Introduction to Computer Graphics (CS360A)

Instructor: Soumya Dutta

Department of Computer Science and Engineering

Indian Institute of Technology Kanpur (IITK)

email: soumyad@cse.iitk.ac.in

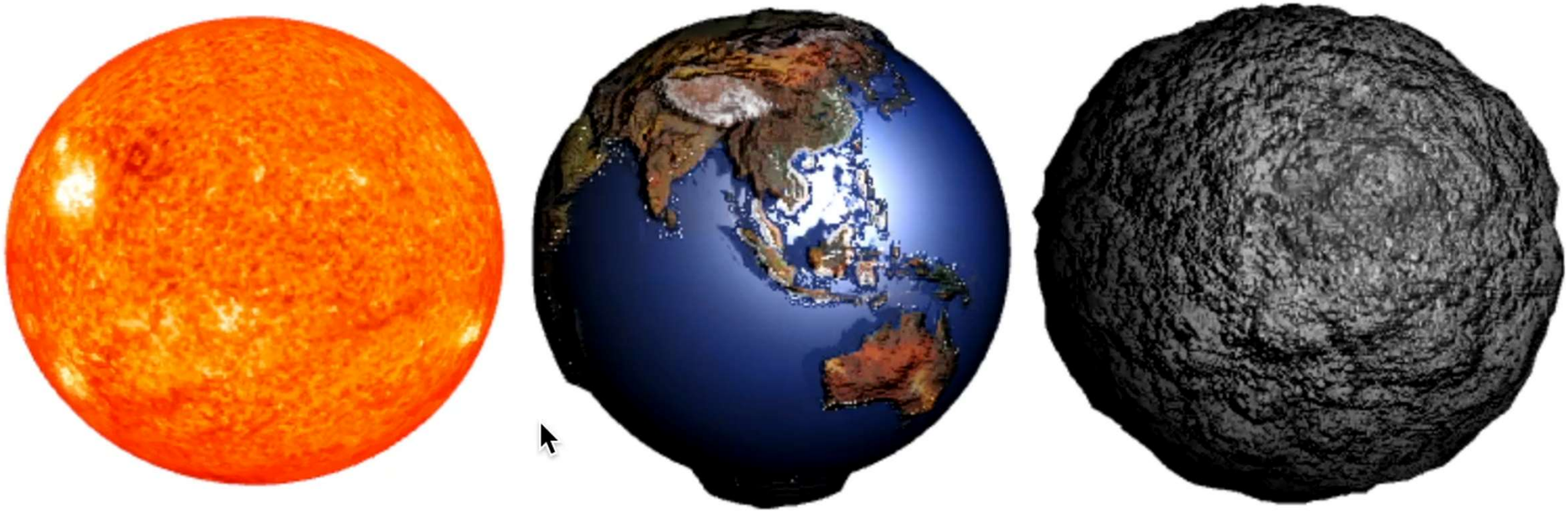
Acknowledgements

- A subset of the slides that I will present throughout the course are adapted/inspired by excellent courses on Computer Graphics offered by Prof. Han-Wei Shen, Prof. Wojciech Matusik, Prof. Frédo Durand, Prof. Abe Davis, and Prof. Cem Yuksel, Prof. Kai Lawonn

Topics for Quiz

- Transformations (2D, 3D, and Hierarchical) & Viewing
- Shading and Shading transformations
- Textures, FBO, Reflection, and Refraction modeling
- Everything from lecture 10 up to 16th October's lecture

Bump Mapping

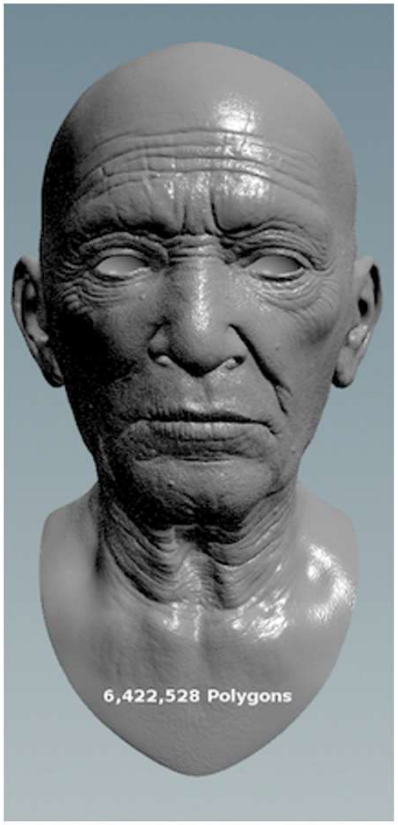


Examples of Parallel Occlusion Mapping

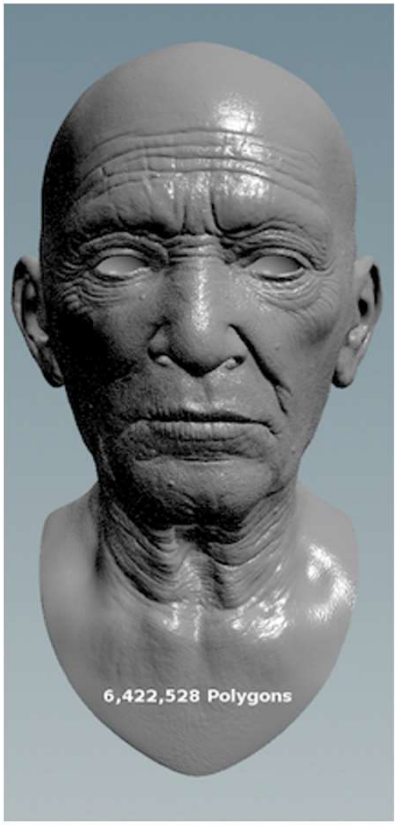
- The Elder Scrolls IV: Oblivion

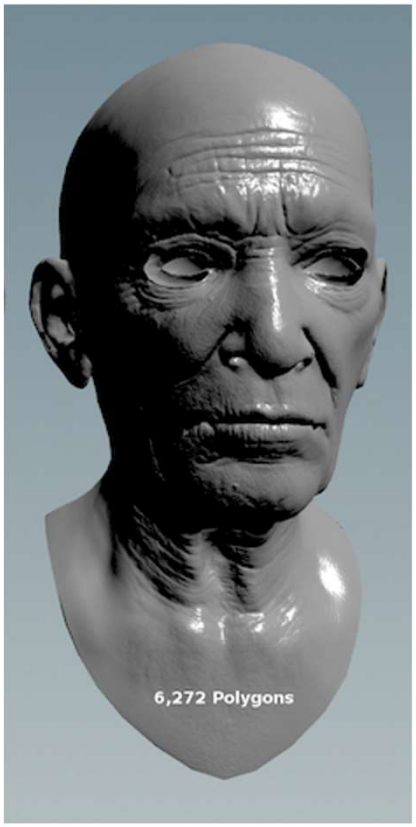
Bump Mapping: Simple Model, Complex View

Bump Mapping: Simple Model, Complex View



Bump Mapping: Simple Model, Complex View



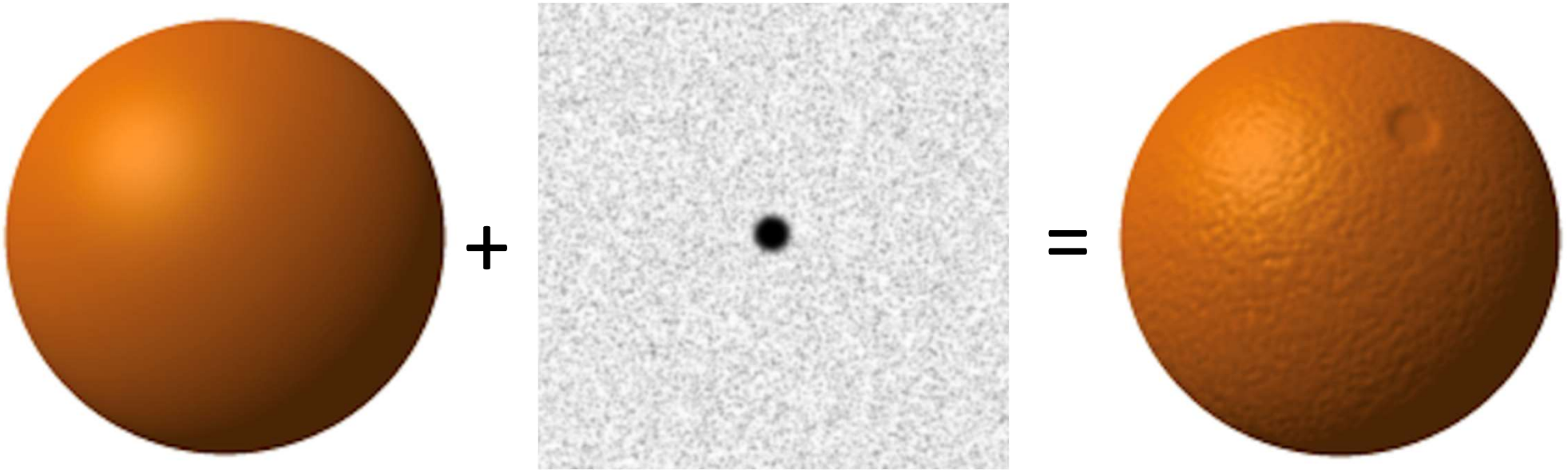


Bump Map

Bump Mapping: Simple Model, Complex View

Bump Map

Bump Mapping

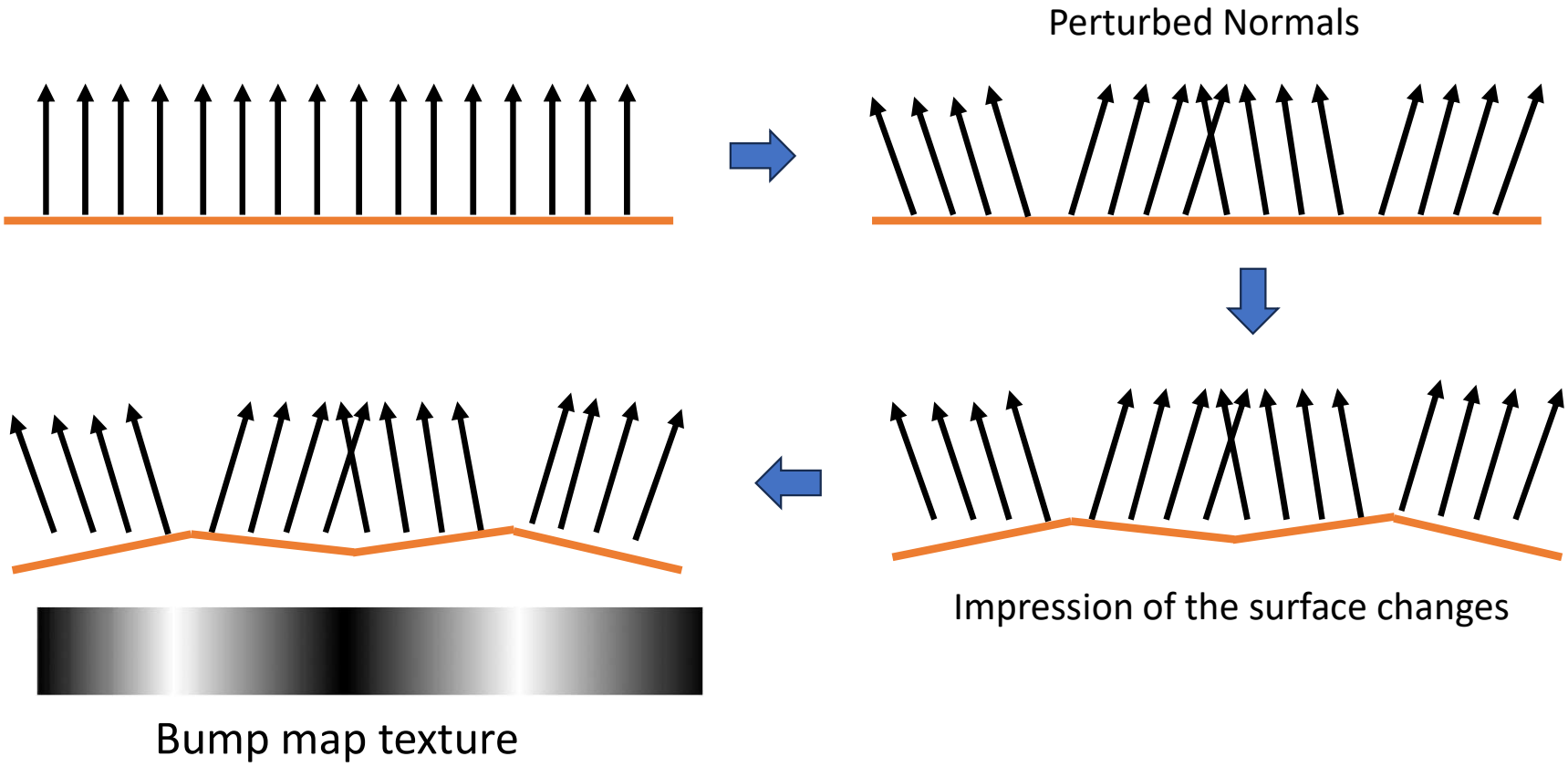


Object

Bump Map Texture

Bump Mapped Sphere

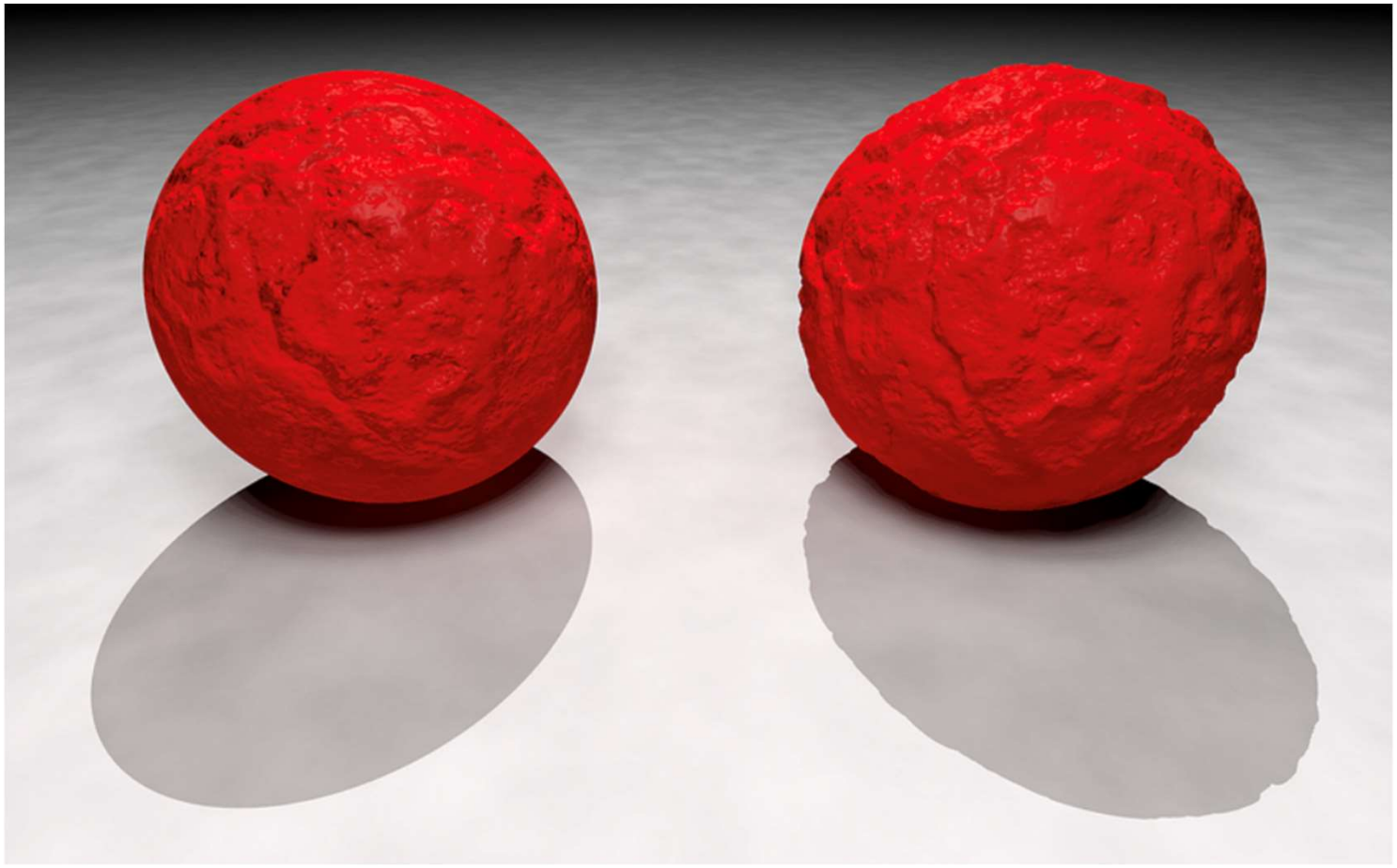
Bump Mapping



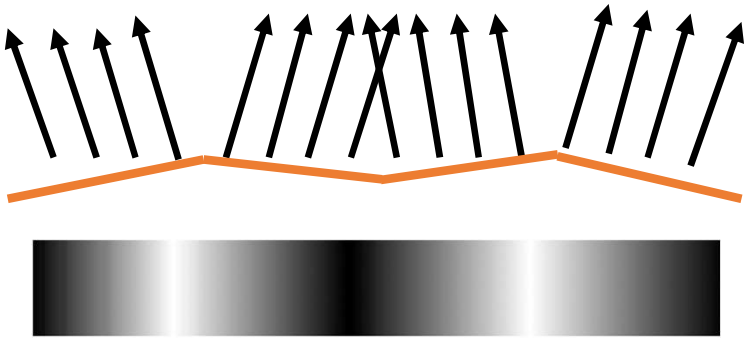
Bump Mapping

Bump Mapping

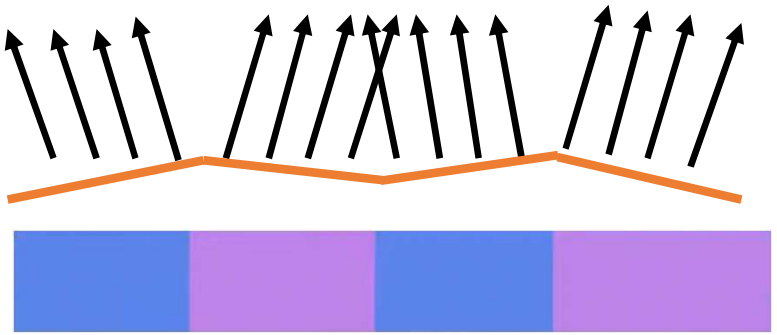
Bump Mapping Illusion



Bump Mapping → Normal Mapping



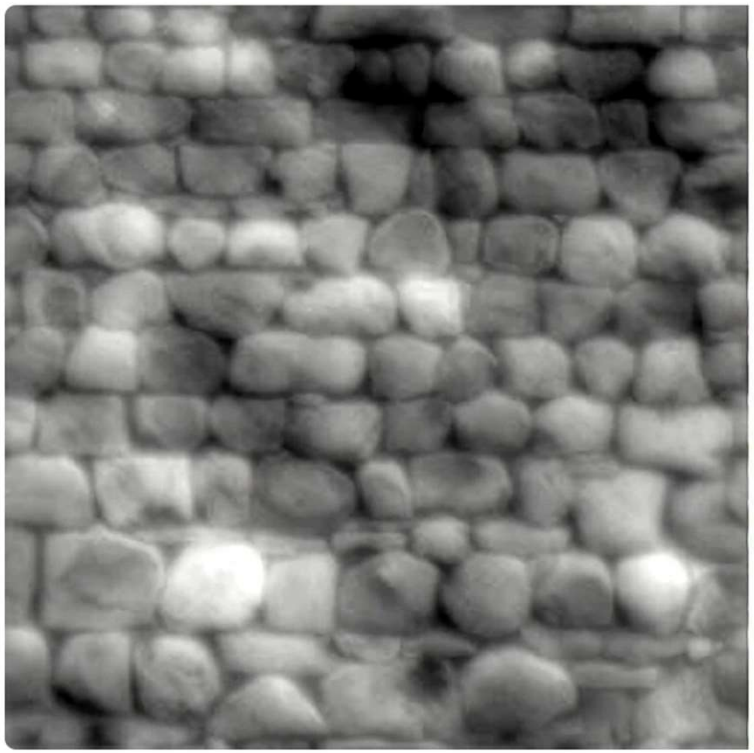
Bump map texture



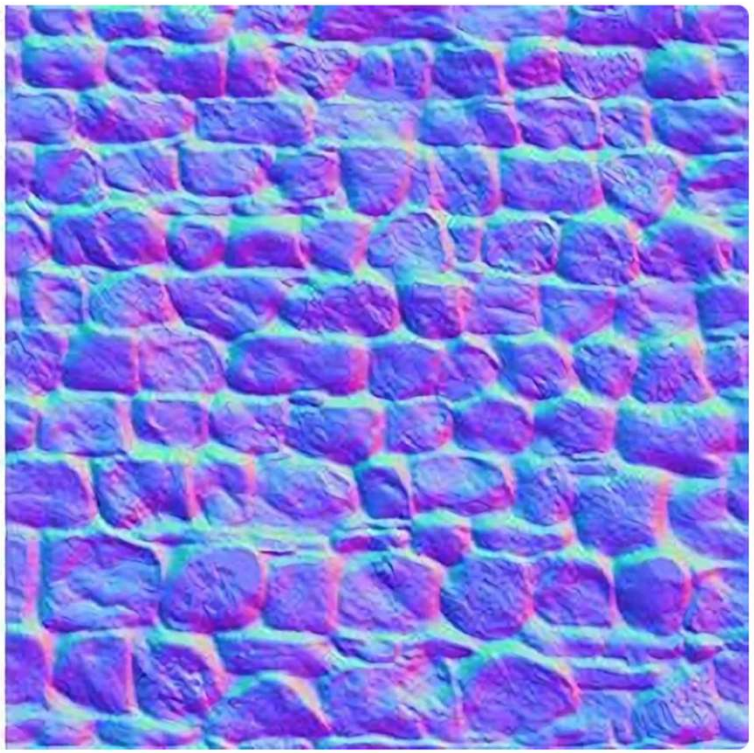
Normal map texture

- In normal map texture, colors represent the normal vectors
- Normal vectors are pre-computed and stored in images

Bump Mapping → Normal Mapping



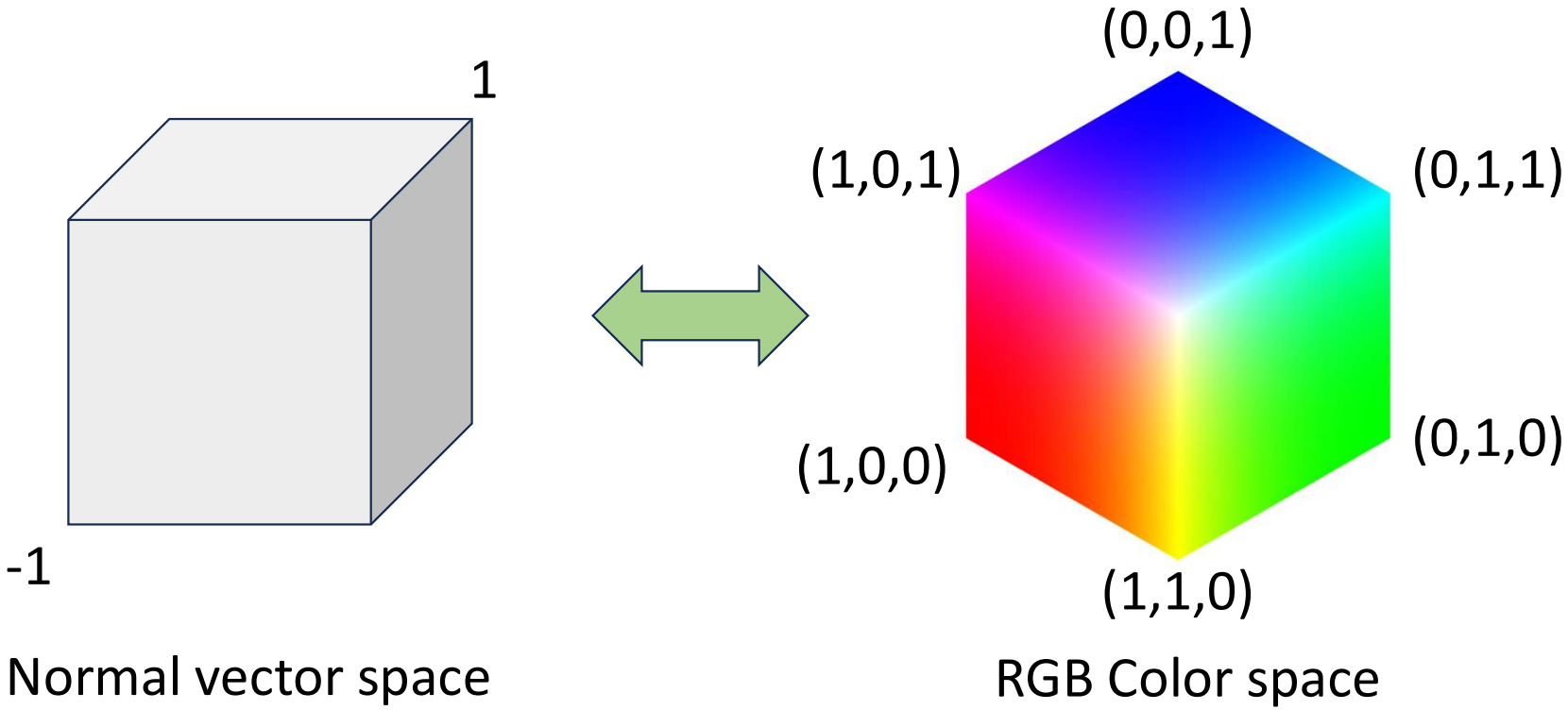
Bump Map Texture



Normal Map Texture

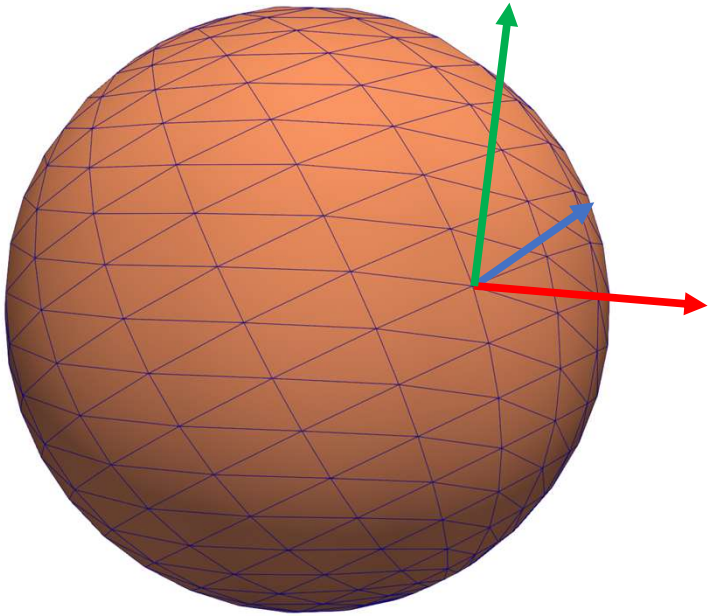
Image source: unity

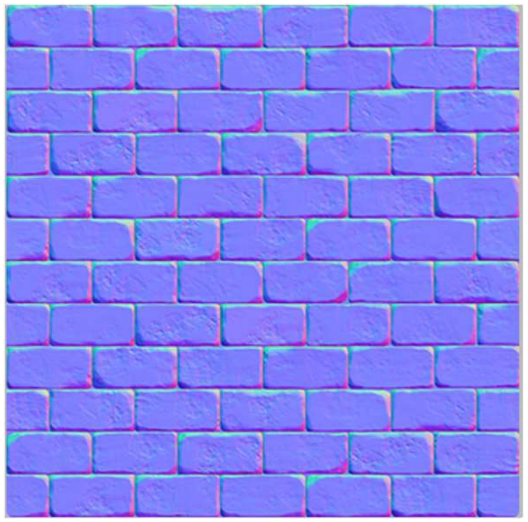
How to Represent Normal Vectors as Colors?



Normal Mapping

- Normal map's coordinates are defined in the local space of the point on an object
  - This allows normal to be represented correctly during animation
  - These normal vectors roughly point towards positive Z-direction





Normal Map Texture

Normal Mapping: Tangent Space

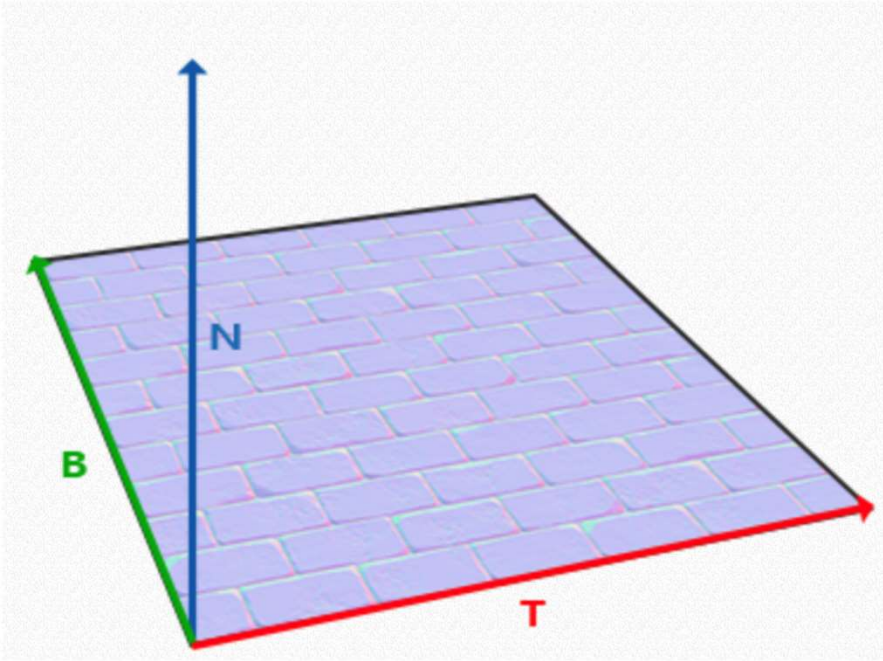
- Vectors in a normal map are expressed in tangent space
- Tangent space is local to the surface
  - Normal vectors are relative to the local reference frame
- Using specific transformation matrices, we can transform normal vectors from this local tangent space to world or other coordinate spaces

Normal Mapping: TBN Matrix

- TBN matrix: tangent, bitangent and normal matrix
- ‘Change-of-basis’ matrix that transforms tangent-space vector to different coordinate space needs three orthogonal vectors
  - Like forming camera transformation matrix
- In TBN matrix, the tangent space’s Z direction is aligned with the surface's normal direction

Normal Mapping: TBN Matrix

- Calculating the tangent and bitangent is not as straightforward as the normal vector
- Direction of tangent and bitangent vectors align with surface's texture coordinates
- Use this idea to calculate tangent and bitangent vectors



Tangent Space Normal Mapping

- How to use TBN Matrix?
- Use inverse(TBN) matrix to transform all the relevant vectors needed for shading to TBN space
  - Note: TBN matrix is orthogonal so *inverse(TBN) = (TBN)^T*
- Finally, compute shading/illumination in TBN space

Implementation Details

- Assume per vertex tangent vectors are given
- Vertex Shader:

```
// Normal map: Construct TBN matrix
biTangent = cross(aTangent, aNormal);
vec3 t = normalize(mat3(uNMatrix) * aTangent);
vec3 b = normalize(mat3(uNMatrix) * biTangent);
fNormal=vec3(uNMatrix*vec4(aNormal,0.0));
mat3 TBN = inverse(mat3(t, b, fNormal));

tangentSpaceLightPos = TBN*vec3(lightPos);
tangentSpaceEyePos = TBN*eyePos;
tangentSpaceFragPos = vec3(uMMatrix * vec4(aPosition, 1.0)); // first go to
world space
tangentSpaceFragPos = TBN*tangentSpaceFragPos;
```

Implementation Details

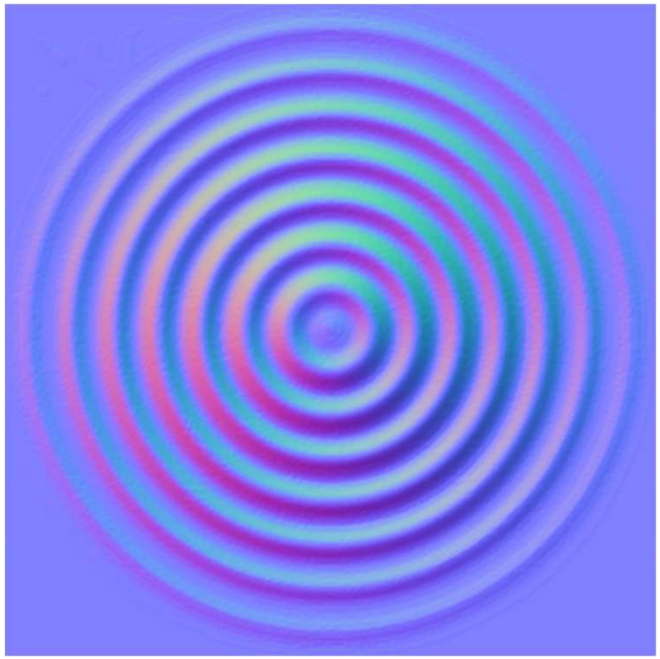
- Fragment Shader:

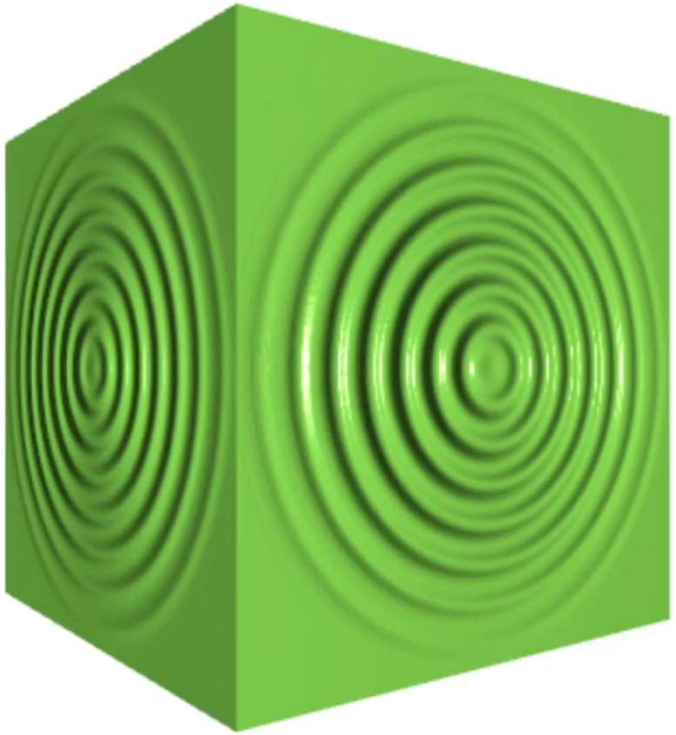
```
// for normal map
in vec3 tangentSpaceLightPos;
in vec3 tangentSpaceEyePos;
in vec3 tangentSpaceFragPos;

// normalmap in tangentspace using normalmaps
norm = normalize(texture(bumpTexture, texCoordinate).rgb * 2.0 - 1.0);
```

  - Normals are converted from (0 to 1) range to (-1 to 1) range by multiplying by 2.0 and then subtracting 1.0

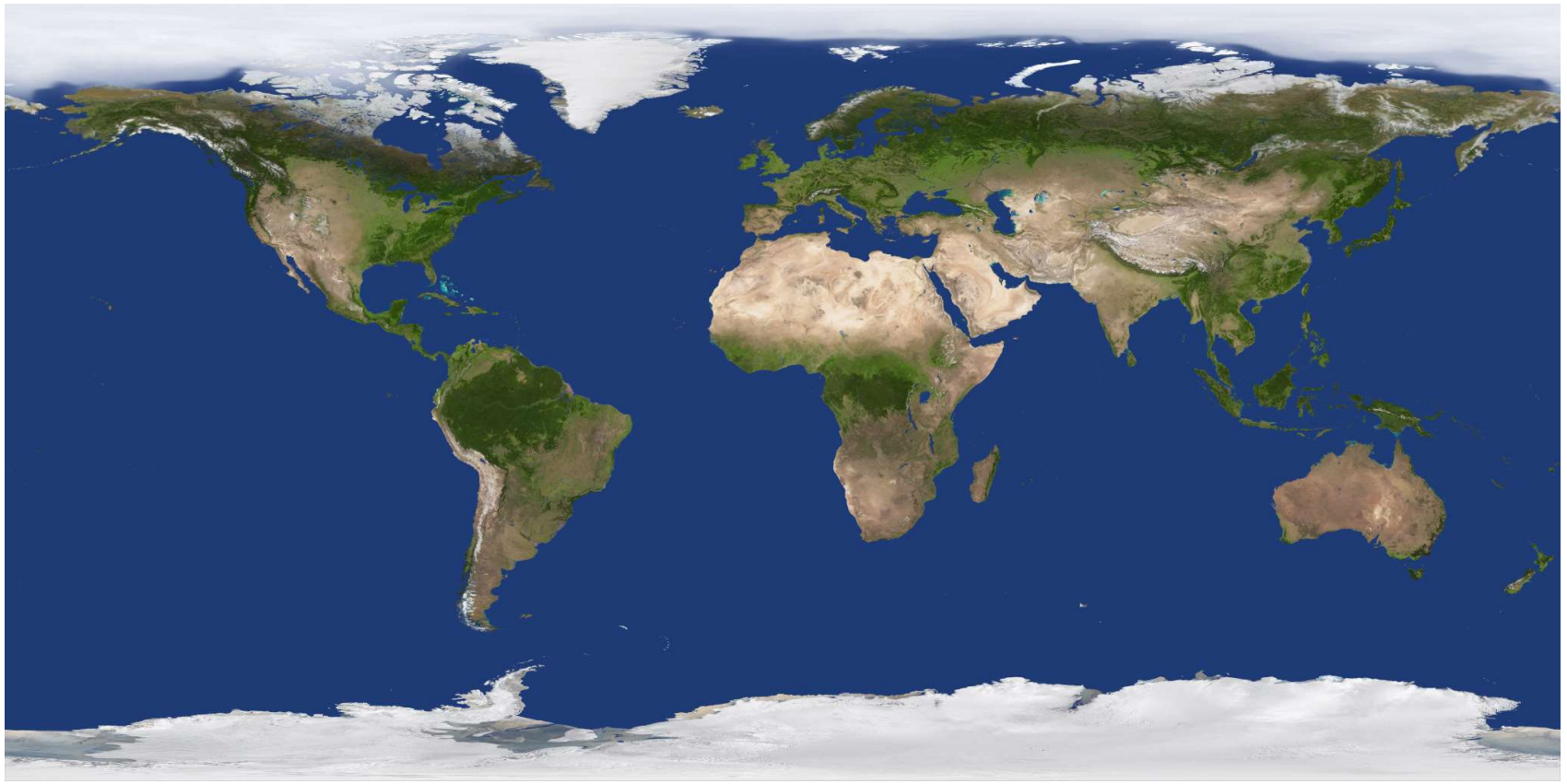
```
// compute the Phong shading now
.....
```

Normal Map on Cube

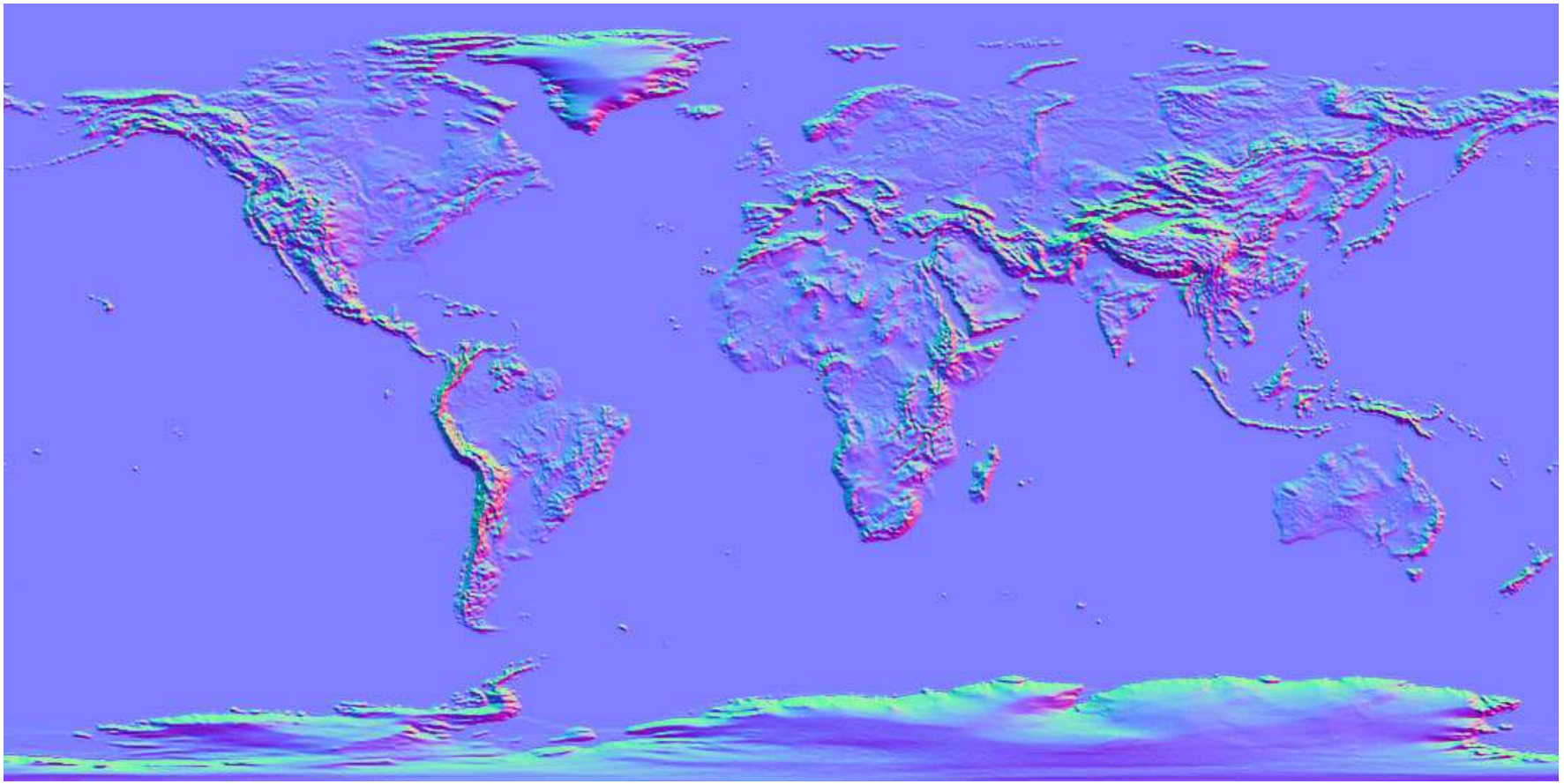




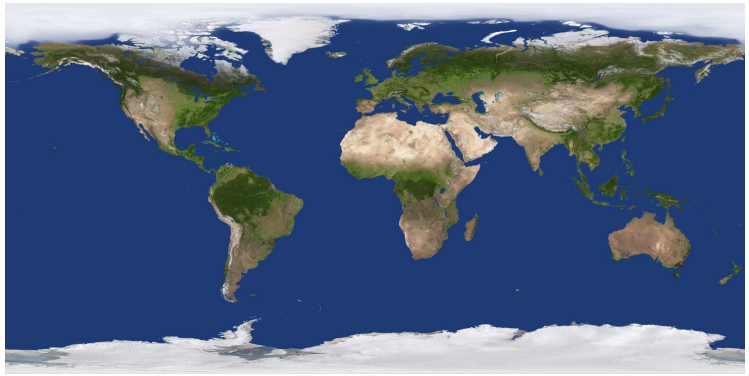
Normal Map on Sphere + Texture



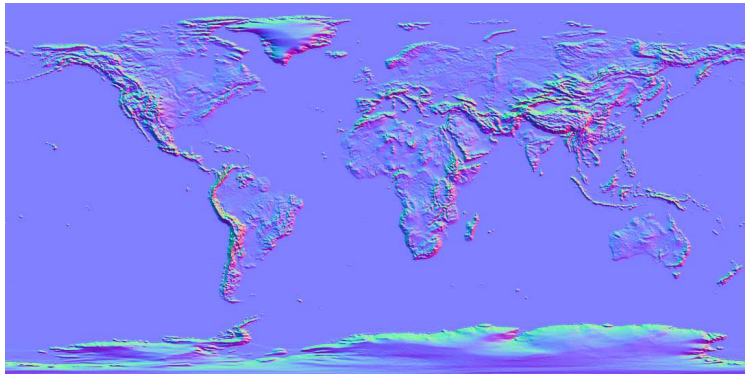
Normal Map on Sphere + Texture

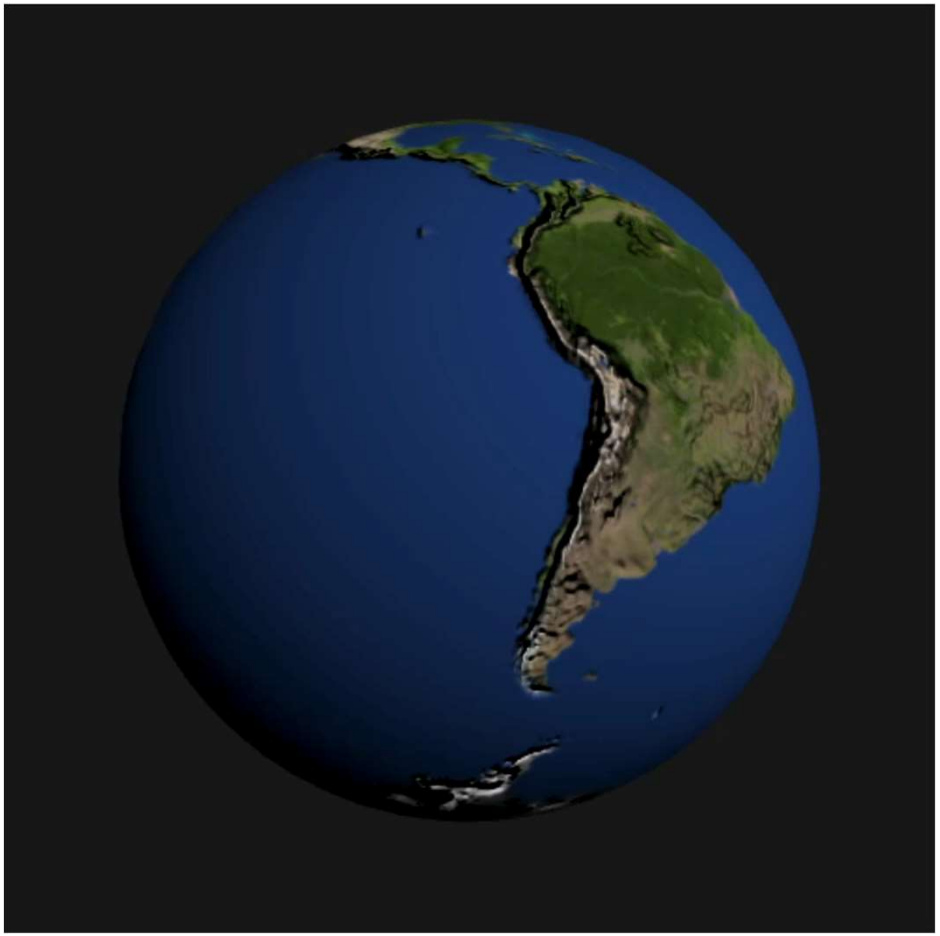


Normal Map on Sphere + Texture

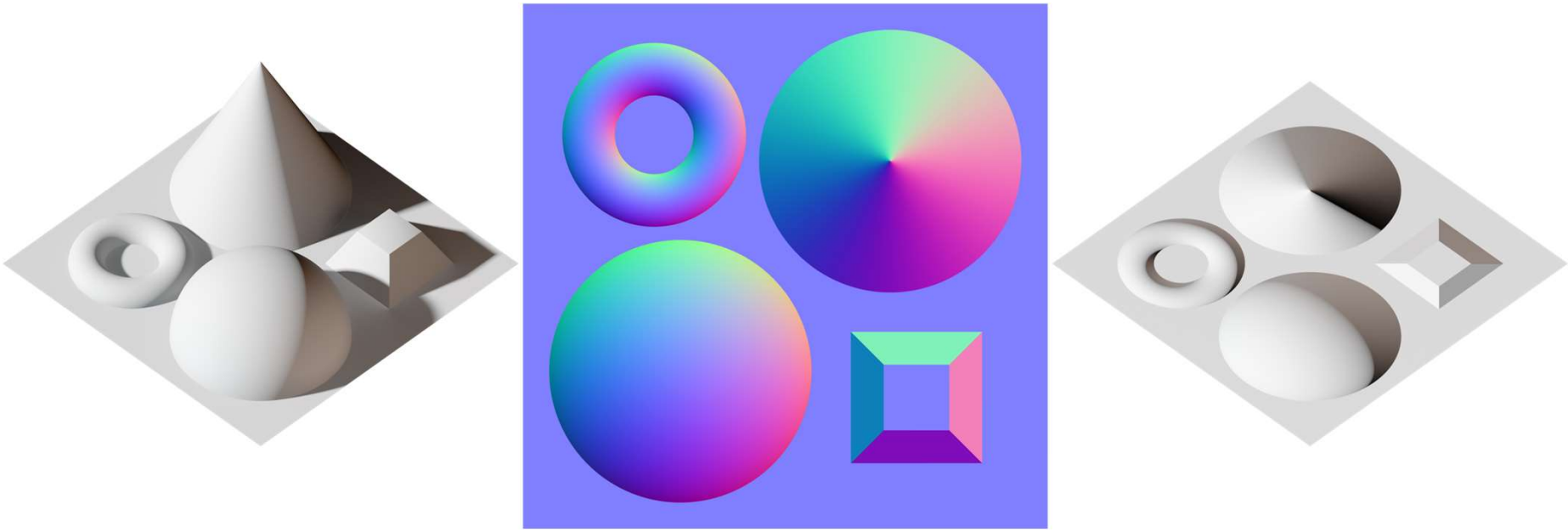


+

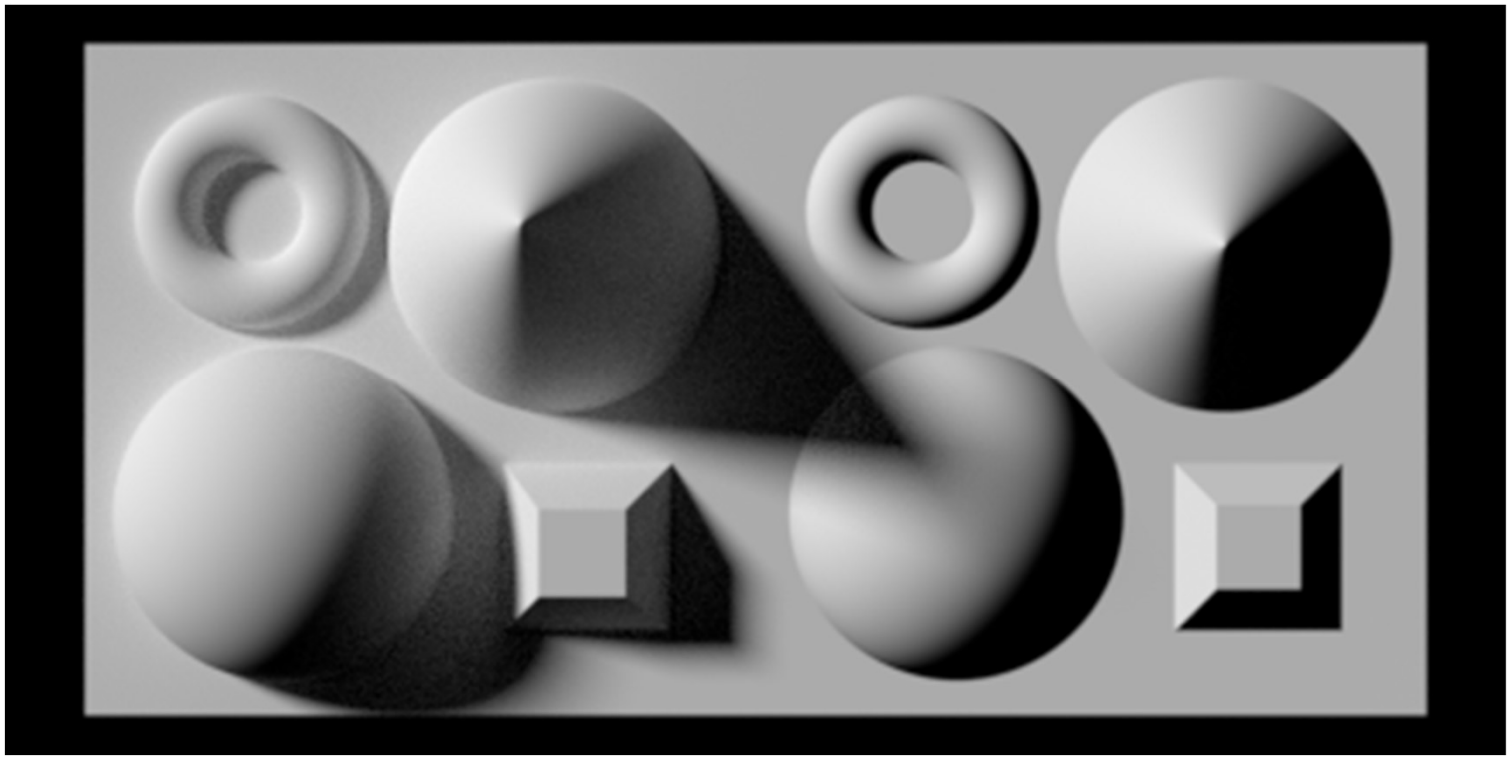




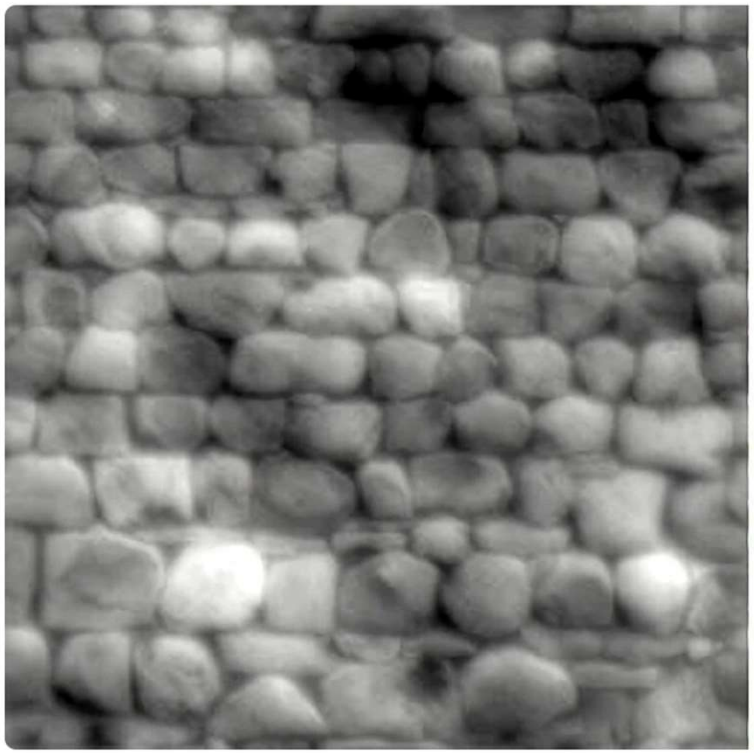
Normal Mapping



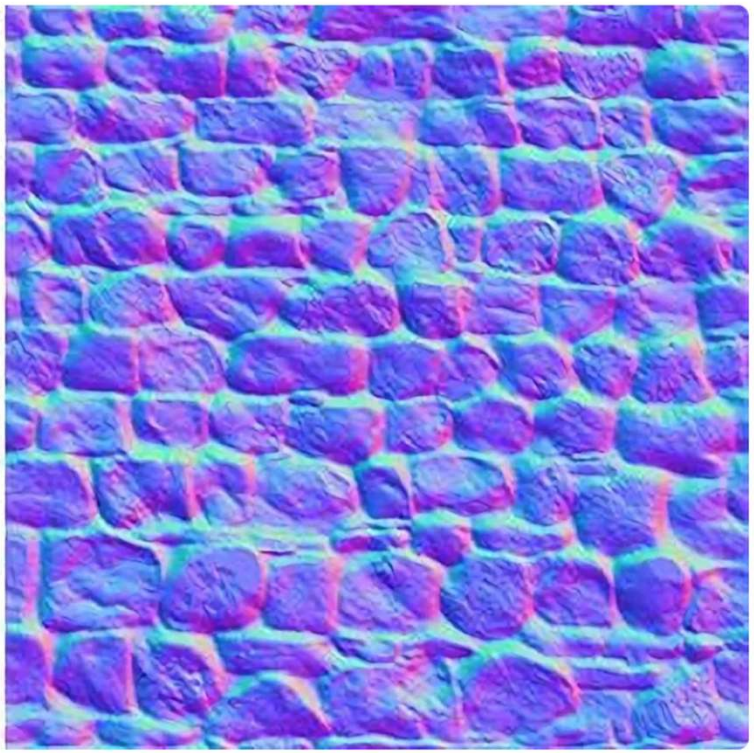
Normal Mapping



Bump Mapping vs. Normal Mapping



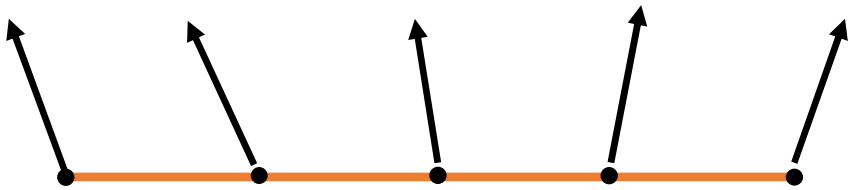
Bump Map Texture



Normal Map Texture

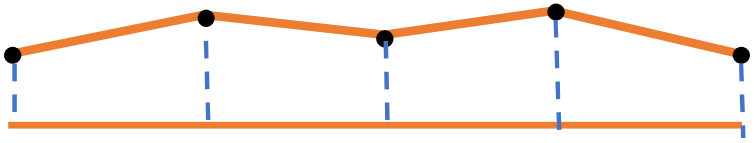
Image source: unity

Displacement Mapping



Normal Mapping

Normals are perturbed

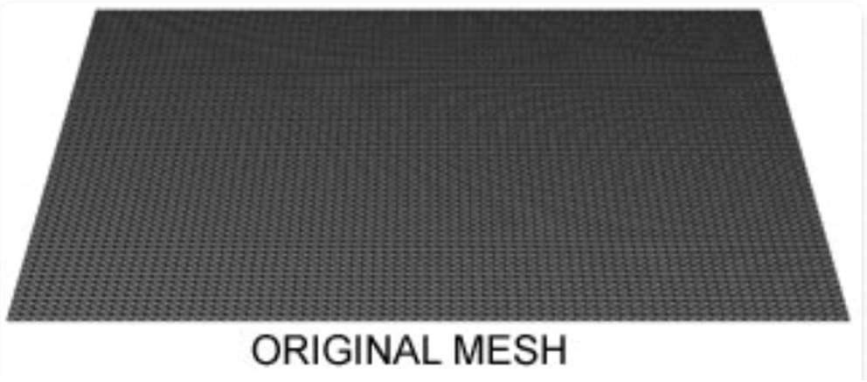


Displacement Mapping

Vertices are displaced

Displacement Texture Map

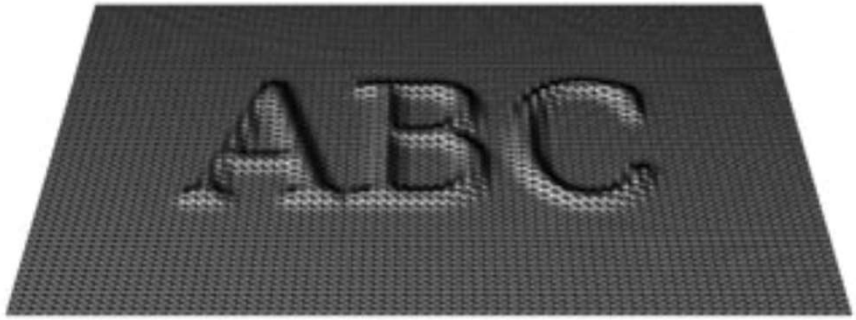
Displacement Mapping



ORIGINAL MESH

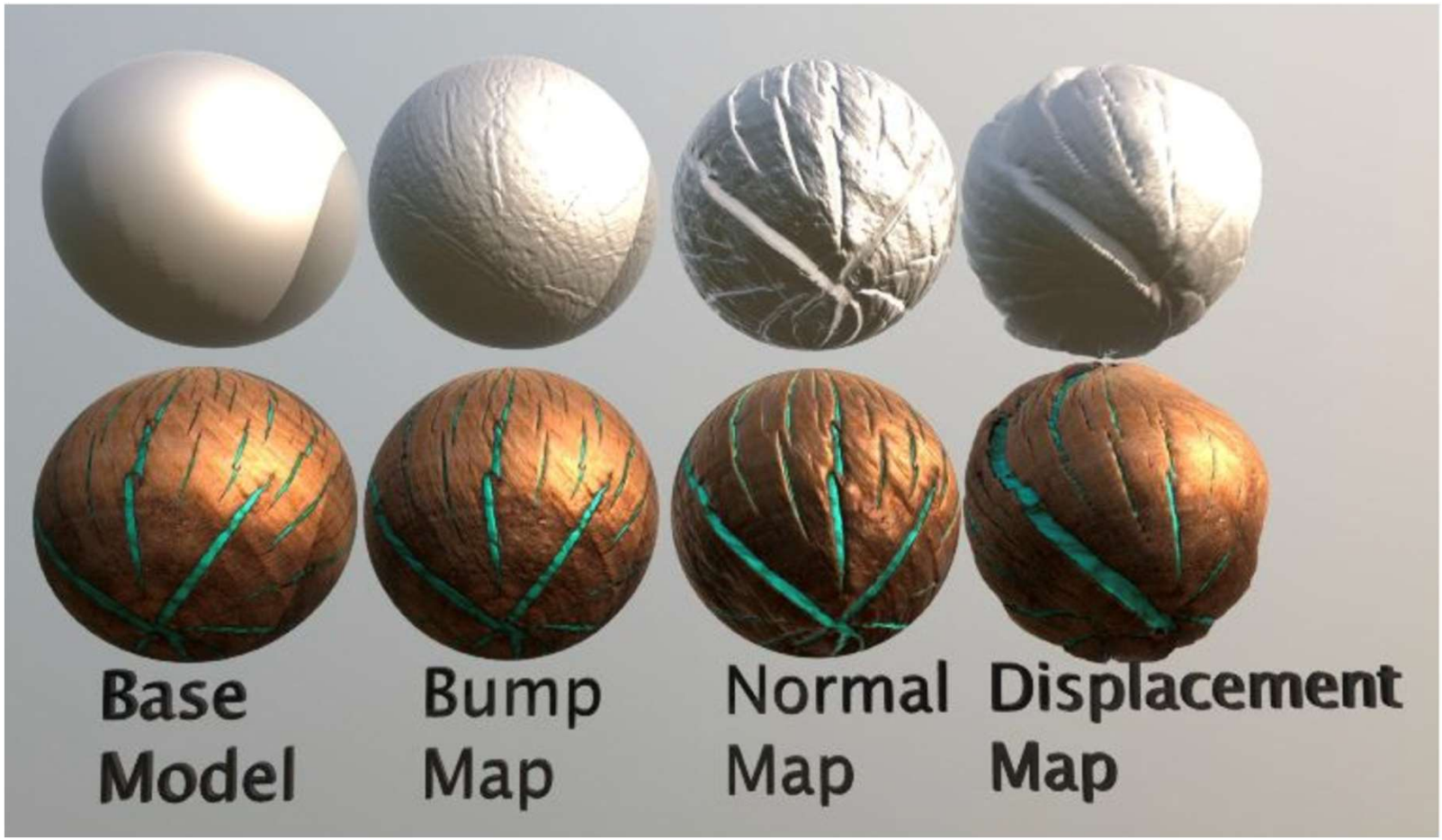
+

DISPLACEMENT MAP

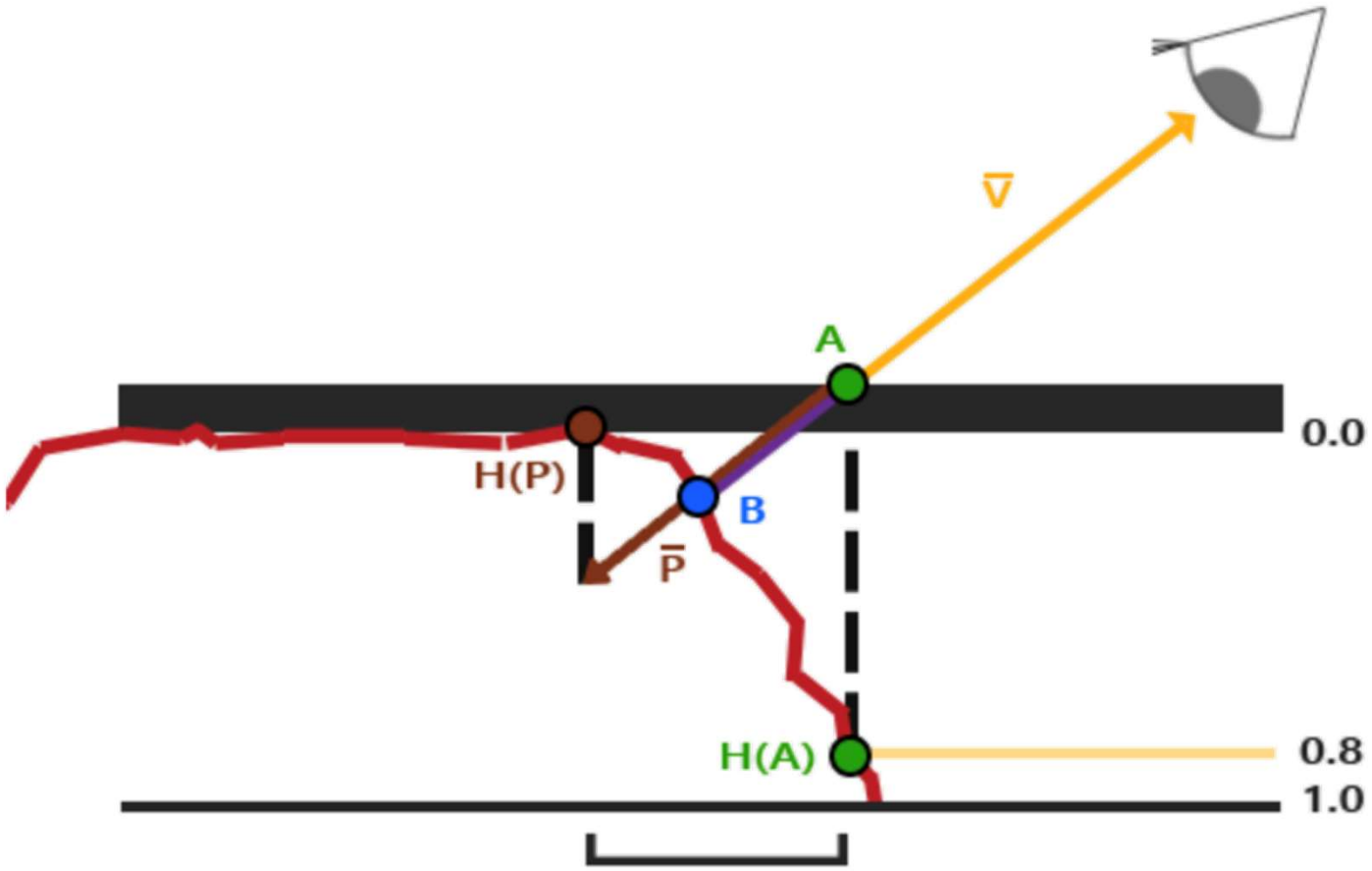


MESH WITH DISPLACEMENT

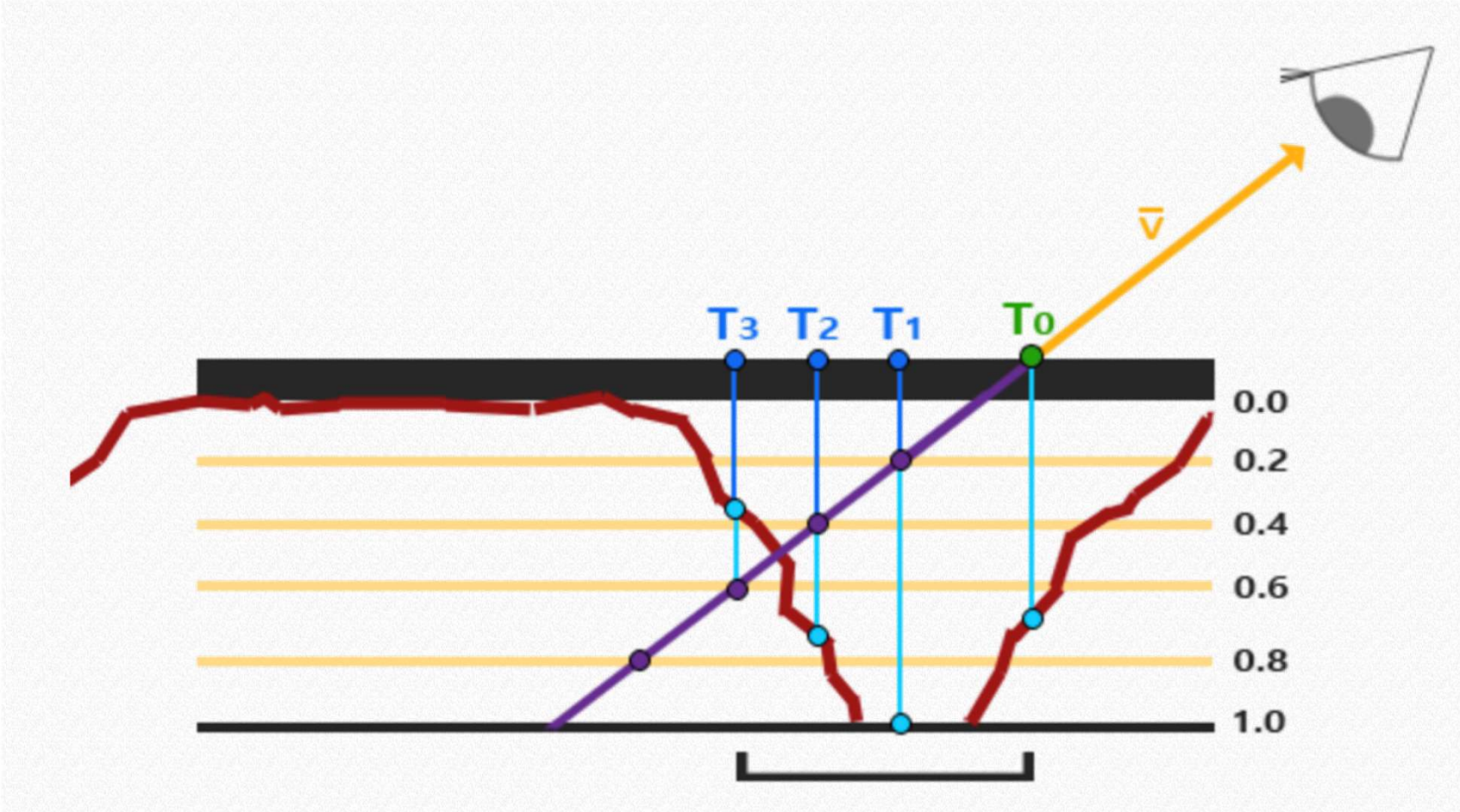
Displacement Mapping



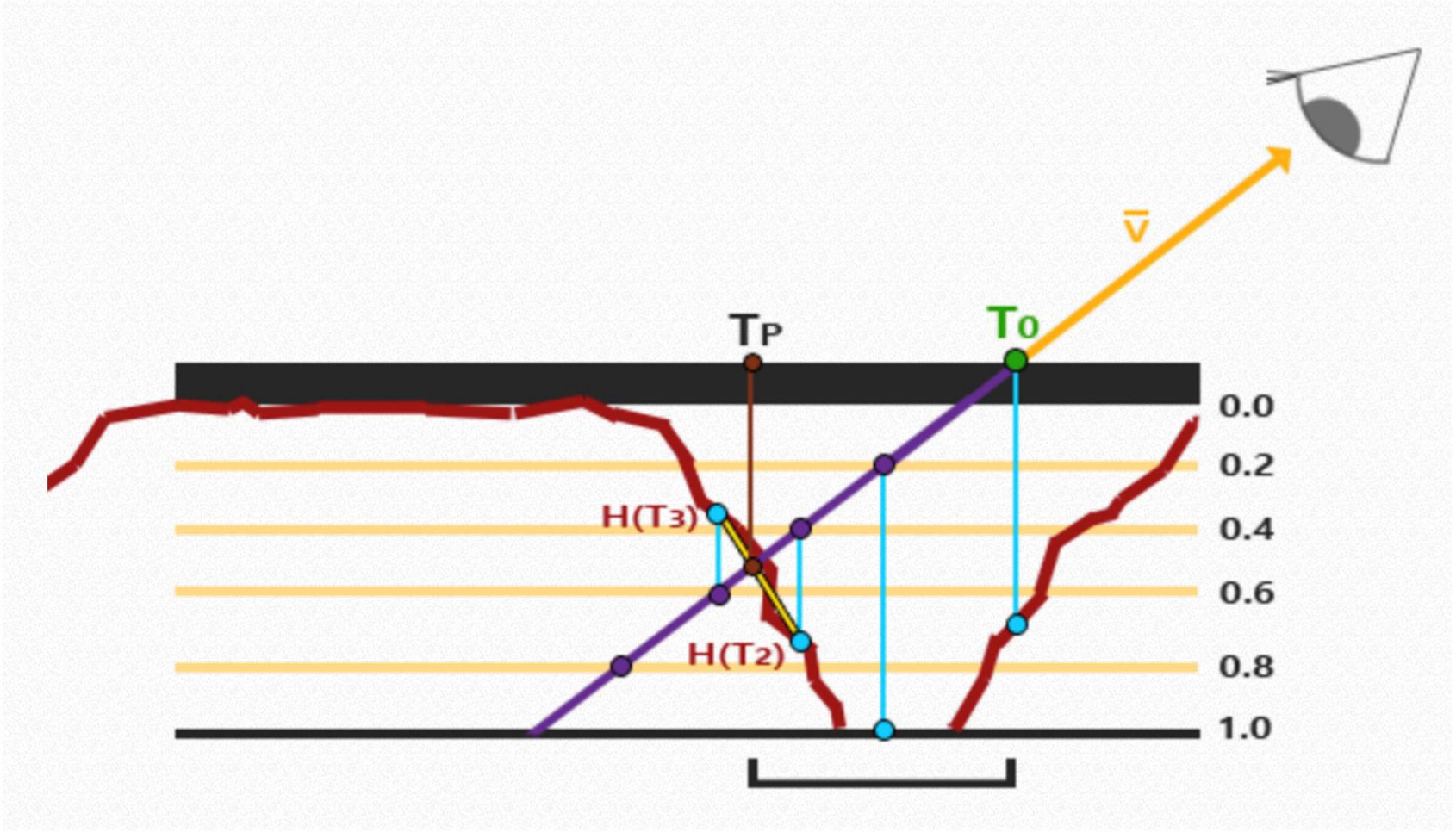
Parallax Mapping



Steep Parallax Mapping



Parallax Occlusion Mapping (POM)



Examples of POM

- Mass Effect

Examples of POM

- The Elder Scrolls IV: Oblivion

Examples of POM

- Crysis

Examples of POM

- S.T.A.L.K.E.R.

Various Types of Bump Mapping

Texture Mapped

Normal Mapped

Parallax Mapped

Steep Parallax Mapped