
Computer Graphics

12 - More Lighting, Texture

Yoonsang Lee
Hanyang University

Spring 2026

Final Exam Announcement

- Date & time: **June 22 (Mon), 10:00 - 11:30 AM**
- Place: IT.BT Room **508** & Room **510**
 - **Check the announcement in LMS** for your assigned exam room:
https://learning.hanyang.ac.kr/courses/201420/discussion_topics/501477
- Scope: Lecture, Lab, Lab Check **8 ~ 13, Project 2 & 3**
- All other exam policies remain unchanged.

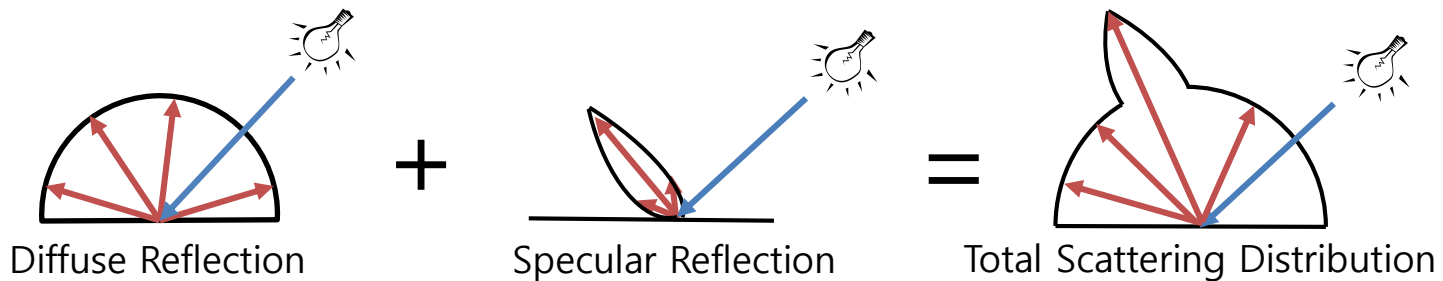
Outline

- More Lighting
 - BRDF
 - Local & Global Illumination
- Texture Mapping
 - Concept
 - UV Mapping
 - Texture Mapping Process
 - Defining Texture Coordinate Function
 - Rendering Texture-Mapped Objects
 - Diffuse, Specular, Normal Maps
 - Various Uses of Texture Maps

More Lighting

Recall: Reflection of General Materials

- Most materials have both diffuse and (non-ideal) specular reflection.



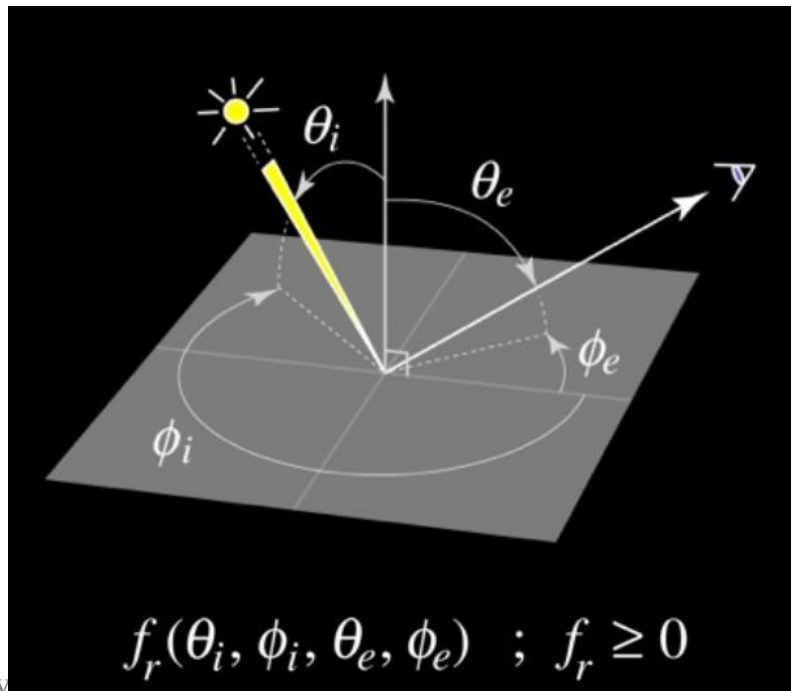
- Different materials have different combinations of diffuse and specular reflection.
- We can represent **how a material reflects light** as a function.
- **BRDF** (Bidirectional Reflectance Distribution Function)

* This image is from the slides of Prof. Andy van Dam (Brown Univ.)

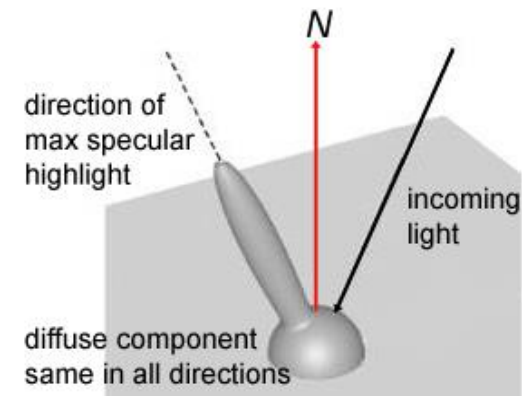
<http://cs.brown.edu/courses/csci1230/lectures.shtml>

Bidirectional Reflectance Distribution Function (BRDF)

- A BRDF describes **how incoming light is distributed into outgoing directions**
 - Incoming direction: θ_i, ϕ_i
 - Outgoing direction: θ_e, ϕ_e
 - f_r gives the **fraction of incoming light reflected toward the outgoing direction**.



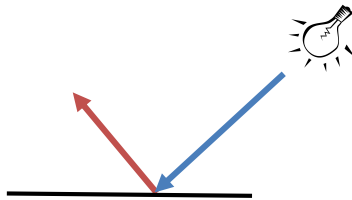
An example BRDF



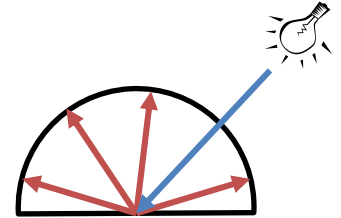
Examples of BRDF



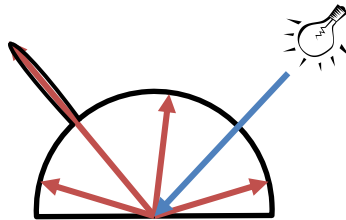
Mirror - Ideal Specular



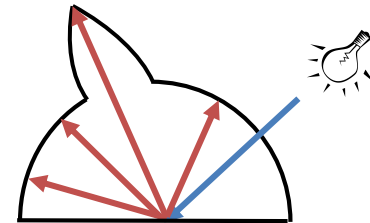
Chalk – Ideal Diffuse



Polished Marble Floor – Strong Specular

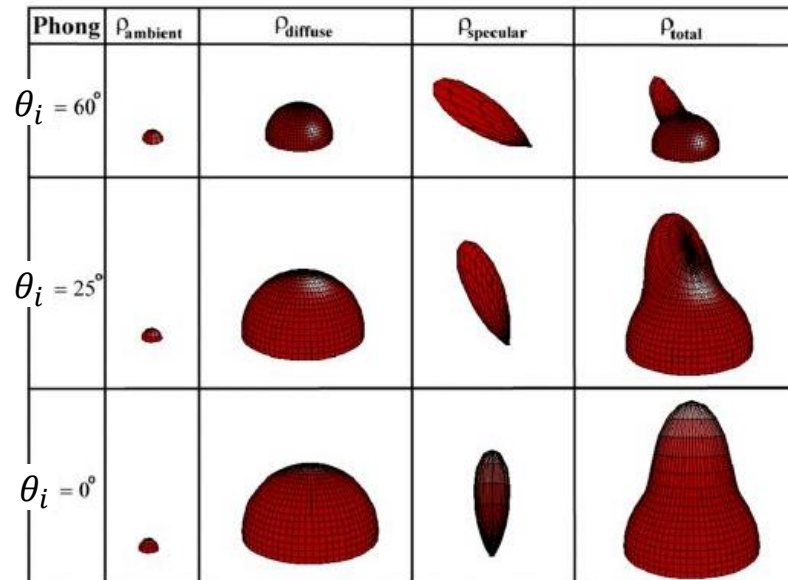


Apple – Broad Specular



Phong Illumination Model as a BRDF

- Phong illumination can be interpreted as a simple BRDF model:
 - A hemisphere for diffuse reflection
 - A specular lobe modeled by $\cos^n(\alpha)$



Measuring BRDF

- Phong's model is only an approximation.
Real material BRDFs can be measured directly.
- BRDFs of real materials can be measured using devices like this:
 - Basic idea: rotating light source & rotating sensor

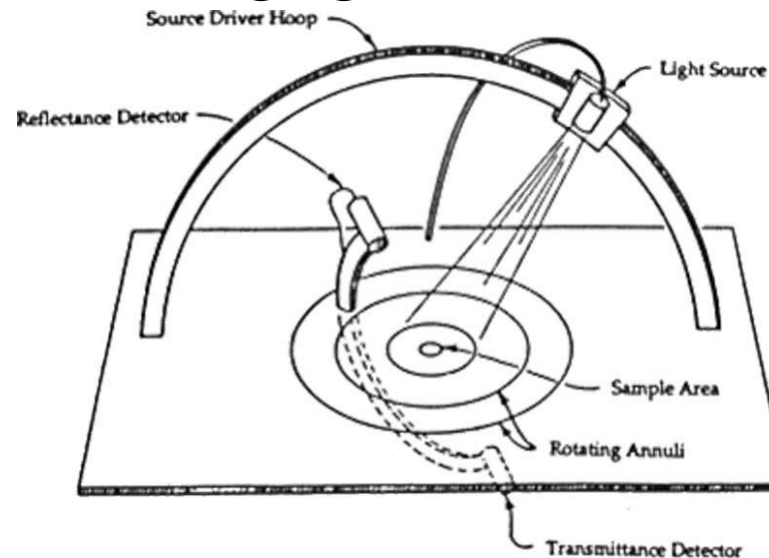
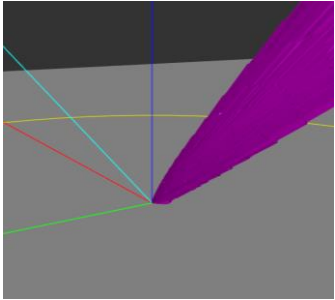
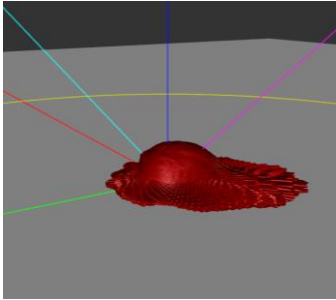
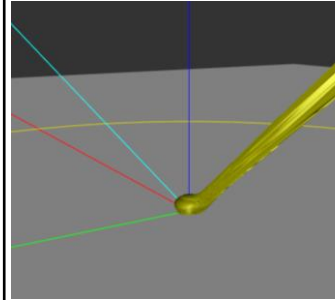
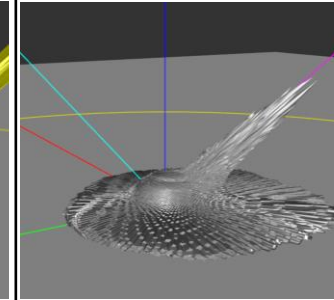
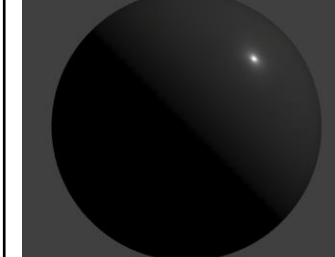
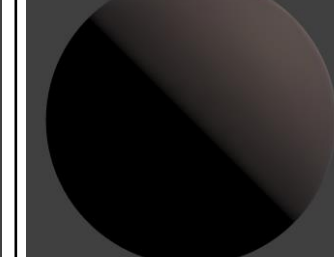
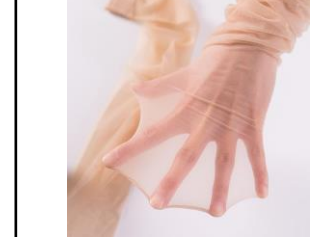


Image credit: [Chuck Moidel](#)

Using Measured BRDF for Rendering

- Measured BRDFs can be used for rendering.

	Nickel	Pink fabric	Gray plastic	Nylon
Measured BRDF				
Rendered object				
Real object example				

Quiz 1

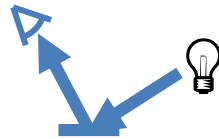
- Go to <https://www.slido.com/>
- Join #cg-ys
- Click "Polls"

- Submit your answer in the following format:
 - **Student ID: Your answer**
 - e.g. **2024123456: 4.0**

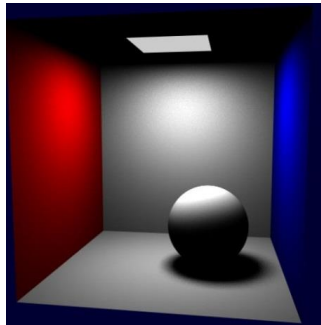
- Note that your quiz answer must be submitted **in the above format** to be counted for attendance.

Local vs. Global Illumination

- **Local (or direct, or non-global) illumination**
 - Models light that **comes directly from a light source**
 - Can be rendered fast, but less realistic (unrealistic)
 - e.g. Phong illumination model



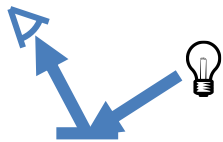
Direct illumination



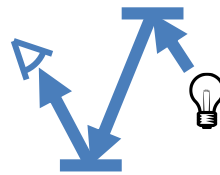
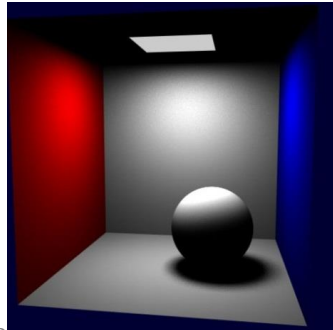
Local vs. Global Illumination

- **Global illumination**

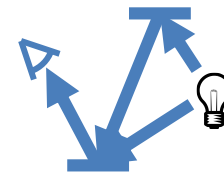
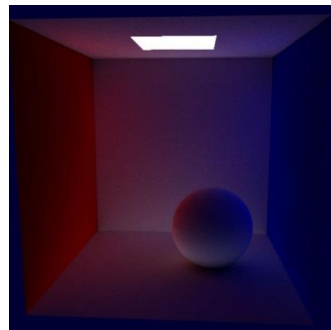
- Direct illumination +
- Indirect illumination that models light **reflected by other surfaces** (inter-object reflections)
- Slow, but much more realistic
- e.g. Ray tracing, Path tracing, Radiosity



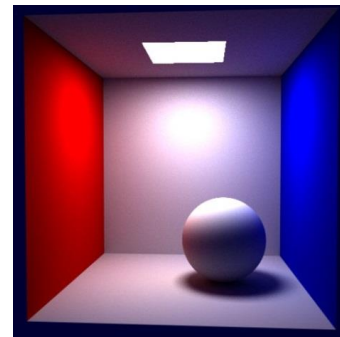
Direct illumination



Indirect illumination



Global illumination



+

=

Phong Illumination Model & Local, Global Illumination

- Phong illumination model is basically a local illumination model.
- Instead of explicitly computing indirect illumination, Phong's mode uses a constant ambient term.



Ambient

- → Fast, but cannot reproduce realistic global lighting effects.

Texture Mapping

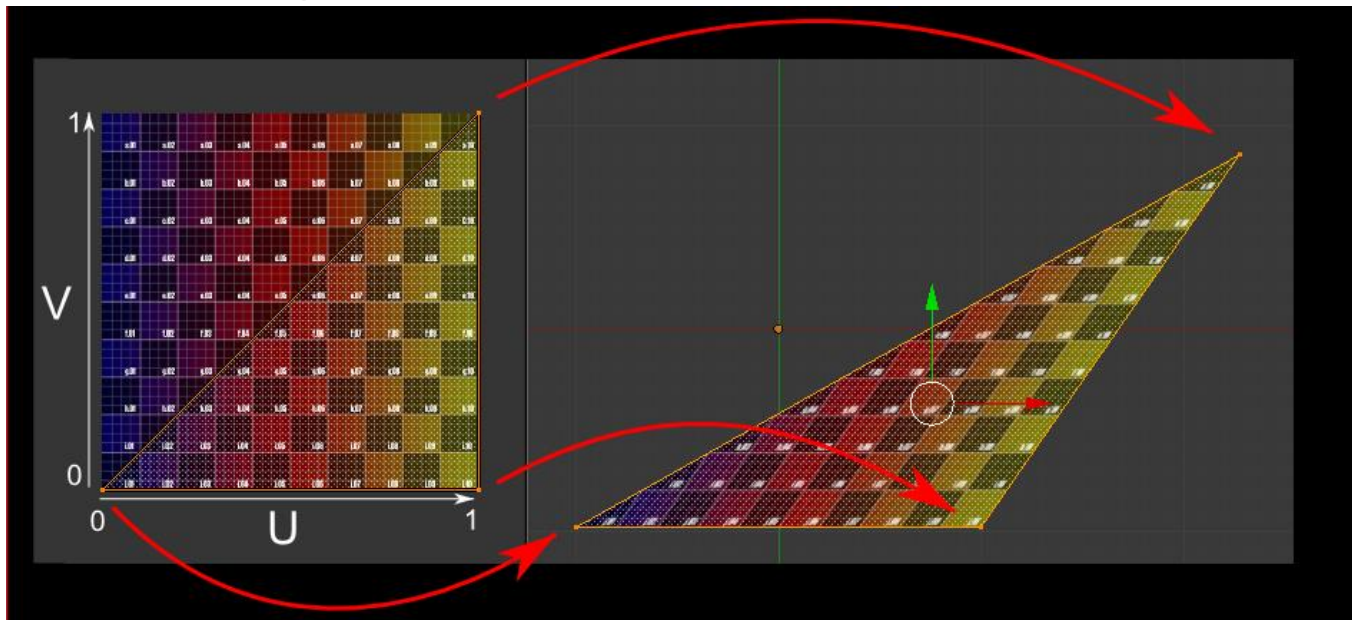
Texture Mapping

- A technique of defining surface properties **as a function of position on a surface**
 - Usually, a 2D texture image (texture maps) is applied to a 3D object surface.
- Main objective: Adding surface detail efficiently
 - Rich visual detail without increasing geometric complexity
 - Compared to high-poly models, fewer polygons are required and rendering is significantly faster

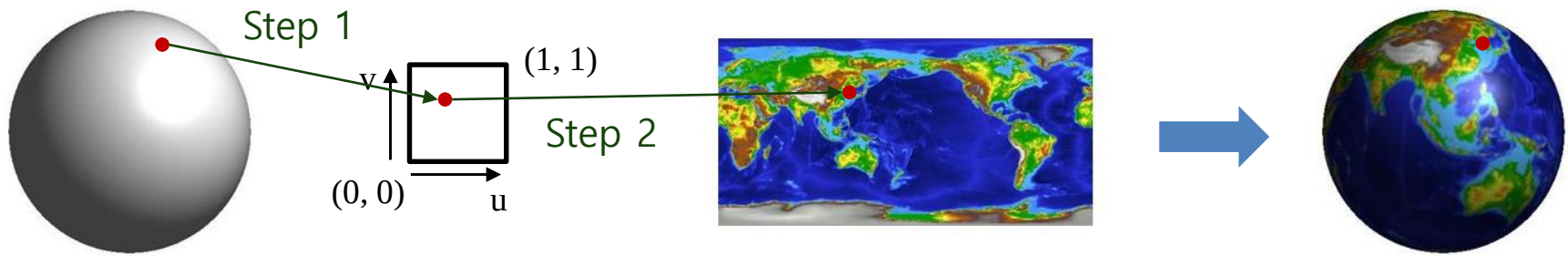


UV Mapping

- Texture mapping is commonly defined using *UV mapping*.
- UV mapping assigns 2D texture coordinates (u, v) to each vertex of a 3D model.
 - Specifies where each vertex maps onto the 2D texture image.
- The UV mapping function is also called the *texture coordinate function*.



Texture Mapping Process

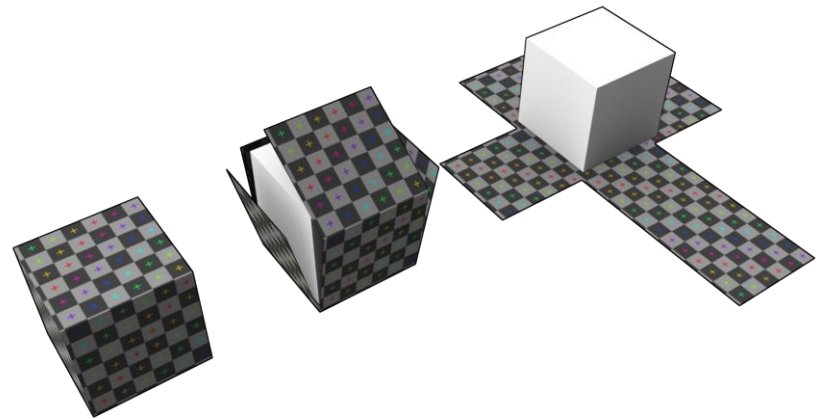


- 1. Map a surface point to a point in UV space (UV mapping).
- 2. Map the UV coordinates to a point on texture.
 - Step 2 is simply a scaling, automatically done by system.
- Step 1 is the *texture coordinate function* we need to define.

Defining Texture Coordinate Function:

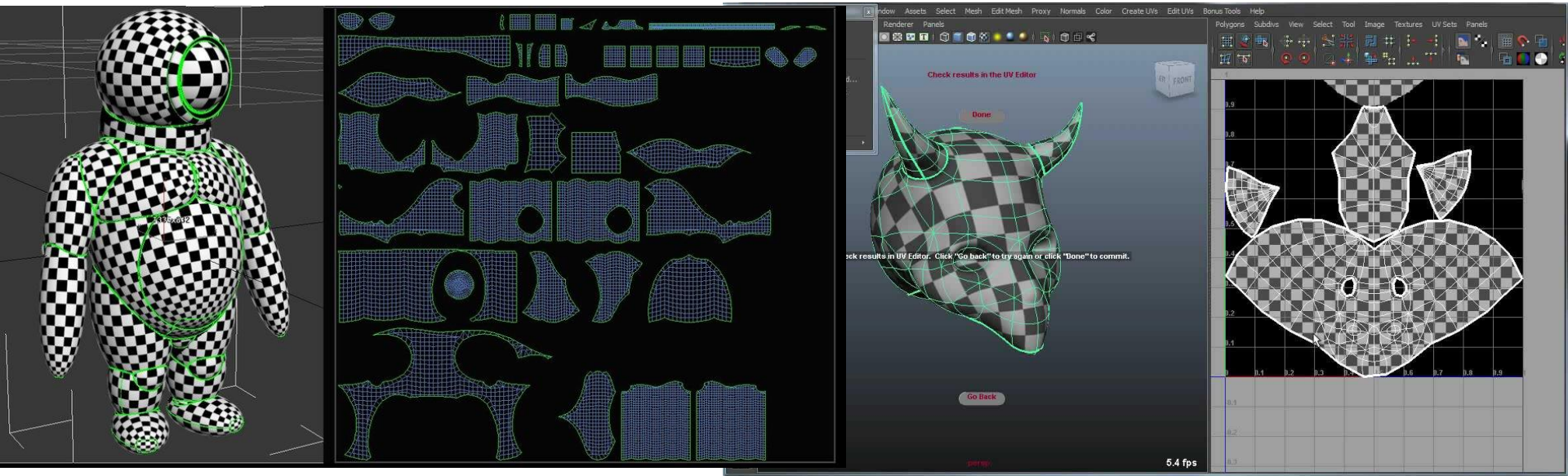
1) Creating "UV map"

- "Unwrap" a 3D surface into a 2D domain.
 - 3D modeling software such as Blender often automates this process.
 - Artists often refine automatically generated UV maps through manual editing.

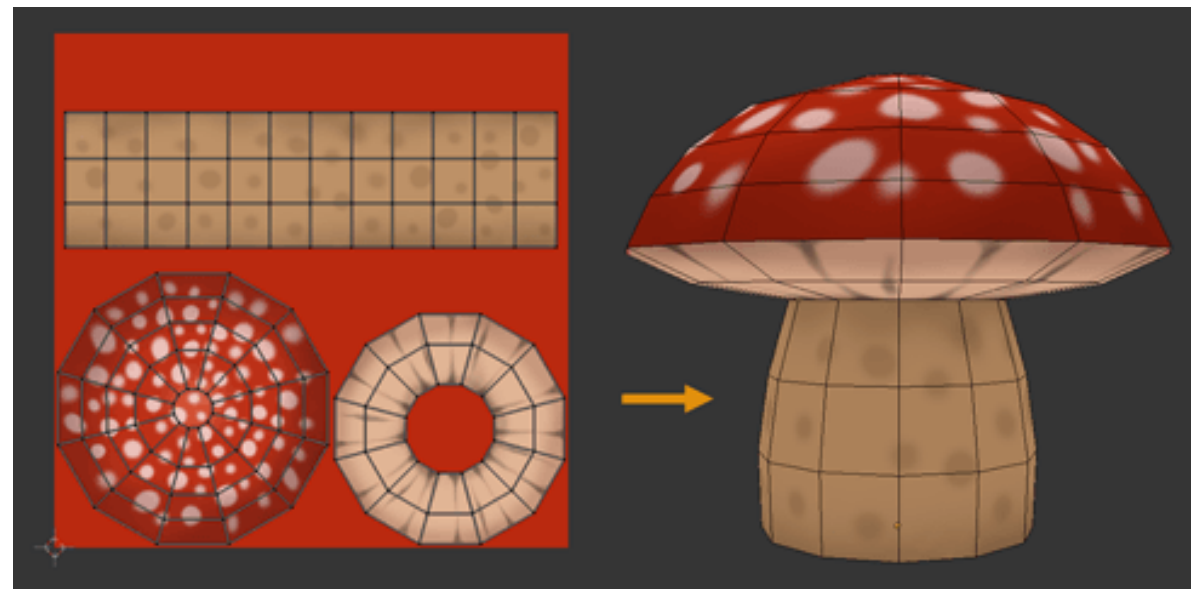
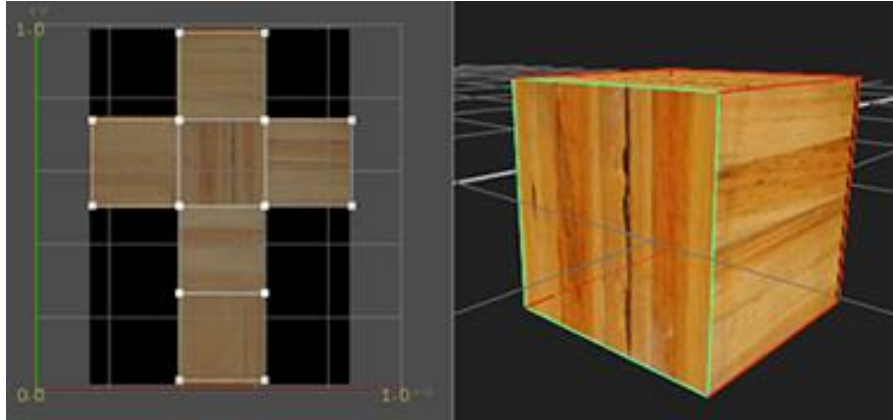


"Unwrapping"

- Automatic “unwrapping” algorithms
 - Assign UV coordinates to minimize texture distortion.
 - Different modeling software, such as Blender or Maya, uses different unwrapping algorithms.



UV Map Examples

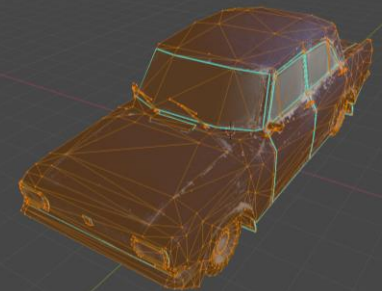
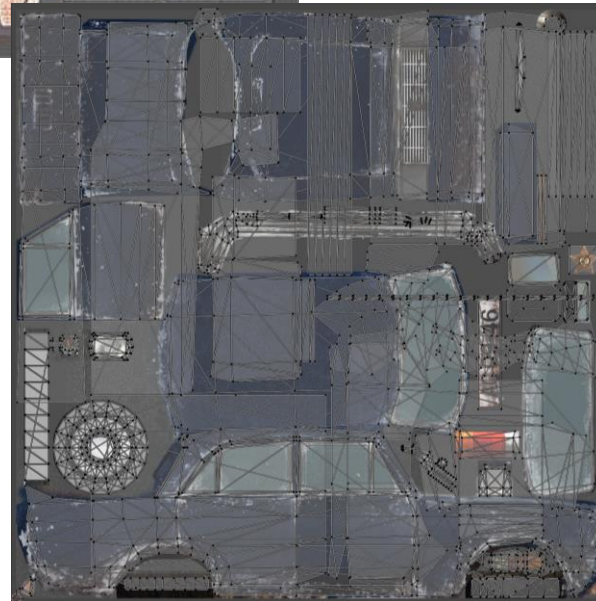
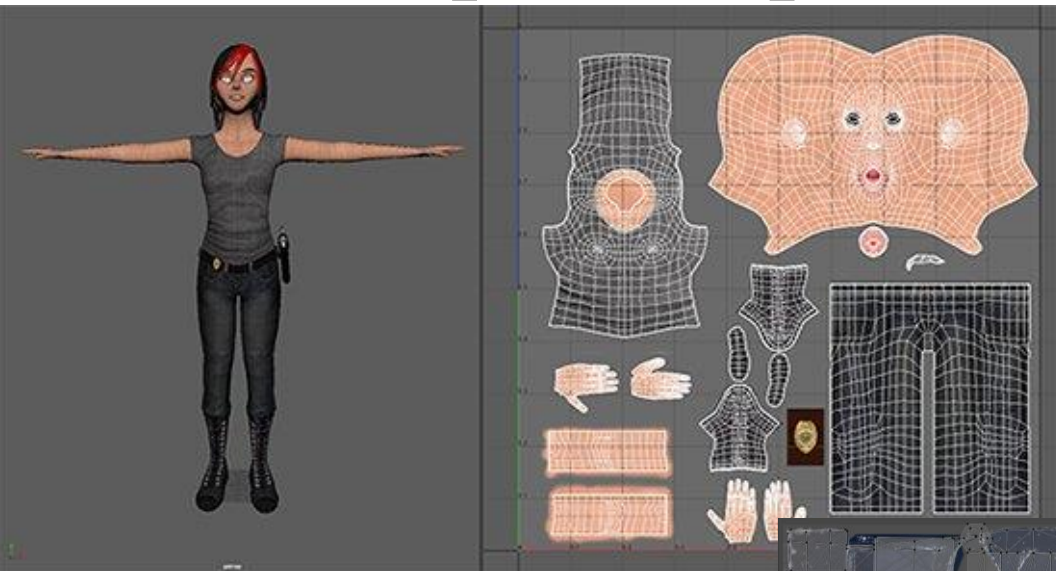


* The images are from

https://learn.foundry.com/nuke/content/comp_environment/modelbuilder/uv_unwrapping.html,

<https://www.blendernation.com/2017/04/22/tutorial-modeling-uv-unwrapping-texturing-mushroom/>

UV Map Examples



* The images are from

<https://help.autodesk.com/view/MAYAUL/2023/ENU/?guid=GUID-FDCD0C68-2496-4405-A785-3AA93E9A3B25>,

Blender rendering of a model downloaded from <https://free3d.com>

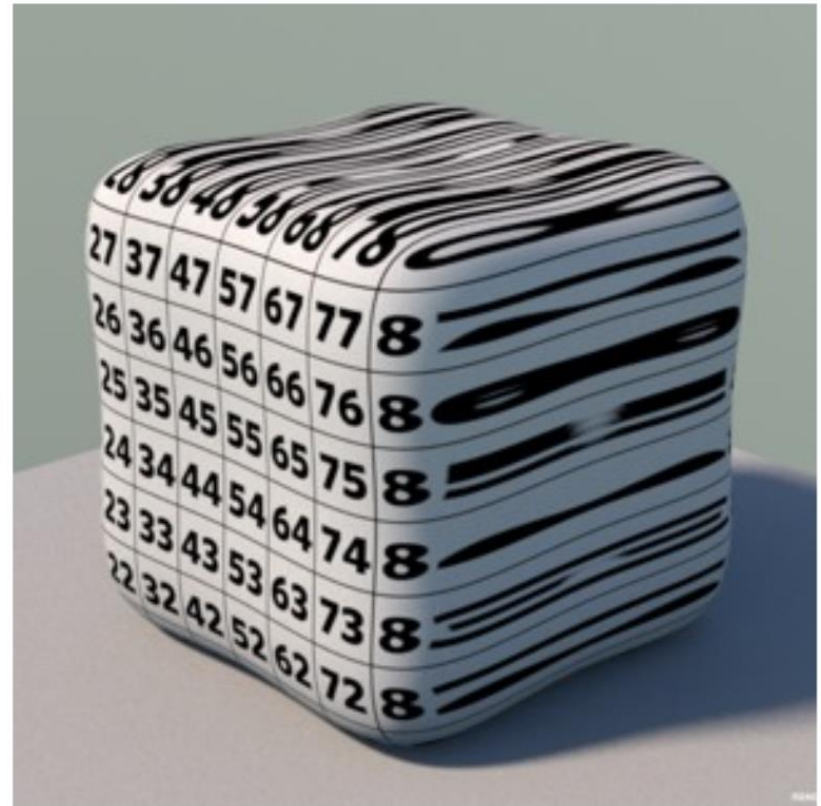
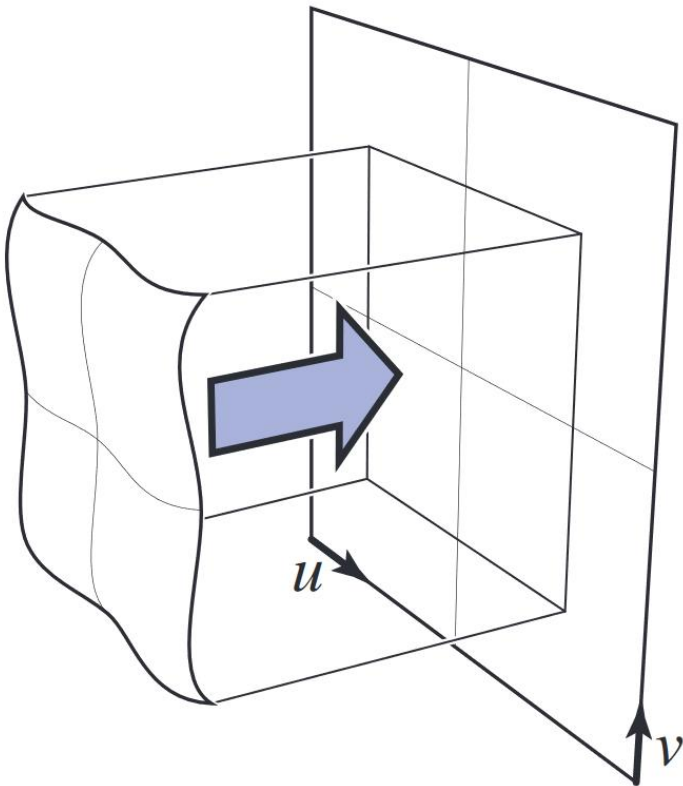
Defining Texture Coordinate Function:

2) Projection-Based Methods

- For objects that resemble simple parametric surfaces (rectangles, spheres, cylinders, etc.),
- Texture coordinates can be generated by projecting object surface points onto a simple parametric surface.
- Common projection methods:
 - Planar projection
 - Cylindrical projection
 - Spherical projection

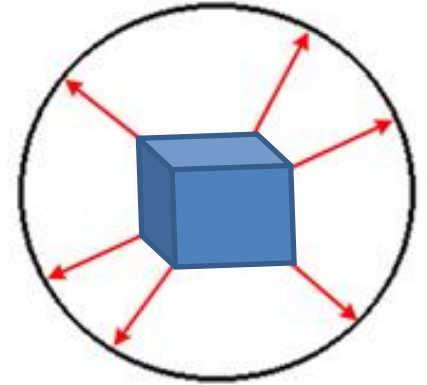
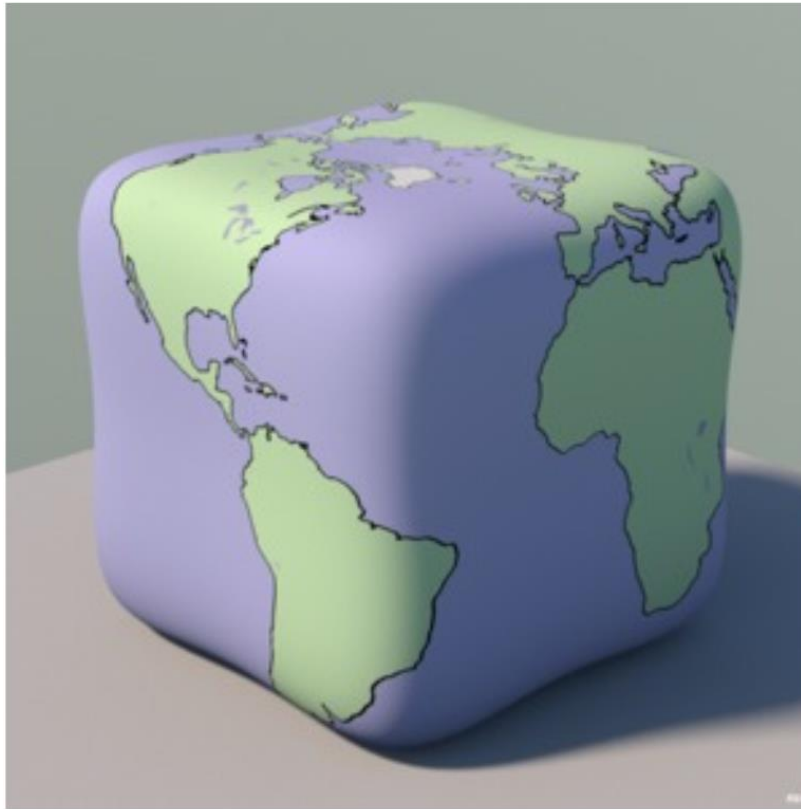
Examples of coordinate functions

- Planar projection



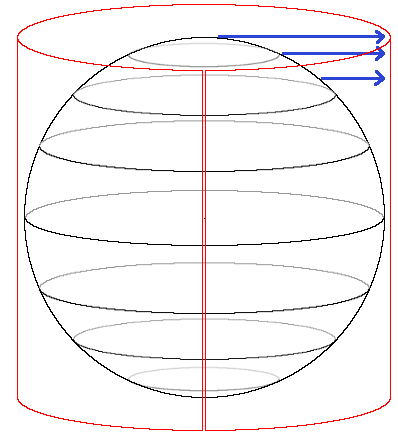
Examples of coordinate functions

- Spherical projection

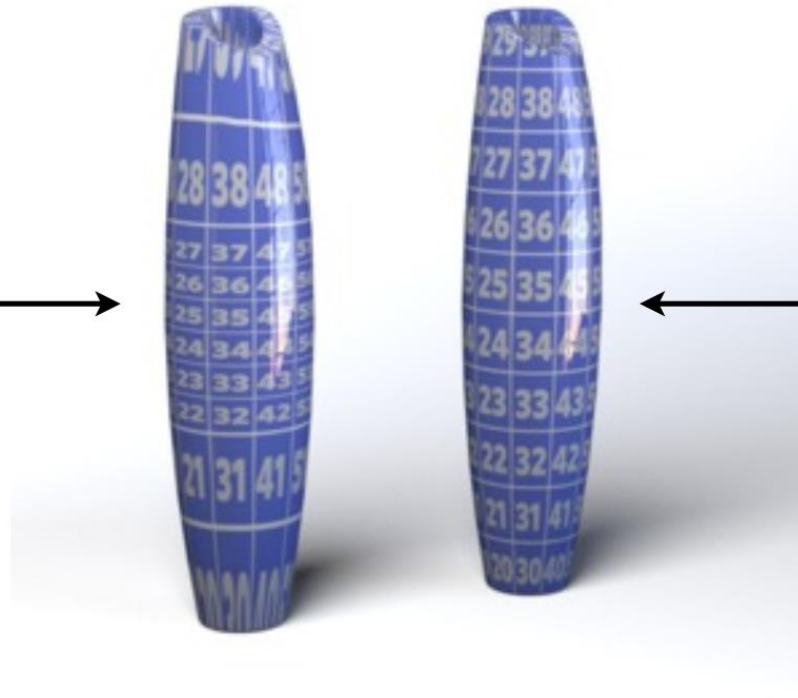


Examples of coordinate functions

- Cylindrical projection



spherical →



← cylindrical

Quiz 2

- Go to <https://www.slido.com/>
- Join #cg-ys
- Click "Polls"
- Submit your answer in the following format:
 - **Student ID: Your answer**
 - e.g. 2024123456: 4.0
- Note that your quiz answer must be submitted **in the above format** to be counted for attendance.

Rendering Texture-Mapped Objects

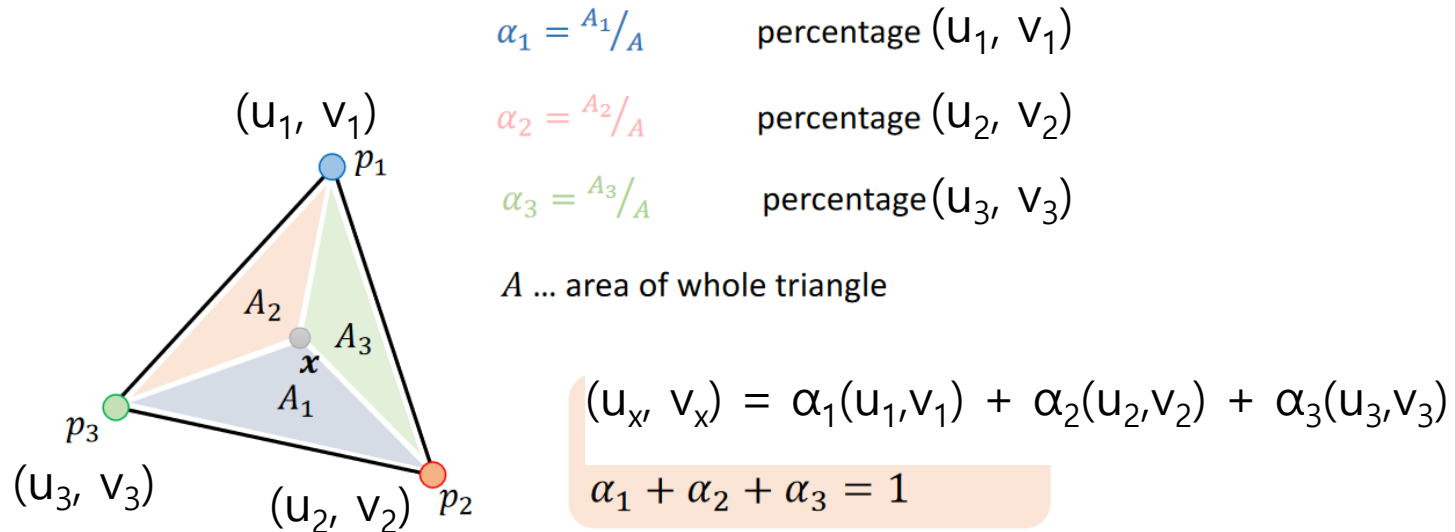
for each fragment:

```
(u, v) = texture coordinates at fragment  
color = texture.get_color(u, v)  
pixel.color = color
```

- However, texture coordinate functions define texture coordinates for each vertex.
- How do we obtain texture coordinates for every fragment?

Rendering Texture-Mapped Objects

- Texture coordinates are barycentrically interpolated across a triangle.
 - (u, v) for each pixel is computed from its barycentric coordinates inside the triangle.



Texture Maps for Different Surface Properties

- Texture maps can store many types of surface properties:
 - **Diffuse color**
 - **Specular color / reflectance**
 - Shininess (specular exponent)
 - Transparency
 - **Surface normals**
 - Lighting information
 - Reflection information

Examples of Diffuse, Specular, Normal Map



diffuse map

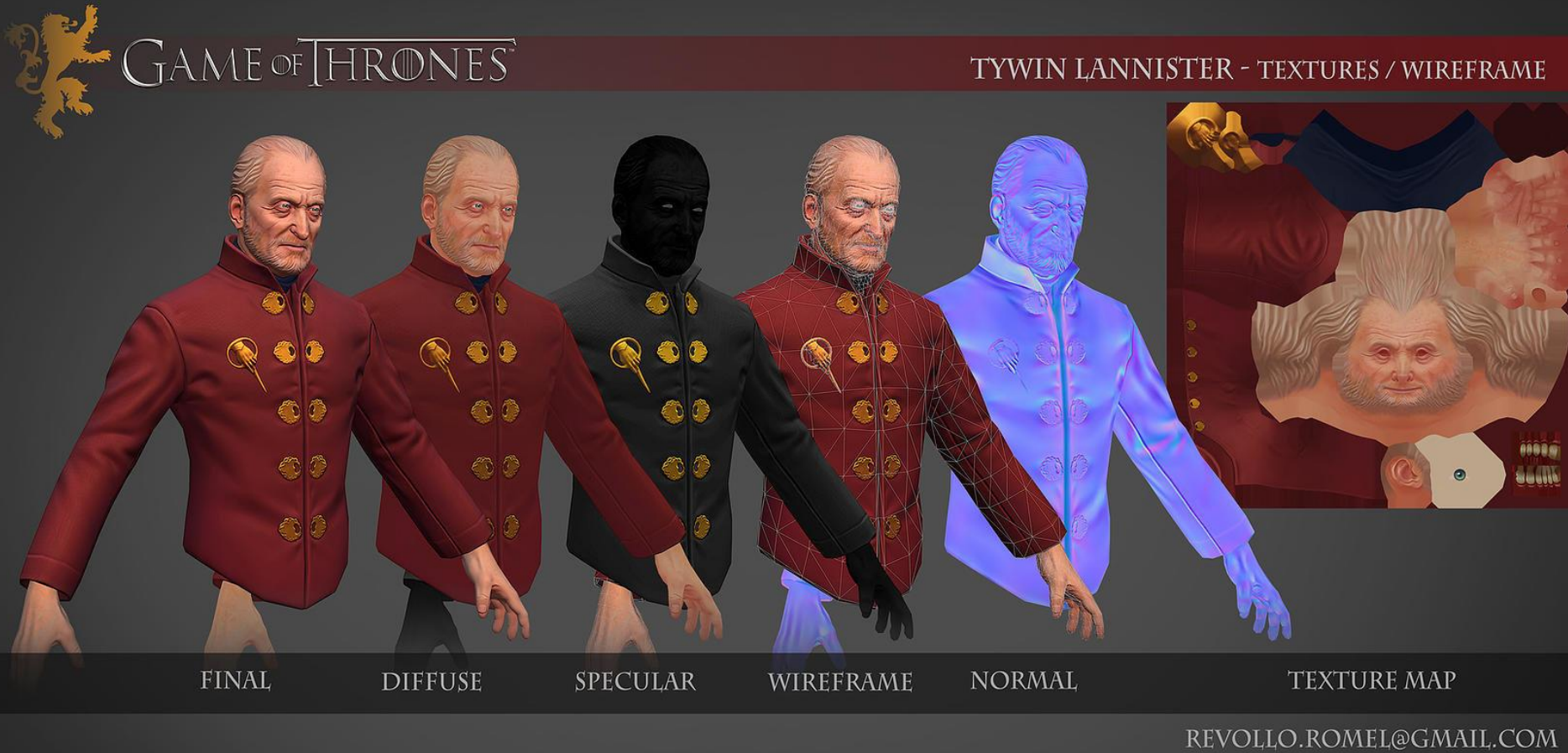


specular map

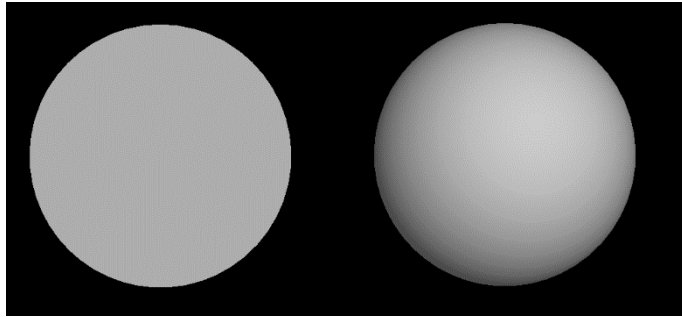


normal map

Examples of Diffuse, Specular, Normal Map

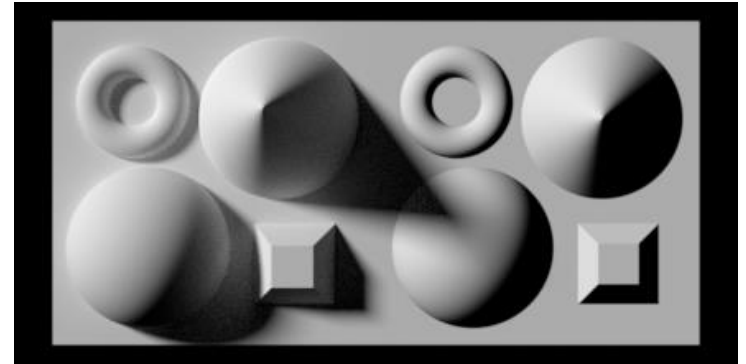


Normal Mapping - Motivation



circle

sphere

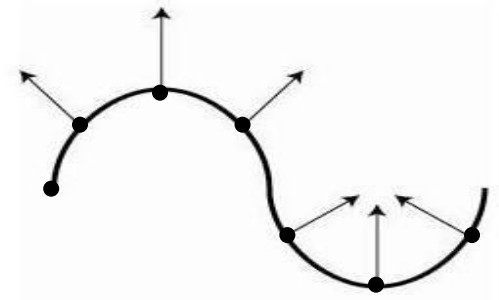


https://commons.wikimedia.org/wiki/File:Rendering_with_normal_mapping.gif

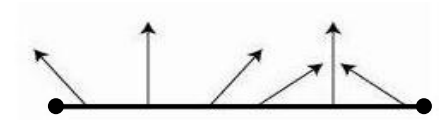
- Rendering a circle would be faster than a sphere, because a circle would have fewer triangles.
- The circle can appear like a sphere if it has the sphere's normal vectors!
- Because the human visual system infers the shape based on the patterns of **light and dark areas**, a process known as "**shape from shading**", and this is **determined by the normal** rather than the actual geometry of the model.

Normal Mapping - Basic Idea

- Goal: Getting a low-poly model that "looks like" a high-poly model.
- 1. Encode the detailed surface normals of a high-poly model into a texture image.
- 2. Use the encoded normals from the texture during lighting calculations on a low-poly model.



Original high-poly model

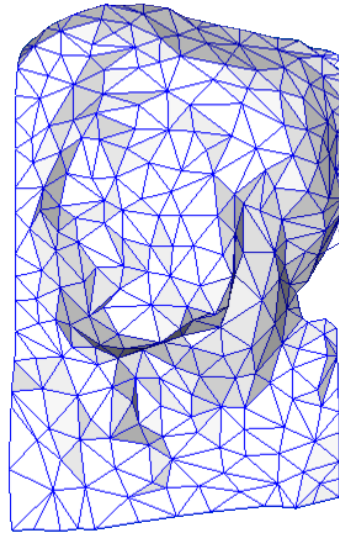


Low-poly model with high-poly model's normals preserved

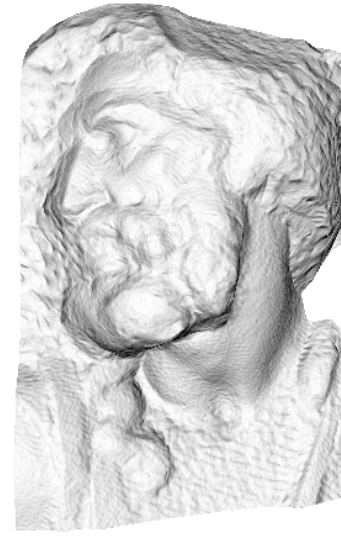
Normal Mapping - Example



original mesh
4M triangles



simplified mesh
500 triangles



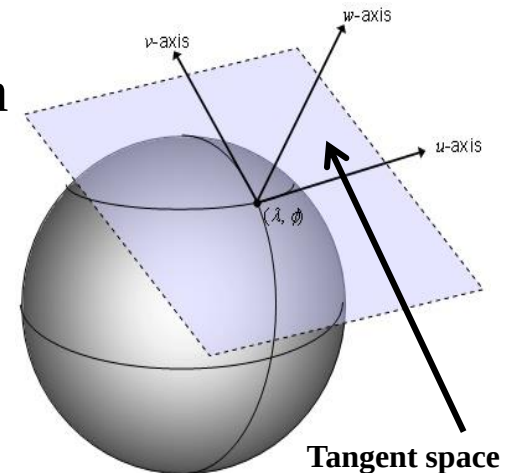
simplified mesh
and normal mapping
500 triangles

* The image is from the slides of Prof. Andy van Dam (Brown Univ.)

<http://cs.brown.edu/courses/csci1230/lectures.shtml>

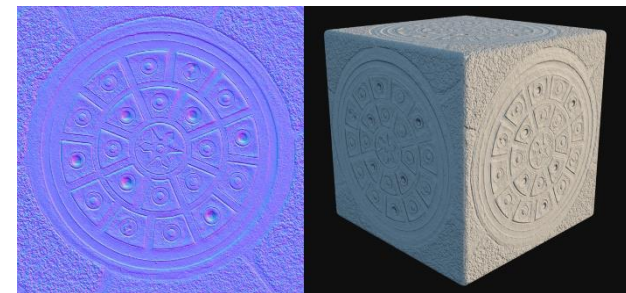
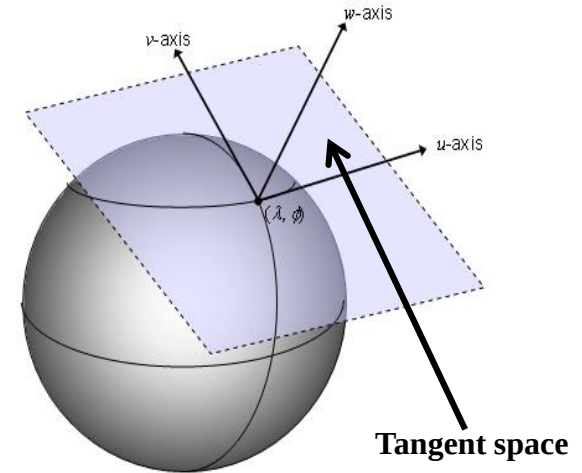
Normal Mapping - Details

- Specifically, normal vectors of a high-poly model are encoded into RGB values of each pixel in a texture map.
- In *tangent space normal map*, RGB values of each pixel indicates the u, v, w component of the normal vector at the corresponding location on an object surface.
- i.e., $R = N_U$, $G = N_V$, $B = N_W$
 - u: u-axis of texture coordinates
 - v: v-axis of texture coordinates
 - w: surface normal at that surface point
 - Refer <http://foundationsofgameenginedev.com/FGED2-sample.pdf> for the computation of u and v vector



Normal Mapping - Details

- $R = N_U$, $G = N_V$, $B = N_W$
 - U: -1 to +1 $\rightarrow$ Red: 0 to 255
 - V: -1 to +1 $\rightarrow$ Green: 0 to 255
 - W: 0 to +1 $\rightarrow$ Blue: 128 to 255
- That's why a tangent space normal map is usually *bluish*.
 - Normals point outward, usually around (0, 0, 1) in tangent space, which is (128, 128, 255) in RGB space.

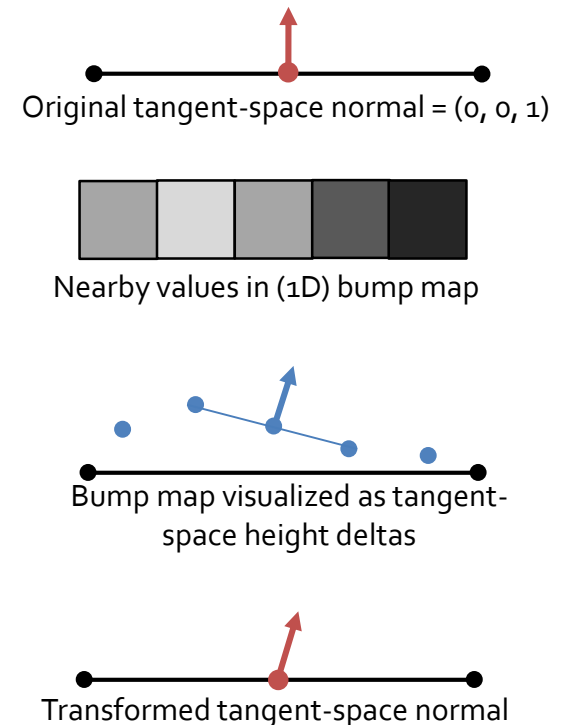


Quiz 3

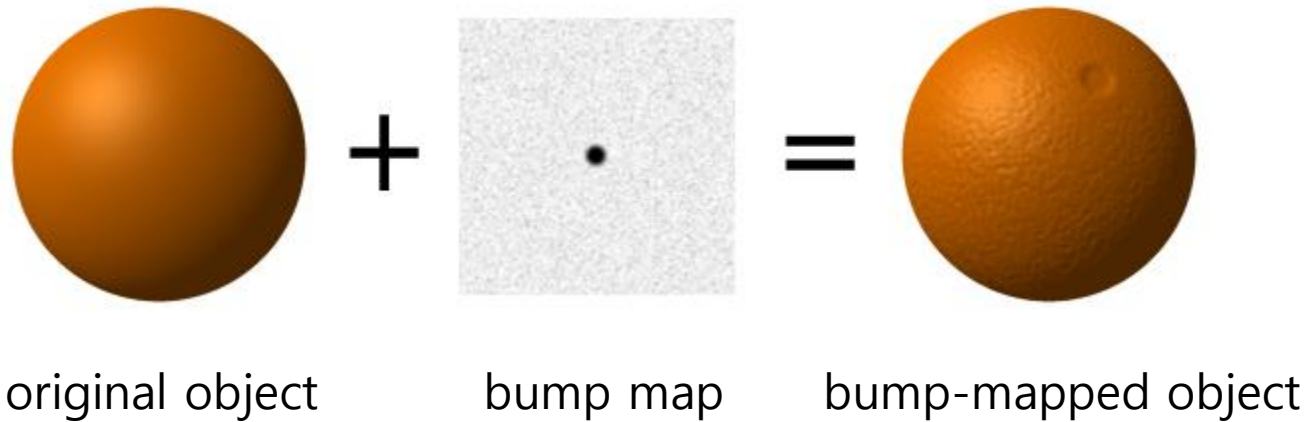
- Go to <https://www.slido.com/>
- Join #cg-ys
- Click "Polls"
- Submit your answer in the following format:
 - **Student ID: Your answer**
 - e.g. 2024123456: 4.0
- Note that your quiz answer must be submitted **in the above format** to be counted for attendance.

Bump Mapping

- Same goal as normal mapping.
- Idea: Instead of encoding normal vectors in a texture map, encode **relative heights**.
 - Black: minimum height delta
 - White: maximum height delta
- Normals are computed from the height map, and then applied.

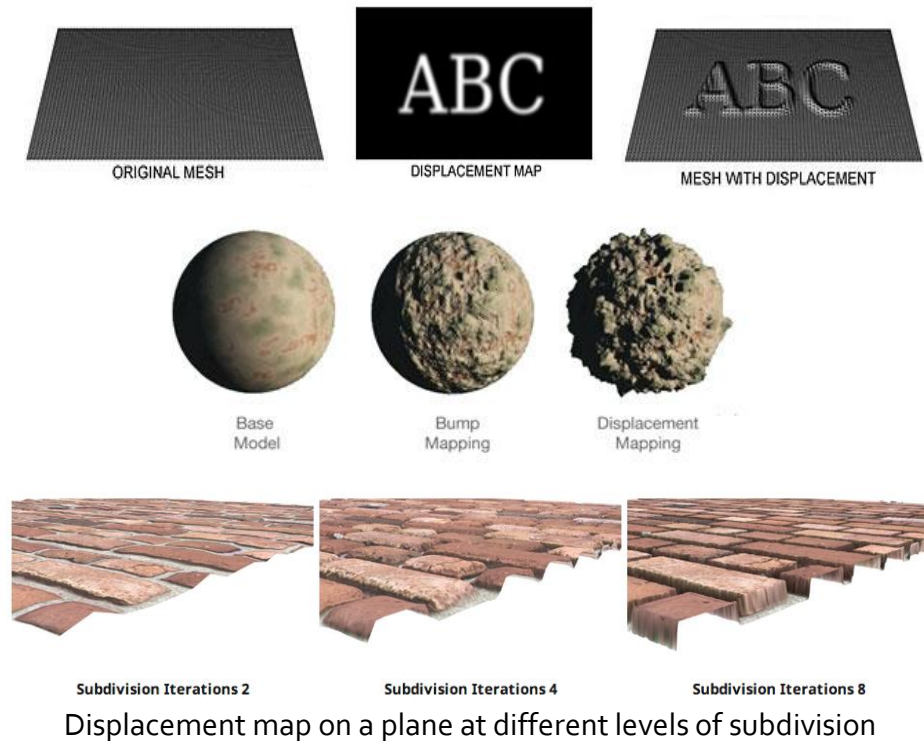


Bump Mapping Example



Displacement Mapping

- *Actually move the vertices* by looking up height deltas in a height map.
 - Unlike bump/normal mapping, it produces **correct silhouettes** and self-shadowing.
- By default, it does not provide detail between vertices like normal/bump mapping.
 - To increase detail, we have to add more vertices, thus it can become very costly.



http://en.wikipedia.org/wiki/Displacement_mapping
<http://www.nvidia.com/object/tessellation.html>
<https://support.solidangle.com/display/AFMUG/Displacement>

[Demo] Normal / Bump / Displacement Mapping

- Normal mapping / Displacement mapping
 - https://threejs.org/examples/#webgl_materials_displacementmap
- Bump mapping
 - https://threejs.org/examples/#webgl_materials_bumpmap

Light Map

- Light maps are used to store pre-computed illumination

Textures Only



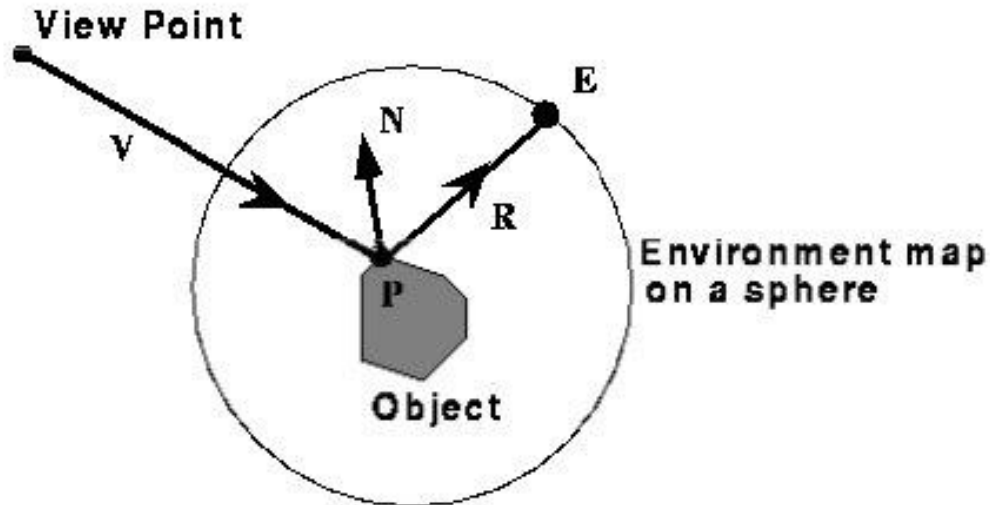
Textures & Light Maps



Light map
image by Nick
Chirkov

Environment Map

- **Simulate complex mirror-like objects**
 - Use textures to capture environment of objects
 - Use surface normal to compute texture coordinates



* This slide is from the slides of Prof. Sung-eui Yoon (KAIST)
<https://sglab.kaist.ac.kr/~sungeui/CG/>

[Practice] Online Demos

- Light mapping
 - https://threejs.org/examples/?q=lightmap#webgpu_materials_lightmap
- Environment mapping
 - https://threejs.org/examples/#webgl_materials_cubemap_dynamic
 - https://threejs.org/examples/?q=refrac#webgl_materials_cubemap_refraction

Lab Session

- Now, let's start the lab today.