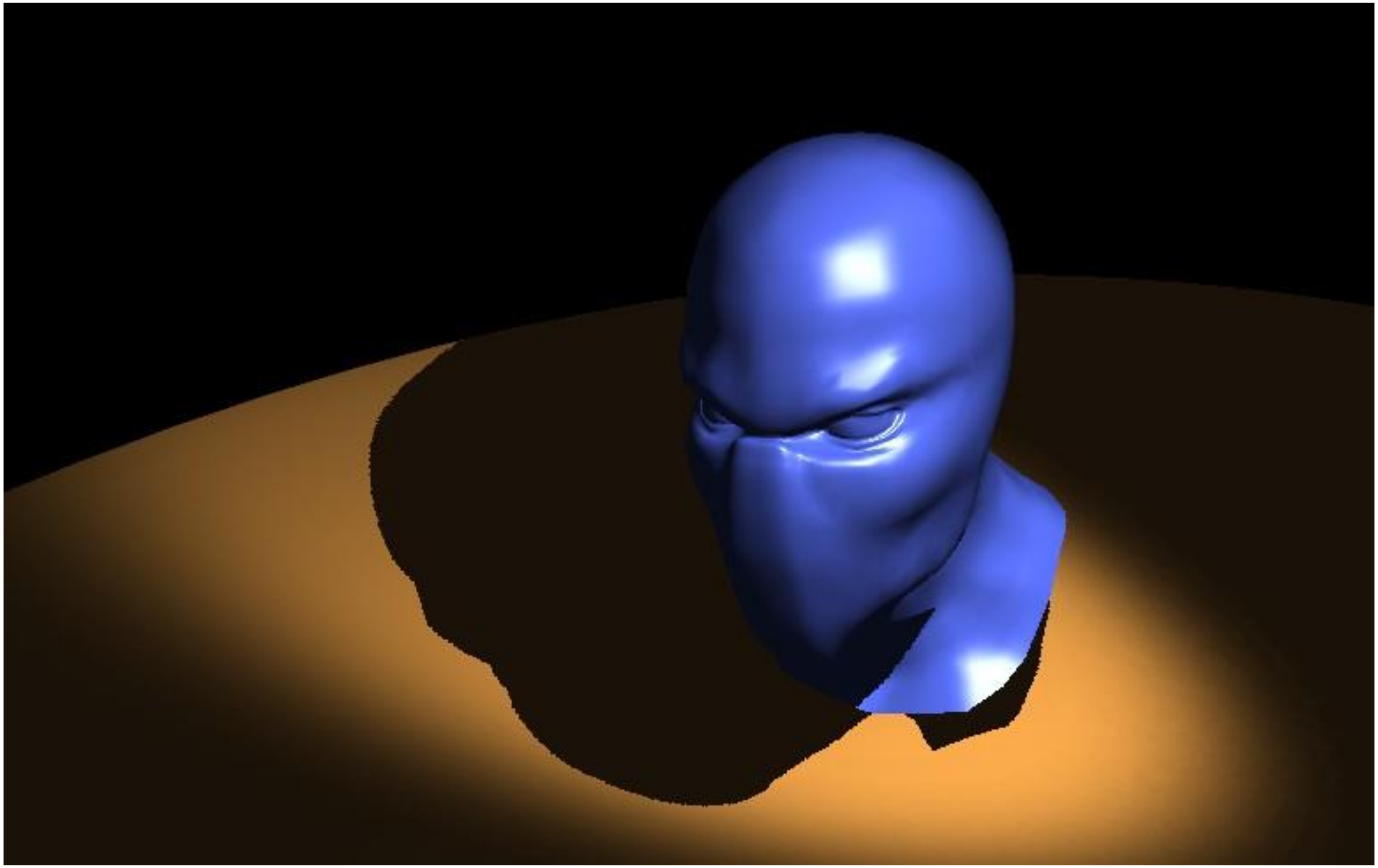
Illustration



Shadow map

- For the Picture, a shadow map is formed by storing the depths to the surface; on the right, the eye is shown looking at two locations.
- The sphere is hit at point *Va*, and this point is found to be located at texture position a on the shadow map. The depth stored in *a* is not less than point *Va* is from light, so the point is not in shadow;
- For point *Vb* distance from point light is farther than stored in shadow map, so the point is in shadow.

Shadow map

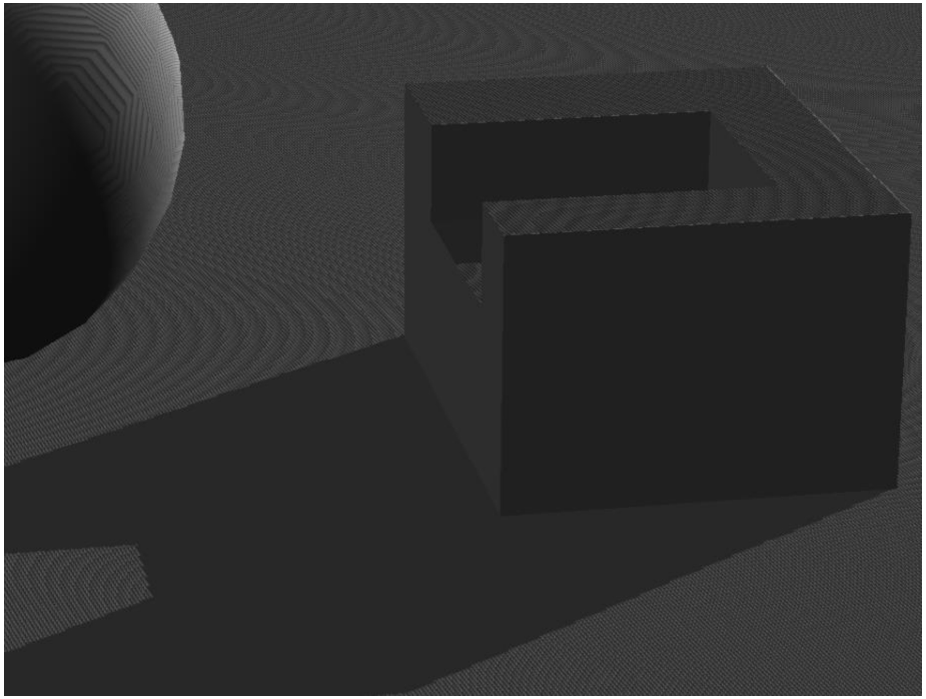


Shadow map

- Advantages
  - Most hardwares directly support shadow map, and it can be used to render arbitrary geometry.
  - It is fast. The cost of building the shadow map is linear to the number of rendered primitives and access time is constant.
- Disadvantages
  - Because shadow map is image-based, so the quality depends on the resolution of the shadow map, and the numerical precision of the Z-buffer.

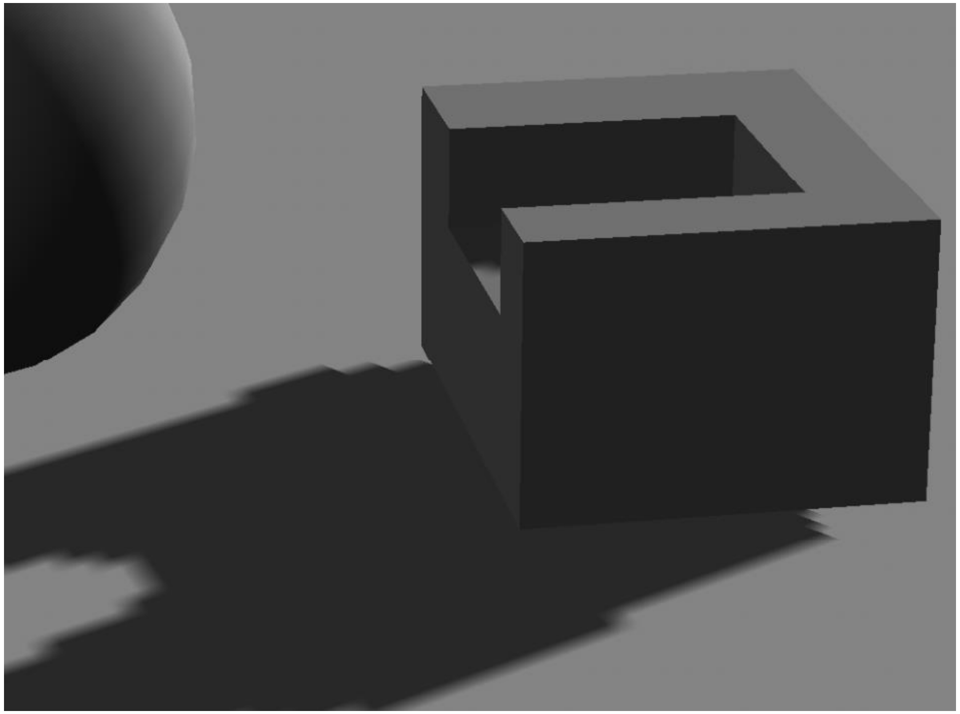
Shadow map

- Shadow map Problems
  - If the epsilon value for comparison is low, it produces a pattern on object's surfaces.



Shadow map

- Shadow map Problems
  - If the epsilon value is set too high, we will see shadow “creeps” from under the block object.



Shadow map

- Some conclusion
  - Shadow map and shadow volume are the most widely used shadow genereation methods(especially, shadow map is used more).
  - There are various extensions for these 2 techniques.
  - For more information:
    - nVidia/ATI websites
    - http://www.realtimerendering.com
    - SIGGRAPH and Eurographics websites

Recent work

- **Overview of the approach**
  - **Base on *shadow map* method.**
  - **Develop *Single-layered algorithm* to simulate highly accurate soft shadows, but exist overlap error.**
  - **Develop *Multi-layered algorithm* to reduce such errors, and choose the number of layers to trade off accuracy with performance.**

Advanced Topic

- **Soft shadows under Global Illumination**
  - **Precomputed Radiance Transfer**
    - **Spherical Harmonics**
    - **Wavelet**
    - **Spherical Piecewise Constant Basis Function**

Advanced Topic

- **Spherical Harmonics**
  - **Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments [Sloan02]**

Advanced Topic

- **Wavelet**
  - **All-Frequency Shadows Using Non-linear Wavelet Lighting Approximation [Ng03]**

- **Spherical Piecewise Constant Basis Function**
  - **Spherical Piecewise Constant Basis Functions for All-Frequency Precomputed Radiance Transfer**

    Kun Xu, Yun-Tao Jia, Hongbo Fu, Shi-Min Hu and Chiew-Lan Tai

Reference

- http://courses.csail.mit.edu/6.837/F06-www/lectures/22_shadows.pdf
- *J.-M. Hasenfratz, M. Lapierre, N. Holzschuch, F.X. Sillion* A survey of Real-Time Soft Shadows Algorithms
- http://www.ce.chalmers.se/edu/year/2006/course/EDA425/lectures/shadrefl.pdf
- *Xiao-Hua Cai, Yun-Tao Jia, Xi Wang, Shi-Min Hu and Ralph R. Martin* Rendering Soft Shadows using Multi-layered Shadow Fins
- Sloan, Peter-pike and Kautz, Jan and Snyder Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments
- Ren Ng, Ravi Ramamoorthi and Pat Hanrahan All-Frequency Shadows Using Non-linear Wavelet Lighting Approximation
- *Kun Xu, Yun-Tao Jia, Hongbo Fu, Shi-Min Hu and Chiew-Lan Tai* Spherical Piecewise Constant Basis Functions for All-Frequency Precomputed Radiance Transfer

Reference

- **http://www.realtimerendering.com**
- **http://www.nvidia.com/page/home.html**
- **http://ati.amd.com/products/index.html**
- http://www.siggraph.org
- https://www.eg.org/
- http://developer.nvidia.com/object/shadow_mapping.html
- Lance Williams Casting curved shadows on curved surfaces
- http://developer.nvidia.com/object/doc_shadows.html