CS559 Lecture 19-20: More Texture
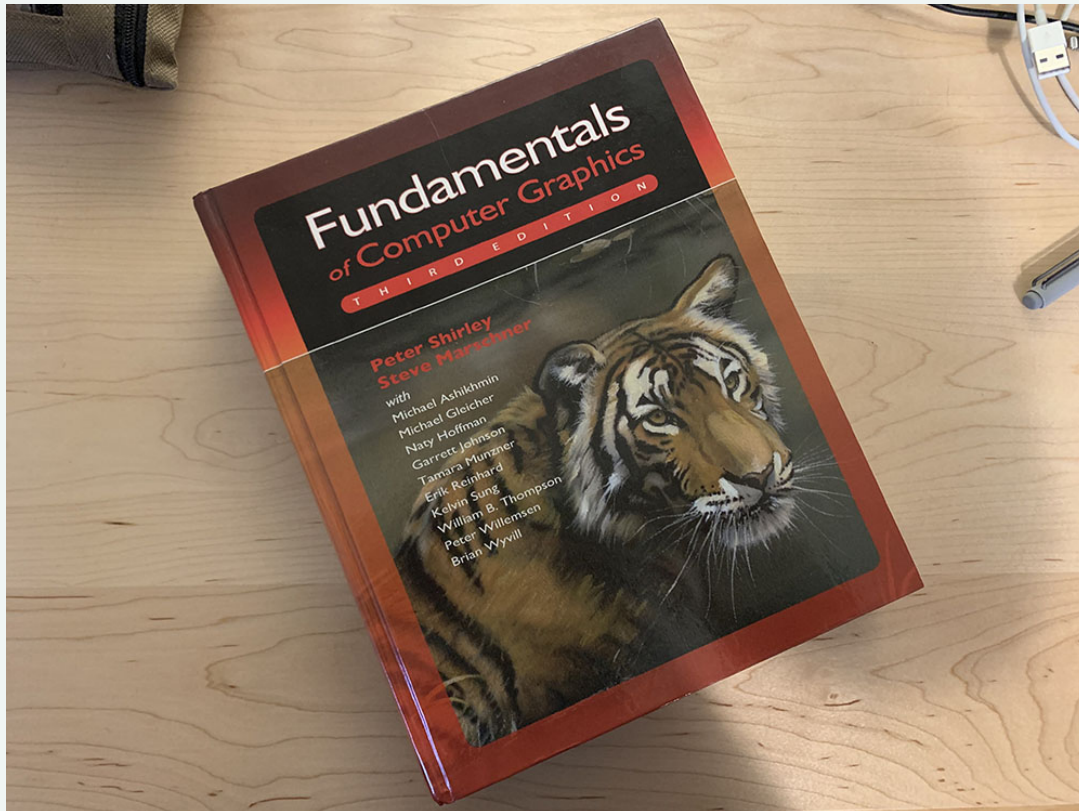
Part 1: Basic Texture Review
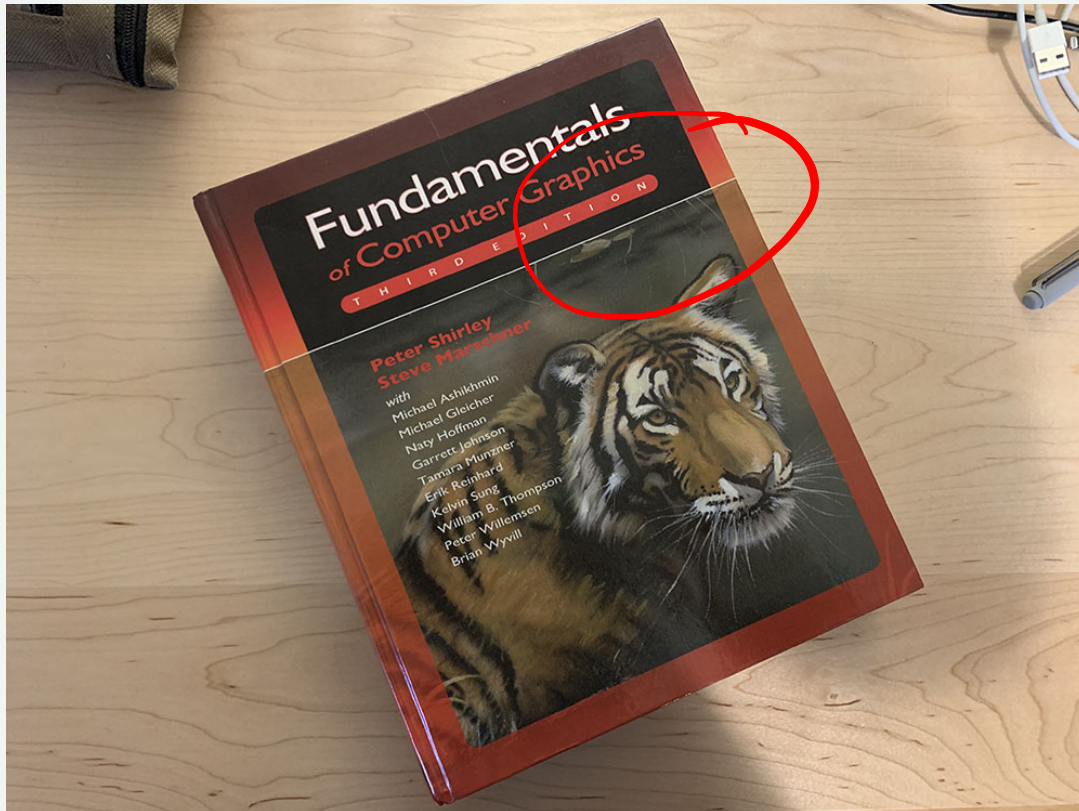
Motivation and Review

Why Basic Textures?

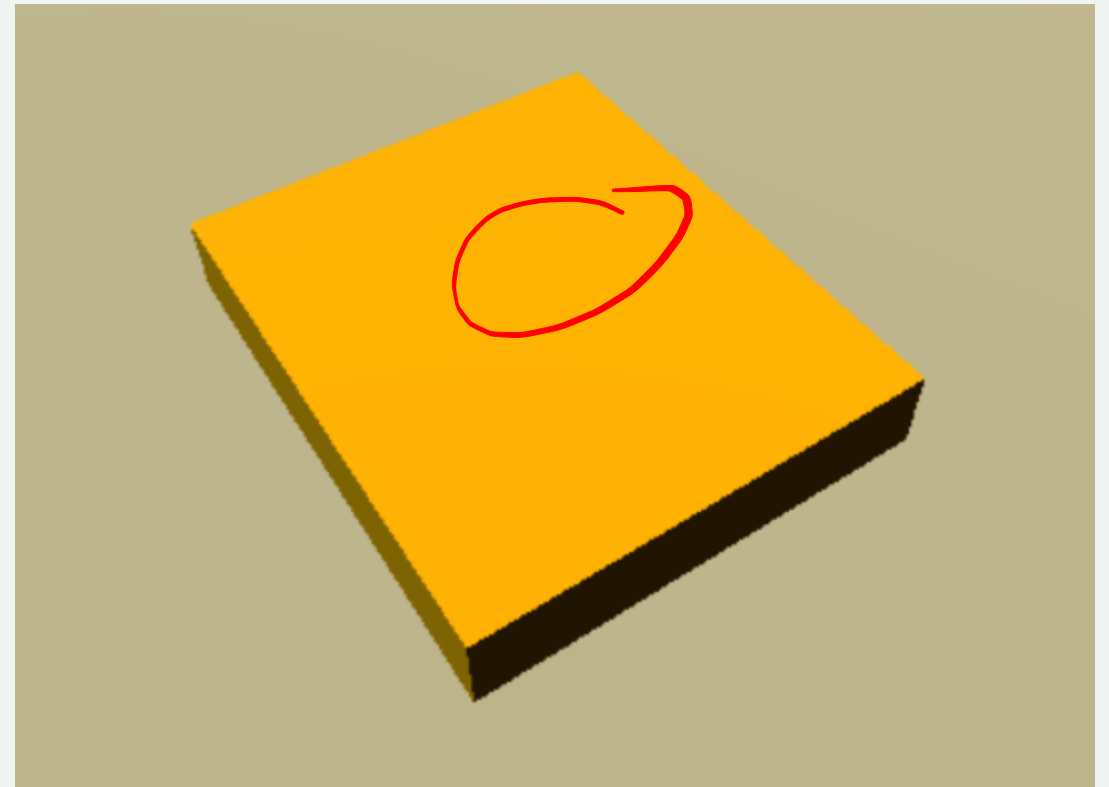Because real objects are interesting

Why Basic Textures?

Real objects are interesting

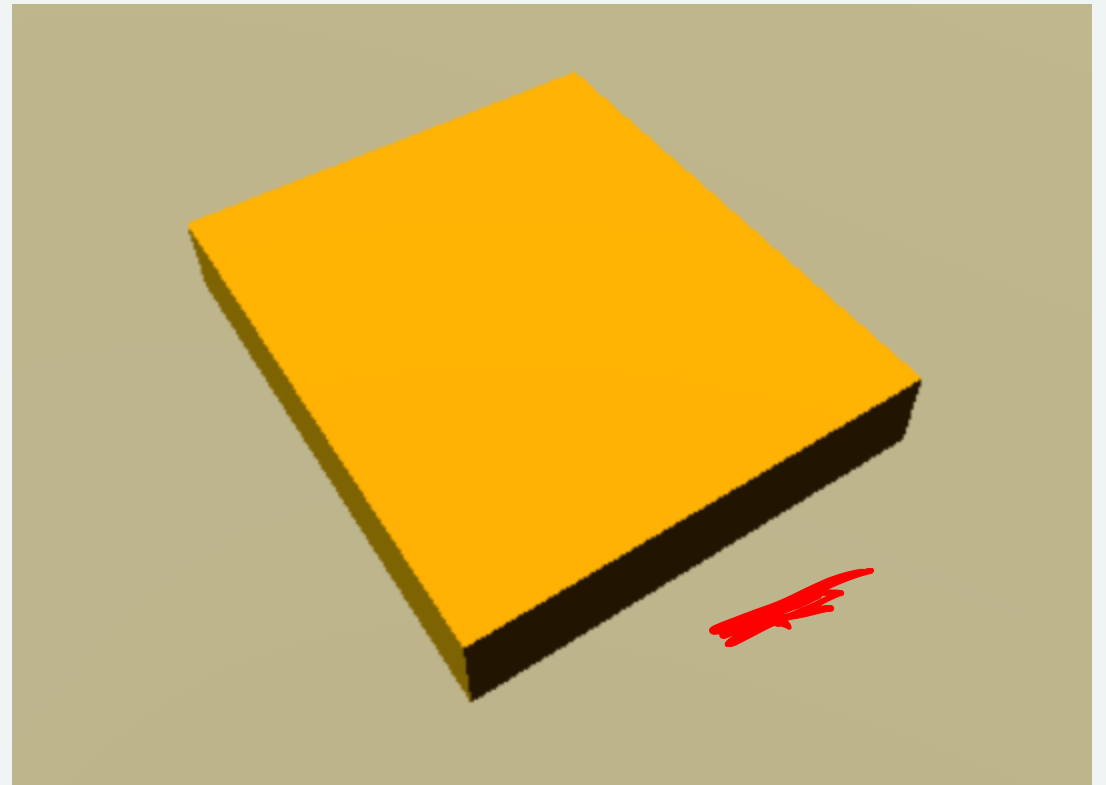Computer Graphics can be boring...

Why Basic Textures?

Even Colors can Help

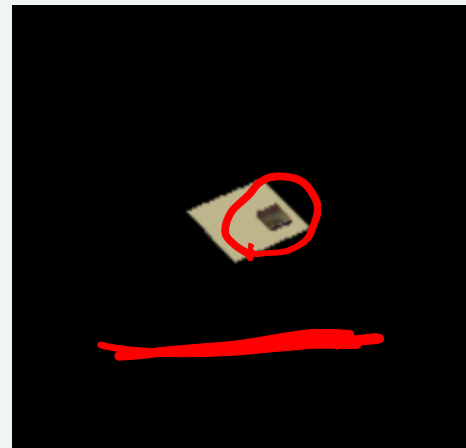Computer Graphics can be boring...

But Why Textures?

Even Colors can Help

- Easy to get image
- Hard to model detals
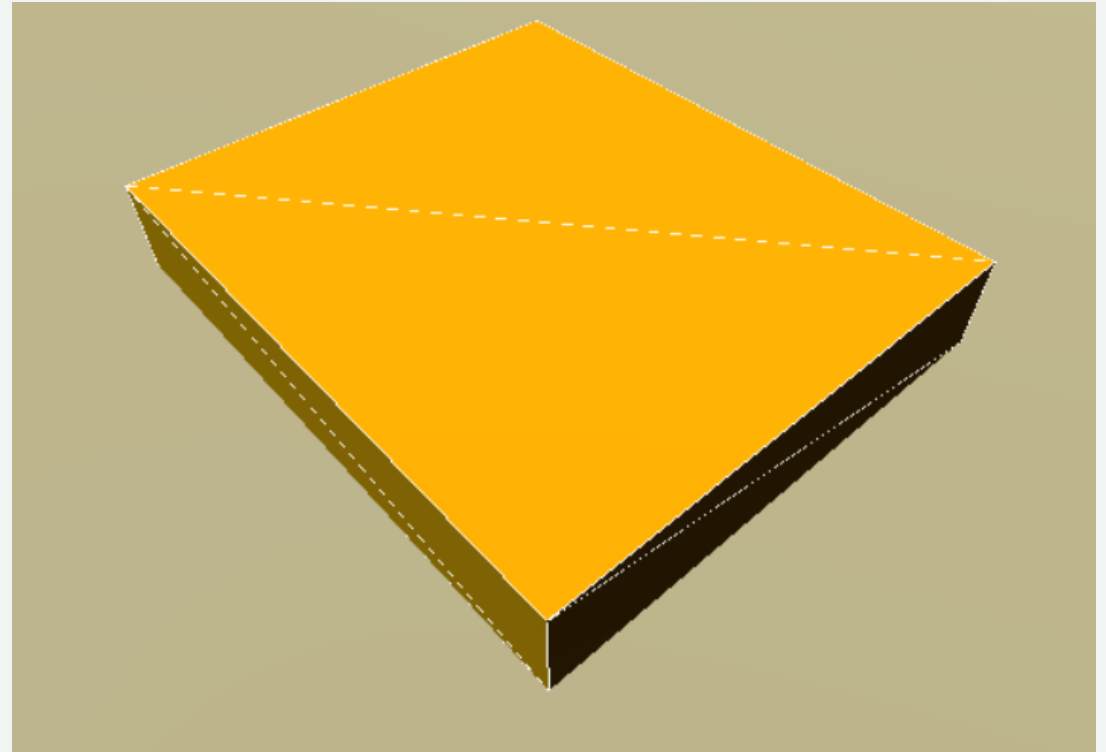- Easy to make simple geometry
- Proper sampling

How To Do Basic Textures?

1. Make Some Geometry

2. Get a Picture

3. Get the picture in the right form

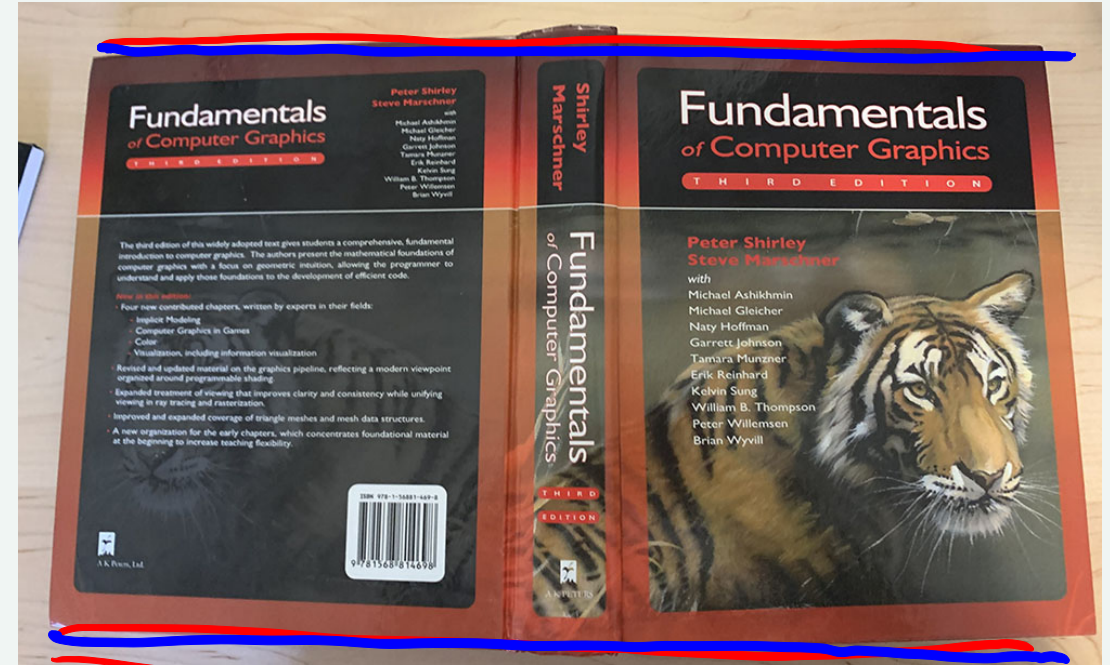4. Assign UV values to vertices

5. Enable Texturing

Geometry

1. **Make Some Geometry**

2. Get a Picture

3. Get the picture in the right form

4. Assign UV values to vertices

5. Enable Texturing

A Picture

1. Make Some Geometry

2. **Get a Picture**

3. Get the picture in the right form

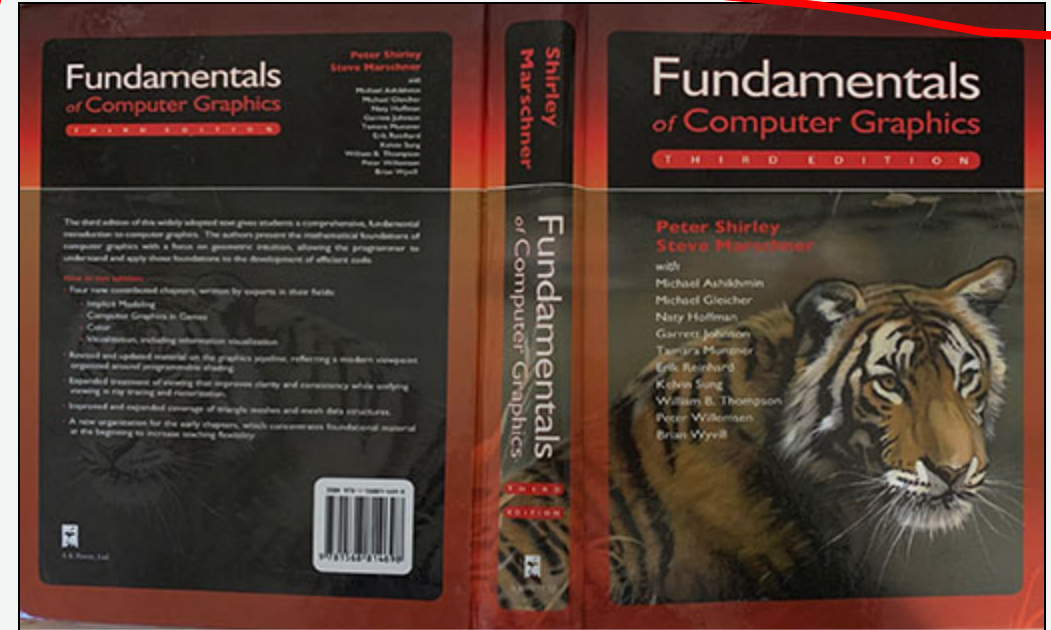4. Assign UV values to vertices

5. Enable Texturing

Can paint it yourself

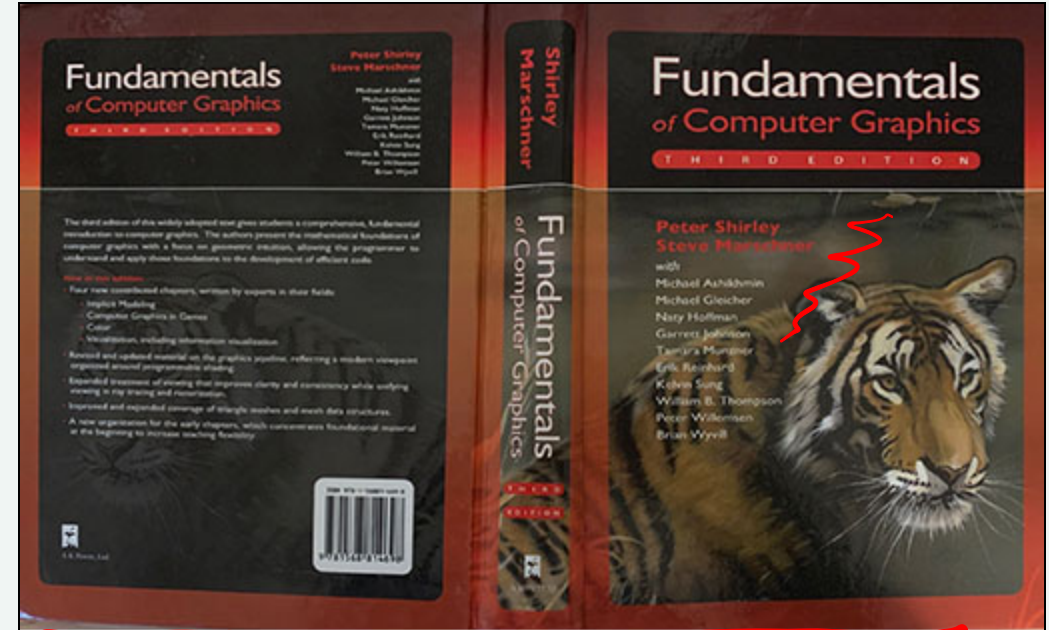Need to get things to match simple geometry

Process the Picture

1. Make Some Geometry

2. Get a Picture

3. **Get the picture in the right form**

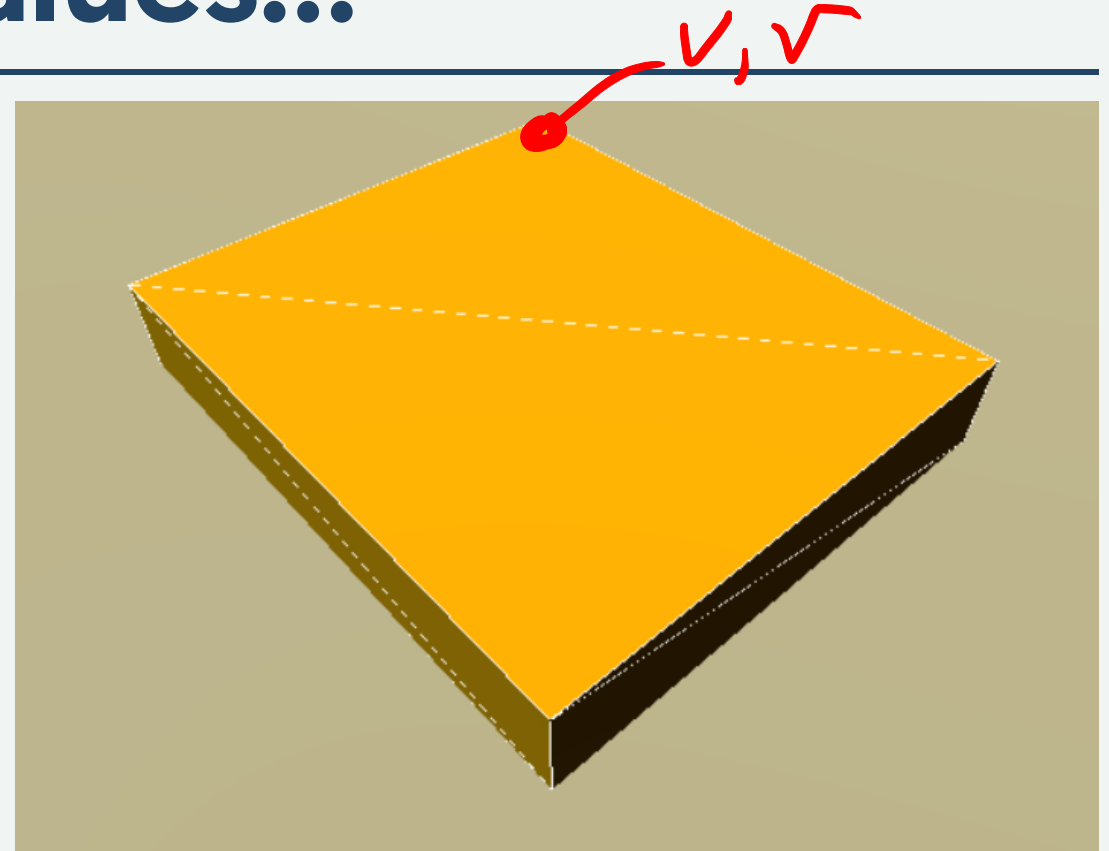4. Assign UV values to vertices

5. Enable Texturing

What do we need from a texture?

1. Square
2. Matches Simple Geometry
3. Minimal lighting
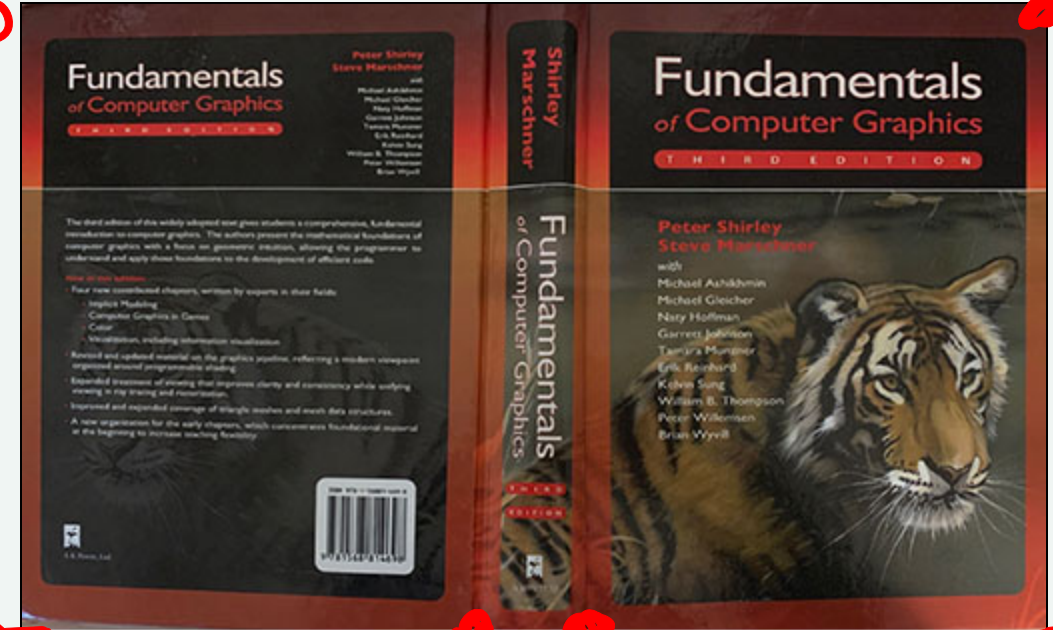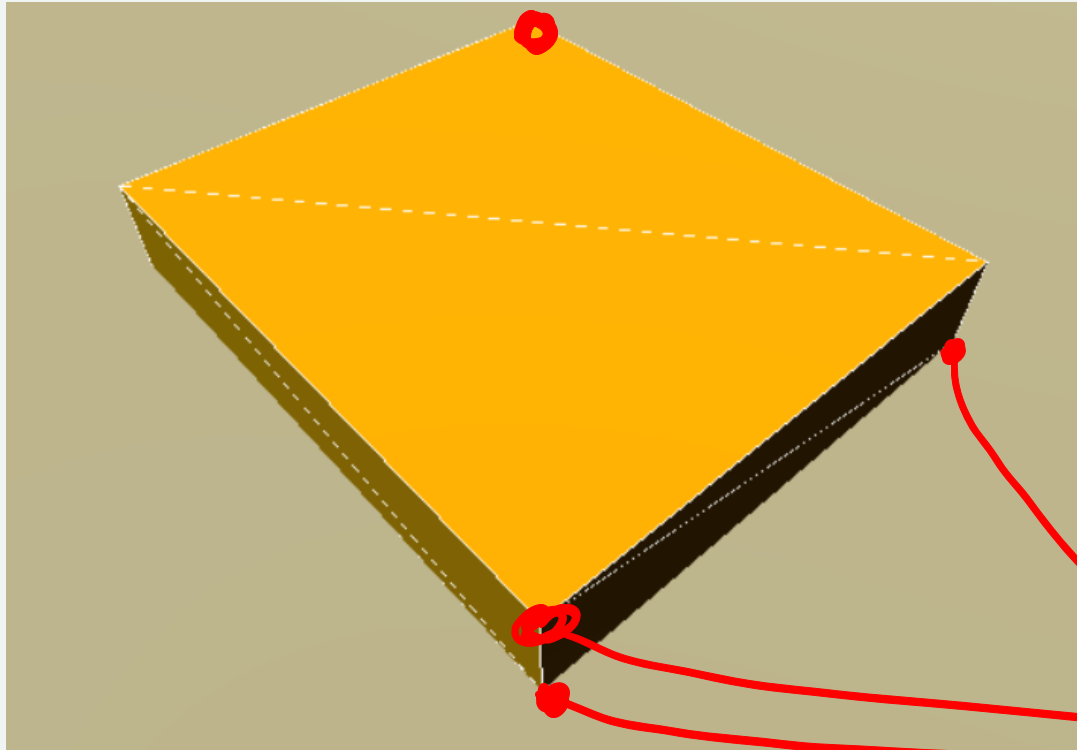4. Put lots of parts in one image

11

Getting those UV Values...

1. Make Some Geometry

2. Get a Picture

3. Get the picture in the right form

4. **Assign UV values to vertices**
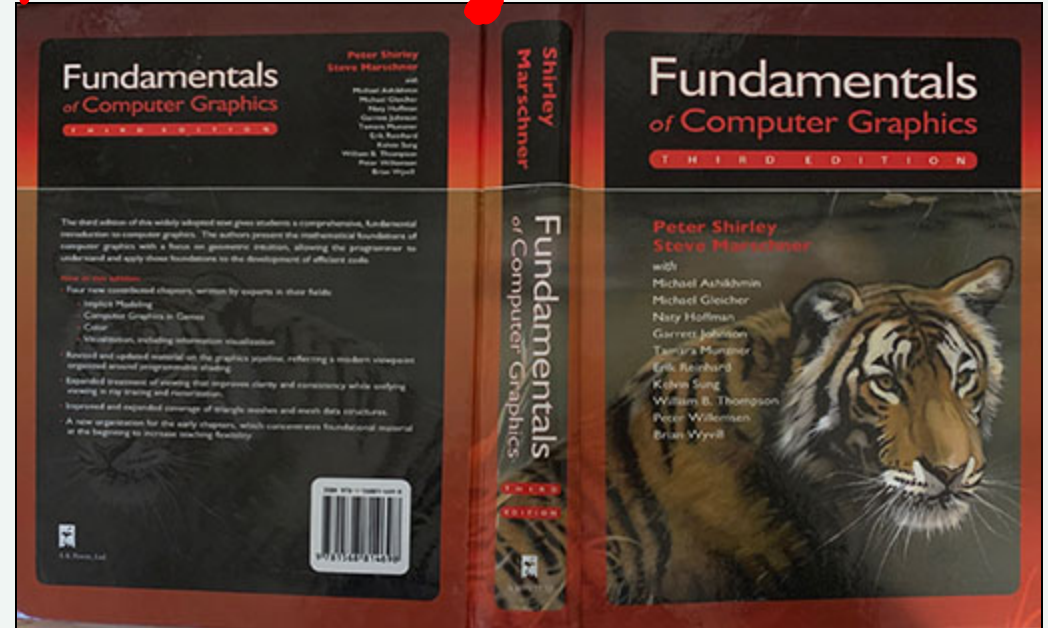
5. Enable Texturing

Finding UVs

Assign UV values to vertices

```
const vertexUVs = [
    // bottom (back of book)
    new T.Vector2(232/512,0),
    new T.Vector2(0       ,0),
    new T.Vector2(0,       311/512),
    new T.Vector2(232/512,311/512),
    // top (front of book)
    new T.Vector2(282/512, 0),
    new T.Vector2(512/512, 0),
    new T.Vector2(512/512,311/512),
    new T.Vector2(282/512,311/512),
]
```


14

Put into the (weird) THREE structures

```
const vertexUVs = [
    // bottom (back of book)
    new T.Vector2(232/512,0),
    new T.Vector2(0        ,0),
    new T.Vector2(0,        311/512),
    new T.Vector2(232/512,311/512),
    // top (front of book)
    new T.Vector2(282/512, 0),
    new T.Vector2(512/512, 0),
    new T.Vector2(512/512,311/512),
    new T.Vector2(282/512,311/512),
]
```

```
let face1V = [vertexUVs[0],
              vertexUVs[1],
              vertexUVs[2]
              ];
// ...
let faceVs = [face1V, face2V, ...];

//
geom.faceVertexUvs = [faceVs];
```

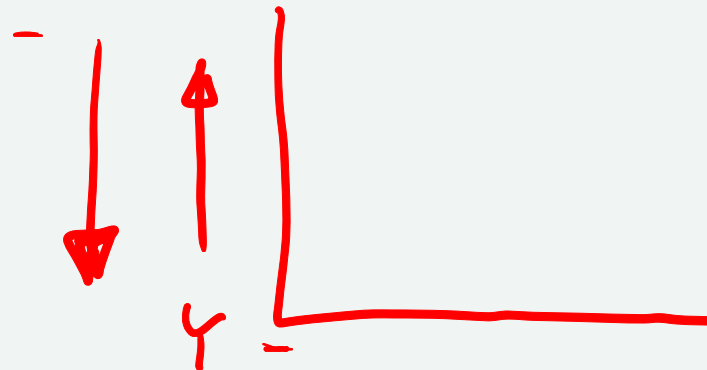# Why Per Face? - Vertex Splitting!
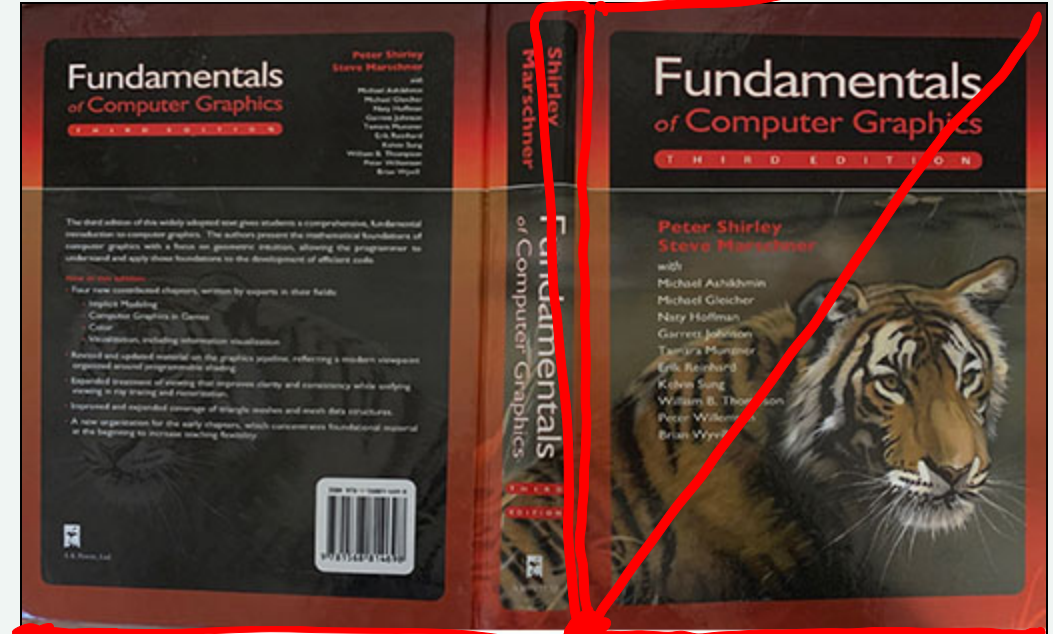
Put it together...

1. Make Some Geometry

2. Get a Picture

3. Get the picture in the right form

4. Assign UV values to vertices
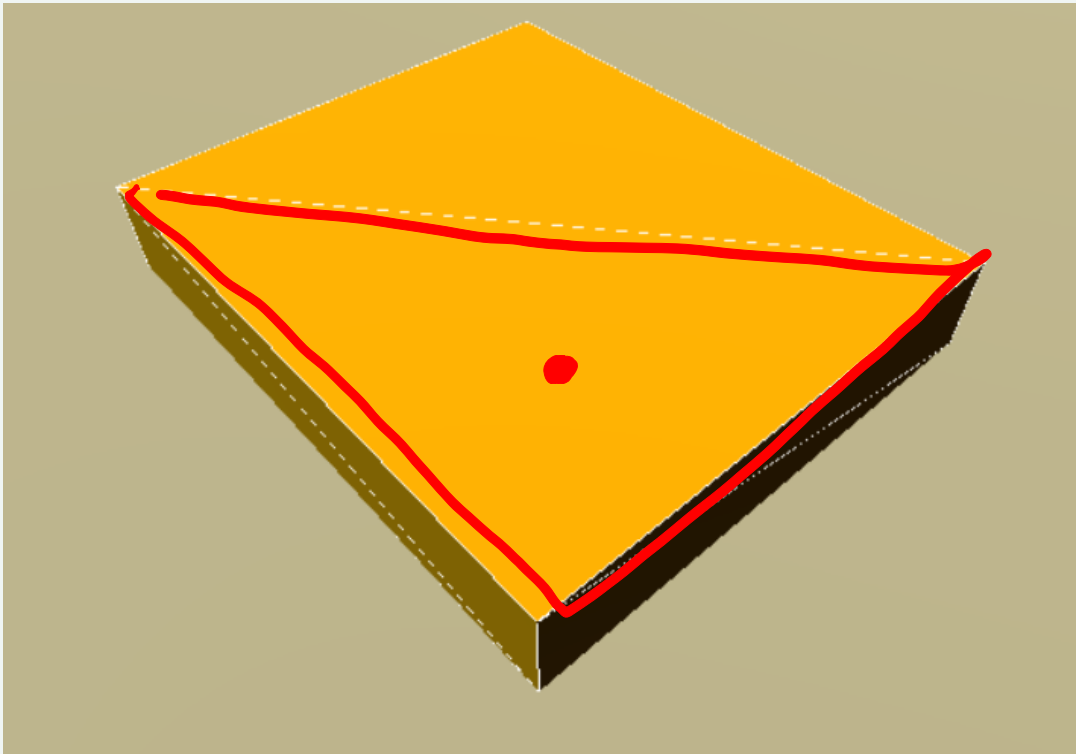
5. **Enable Texturing**

```
// load in the cover texture
let fcg = new T.TextureLoader().load("fcg-texture.jpg");
fcg.flipY = false;
```

```
let mat = new T.MeshStandardMaterial(
    {color:"white", map:fcg}
);
```
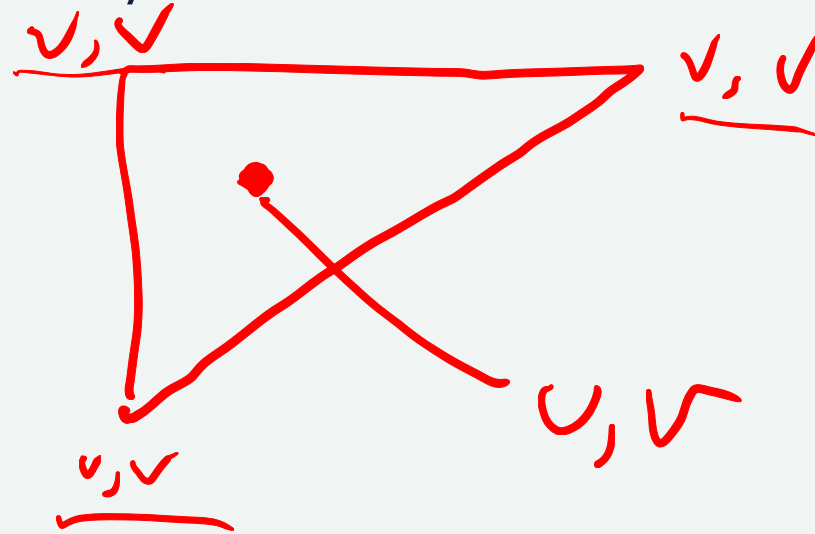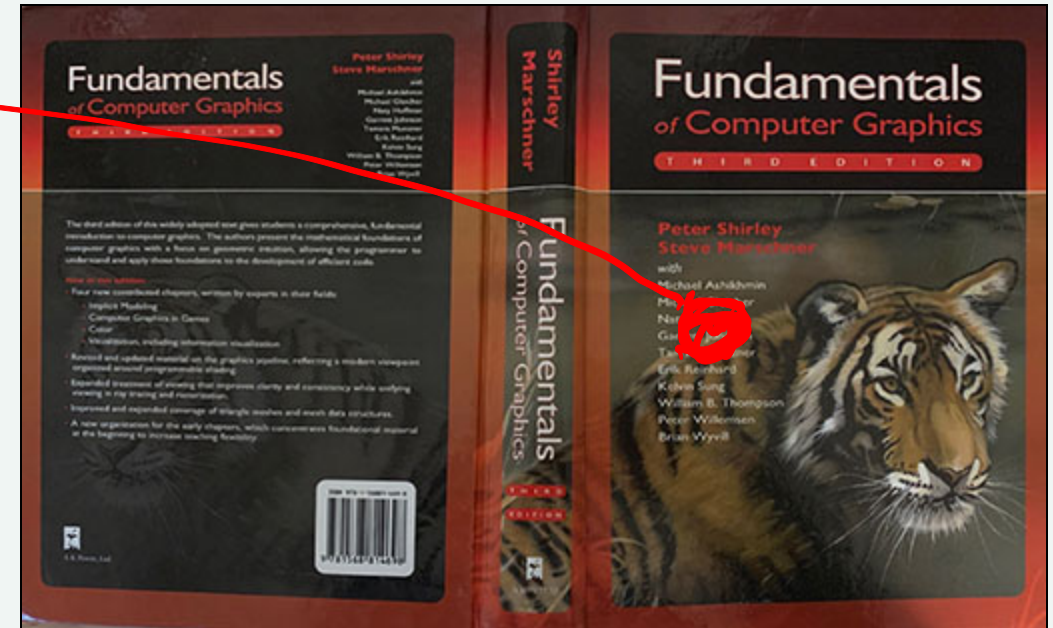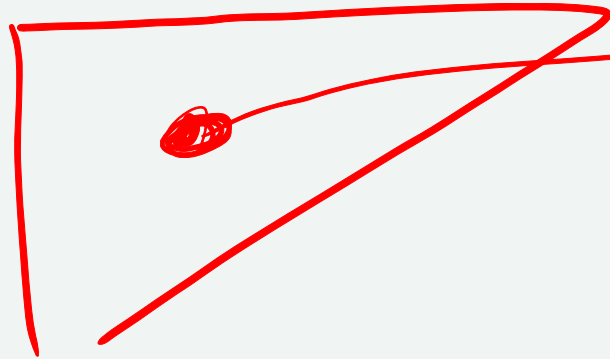
What the hardware does...
1. UV coordinates per pixel

Barycentric Coordinates

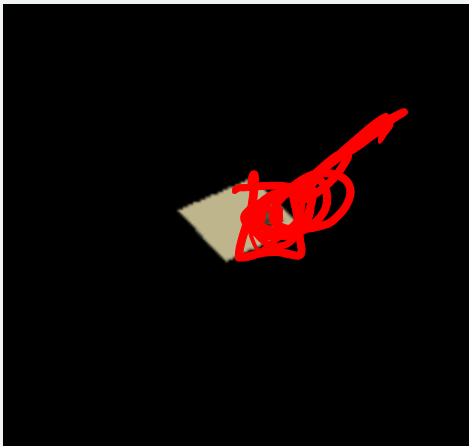# What the hardware does...
2. Texture Lookup

What the hardware does...
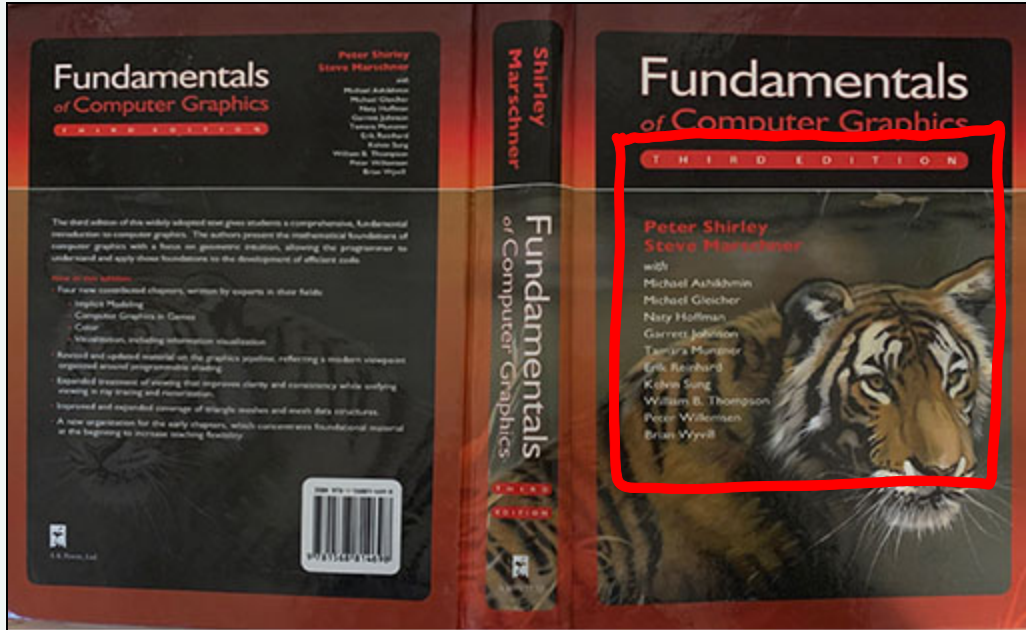# 3. Texture Filtering
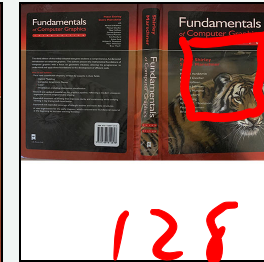
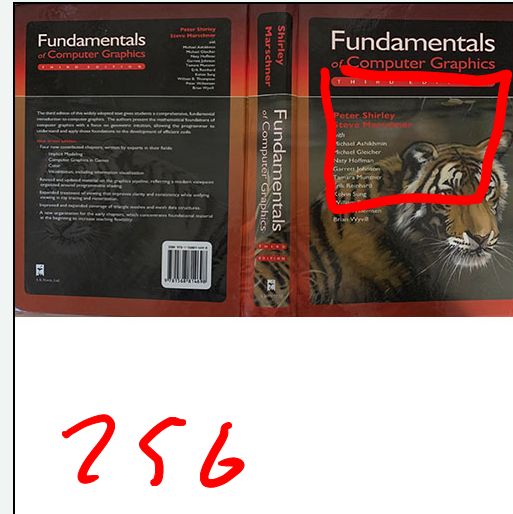Each pixel maps to many texels

Can't pick one!

Average region together!

20

Filtering Fast... Mip Maps

512 x 512

256

128

Once you have the color...

Use as the material color (for lighting)