Computer Graphics

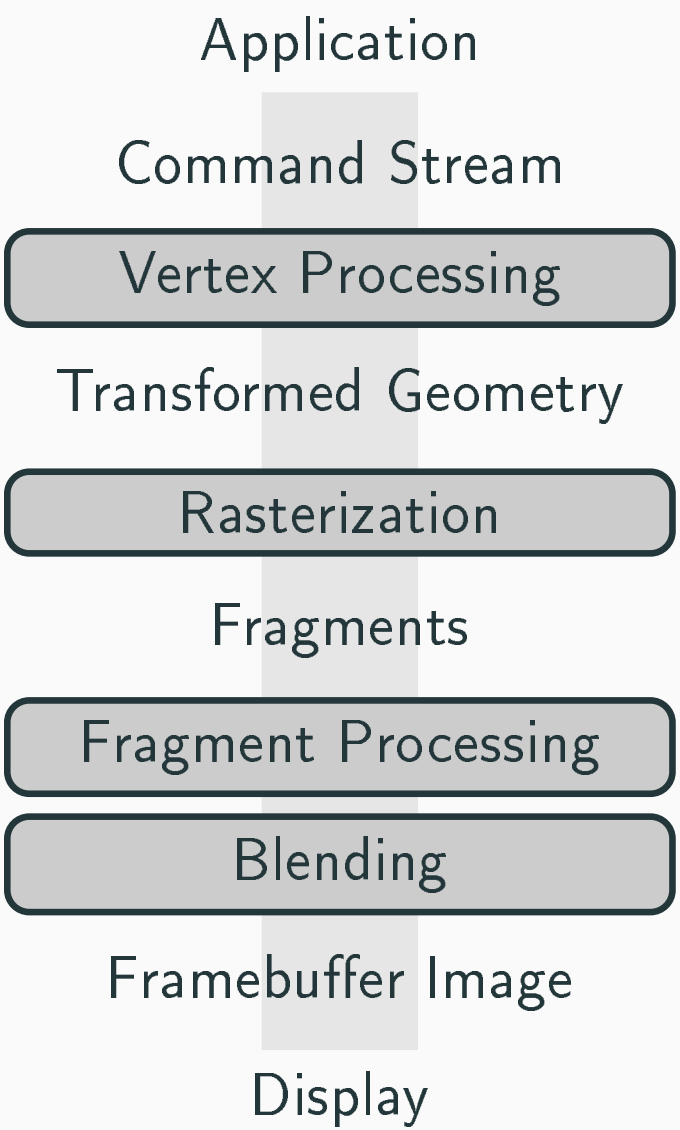
Dr. rer. nat Martin Möddel

April 6, 2021

Institut für Biomedizinische Bildgebung

Object Ordered Rendering

Fragment Shader



Fragment Processing

- Processes the fragments each individually
- Usually programmable
- Passes one or more colors to the blending stage

Fragment Shader

- The rasterizing stage creates fragments
- The fragments have screen space coordinates, z-value and various other shading information such as the normal direction or color associated
- These fragments are then further processed within the fragment shader, e.g. texturing, lighting
- Fragments are processed individually
- Crosstalk is not allowed (if required it must be postponed to an extra image processing step)

Fragment Shader - Illumination

- Once the basic color of the fragment is determined we can modify it by applying a suitable lighting model
- due to the fact that each fragment carries information about its normal we can reuse many of the illumination models we discussed in ray tracing
  - Ambient
  - Lambert
  - Blinn-Phong
- Implementation of shadow is more complicated

Fragment Shader - Texturing

- In CG it is often required to model surface properties in detail finer than allowed by the vertex geometry
- This can be achieved by a technique called texturing

Definition
Texturing is the process of modifying the appearance of a surface using an image like data source e.g.

- Coloring
- Bump/Normal mapping (modification of surface normals)
- Parallax mapping (illusion of depth in e.g. stone surfaces)

Texturing

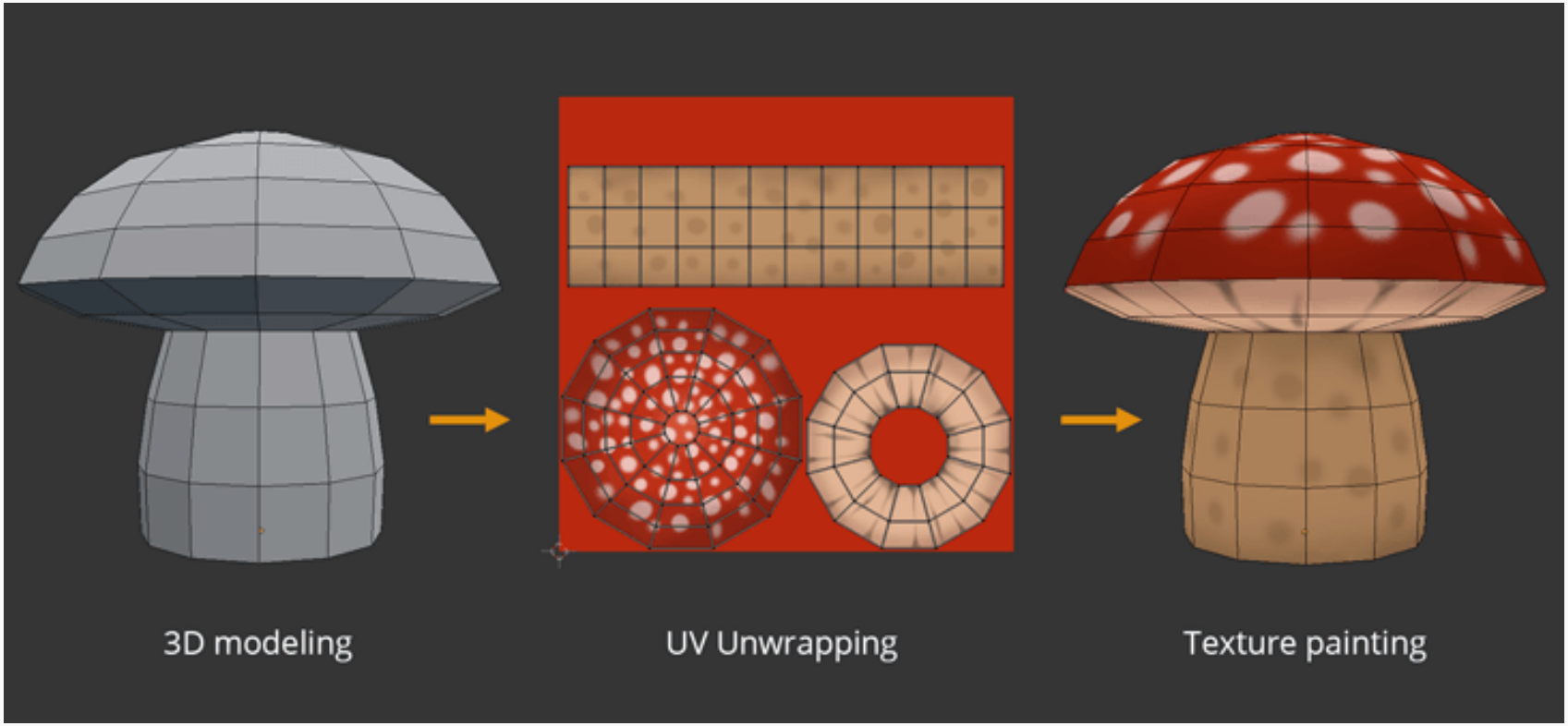
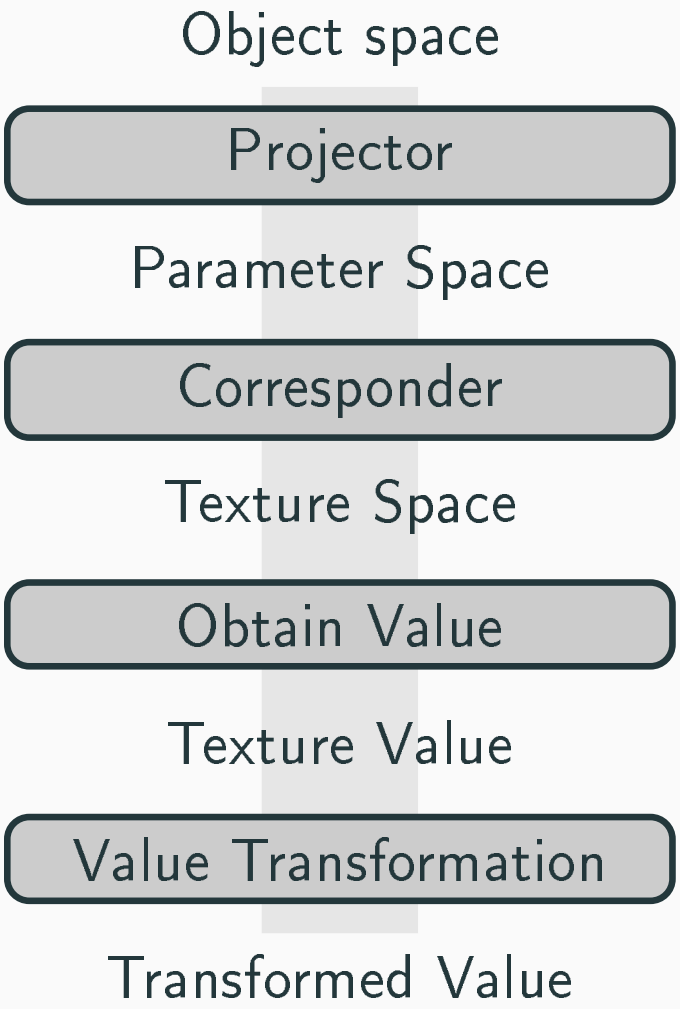


Image Source: www.blendernation.com

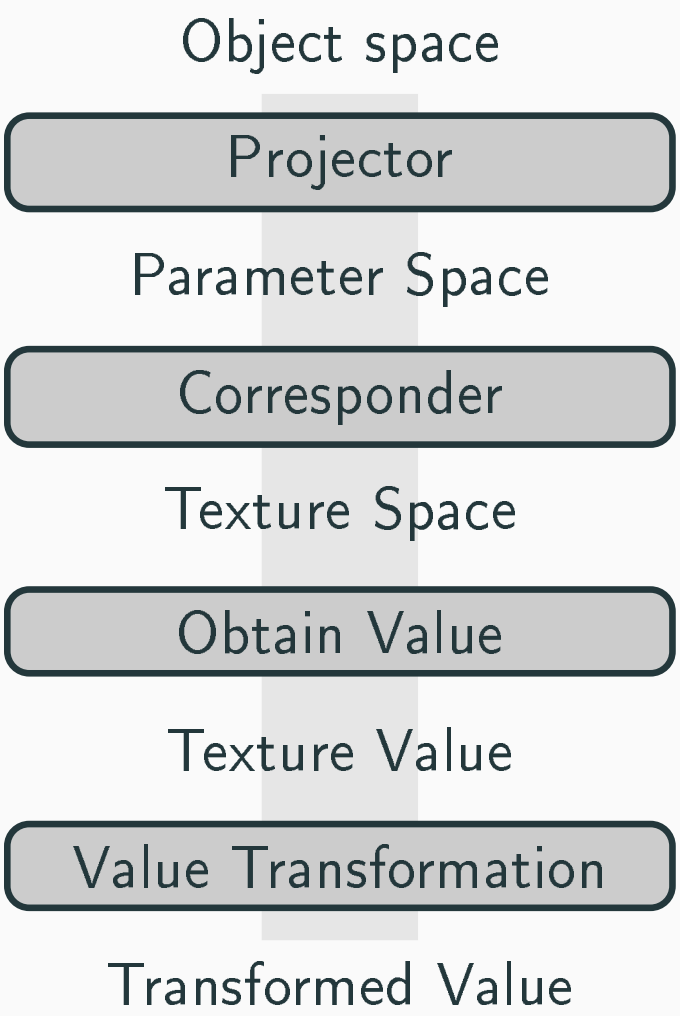
Texturing



Texturing

- Texturing is performed in a modular pipeline
- Usually not all steps might be under explicit user control
- Aim is to output some texture value, e.g.
  - Color
  - Normal vector
- Basically the same in IOR and OOR

Texturing



Projector
The projector function P is a mapping from a surface points in object space to 2D points in parameter space $\mathcal{P} \subset \mathbb{R}^2$

- Sometimes projector function directly maps to texture space in which case no corresponder function is required

Texturing - Projector Function

- For mesh objects this mapping is usually modelled, i.e. the (u, v) coordinates are predefined for each vertex and later **perspectively correct** interpolated
- In ray tracing one can invert the parameter function of an explicit surface to define projector function
- Alternatively one can use projections, e.g. along the surface normals onto an enclosing surface (sphere, cube, cylinder)
- There are not many restrictions on the projector function, but some properties make handling easier
  - In most cases it is helpful if the mapping is bijective
  - Distortions should be minimized
  - Neighboring points on a surface should map to neighboring parameter points
- If the geometry is too complex to handle than it is commonly subdivided into smaller parts, which each have their projector

Texturing - Spherical Projector Function

- Surfaces that are roughly spherical can be parametrized using spherical coordinates
- If the object is centered around the origin 0 then the projector function is given by the longitude and latitude of the point projected onto the sphere

$$P(x, y, z) = \left(\frac{\pi + \text{atan}(y, x)}{2\pi}, \frac{\pi - \text{acos}\left(\frac{z}{\|\text{x}\|_2}\right)}{\pi} \right) \quad (1)$$

  where $\text{x} = (x, y, z)$ is the surface point
- The spherical coordinate map is not bijective at the poles, which might lead to distortions

Texturing - Cubic Projector Function

- Instead of projection onto a sphere we might as well project the surface onto a cube
- Projection along the normal or projection towards the nearest side of the cube
- To project along the positive and negative x-direction one might use

$$P_{-x}(x,y,z) = \frac{1}{2}\left(1 + \frac{z}{\|\mathrm{x}\|_2}, 1 - \frac{y}{\|\mathrm{x}\|_2}\right) \tag{2}$$

$$P_{+x}(x,y,z) = \frac{1}{2}\left(1 - \frac{z}{\|\mathrm{x}\|_2}, 1 - \frac{y}{\|\mathrm{x}\|_2}\right) \tag{3}$$

- A suitable texture has six square pieces
- Often used for textures that are viewed from the inside of a cube (environment mapping)

Texturing - Perspectively Correct Projector Function

- Consider two points $q, p \in \mathbb{R}^3$ in world space
- Two values c_q, c_p may be interpolated along the line segment by convex combination, i.e. by

$$c_q + \alpha(c_p - c_q) \tag{4}$$

at position

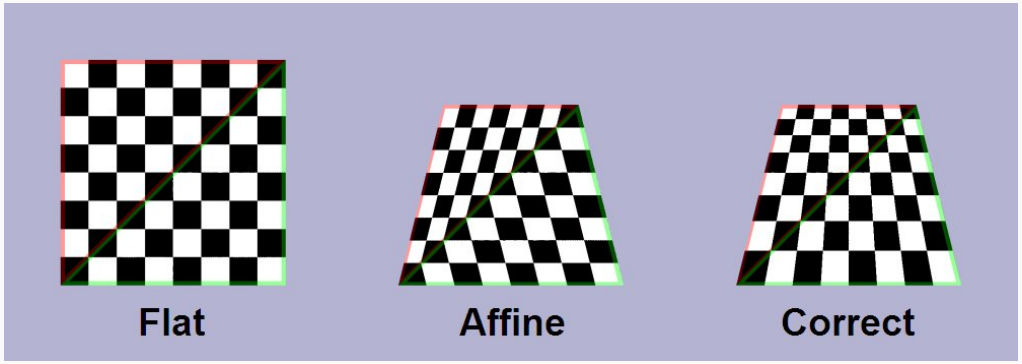
$$q + \alpha(p - q) \tag{5}$$

- This is the basis for Gouroud shading and texturing

Texturing - Perspectively Correct Projector Function

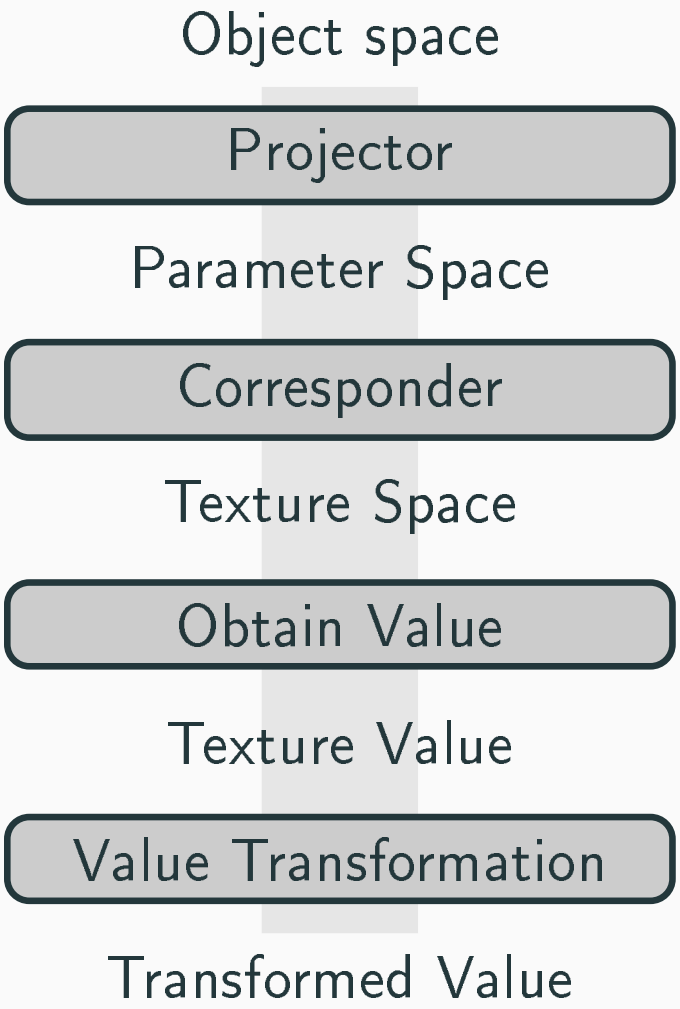
- A problem arises if we apply perspective projections P, in which case

$$P(\mathsf{q} + \alpha(\mathsf{p} - \mathsf{q})) \neq P(\mathsf{q}) + \alpha(P(\mathsf{p}) - P(\mathsf{q})) \tag{6}$$



- Texture coordinates and other values need to be transformed using perspectively correct projector functions
- This can be done using homogeneous coordinates, which will be introduced at the end of the lecture

Texturing - Corresponder Function



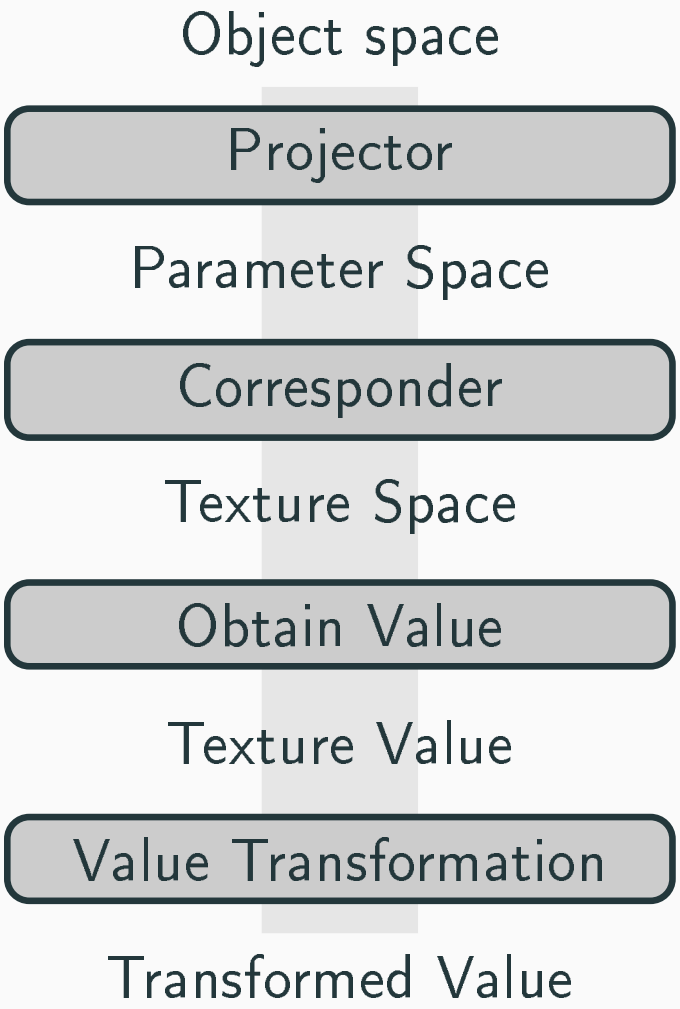
Corresponder
The corresponder function transforms points $\mathrm{p} \in \mathcal{P}$ to texture space $[0,1]^d$, $d \in \mathbb{N}$

- Textures might be images $d = 2$ or higher dimensional objects $d > 2$
- Determine the behaviour if coordinates are outside the texture space
- Mapping to pixel coordinates

Texturing - Corresponder Function

- Mapping to pixel coordinates is the only part that depends on the resolution of the texture
- Determining the behaviour outside the texture space, e.g. to repeat a texture multiple times
- Common behaviours are
  - Repeating of the texture by dropping the integer part of the texture coordinate
  - Mirroring the texture repeatedly at its edges
  - Clamp the texture to edge by repeating the texture values at the borders
  - Clamp the texture to border by using a separately defined border color
- These behaviours can often be assigned independently to the different canonical directions in texture space

Texturing - Obtain Value

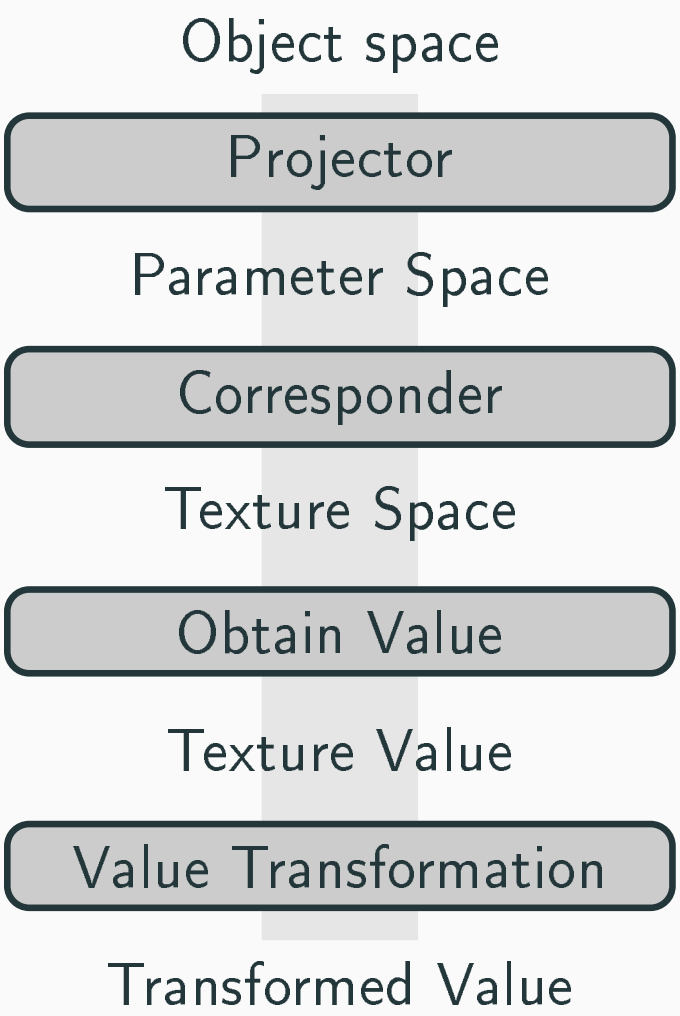


Definition
A texel refers to a pixel in texture space, whereas a pixel often refers to a pixel in screen space.

Obtain Value

- In the simplest case it is as simple as reading the value of a texel
- Most often it is not as simple as explained later in image texturing

Texturing - Value Transformation



Value Transformation

- Depending on the application the retrieved value can be processed further
- Modified by the light conditions at the surface for example

Image Texturing

- In image texturing an image is effectively glued onto a surface
- Idealy the texture resolution is roughly that or a little bit larger than the number of pixels covered on screen, i.e. a pixel to texel ratio of about $1:1$
- If the texture resolution is much lower the texture is magnified
- If the texture resolution is much larger it is minified
- Problem if a texture is observed at a steep angle (normal to viewing direction)
- Problem if the pixel to texel ratio is correct at some parts of the surface and magnification/minification occurs at other parts
- In both cases filtering needs to be applied to avoid aliasing

Image Texturing - Magnification

- When a texture is magnified many pixels are obtain their information from a single texel, i.e. a low frequency texture is sampled at a high frequency
- Aliasing can be reduced by filtering (e.g. convolution)

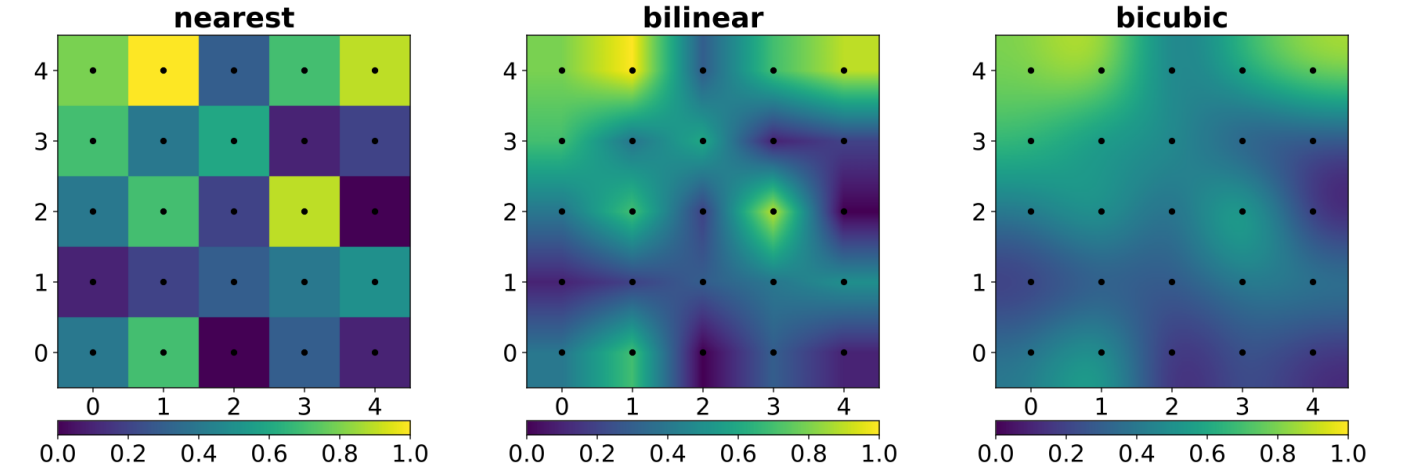


Figure 1: "Illustration of Nearest neighbor/Bilinear/Bicubic interpolation on a random dataset. Compare with other interpolation methods that share the same dataset" by Zykure licensed under CC BY-SA 4.0

Image Texturing - Minification

- When a texture is minimized many texels may contribute to the color of a single pixel, i.e. a high frequency texture is sampled at a low frequency
- For correct results these texels should be integrated
- Precise integration is complicated in real time
- A number of methods to preprocess the texture to compute quick approximations are used
  - Nearest neighbour interpolation
  - Mipmapping
  - Summed-area tables
  - Anisotropic filtering

Image Texturing - Minification - Mipmapping

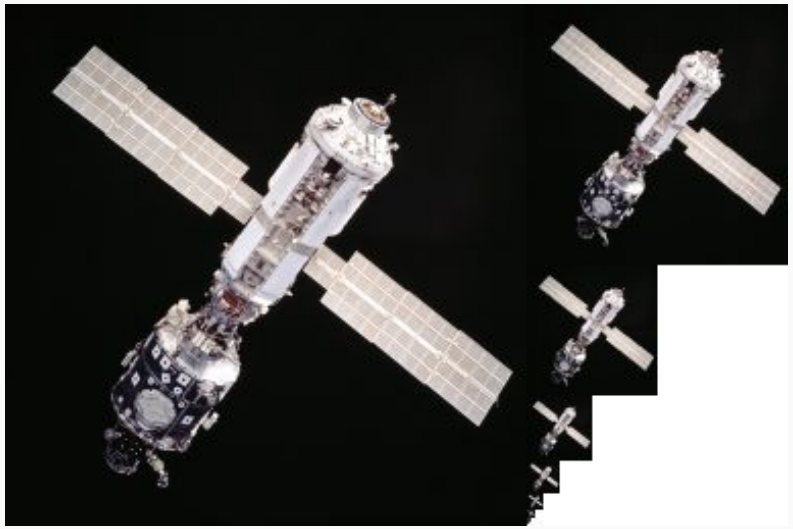


Figure 2: "This image is a example of mip-mapping." by Mulad licensed under CC BY-SA 3.0

- Mip stands for *multum in parvo* (many things in a small place)
- In a preprocessing step using a suitable filter a series of minifications of ratio 2 of the texture are created and stored in mipmap

Image Texturing - Minification - Mipmapping

- Color is sampled from minification with appropriate level of detail
- Appropriate level of detail is where the pixel-to-texel ratio of at least 1:1 to satisfy the Nyquist rate
- Alternatively, a number of minifications are used to obtain the color value
- Extension (ripmap) creates minifications with varying aspect ratios to compensate for blurring along one direction if the texture is observed at a steep angle

Procedural Texturing

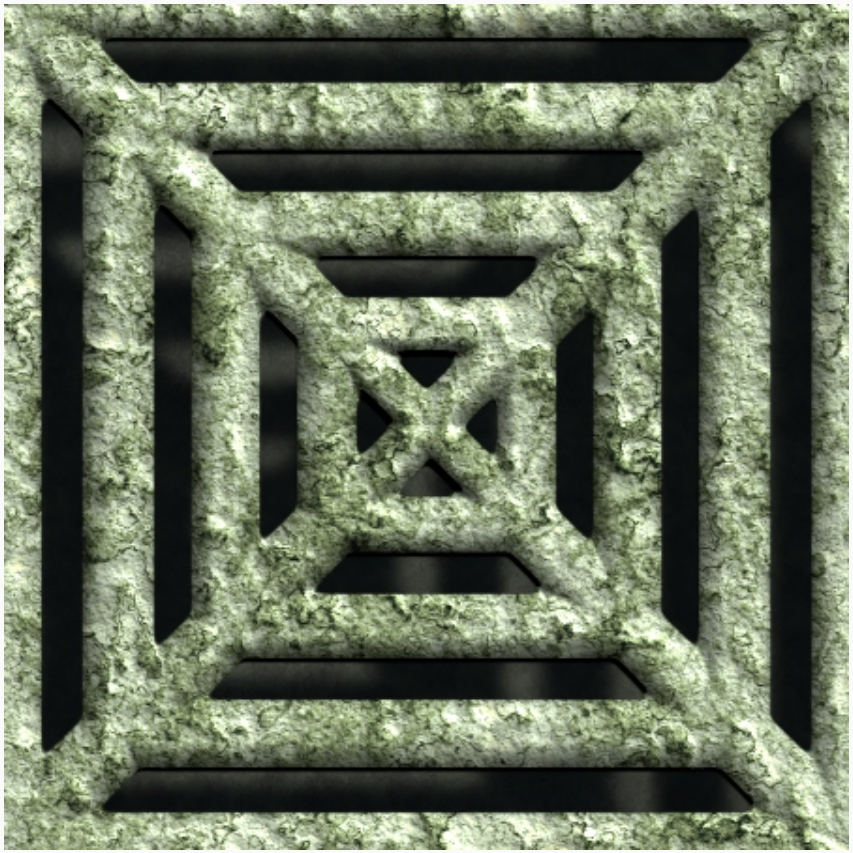


Figure 3: "An example of a procedural texture generated with the professional texture editor Genetica" by Wiksaidit licensed under CC BY-SA 3.0

Procedural Texturing

- Instead of looking up textures one can generate texture values by evaluating functions
- Here too the frequency of the function has to be accounted for to avoid aliasing artifact if the function is sampled at low frequencies
- Direct control over the function allows to remove high frequency components
- Such textures are often extremely smooth and perfect, which creates unnatural appearence
- Noise can be added to avoid these problems

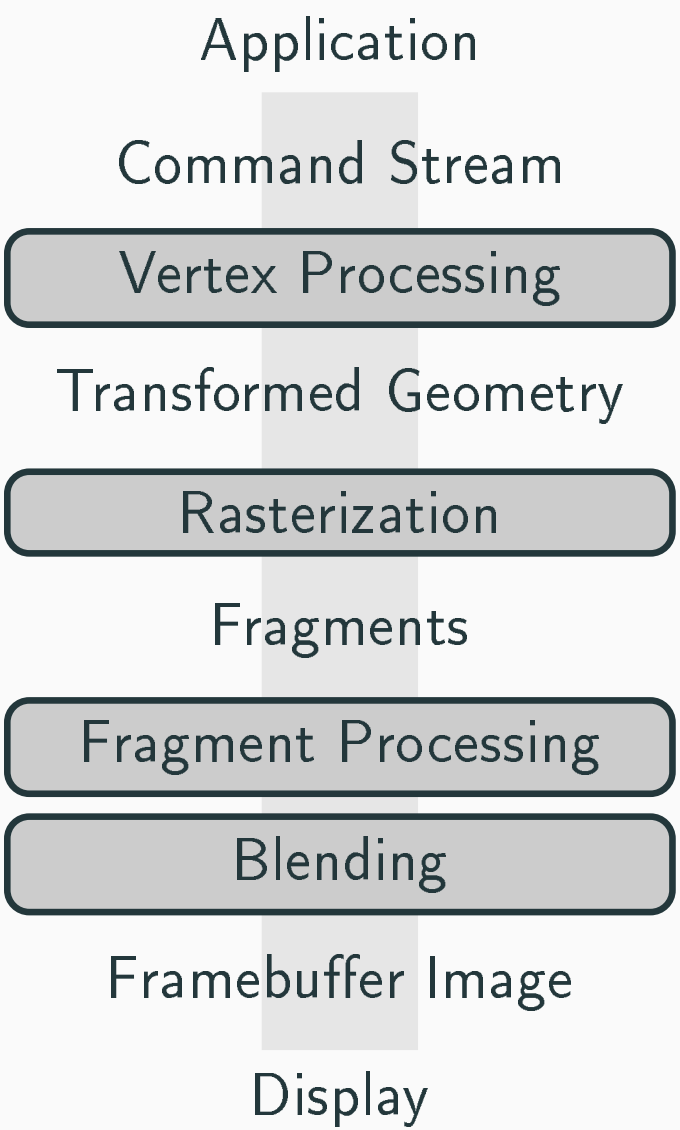
Texturing - Bump Mapping

- Textures may be used to add small-scale surface details to surfaces on coarse-scale geometry meshes
  - Bump mapping - texture contains information about how to vary the normal vector along the surfaces tangent space
  - Normal mapping - texture contains normal vector encoded into RGB channels
  - Paralax mapping - displacement of texture coordinates to account for self-occlusion (e.g. mortar in a brick wall not visible if viewed at a steep angle)
  - Relief mapping - can accurately model self-occlusion, self-shadowing, and parallax
- Can be used to introduce geometric details without having to increase to polygon number

Texturing

- Appart from the methods discussed so far there are many more texturing methods available which can be used to come closer to photo realism
- However, the general idea remains the same, i.e. to store information inside the textures, which can not be obtained in real time otherwise
  - Texture animation
  - Material mapping
  - Alpha mapping
  - Shadow mapping
  - many more

Blending



Blending

- Blend existing pixel values in the frame buffer and colors received from the fragment shader
- Colors might have alpha channels associated
- In this stage the occlusion problem is solved

Blending

- Blending itself is the process of combining a number of colors
- Alpha blending combines a translucent foreground color with a background color
- There are many blending modes available such as
  - Multiply $f(a, b) = ab$
  - Screen $f(a, b) = 1 - (1 - a)(1 - b)$
  - Pegtop $(1 - 2b)a^2 + 2ab$
  - Many more

Blending - Z-Buffering

- The most important step in this stage is to find a pixel inside the framebuffer that needs to be updated or not
- Therefore an additional buffer keeps track of the z-values (camera space) of the pixels drawn
- The stored values are then compared to the z-values of the current fragment
- The framebuffer is updated only if the current fragment is closer to the observer

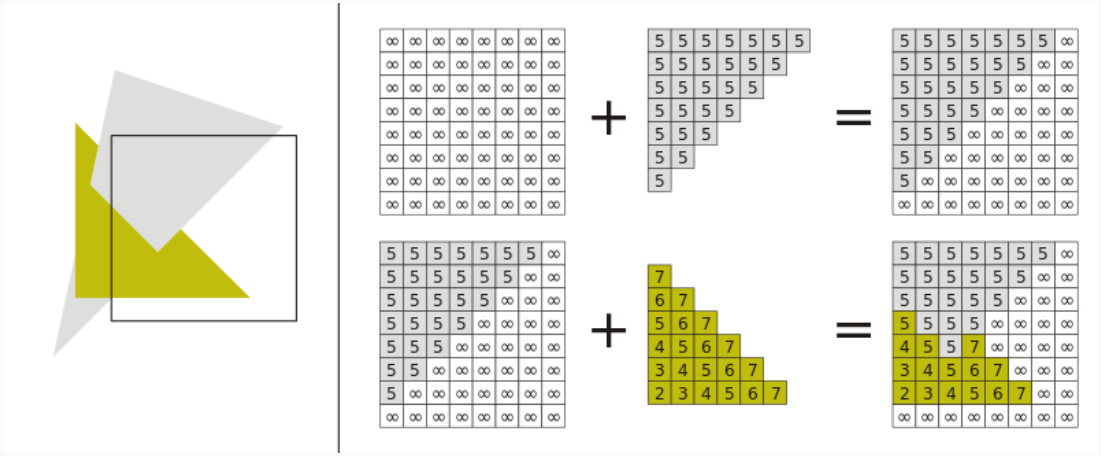


Figure 4: "z-Buffer" by Vierge Marie

Blending - Z-Buffering

Precision

- Storing the z-values equidistantly with b-bit integer precision leads to a depth resolution of

$$\Delta z = \frac{|f - n|}{2^b}, \tag{7}$$

  where $|f - n|$ is the distance between near and far plane

- In world coordinates this bin size is related to its distance from the near plane g_{world} along the gaze direction

$$\Delta z_{\text{world}} = \frac{g_{\text{world}}^2 \Delta z}{nf} \tag{8}$$

- Note that $n = 0$ leads to infinitely large bins in world space at the far plane
- So near and far plane need to be chosen with care