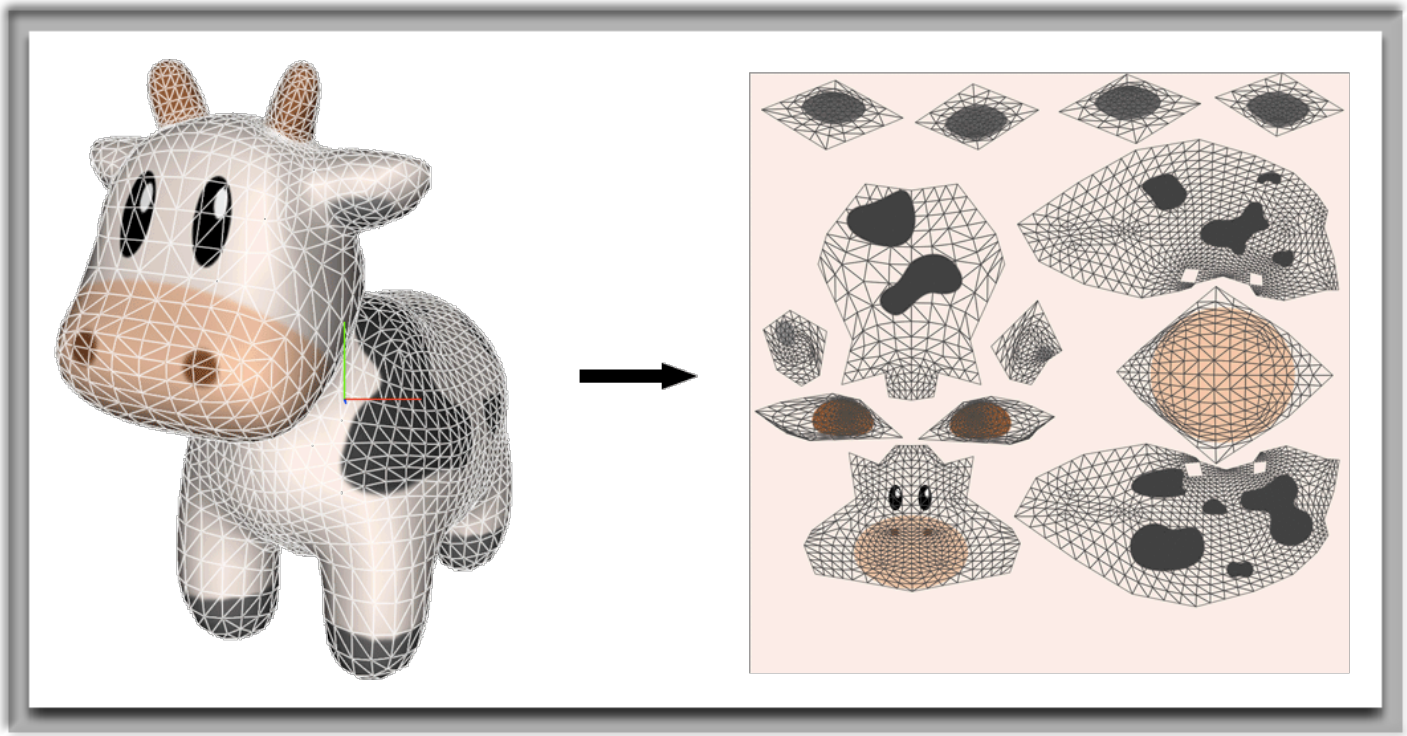
Textures

Let us skin our objects

Texture Mapping

needs for „skinning“ a mesh

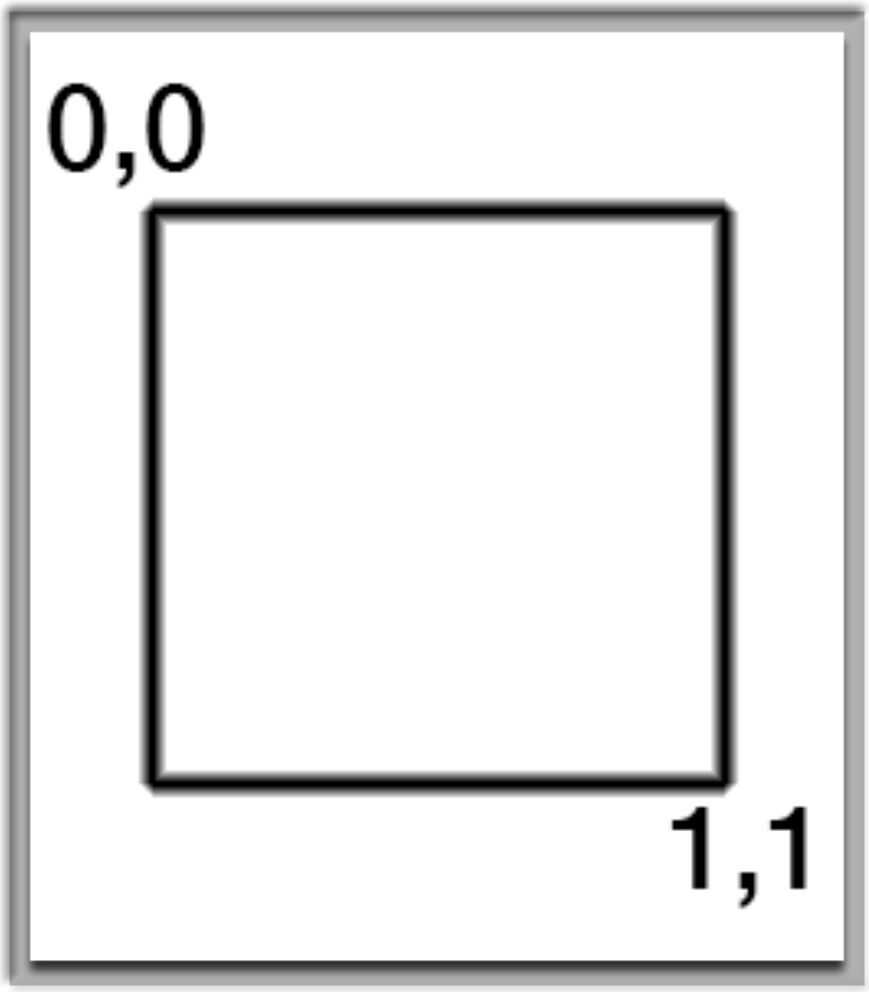


UV Mapping

normalized coordinate system

original in left top corner

every vertex has its own pair of uv coordinates



Direct 3D

every textures needs a ID3DShaderResourceView for sending data to a shader

every textures needs a ID3D11SamplerState for sampling texture for specific coordinates

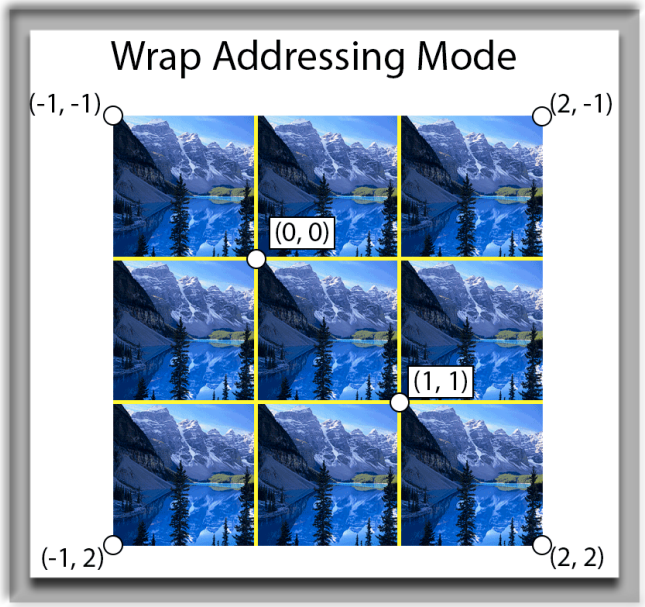
Sampler State I

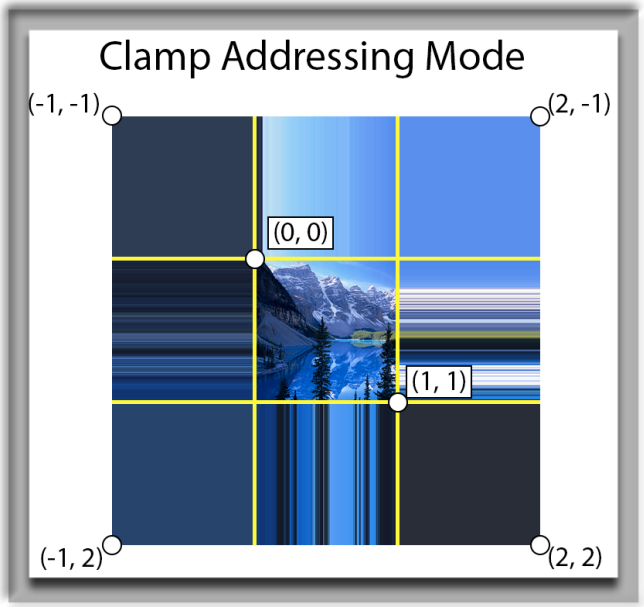
a sampler state consists of many settings for sampling a pixel on a texture referenced by uv coordinates

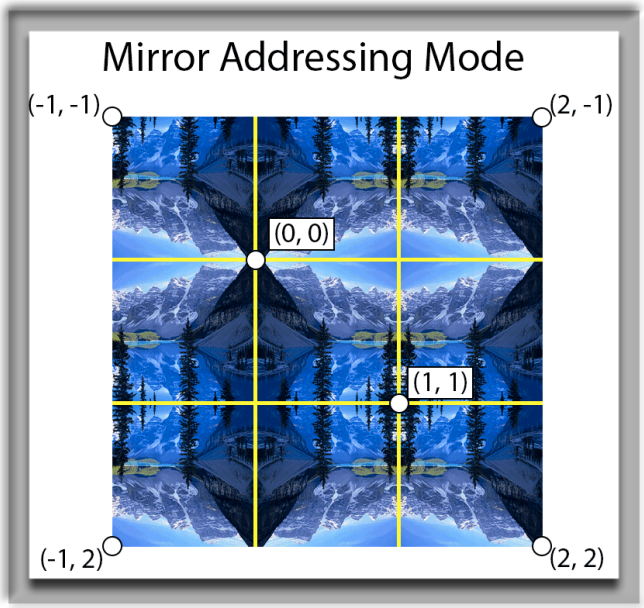
address mode set the behaviour of mapping outside the normalised coordinate system

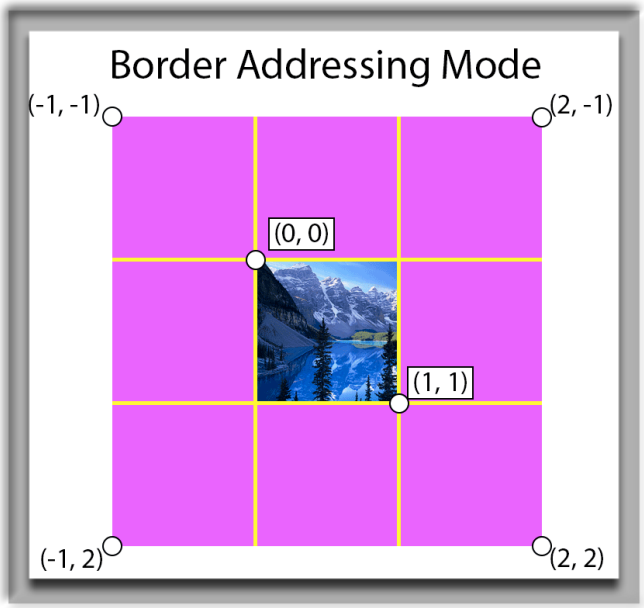
address modes are

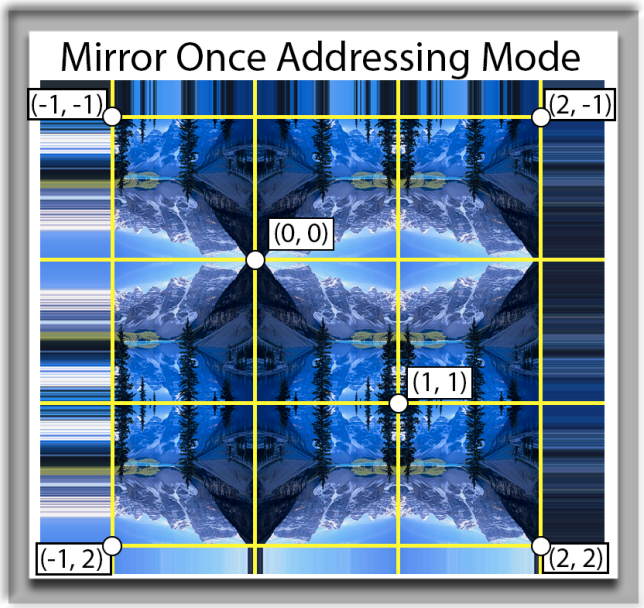
- WRAP
- CLAMP
- MIRROR
- BORDER
- MIRROR ONCE


Wrap Addressing Mode
(-1, -1)
(2, -1)
(0, 0)
(1, 1)
(-1, 2)
(2, 2)


Clamp Addressing Mode
(-1, -1)
(2, -1)
(0, 0)
(1, 1)
(-1, 2)
(2, 2)

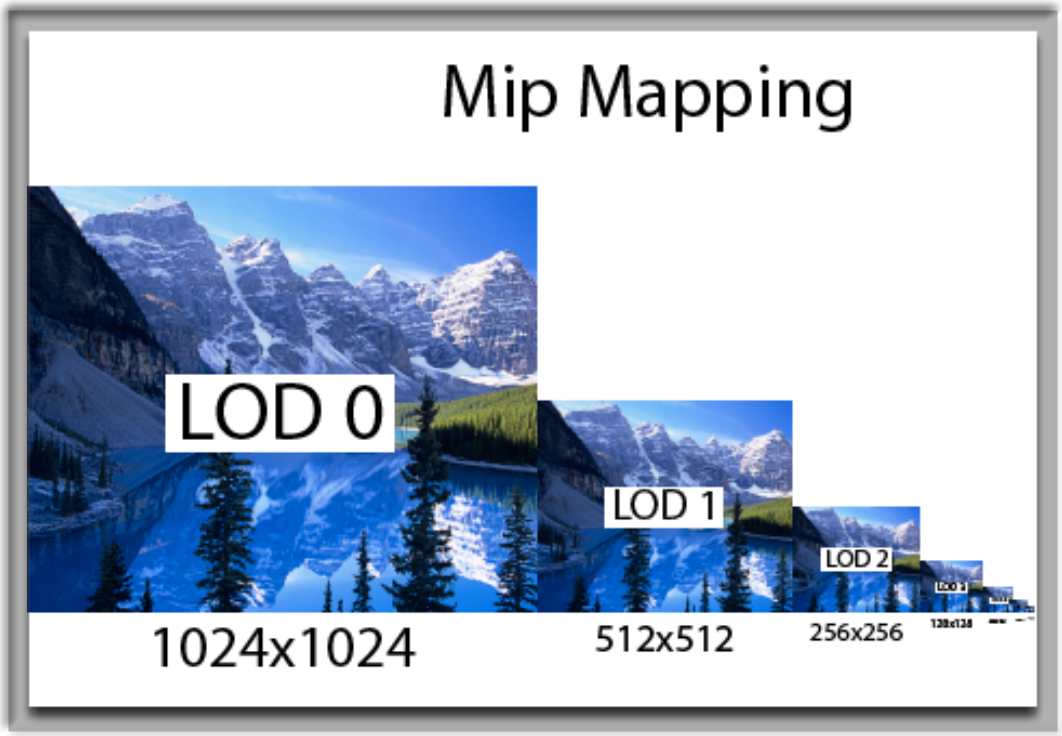

Mirror Addressing Mode
(-1, -1)
(2, -1)
(0, 0)
(1, 1)
(-1, 2)
(2, 2)


Border Addressing Mode
(-1, -1)
(2, -1)
(0, 0)
(1, 1)
(-1, 2)
(2, 2)


Mirror Once Addressing Mode
(-1, -1)
(2, -1)
(0, 0)
(1, 1)
(-1, 2)
(2, 2)

Sampler State II

Mip Mapping creates smaller variants of a texture needs for better performance and caching issues

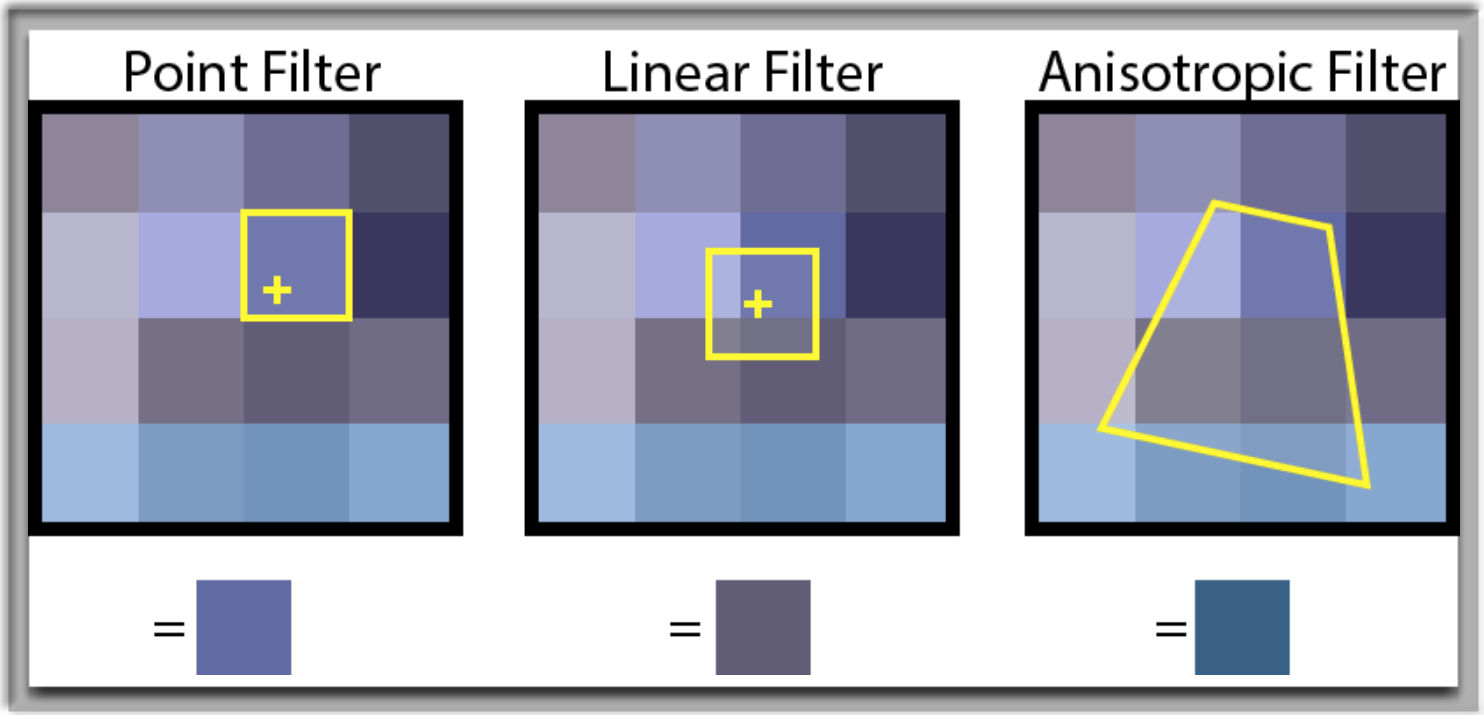


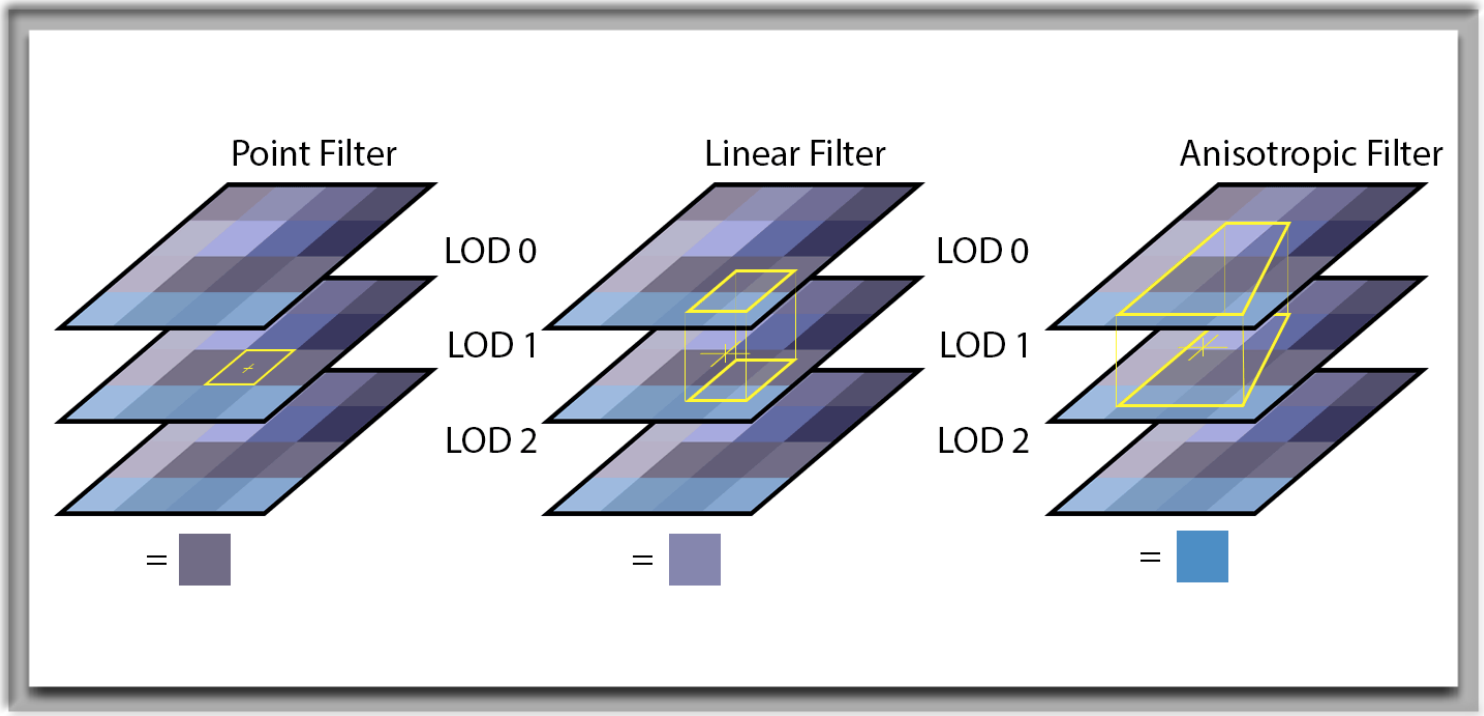
Sampler State III

a filter distinguish the sampled color of pixel

modes are

- point filtering
- linear filtering
- anisotropic filtering


Point Filter
Linear Filter
Anisotropic Filter
=
=
=


Point Filter
Linear Filter
Anisotropic Filter
LOD 0
LOD 1
LOD 2
LOD 0
LOD 1
LOD 2
=
=
=

Coding Time

Let’s texturing a mesh