Advanced Texturing

Computer Graphics
Instructor: Sungkil Lee

Today

- **Advanced texture mapping**
  - Environment mapping
  - Normal/bump mapping
  - Displacement mapping with tessellation
- **Shadow mapping**
  - Hard shadows
  - Soft shadow mapping

Advanced Texture Mapping

Implication in Blinn-Phong Illumination Model

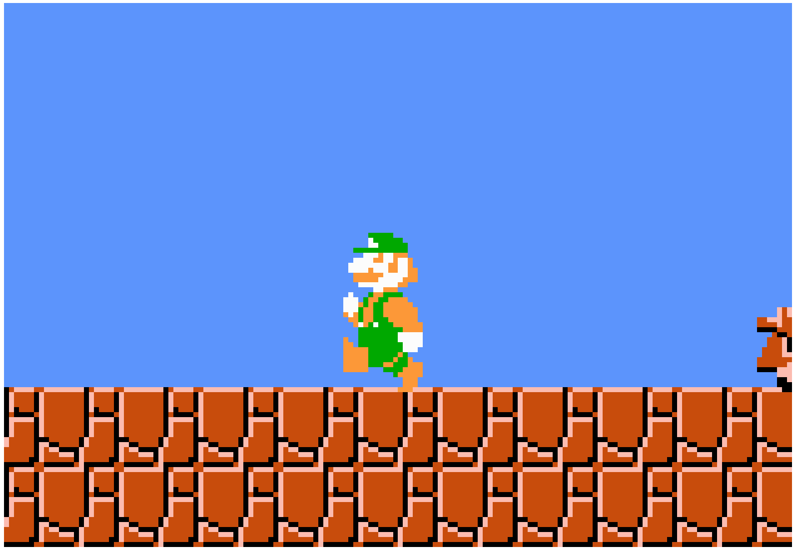
$$\mathbf{I}'_r = \mathbf{k}_a \, \mathbf{I}_a + \max(\mathbf{k}_d (\mathbf{l} \cdot \mathbf{n}) \mathbf{I}_d, 0) + \max(\mathbf{k}_s (\mathbf{n} \cdot \mathbf{h})^{\beta} \, \mathbf{I}_s, 0)$$

- **Surface properties are defined as images.**
  - We usually define $\mathbf{k}_a, \mathbf{k}_d, \mathbf{k}_s$ with manually chosen constants.
  - With texture mapping, we define $\mathbf{k}_a, \mathbf{k}_d, \mathbf{k}_s$ using images.
  - Recall that the image is a sampled representation of a continuous function, which can be efficient to represent varying surface properties.

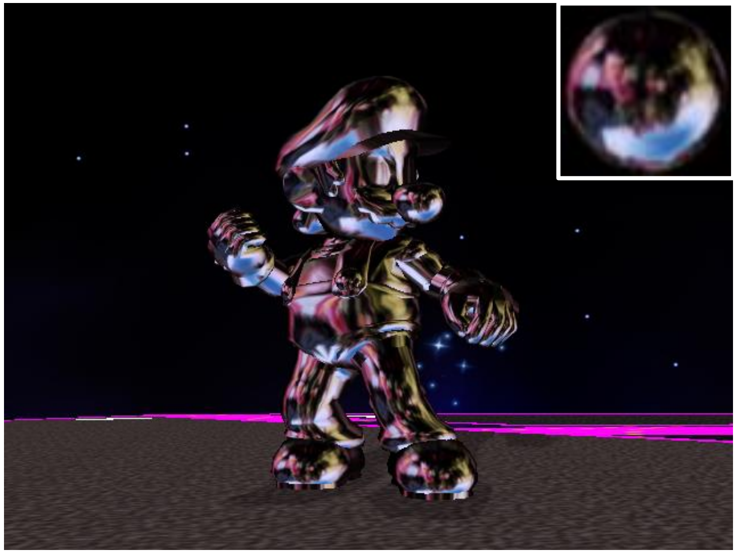
- **Common types of texture mapping**
  - Diffuse mapping ($\mathbf{k}_d$)
  - Environment mapping (reflection mapping) ($\mathbf{k}_s$)
  - Bump/normal mapping ($\mathbf{n}$)
  - Specular Mapping (β), but rare in practice

Environment Mapping

- **Environment mapping (or reflection mapping)**
  - Uses a picture of the environment for highly specular reflection ($\mathbf{k}_s$)
  - The colors of environments are mapped to the surface according to the reflector against the view vector.



Shiny invincible Mario



Emulation using env. mapping

Environment Mapping

- **Omnidirectional texture mapping**
  - Environment mapping requires omnidirectional images, which can encode 360 surroundings (similar to 360 VR).

- **Common ways of omnidirectional texture mapping**
  - Sphere map
  - Cube map
  - (Dual)-paraboloid map

Sphere (Environment) Mapping

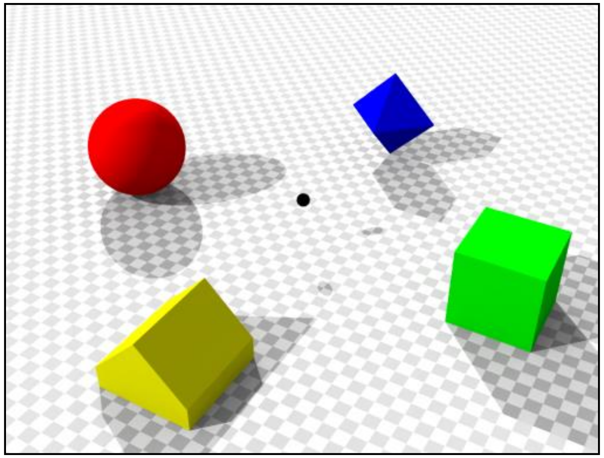
- **Environment mapping:**
  - Environment maps are modeled as a sphere or cube.

sphere map

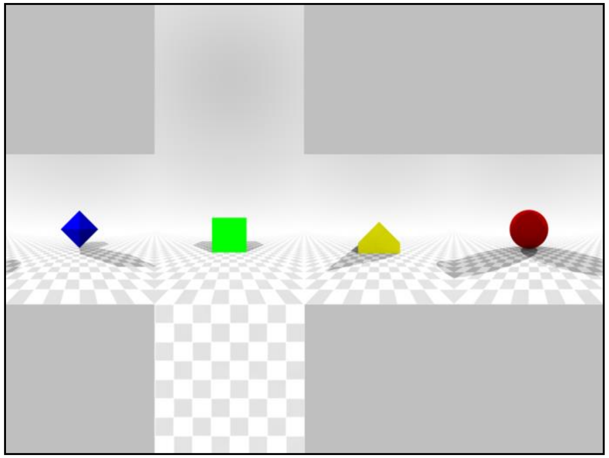
rendering examples with sphere-map reflections

Cube (Environment) Mapping

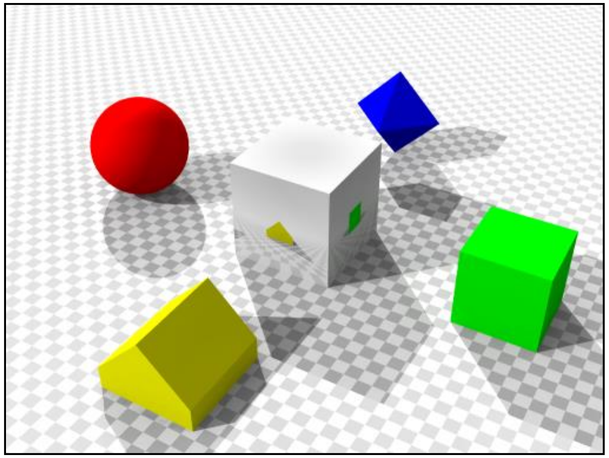
- **Cube mapping:**
  - environment mapping via a 6-face cube map
  - Could be a static texture or dynamically-generated for every frame



place a viewer in a scene



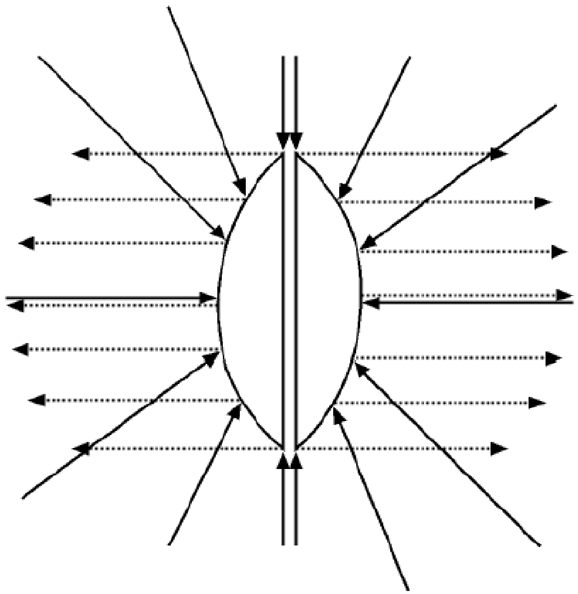
render 6-face cube map



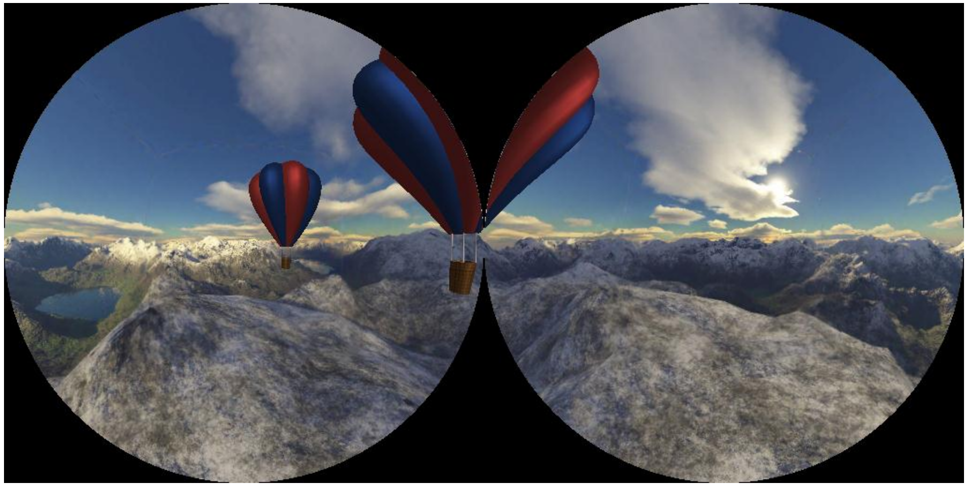
apply texture to the surface

Dual-Paraboloid (Environment) Mapping

- **Paraboloid mapping**
  - encode a hemisphere in a single image using reflection on the paraboloid surface.
  - Two images can represent the whole surrounding environment.



Reflection on dual-paraboloid

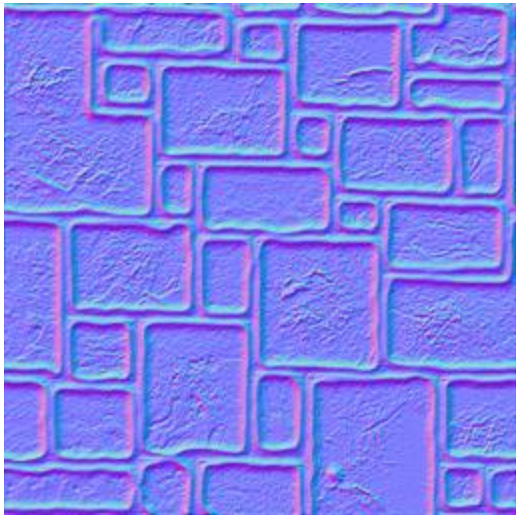


Example dual paraboloid maps.
(C) Imagination Tech.

Bump/Normal mapping

- **Bump/normal mapping**
  - Emulates altering normal vectors during the rendering process
  - Efficient approximation of fine-grained geometries.

diffuse mapped

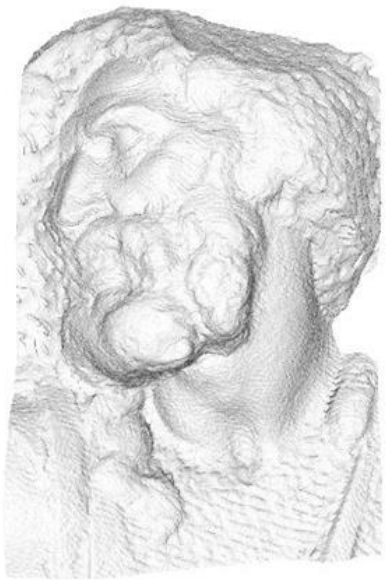


normal map

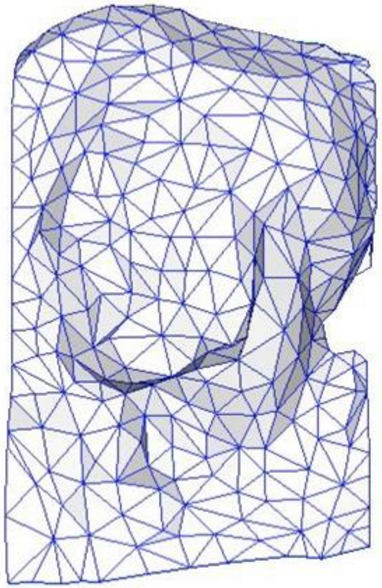
bump mapped

Examples

- **Comparison between fine geometry and normal mapping**



original mesh
4M triangles



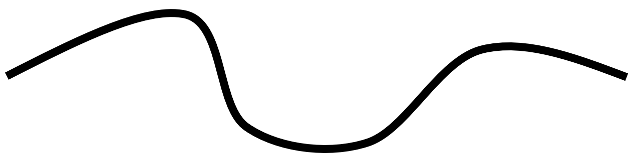
simplified mesh
500 triangles

simplified mesh
and normal mapping
500 triangles

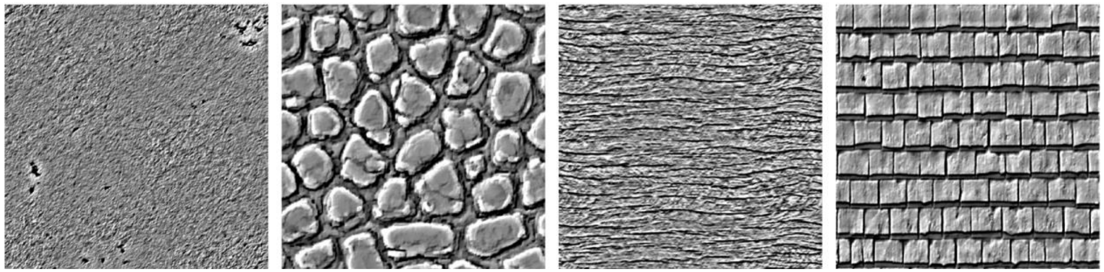
Another Example

Bump Map

- **Bump map:**
  - A 1-D height field, which records the height variation of the surface.

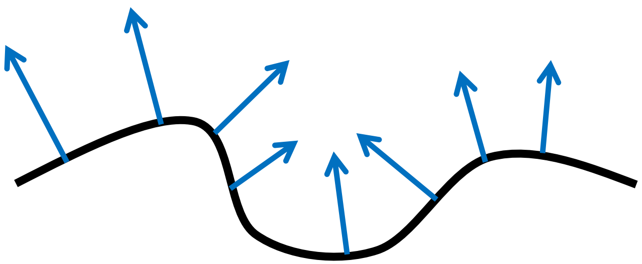


  - Gray-scale image examples

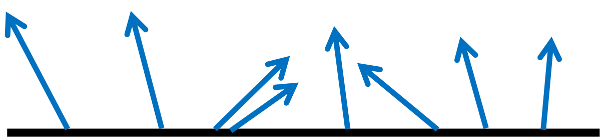


Bump/Normal mapping

- **Basic idea**
  - Cheating your eyes without actual changes in geometric shapes.
  - We change only normals and achieve as if geometry actually changes.



Normals on bumpy surface

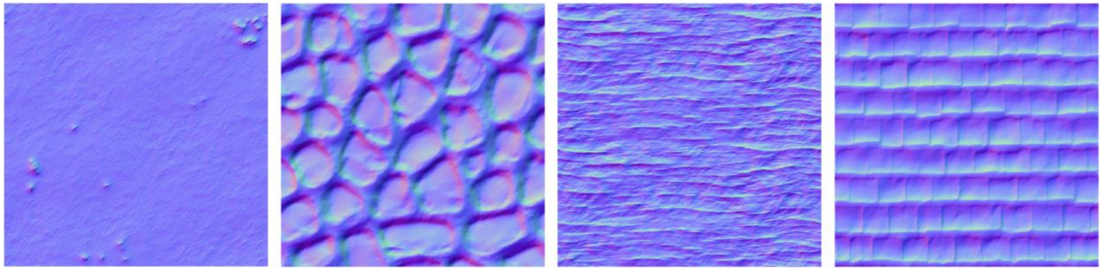


Bumpy normals on flat surface

Normal map

- **Normal map:**
  - encodes normal vectors in images, and is converted from a bump map.
  - Actually, bump mapping refers to the normal mapping in many cases.

- **Encoding of Normals**
  - A normal vector $\mathbf{n}$ can be encoded using 3-channel RGB images
  - (x, y, z) normal vector is encoded in RGB; the ranges are in $[-1,1]^3$.
    - In 8-bit images, [0, 127, 255] is mapped to [-1,0,1].
  - So, the default normal without bumping is (0,0,1), which is encoded in (127, 127, 255) in a bump texture image.



Bump Map vs. Normal Map

- **More on bump and normal maps**
  - A normal map saves computation time needed on the fly, because it provides already computed values of normal vectors from a height (= bump) map, which are needed for the shaded color determination.

  - A single pixel (an RGB value) in a normal map has to be calculated from the surrounding pixels in a height map, i.e. the direction of a normal vector represented by a normal pixel is determined from the surrounding heights in a height map.

Displacement Mapping

- **Bump/normal mapping is a trick that cheats your eyes.**
  - Because the normal vector is not derived from your geometry.
  - A trick is revealed and the most pronounced in the edge of your geometry.

geometry

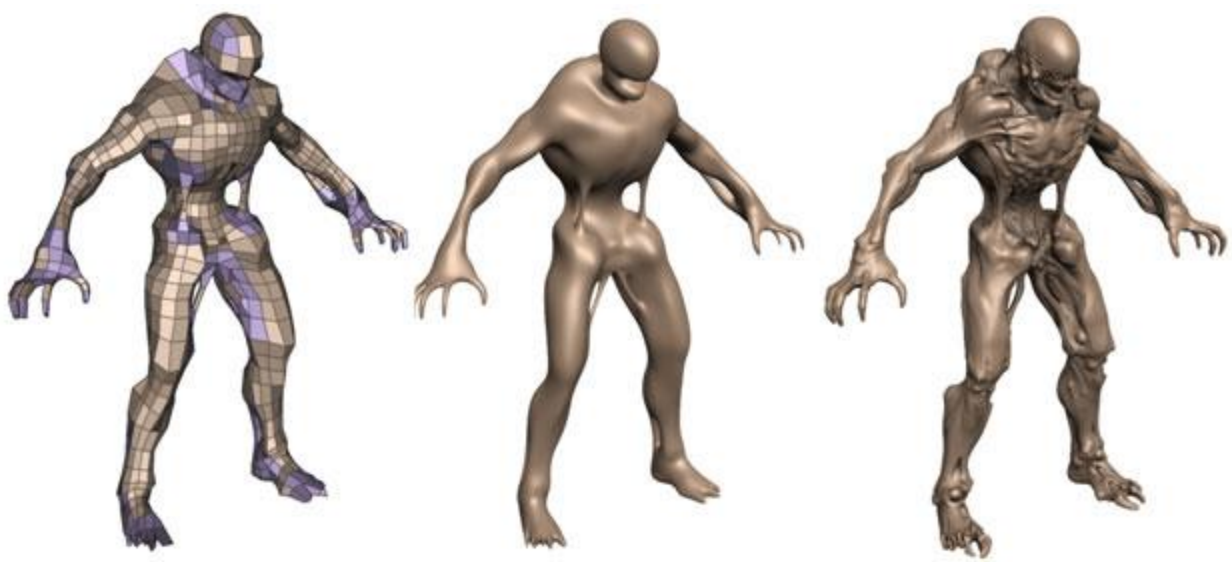
normal mapping

displacement mapping

- **Displacement mapping:**
  - Truly relocate geometry using the height map (displacement/bump map).
  - For displacement mapping, actually you need a tessellation shader, which subdivide your input meshes; this is a modern approach in many real-time applications.

Displacement Mapping + Tessellation Shaders

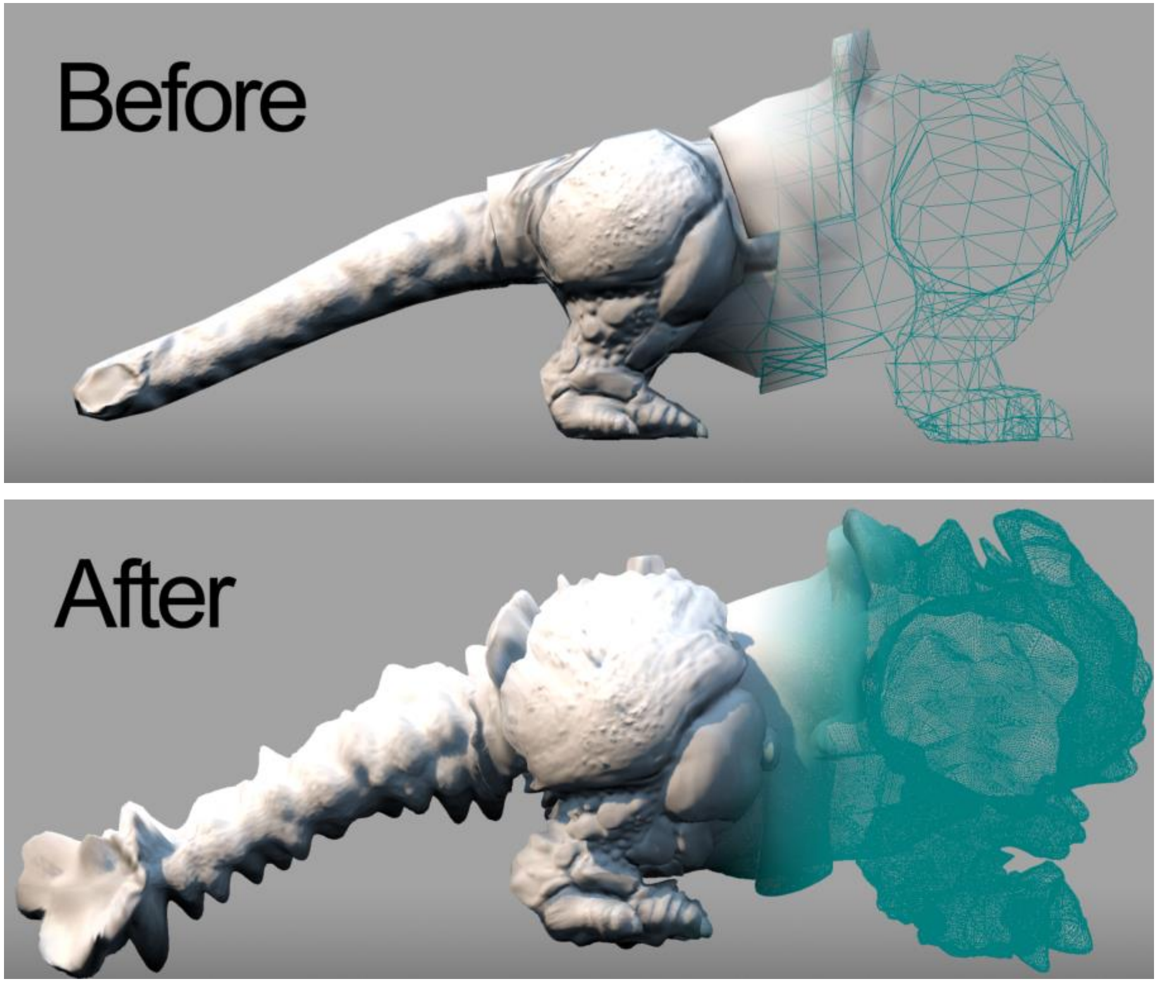
- **Tessellation Shaders**
  - A feature of modern GPUs for on-the-fly tessellation
  - Coarse control patches are tessellated (subdivied) into fine triangles, and then, high-frequency details are elevated with height map.



input control patches | tessellation | tessellation + displacement mapping

Displacement Mapping + Tessellation Shaders

- **More examples**

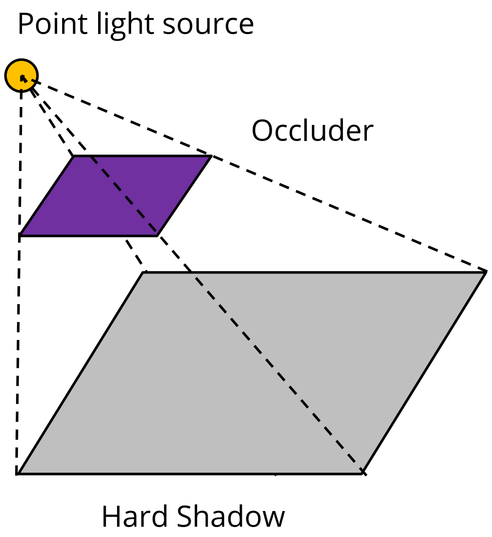


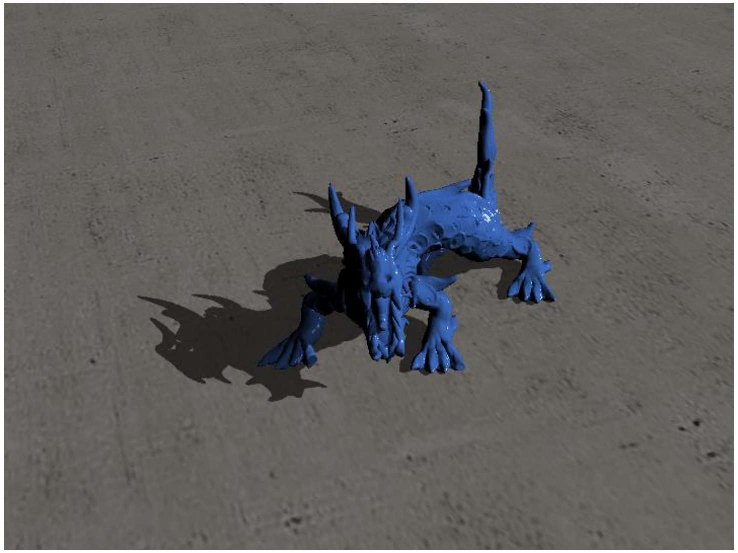
Shadow Mapping

Shadows

- **Shadows**
  - Dark areas where the lights are obstructed by occluders.
  - Shadows create more realistic images.

- **Hard shadows**
  - A point light source makes hard shadows.





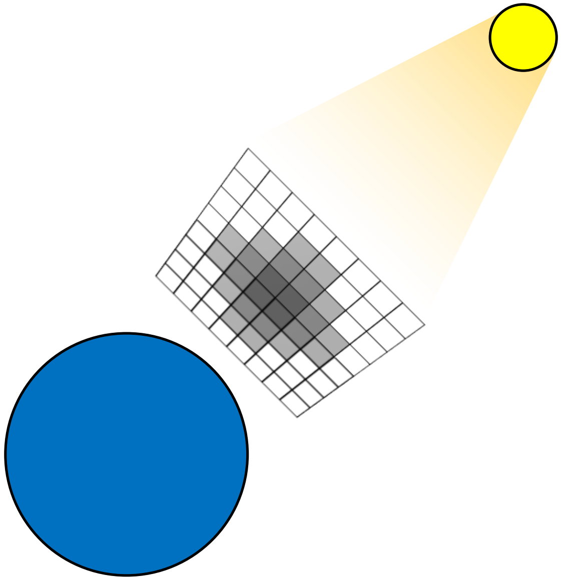
Shadows

- **Two basic approaches**
  - Shadow volume: geometry-based methods
  - Shadow mapping: image-based methods
    - De facto standard in movie productions and 3D games

	Geometry-based	Image-based
Pros	• No aliasing • Unlimited light field of view	• Better performance • Easy to implement as post-processing
Cons	• Poor scene scalability • Fill-rate limited	• Very hard to tweak • Poor resolution scalability • Shadow aliasing and acne

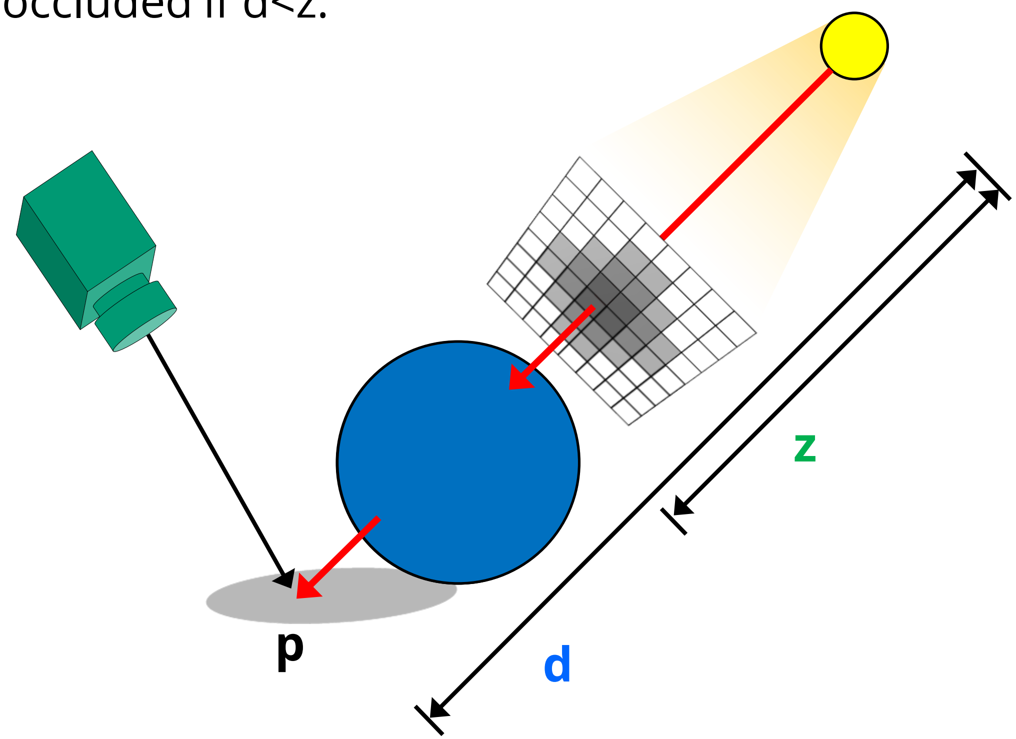
Algorithm

- **First pass: shadow map rendering**
  - Render the depth image (shadow map) from the light viewpoint.



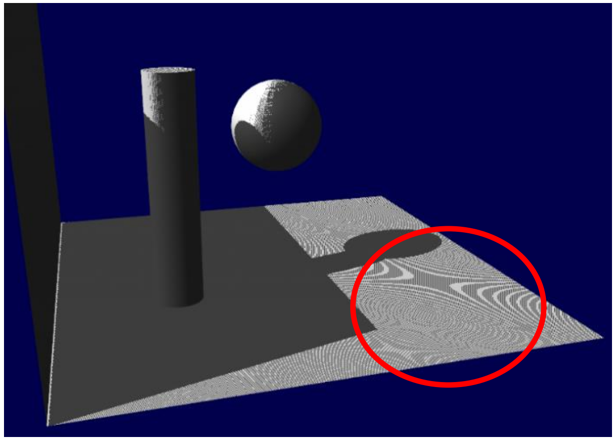
Algorithm

- **Second pass: calculating shadow factors**
  - Calculate shadow factors for each fragment that determines whether the fragment is shaded.
    - Compare depth values of the current fragment (**z**) and the shadow map (**d**).
    - Point p is lit (no shadows) if d>=z.
    - Point p is occluded if d<z.



Limitations

- **Limitations of basic shadow mapping**
  - Shadow acne
  - Aliasing when using low-resolution shadow maps
  - Unrealistic hard shadows



Shadow acne

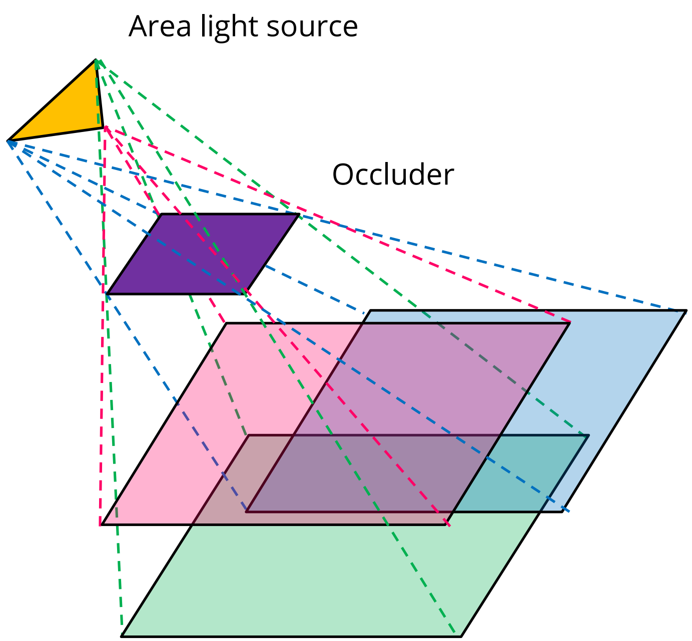
http://www.opengl-tutorial.org/intermediate-tutorials/tutorial-16-shadow-mapping/

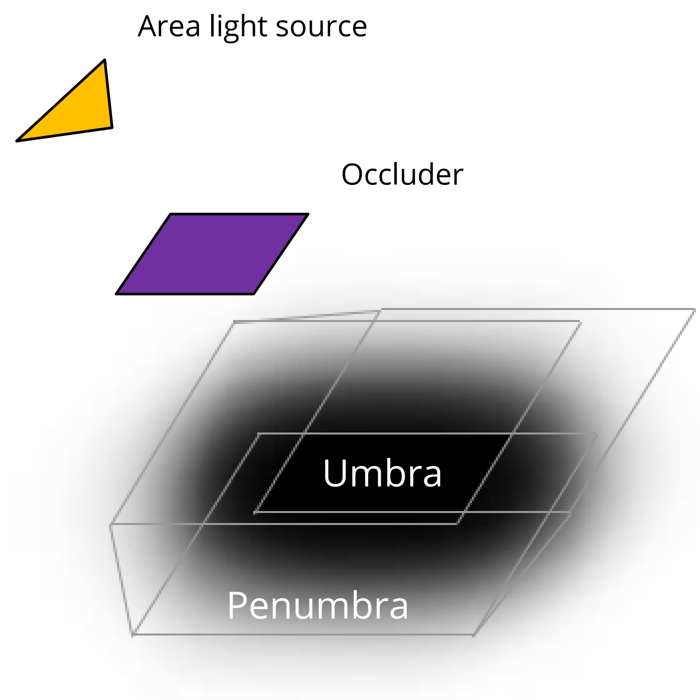
Shadow aliasing

https://msdn.microsoft.com/en-us/library/windows/desktop/ee416324(v=vs.85).aspx

Soft Shadows

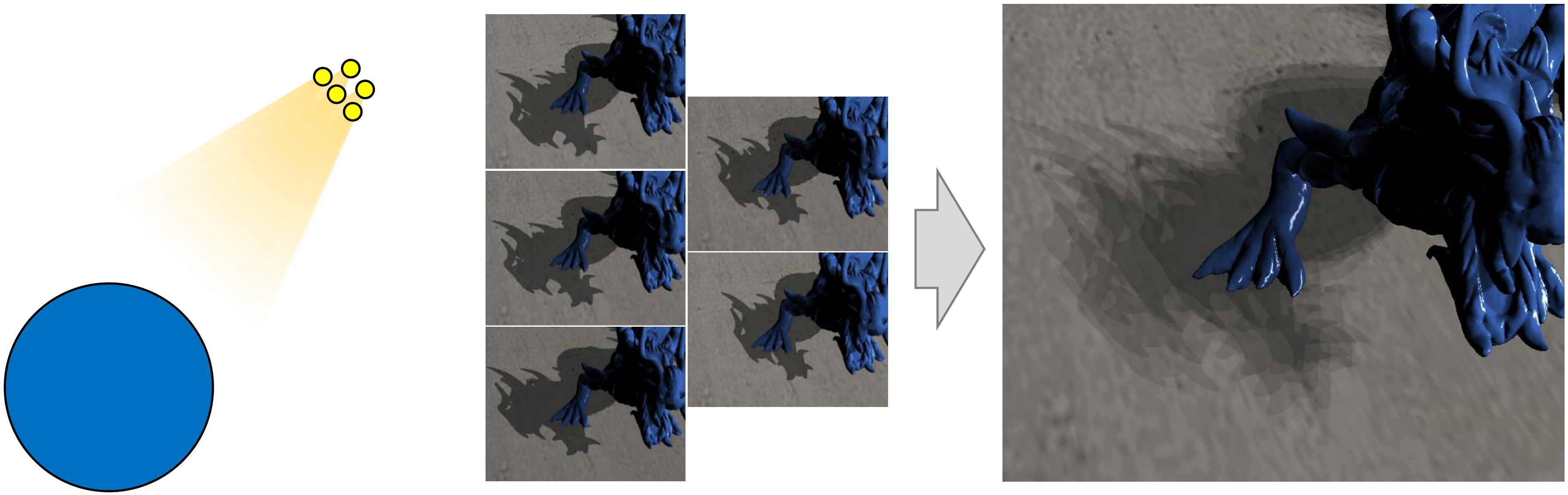
- **Soft shadows**
  - Area light sources make soft shadows.
  - Looks more realistic than hard shadows.
  - Soft shadows create umbra and penumbra areas.





Soft Shadow mapping

- **Basic algorithm: multi-sampling of lights**
  - Sample positions of point lights in the light area.
  - For each point light, acquire hard shadow maps.
  - Blend each shadowed results.



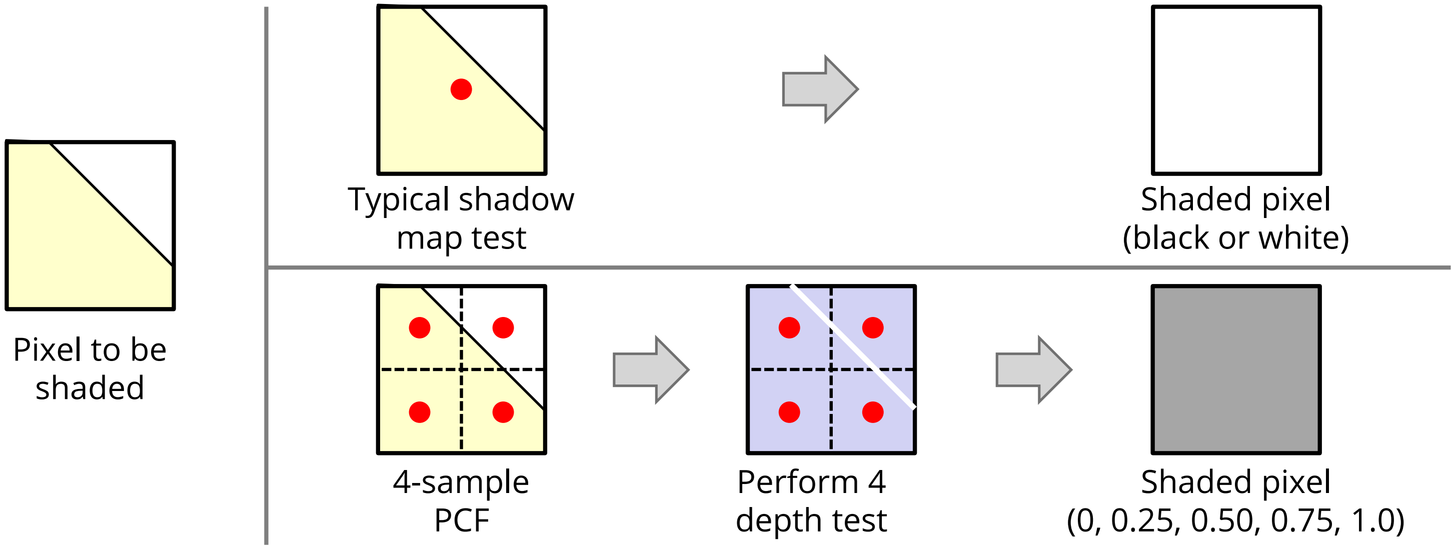
Soft Shadow mapping

- **Limitations**
  - Very costly with many light samples
  - Memory problems

- **Approximate soft shadow mapping**
  - PCSS (Percentage-closer soft shadows)
  - CSSM (Convolution soft shadow maps)
  - VSSM (Variance soft shadow maps)
  - ESSM (Exponential soft shadow maps)

Soft Shadow mapping

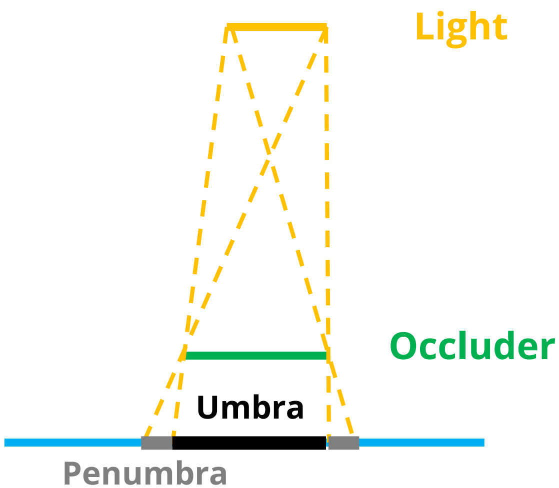
- **Examples of Crytek's shadow method**
  - Percentage-closer filtering (PCF) with Poisson multi-sampling.



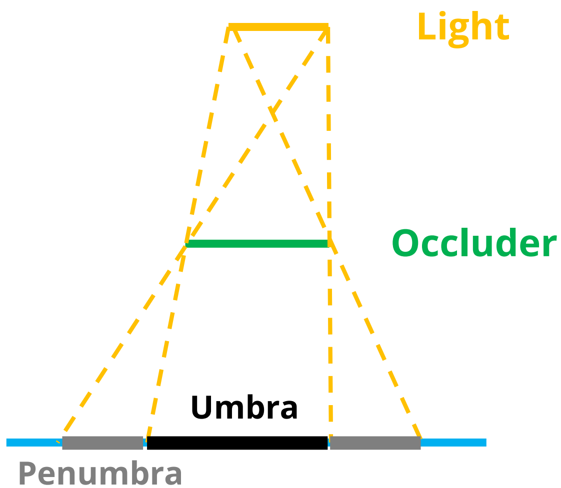
Percentage-closer filtering, Fernando, SIGGRAPH 2005 Course

Soft Shadow mapping

- **Examples of Crytek's shadow method**
  - Adjust the kernel size of PCF to approximate variable penumbra.
  - Kernel size is associated with the average distance ratio to shadow casters.



Small kernel size, less softer shadow



Large kernel size, much softer shadow

Soft Shadow mapping

- **Examples of Crytek’s shadow method**

Playing with Real-Time Shadows
Kasyan, SIGGRAPH 2013

Open Issues of Shadow Mapping

- **Robust**
  - Stable under objects, lights and camera motion
  - No light bleeding and no flickering
- **General**
  - Works with all light types and dynamic geometry
  - Scalable from small to large light sources
- **High quality and high performance**
  - Variable penumbra and no aliasing
  - Sparse sampling and good culling