Shading

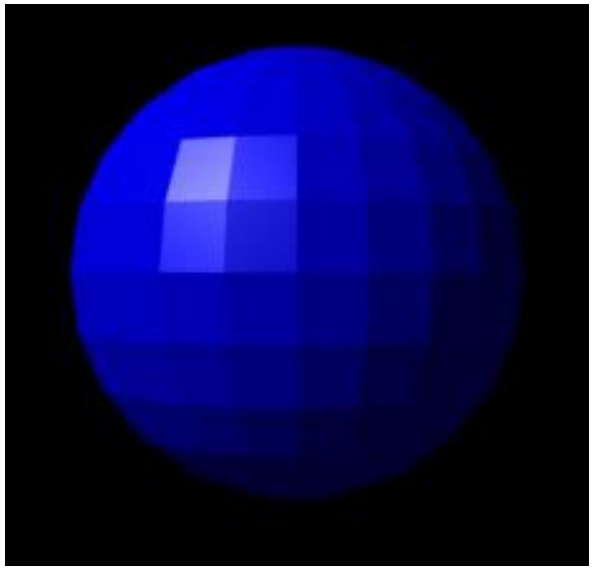
CSU44052 Computer Graphics

Binh-Son Hua

Shading

- Flat shading
  - Compute illumination once per polygon and apply it to whole polygon (per vertex with no interpolation)
- Gouraud shading
  - Compute illumination at borders and interpolate (per vertex)
- Phong shading
  - Compute illumination at every point of the polygon (per fragment)

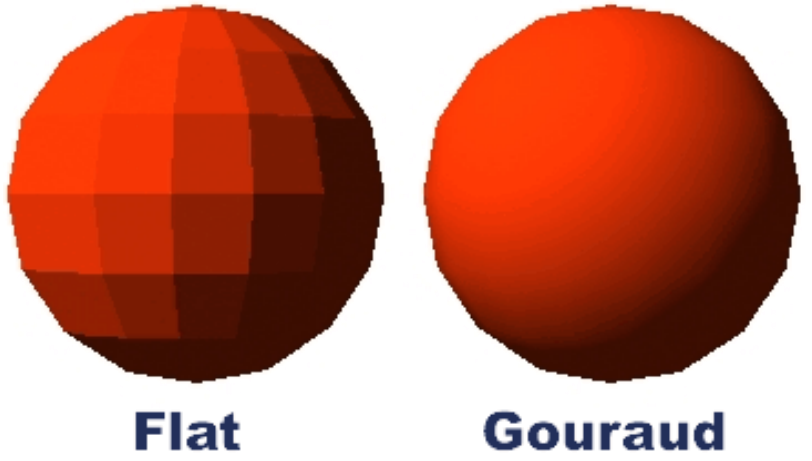
Flat Shading



- Illumination model is applied only once per polygon
- Gives low-polygon models a faceted look.
- Works poorly if the model represents a curved surface.
- Smooth appearance implies large number of polygons.
- Adding more facets helps but this slows down the rendering.
- Advantageous in modeling boxy objects.

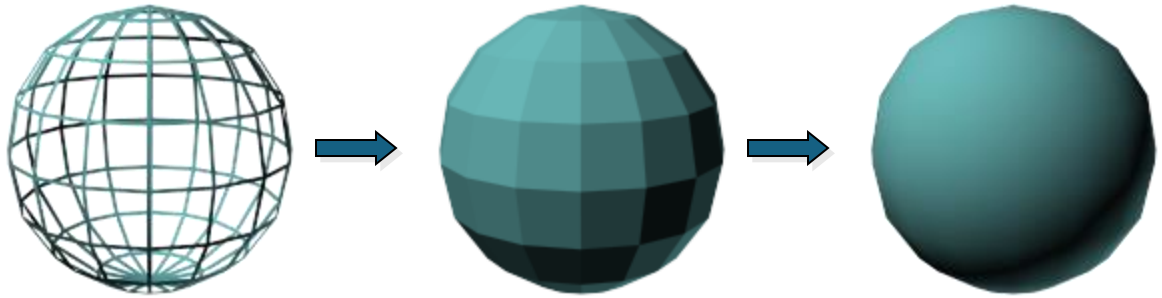
Smooth Shading

- Used to approximate curved surfaces with a collection of polygons.
- Calculate illumination based on approximation of curved surface.
- Does not change geometry, silhouette is still polygonal.



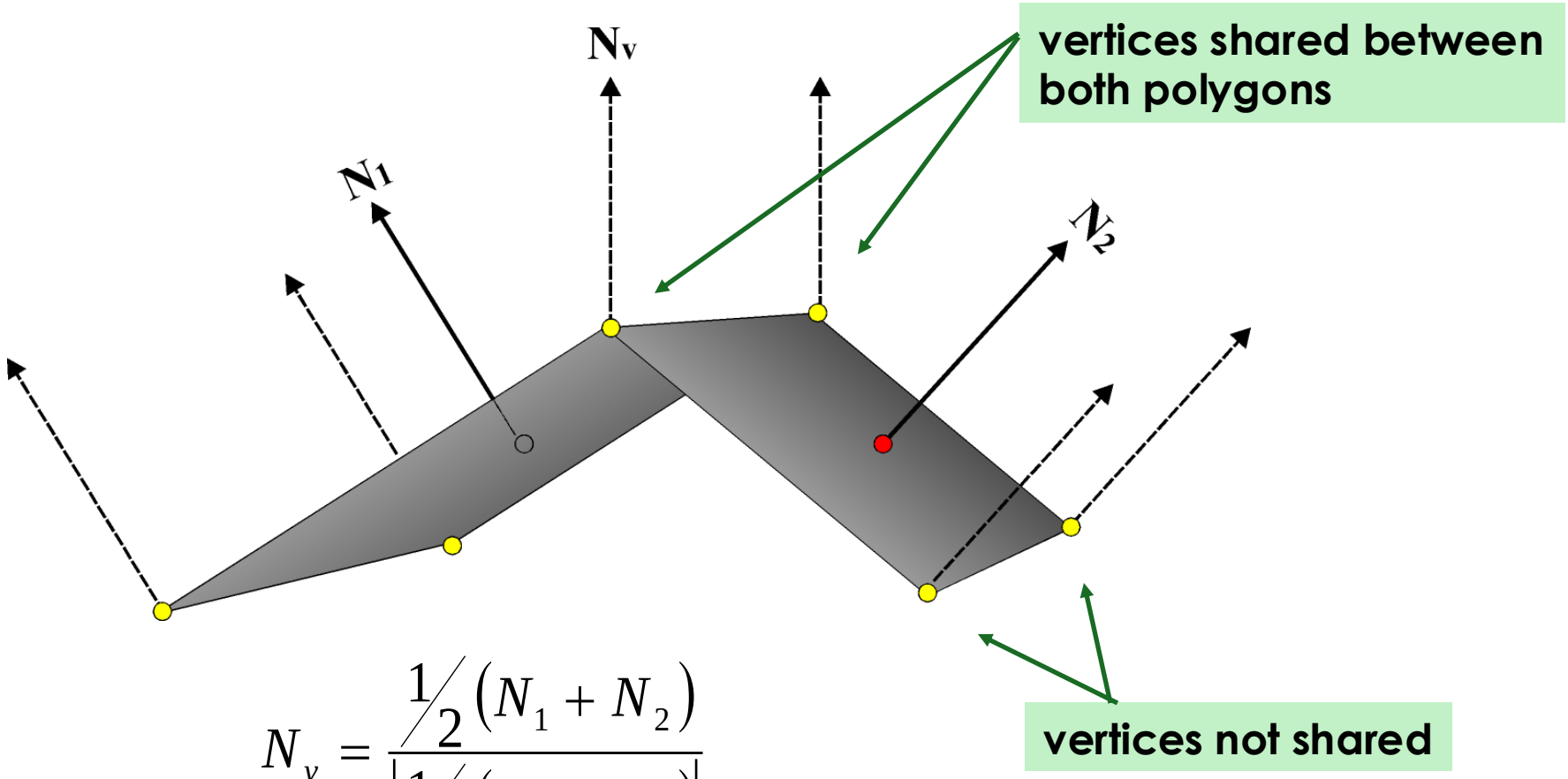
Surface Normal Interpolation

- Often we are approximating curved/smooth surfaces with polygons ⇒ we get *edge artifacts* at polygon boundaries:



- To combat this we determine average normals at the vertices of the polygons by averaging the normals of each polygon that shares a vertex and storing the result with that vertex.

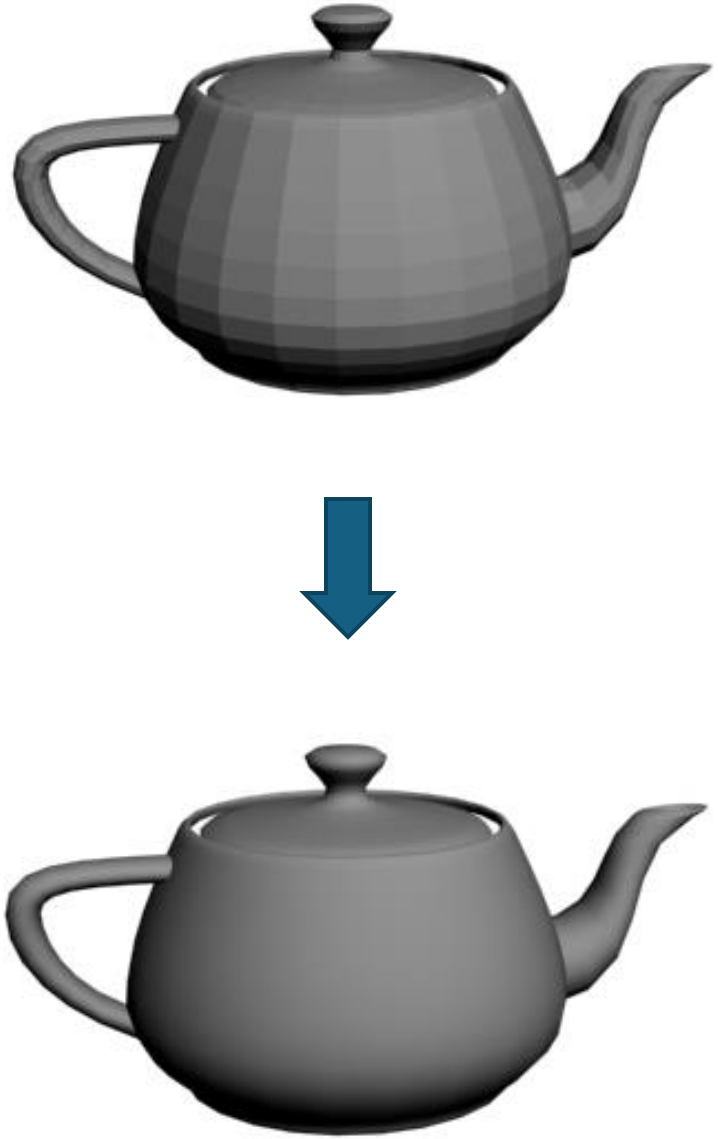
Determining Vertex Normals



$$N_v = \frac{\frac{1}{2}(N_1 + N_2)}{\left|\frac{1}{2}(N_1 + N_2)\right|}$$

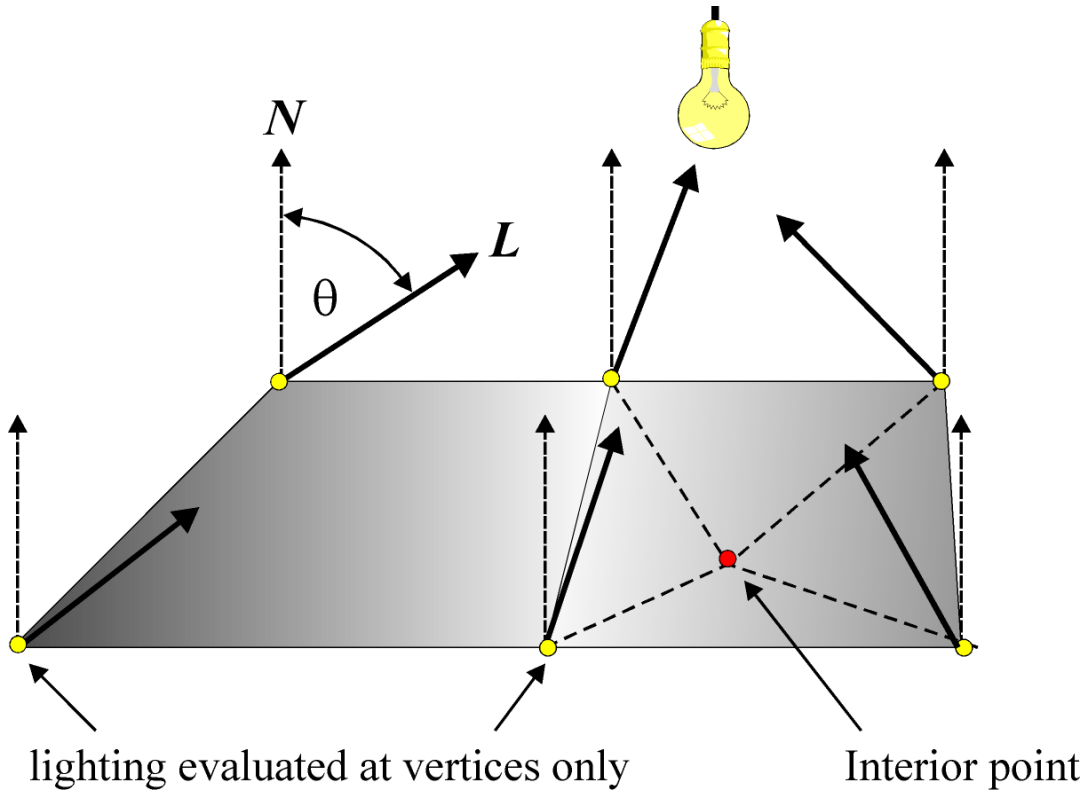
Gouraud Shading

- Gouraud shading is a method for linearly interpolating a colour or shade across a polygon.
- Invented by Henri Gouraud in 1971.
- It is a very simple and effective method of adding a curved feel to a polygon that would otherwise appear flat.

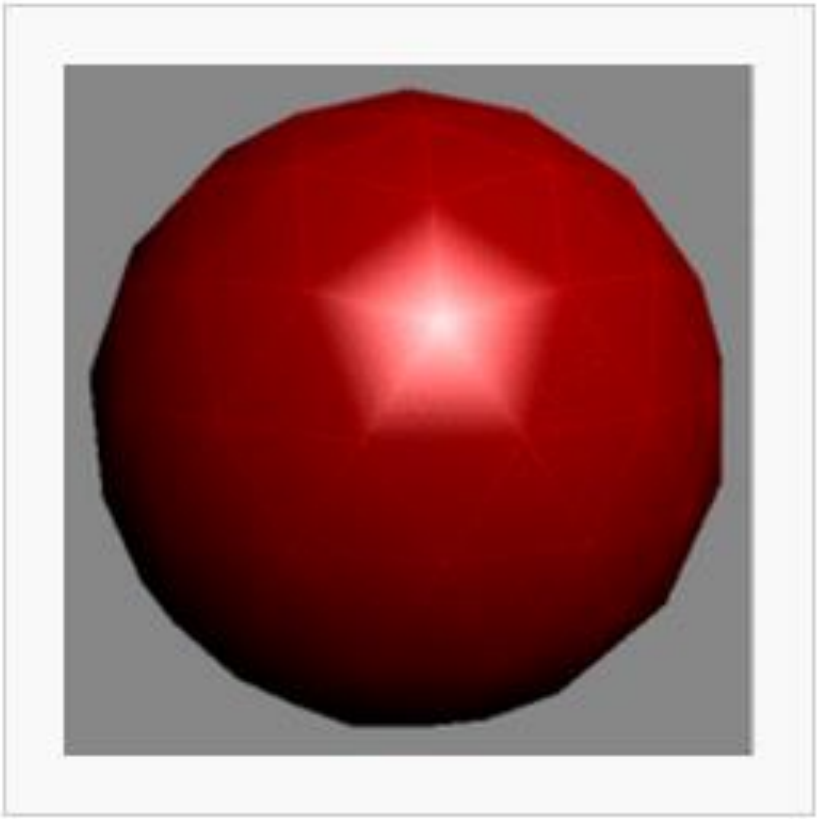


Gouraud Shading

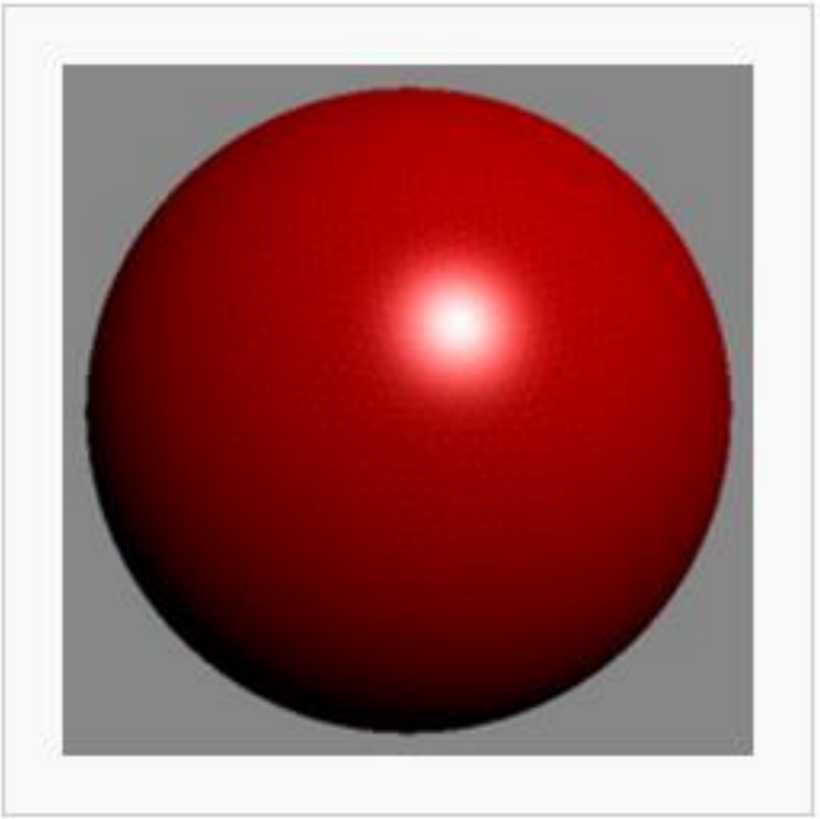
- We need to do lighting calculations at vertices only.
- For each interior point in the polygon being shaded we interpolate the intensity determined at the vertices.
- We do this in exactly the same way that we interpolated colour across the surface of a polygon.
- Lighting is correct at vertices only.
  As polygons increase in size, lighting errors also increase, leading to less accurrate lighting.



Interpolation Errors



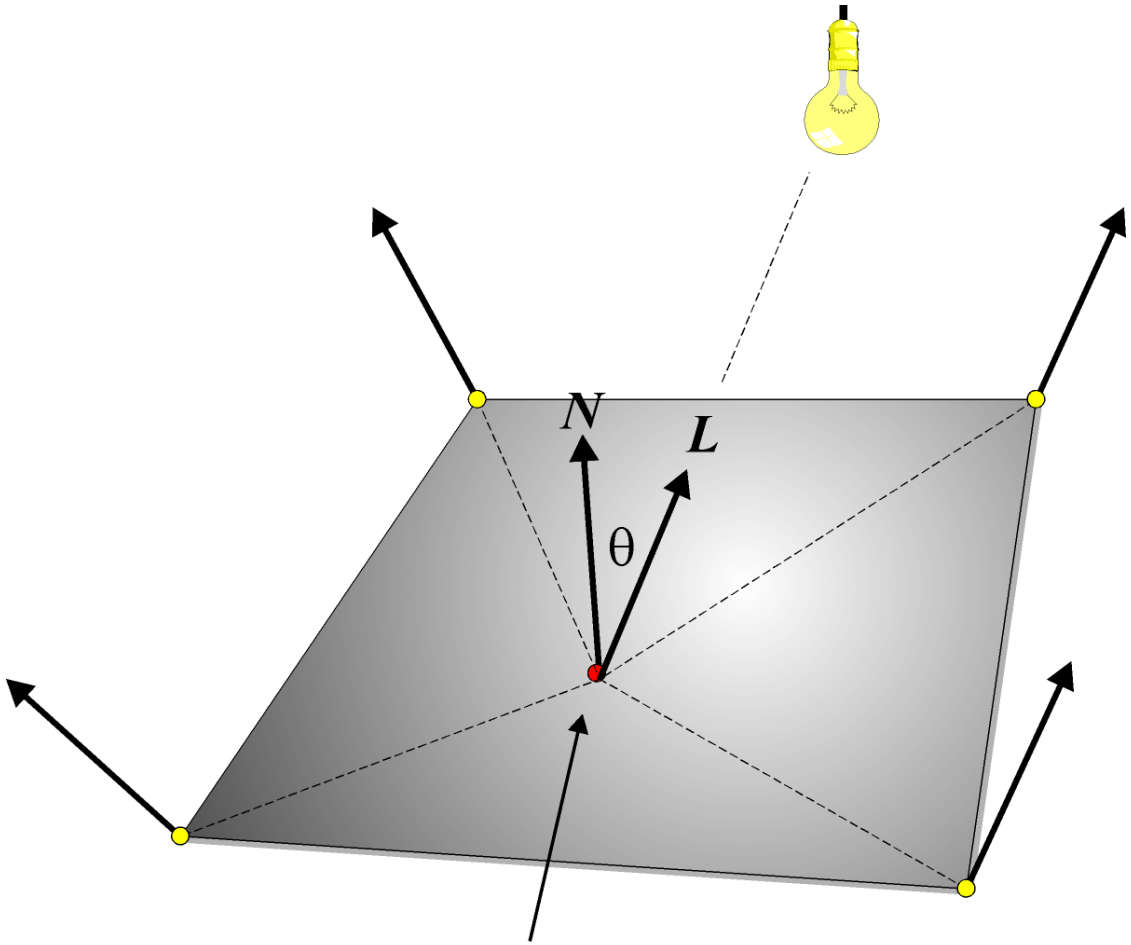
Poor behaviour of the specular highlight



Improvement with very high polygon count

Phong Shading

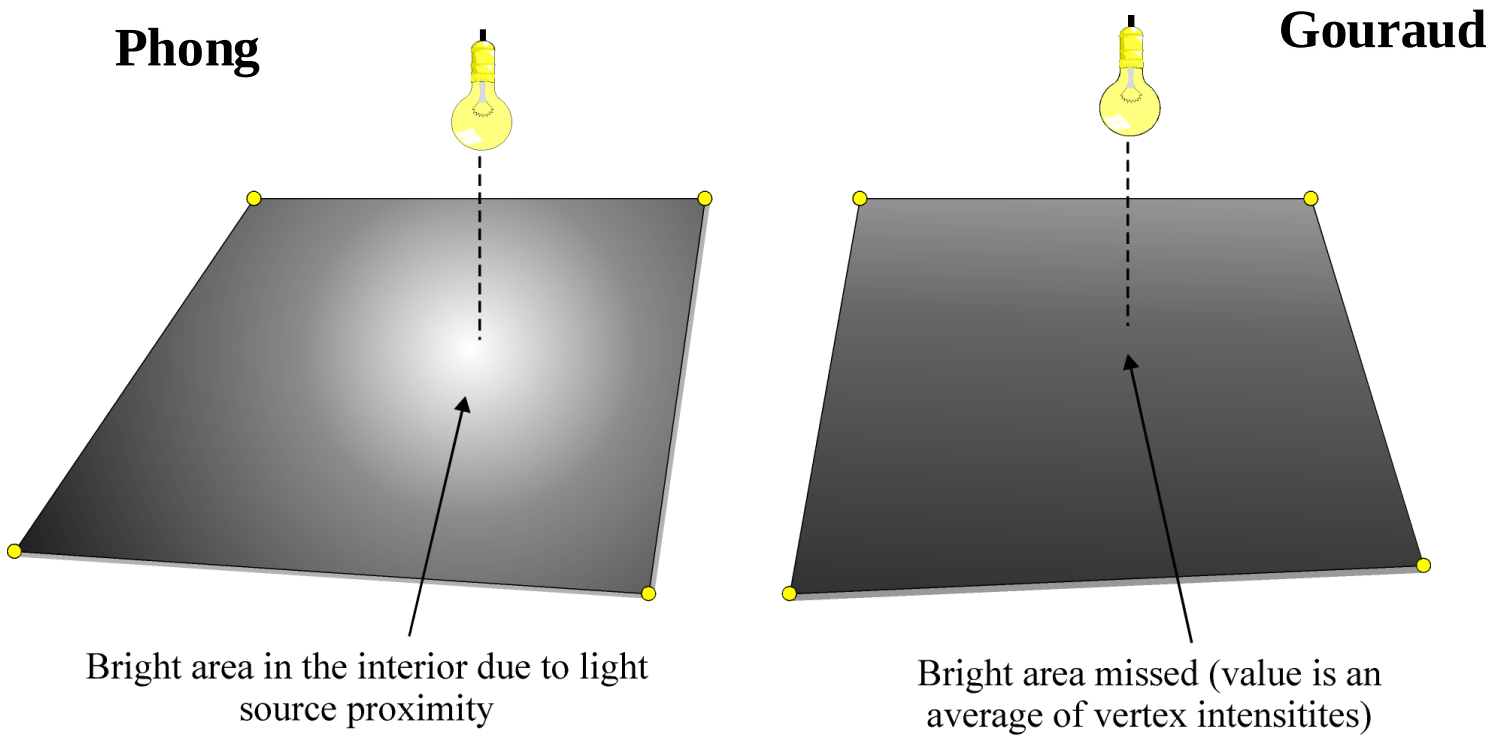
- To improve upon Gouraud shading we can *interpolate the normals* across the surface and apply the lighting model at each point in the interior.
- This assumes we are working with polygonal models.
- Care must be taken to ensure that all interpolated normals are of unit length before employing the lighting model.
- This is known as *Phong shading* (as opposed to the *Phong illumination model*).



Interpolated normal at the interior point

Phong Shading

- Phong shading is capable of reproducing highlights within the interior of a polygon that Gouraud shading will miss:



Which shading was used?

Illumination and Shading

- Illumination: simulating light reflectance, absorption, and transmission.

- Shading: determines how to render the faces of each polygon in the scene, given illumination.

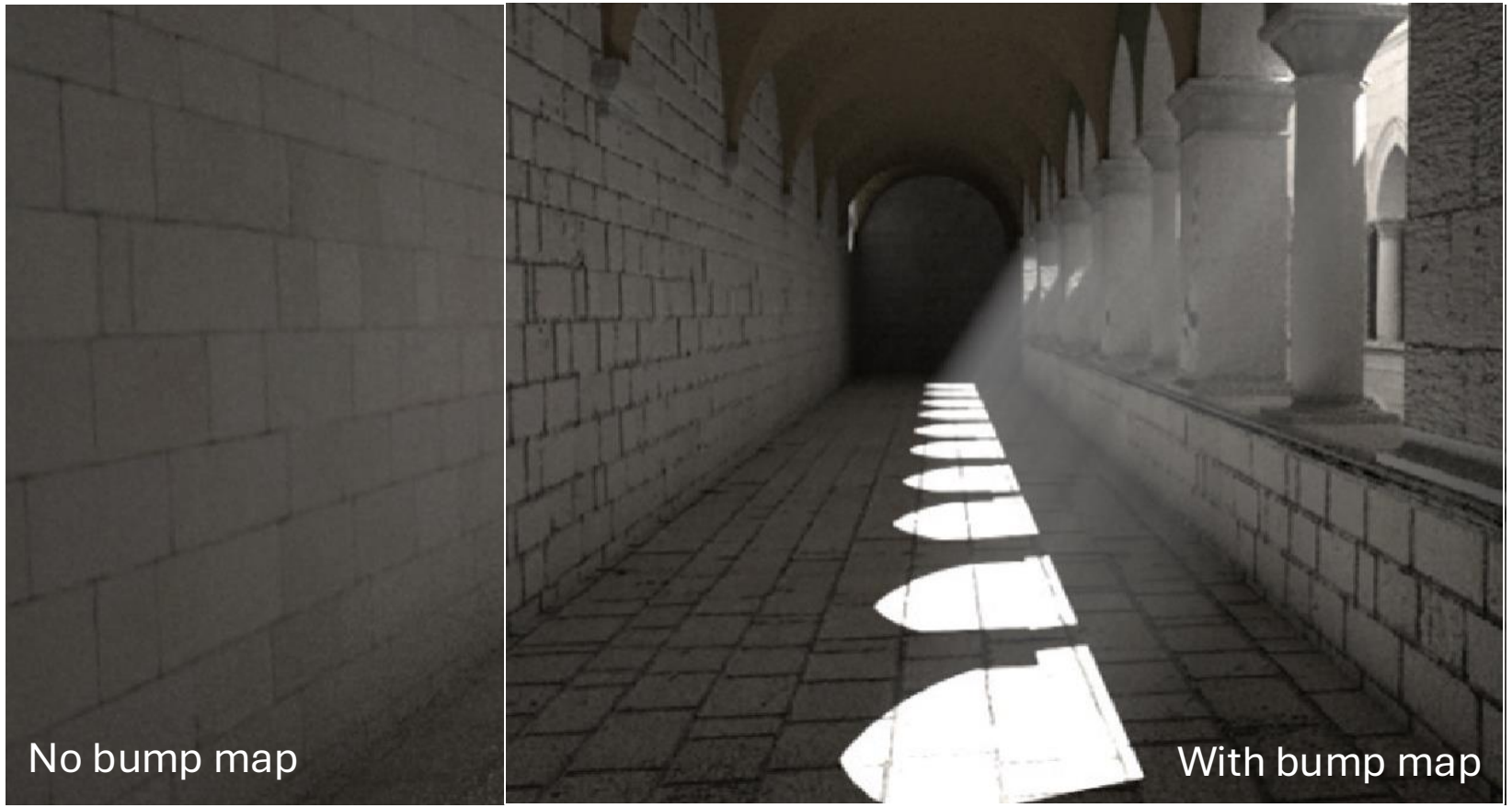
- The shading model depends on the illumination model:
  - Some shading models invoke an illumination model for every pixel.

  - Others only use the illumination model for some pixels and then shade the remaining pixels by interpolation (such as Gouraud shading).

Shading Techniques

Bump Maps

- Texture mapping can be used to model rough surfaces. Real surfaces are not flat but often rough and bumpy.



Bump Maps

- Bump maps create the illusion of depth on the surface of a model using a very simple lighting trick.
- No additional resolution is added to the model as a result of a bump map.
- Values in a bump map imply height.
  It is used to virtually move a surface point up or down along the normal vector:

$$\boldsymbol{p}' = \boldsymbol{p} + d\,\boldsymbol{n}$$

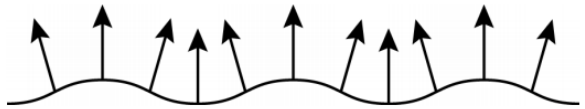
- During shading, we estimate the normal vector from the modified surface point p' and use it for shading.

Normal mapping

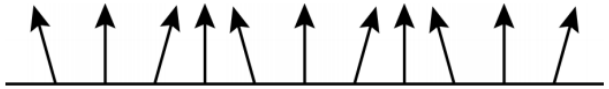
- Normal maps directly stores 2D/3D vectors instead of height values in bump maps.
  - These vectors are combined with the original normals at the points for which we do shading calculations

Flat surface

Real surface



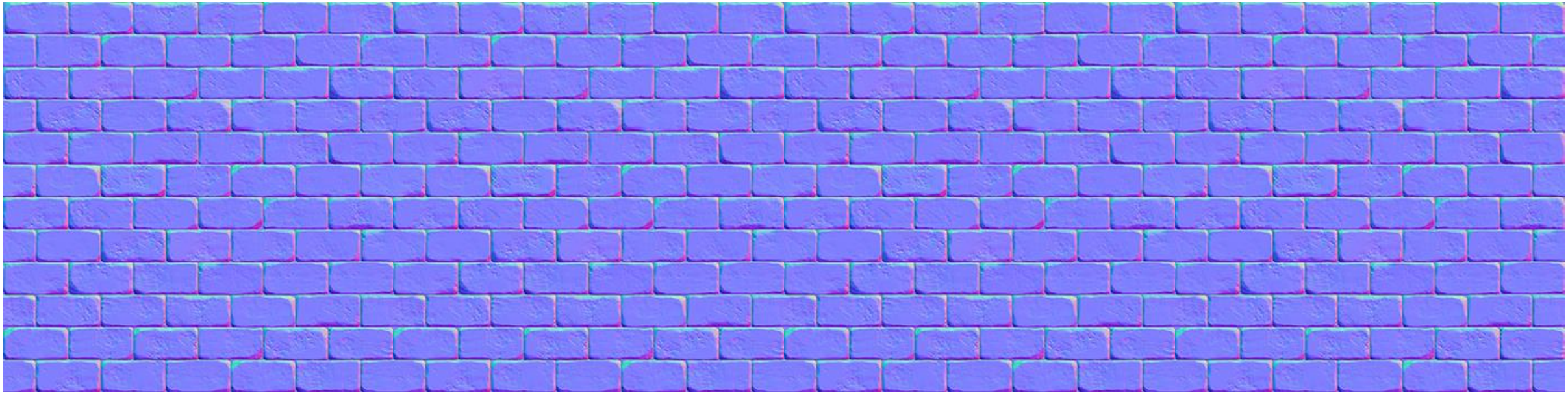
Normal mapped



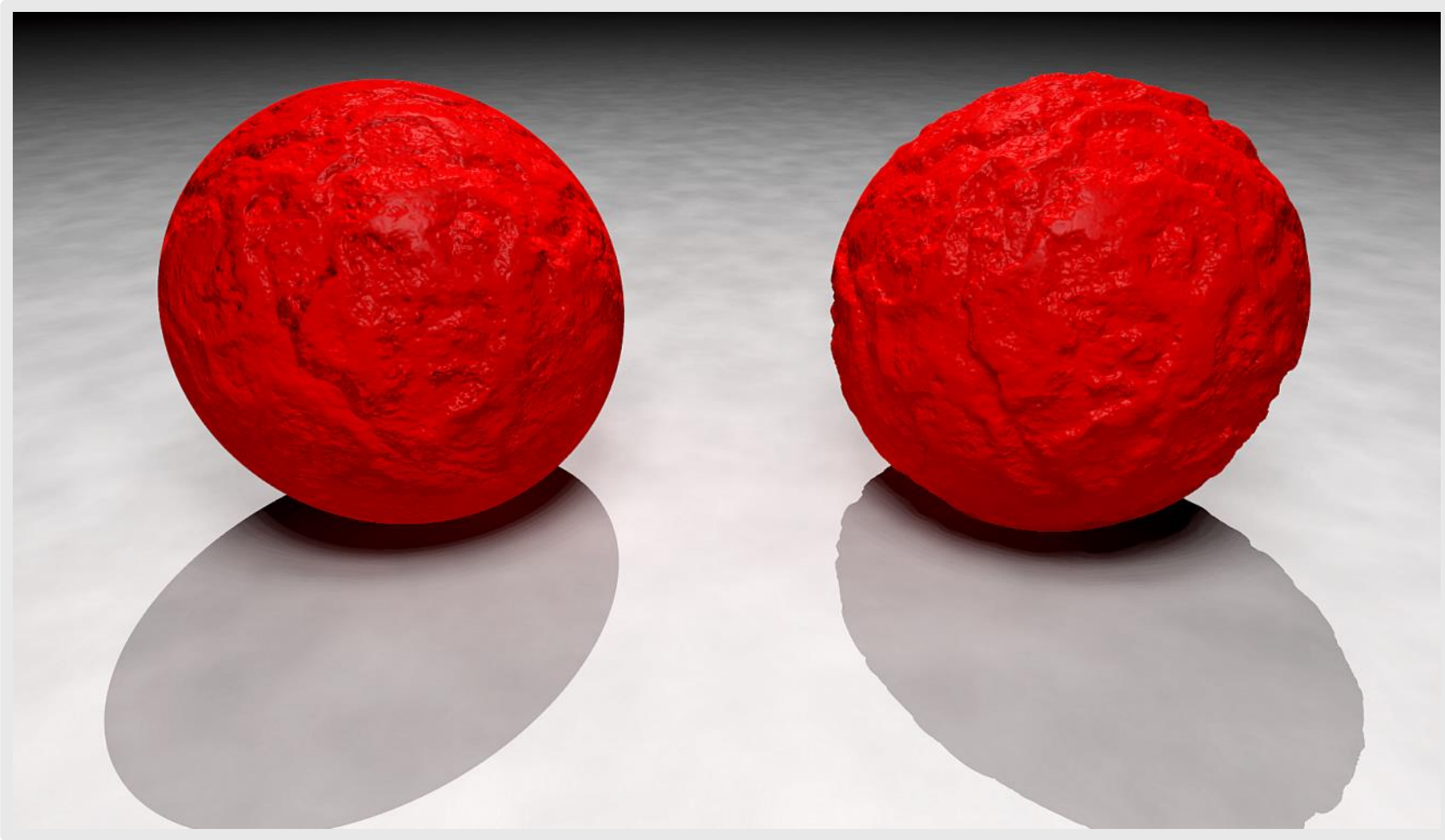
Normal map

Normal mapping

- Implemented by modifying the per-pixel shading routine
- Geometric normal remains the same.
- Why does a normal map tend to have blue/purple colour tone?



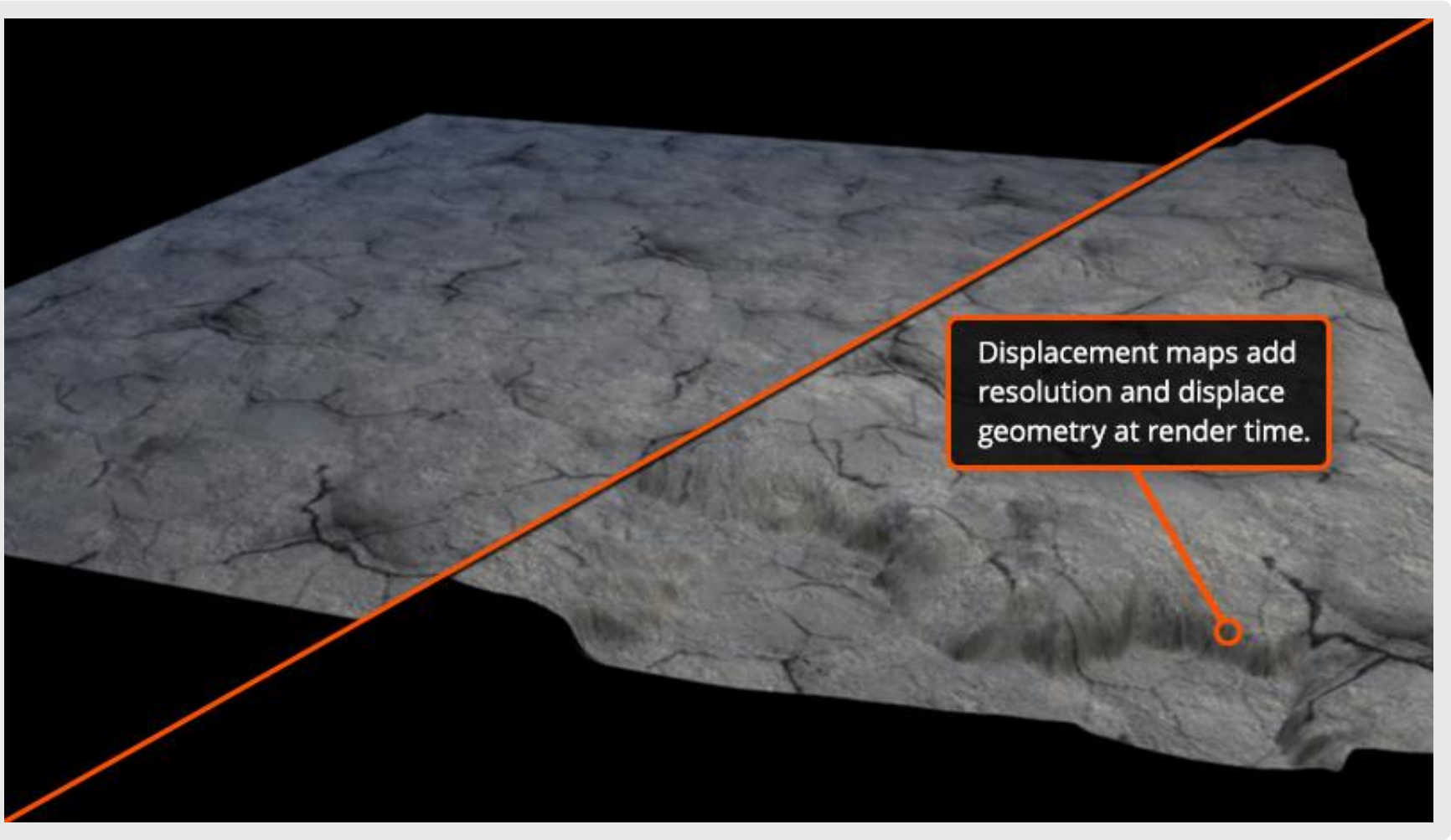
What’s the problem?



Displacement mapping

- To overcome this shortcoming, we can use a displacement map.
  - this is also a 2D or 3D array of vectors, but here the points to be shaded are actually **displaced**.
- Normally, objects are refined using the displacement map, giving an increase in storage requirements

Displacement Maps

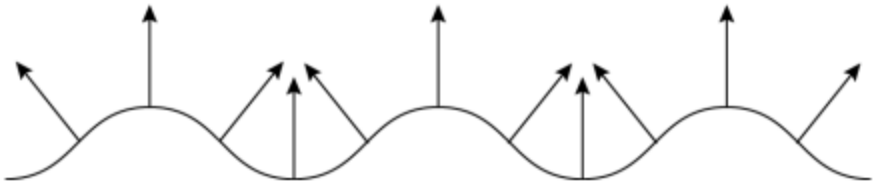


Displacement Mapping

Flat surface

Displacement

Result surface



Problem?

- Collision detection
- Object intersection
- Foot placement for animation

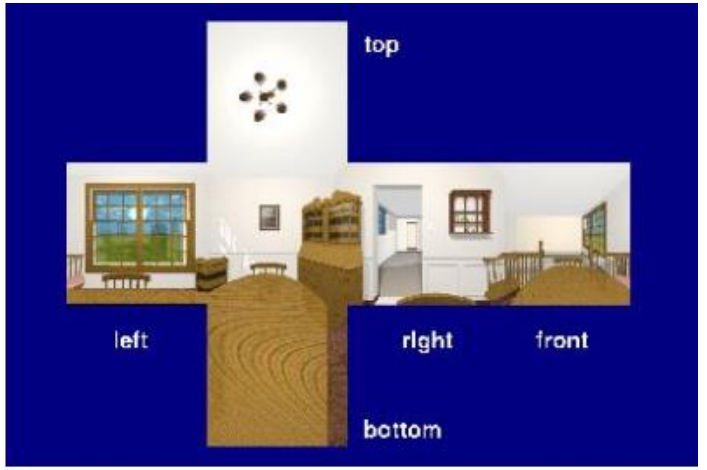
Environment Mapping

- When you look at a highly reflective object such as a chrome sphere, what you see is not the object itself but how the object reflects its environment
- … why not use this to make objects appear to reflect their surroundings specularly?

http://http.developer.nvidia.com/CgTutorial/cg_tutorial_chapter07.html

Environment Mapping

- Idea:
  - place a cube around the object
  - project the environment of the object onto the planes of the cube in a pre-processing stage
  - this is our texture map



- During rendering:
  - compute a reflection vector based on eye position and surface normal,
  - then use the reflection vector to look-up texture values from the cubic texture map.

Environment Mapping

- Ideally, every environment-mapped object in a scene should have its own environment map
  - In practice, objects can often share environment maps with no one noticing
- In theory, you should regenerate an environment map when objects in the environment move, or when the reflective object using the environment map moves significantly relative to the environment
  - In practice, convincing reflections are possible with static environment maps.

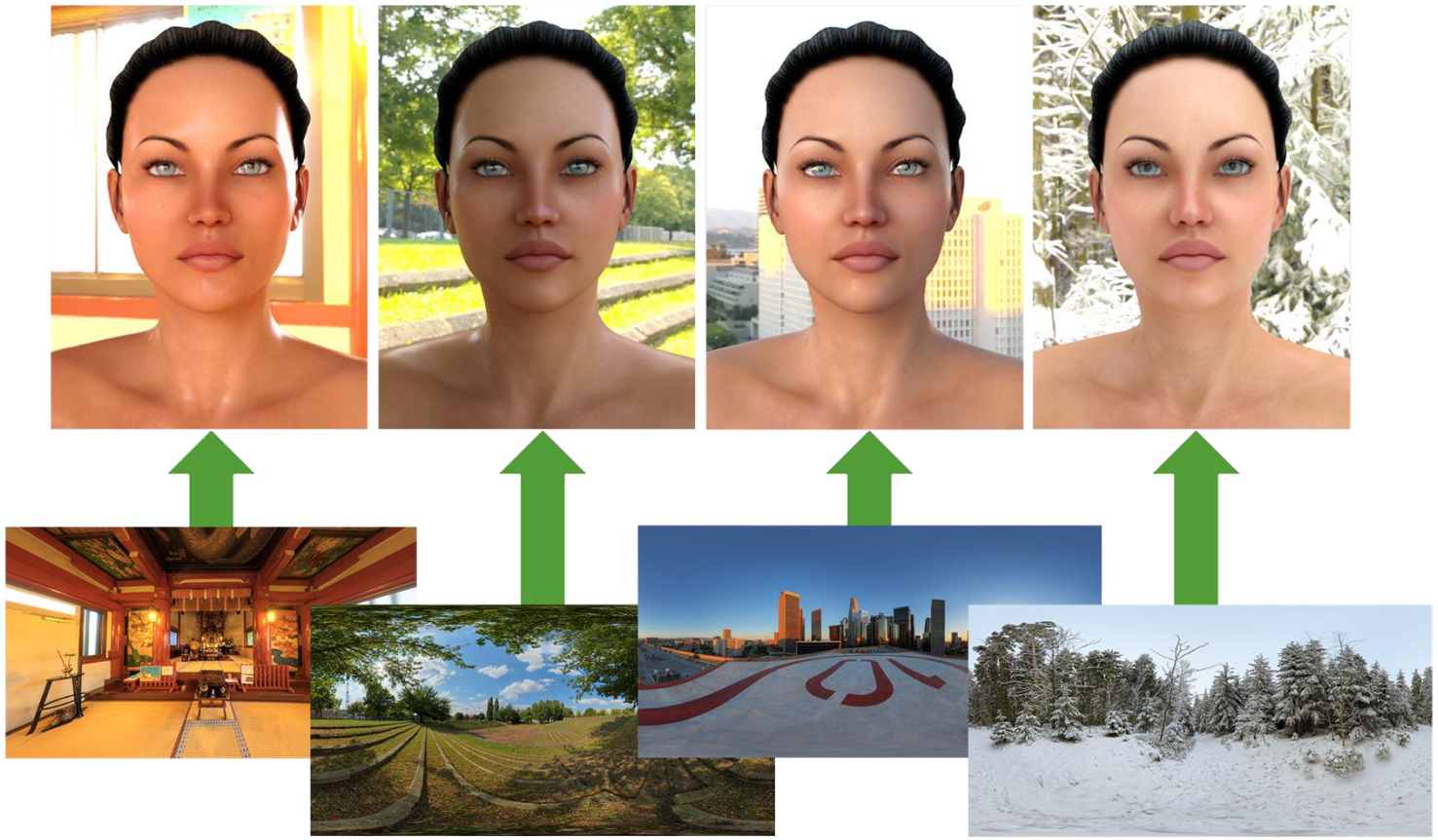
Environment Mapping

- Environment mapping works **best on curved surfaces**.

- It works poorly on flat reflective surfaces such as mirrors, where the reflections depend heavily on position.
  This is because environment mapping depends solely on direction and not on position.

http://www.humus.name/index.php?page=Textures

Environment Lighting



http://docs.daz3d.com/doku.php/public/software/dazstudio/4/new_features/start

Further Reading

- Deferred shading and deferred lighting https://en.wikipedia.org/wiki/Deferred_shading
- Tile-based shading https://www.aortiz.me/2018/12/21/CG.html
- Clustered shading https://github.com/DaveH355/clustered-shading
- Advances in real-time rendering, SIGGRAPH 2024: https://advances.realtimerendering.com/