

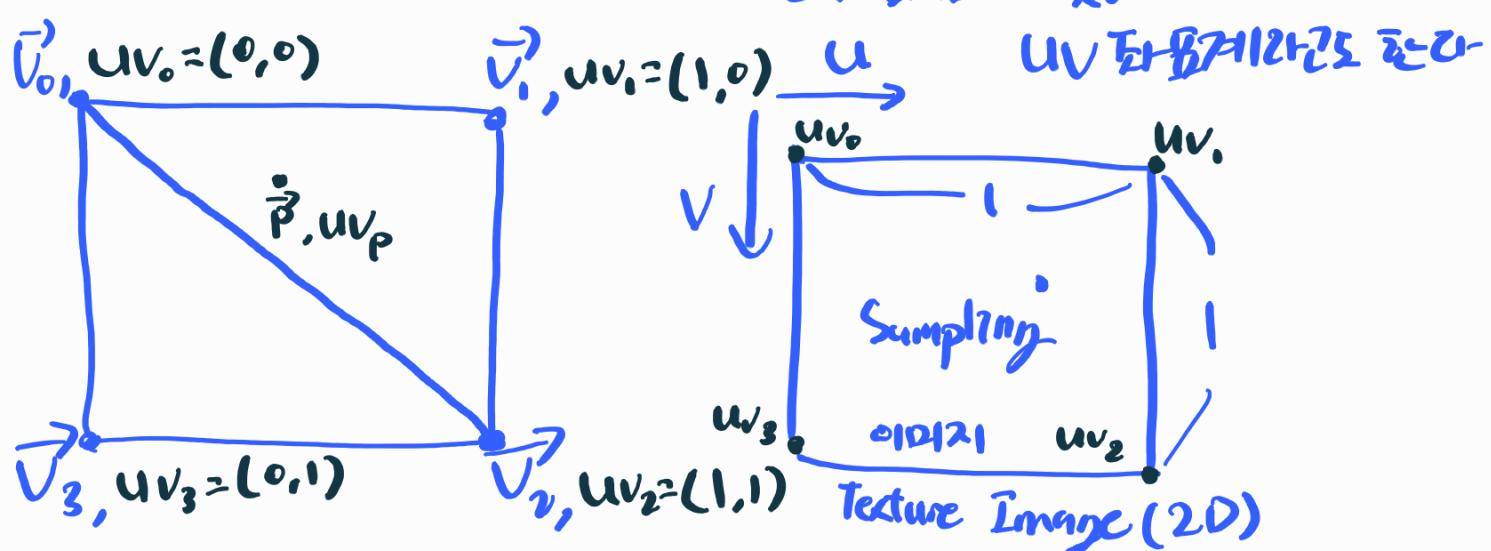
## 텍스처링

기본적인 원리 - 모델 자체를 정교하게 만드는 대신 이미지를 덮친다.

Ex) 2D Box에 사진을 덮여 정교한 물체인 것처럼 표현한다.  
정교한 모델을 렌더링하는 것보다 훨씬 빠르다.

이전 득지중심좌표에서 득지중심을 Interpolate하여 색을 결정한다면  
Texturing에서 득지중심좌표를 이용해 이미지의 어느 위치에서  
색깔을 읽을지 결정한다.

이때 사용하는 좌표를 Texture Coordinate(텍스처 좌표)라 한다.  
스크린좌표와 비슷하다



Texturing은 각각의 Vertex에 UV 좌표를 지정한다

UV<sub>p</sub>는 UV<sub>0</sub>, UV<sub>1</sub>, UV<sub>2</sub>의 득지중심좌표를 이용해 계산된다.

2단계 Texture Image의 UV<sub>p</sub> 좌표의 색을 가져온다.

V<sub>0</sub>, V<sub>1</sub>, V<sub>2</sub>를 이용해 Weight(가중치)를 구한뒤 UV<sub>p</sub> 좌표를 계산한다.

Sampling은 Texture의 좌표에서 색을 가져오는 것을 의미한다

UV 좌표계(Texture 좌표계)에서는 이미지 해상도마다 계산되는 걸  
방지하기 위해 해상도와 관계없이 Height, Width를 1로 가정한다  
(이미지 해상도와 상관없이 Texture를 사용하기 위해 병행화 처리)

그러나 Aspect Ratio(비율)이 1:1이 아닌 이미지는 원래 이미지의 비율이  
유지되지 않고 강제로 1:1로 바뀐다

샘플링(Sampling)은 크게 2가지 방식으로 구별할 수 있고

↳ Texture 좌표에서 색을 가져온다.

## 1) Point Sampling (Nearest-Neighbor Sampling)

↳ 픽셀들이 있는 모서리에 대한 색(도트 이미지, 픽셀아트와 비슷)

텍스처 좌표계 범위 UV  $[0, 1] \times [0, 1]$ 로 이미지 배경으로 상관없이 사용하기 위해 지정된다. 이를 실제 Index 값으로 사용하기 위해 좌표계 변환을 해주어야 한다.  
텍스처 좌표계  $\rightarrow$  이미지 좌표계 XY  $[-0.5, width-1+0.5] \times [-0.5, height-1+0.5]$   
배열 Index의 정수 범위  $i, j [0, width-1] \times [0, height-1]$

UV 좌표계

4x4의 간단한 텍스처 이미지

UV, (0,0), (-0.5, -0.5) 이미지 좌표 중심 (텍스처 좌표계  $\rightarrow$  이미지 좌표계)

UV, (1,0)

각 Pixel의 중앙에 색이 저장되어 있다면 색을

0~3까지의 Index 번호

0.5씩 확장한 범위

샘플링을 해야될  
UV 좌표

이미지 좌표계에서  
좌표  
Round()로  
배열 index ij로  
변환해 색을  
가져온다

UV, (0,1)

UV, (1,1),

(width-1+0.5, height-1+0.5)

UV 좌표계의 어떤 지점이든지 샘플링을 해야하는 것이다.

Point Sampling - 어떤 지점이든지 가장 가까운 Pixel의 색상을 가져온다.

↳ UV의 색상값으로 첫번째 Pixel 색을 가져온다(반올림, round() 사용)

2차원 공간에서 사용할 Pixel의 가운데에 색깔 값이 저장되어 있으므로 가장하고 앞가운데 실제로 Indexing의 범위는 [0~3]이 아니라 전부다 0.5씩 범위가 넓어져야 한다.

임의의 UV 좌표를 입력 받으면 UV 좌표에 해당하는 Pixel 값을 샘플링해서 색상을 해주어야 한다. 이를 위해 UV 좌표계의 범위를 Image 좌표계의 범위로 좌표계 변환을 해주어야 한다. (2차원 좌표계 변환과 비슷하다)

E-UV 좌표계 |  $\rightarrow$  이미지 좌표계 변환

$$0 \leq u, v \leq 1, -0.5 \leq x, y \leq \text{width}-1+0.5, \text{height}-1+0.5$$

$$0 \leq u, v \leq 1, -0.5 \leq x, y \leq \text{width}-0.5, \text{height}-0.5$$

$$uv = (0,0) \rightarrow xy = (-0.5, -0.5)$$

$$uv = (1,1) \rightarrow xy = (\text{width}-0.5, \text{height}-0.5)$$

$$x = u \times \text{width}-0.5, x \in [-0.5, \text{width}-1+0.5]$$

$$y = v \times \text{height}-0.5, y \in [-0.5, \text{height}-1+0.5]$$

$$u=0, x=0 \times \text{width}-0.5 \quad v=0, y=0 \times \text{height}-0.5 \\ = -0.5 \quad \quad \quad = -0.5$$

$$u=1, x=1 \times \text{width}-0.5 \quad v=1, y=1 \times \text{height}-0.5 \\ = \text{width}-0.5 \quad \quad \quad = \text{height}-0.5$$

$$uv \times \text{vec2}(\text{width}, \text{height}) - \text{vec2}(0.5) = xy$$

결과적으로 x, y의 범위는 width, height이 됨.

x의 범위(길이는 width-1+0.5 - (-0.5), End - begin  
 $= \text{width}-1+1 = \text{width}$ )

y의 범위(길이는 height-1+0.5 - (-0.5) = height)

이미지 좌표계  $\rightarrow$  배열 인덱스 (색이 Pixel의 중심에 있다고 가정)

배열의 범위  $[0, \text{width}-1] \times [0, \text{height}-1]$

$$\bar{x} = \text{round}(x)$$

$$\bar{y} = \text{round}(y)$$

## 2) Linear Sampling - Linear Interpolation 이용

선형보간으로 2차원 이미지 색상 부드럽게 표현된다.

UV 좌표계

4x4의 간단한 텍스쳐 이미지

UV<sub>0</sub>(0,0), (-0.5, -0.5)

이미지 좌표 중심 (텍스처 중심) → 0.5(2, 2)  
좌표계(1,1)

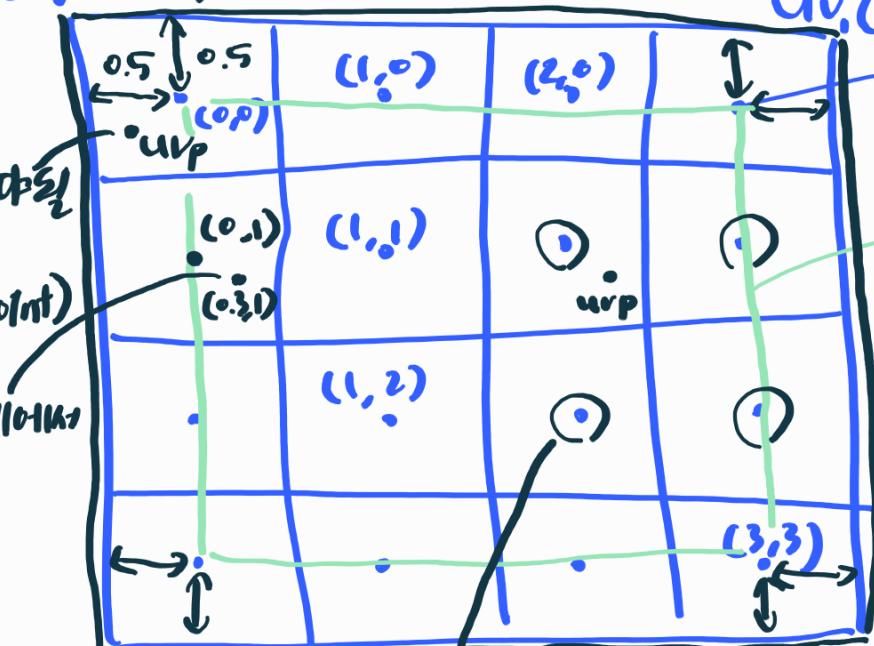
UV<sub>1</sub>(1,0)

각 Pixel의 중심에 색상  
저장되어 있으므로 색상

샘플링을 해보자

UV 좌표  
(Sampling Point)

이미지 좌표계에서  
좌표



0~3까지의  
Index 번호

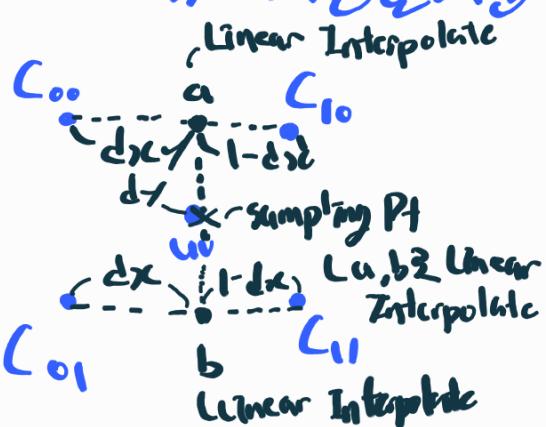
0.5씩 늘어나는 높이

UV<sub>2</sub>(1,1)

(width-1+0.5, Height-1+0.5)

Point Sampling에서 가장 가까운 Pixel 수만 가져온다

Linear Sampling에서 주변에 있는 Pixel들을 Linear Interpolation을 통해 섞어서 보여준다



1) UV 좌표를 Image 좌표로 변환

2) 변환된 좌표와 가까운 Pixel들 중 가장  
작은 Pixel을 찾는다 (floor() 함수 사용)  
내장

3) 결국 L00의 값을 알게 되는데 L00을 얻면  
L10, L11, L01의 값을 알 수 있다

$$L_{00} = (i, j), L_{10} = (i+1, j), L_{11} = (i+1, j+1), L_{01} = (i, j+1)$$

Linear Interpolation하는 방식

L00와 L10을 interpolate하는 값 a와 L01과 L11을 interpolate하는 값 b를  
interpolate하는 Sampling Point의 값을 계산한다

Interpolation을 2번 반복해 2D interpolation을 Bilinear Interpolation이라 한다