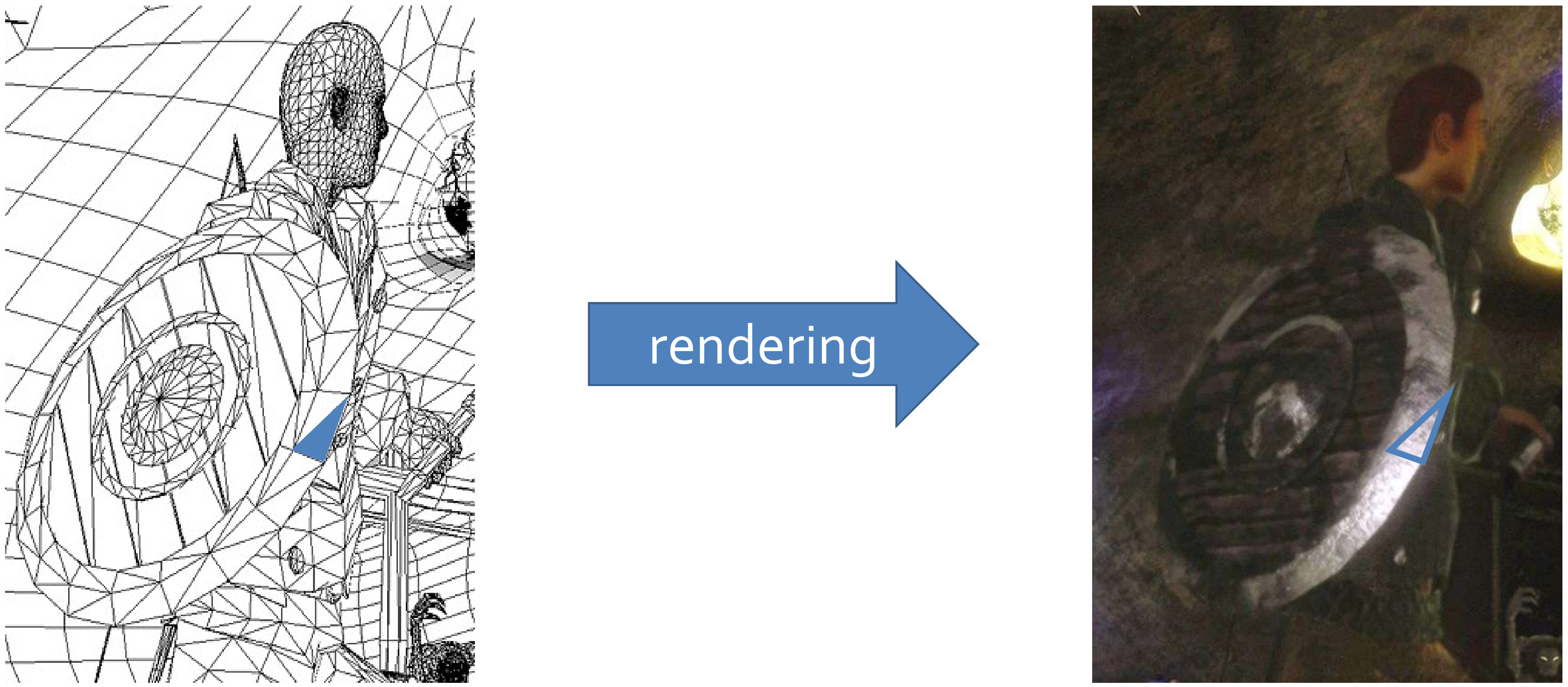
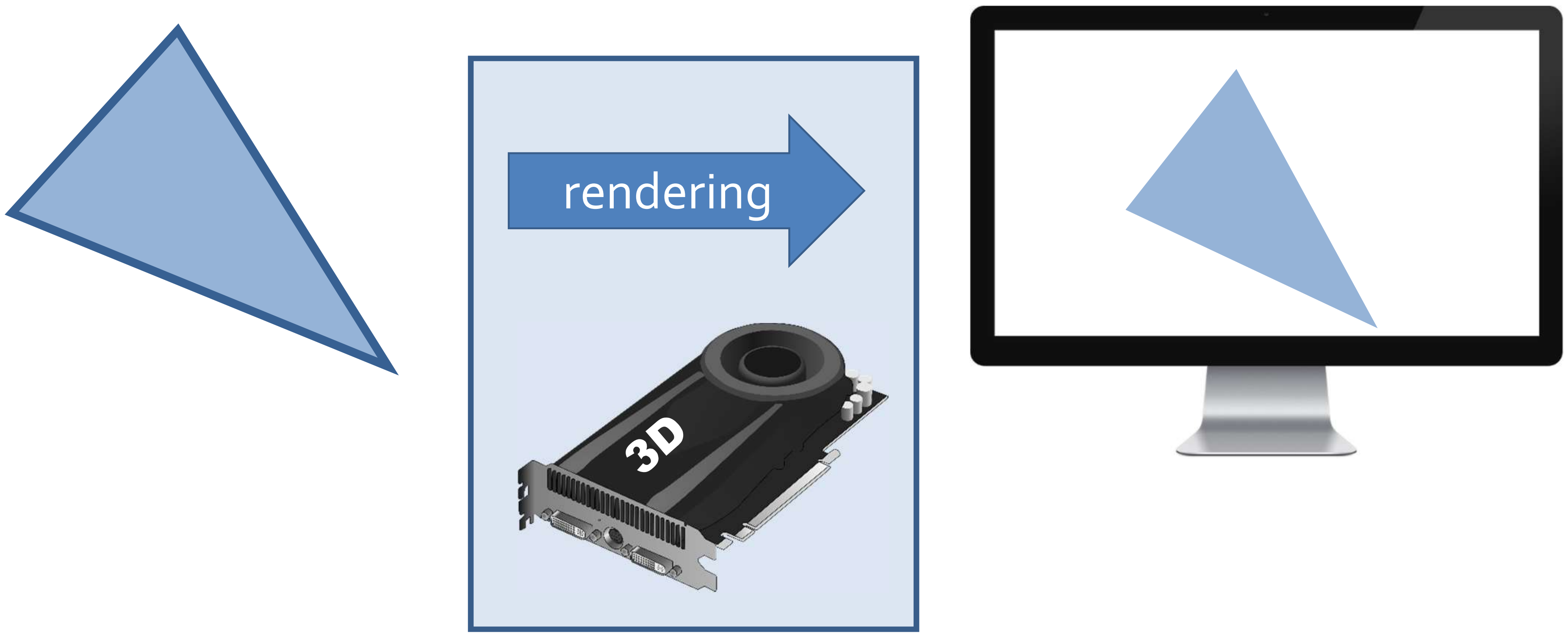
What is a Shader?

Every scene is constructed out of primitives

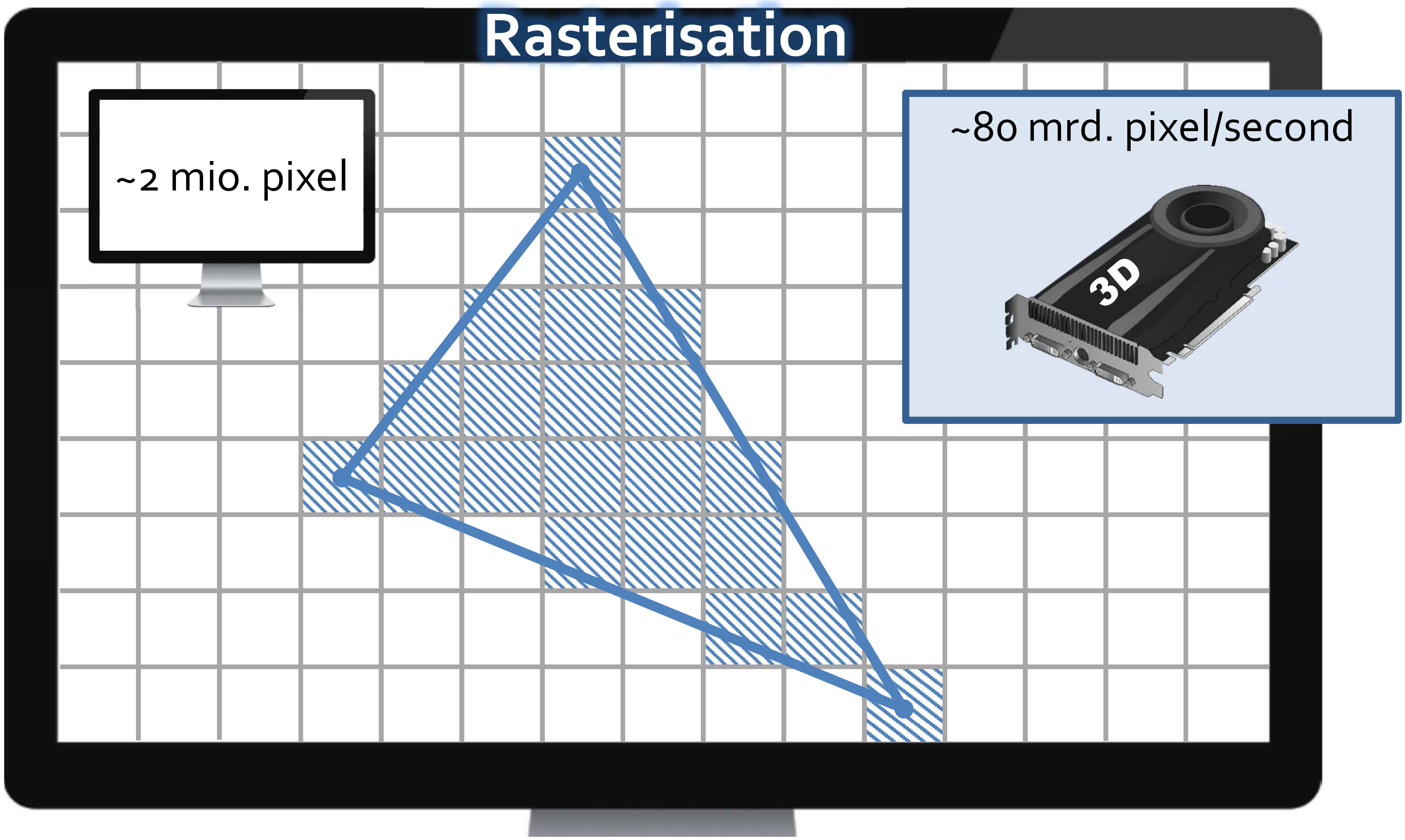
Rendering = Turn Primitives into Image



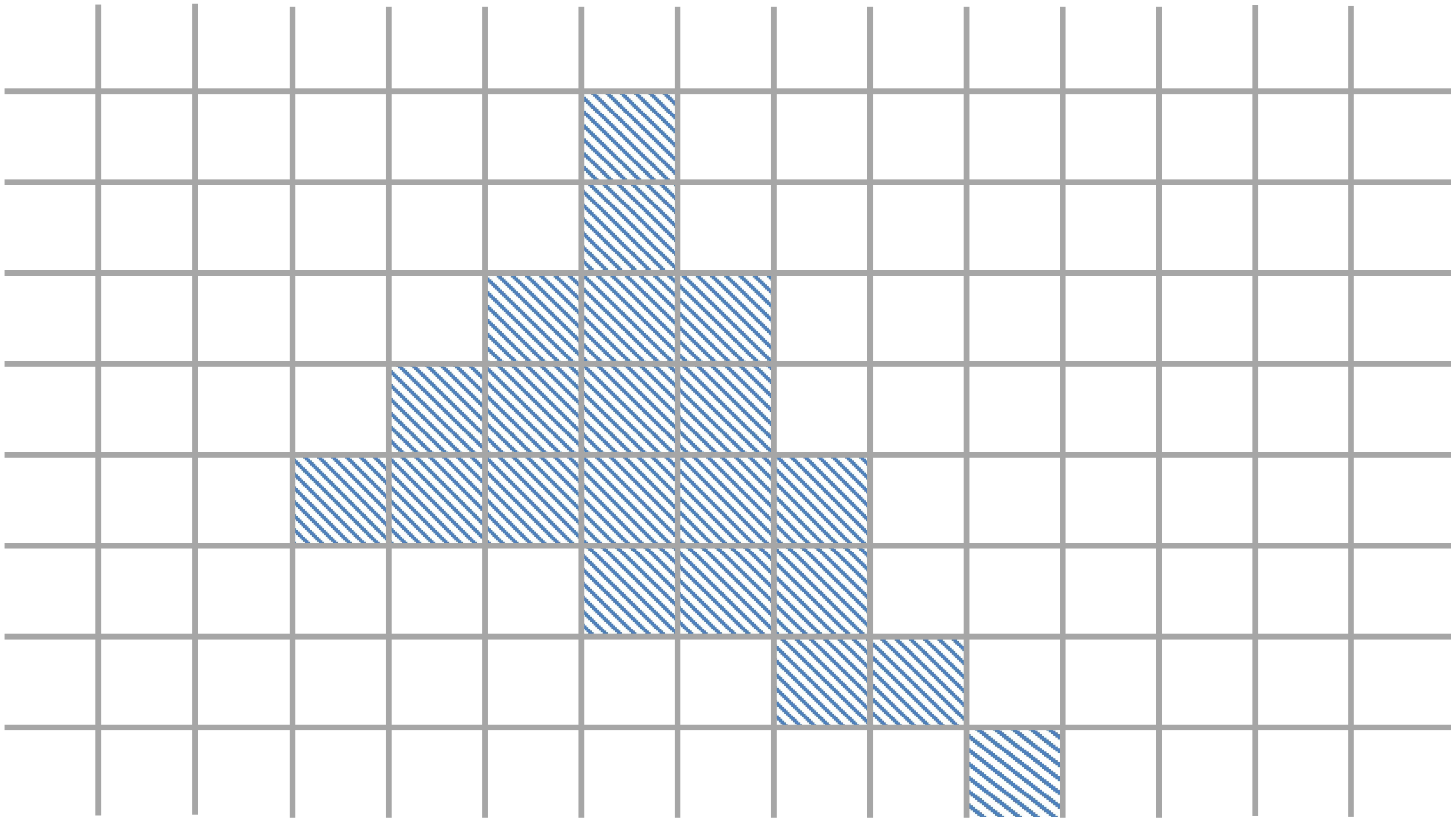
Rendering = Geometry into Image



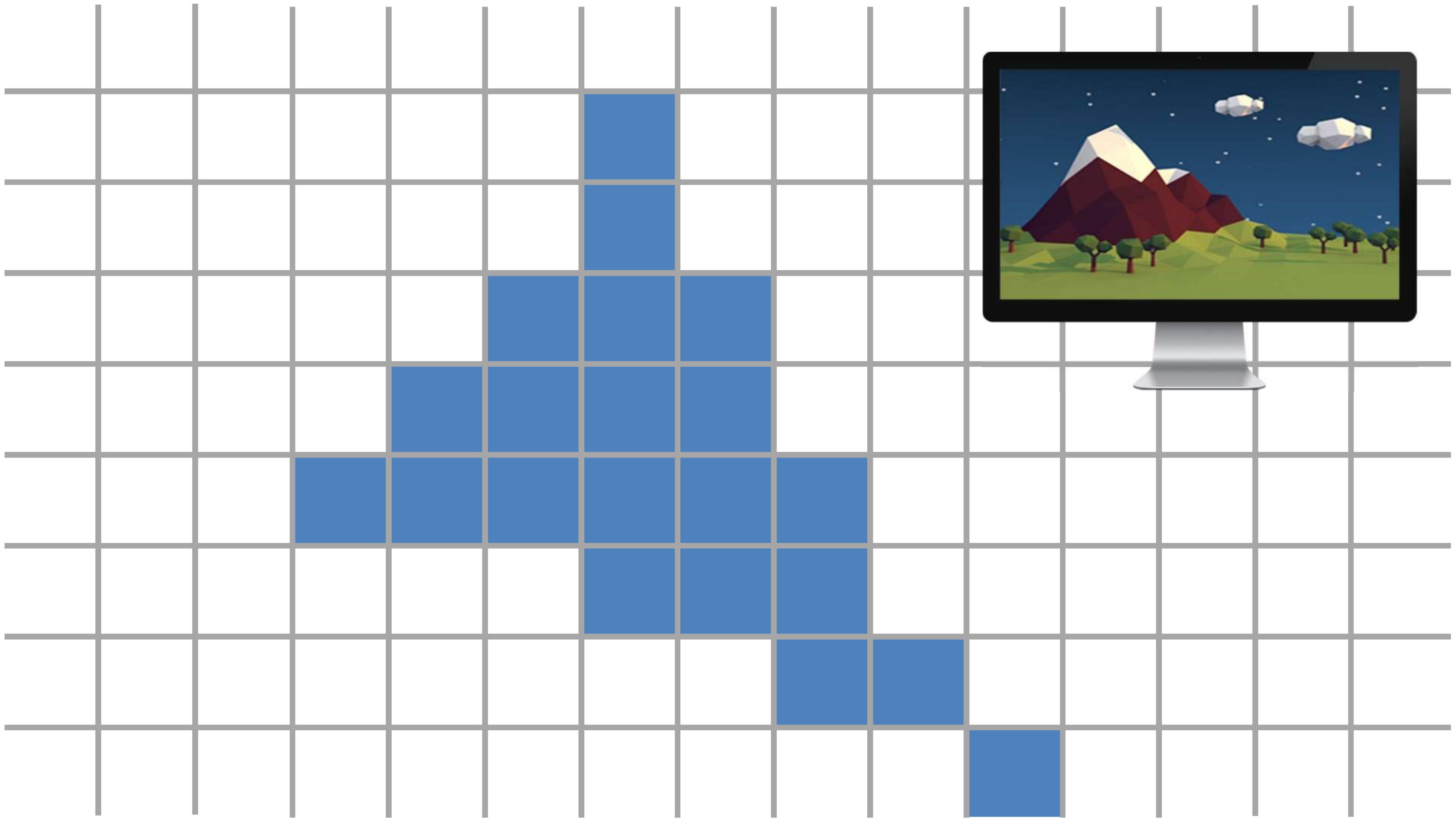
Rasterisation



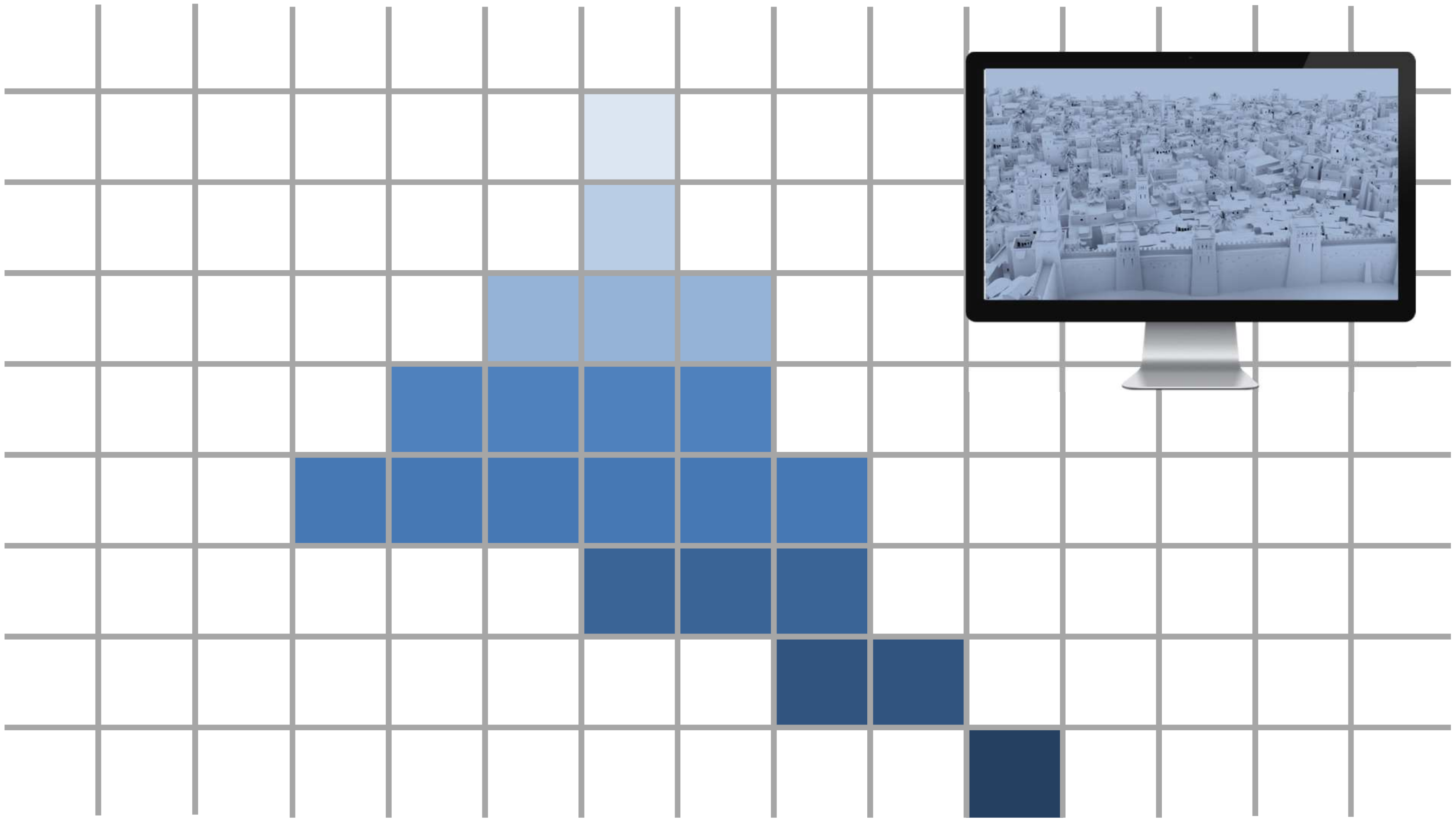
Color of a Pixel?



Color of a Pixel?



Color of a Pixel?



Color of a Pixel

- Great freedom required
  - Programmable
  - Fragment/Pixel shader

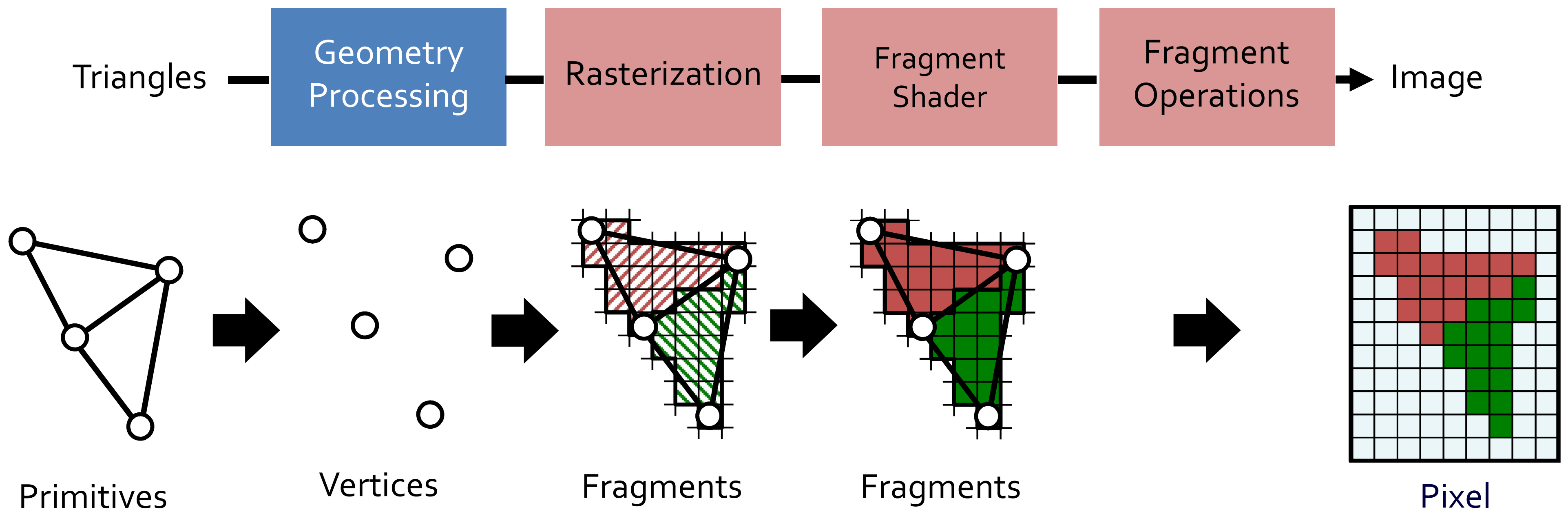
(Fragment) Shader decides the Color of a Pixel

- Program on graphics hardware

Rendering by Graphics Hardware

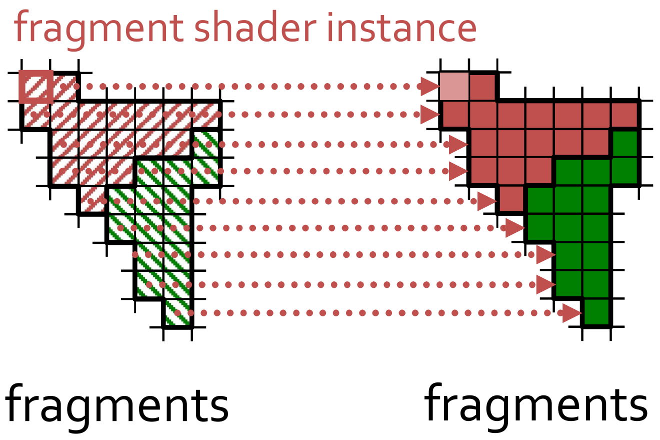
$$\begin{tikzcd}\text{Triangles} \arrow[r] & \text{Image}\end{tikzcd}$$

Rendering by Graphics Hardware

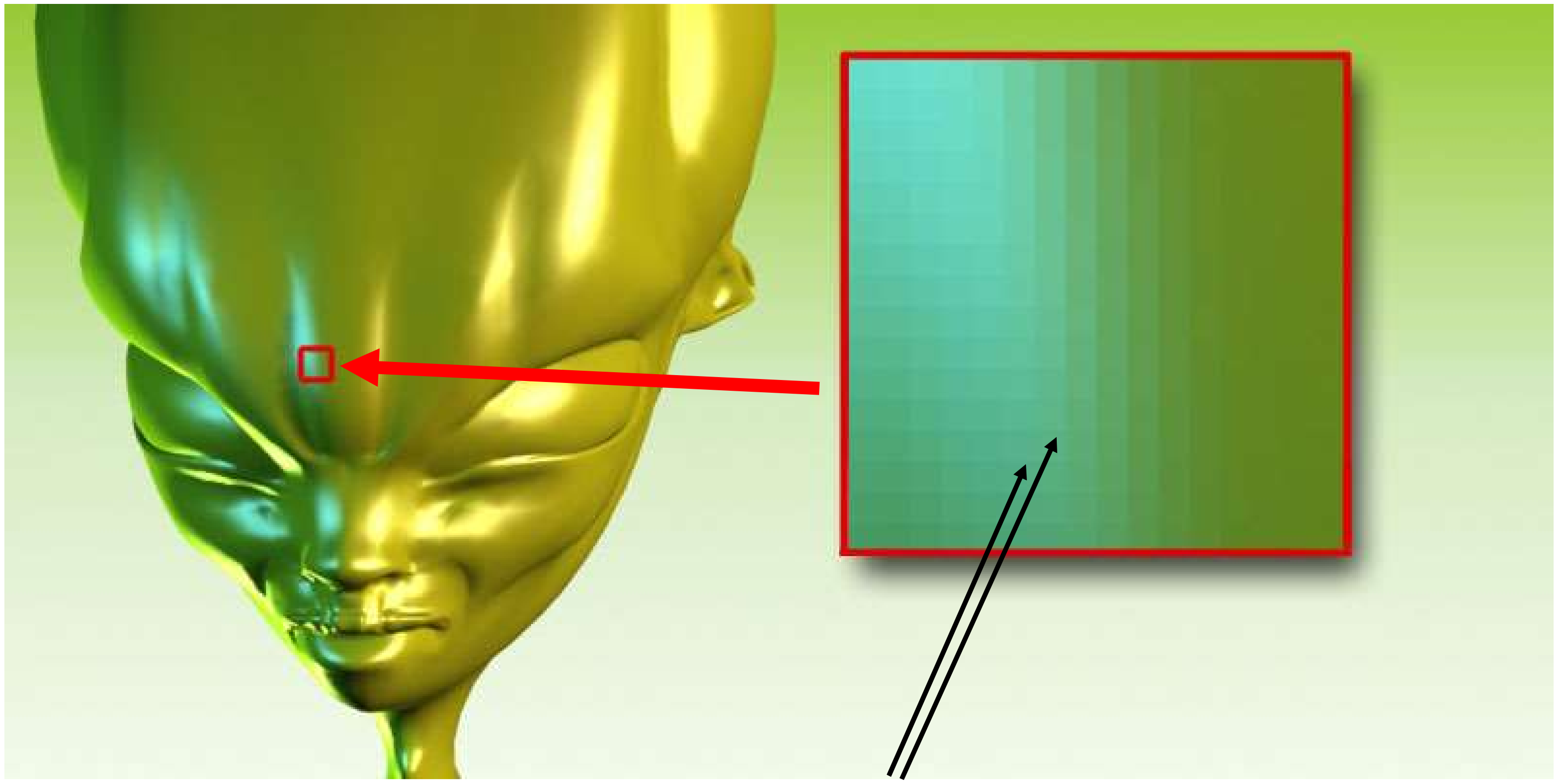


Fragment shader

- One instance processes one fragment
- No knowledge of neighbouring fragments



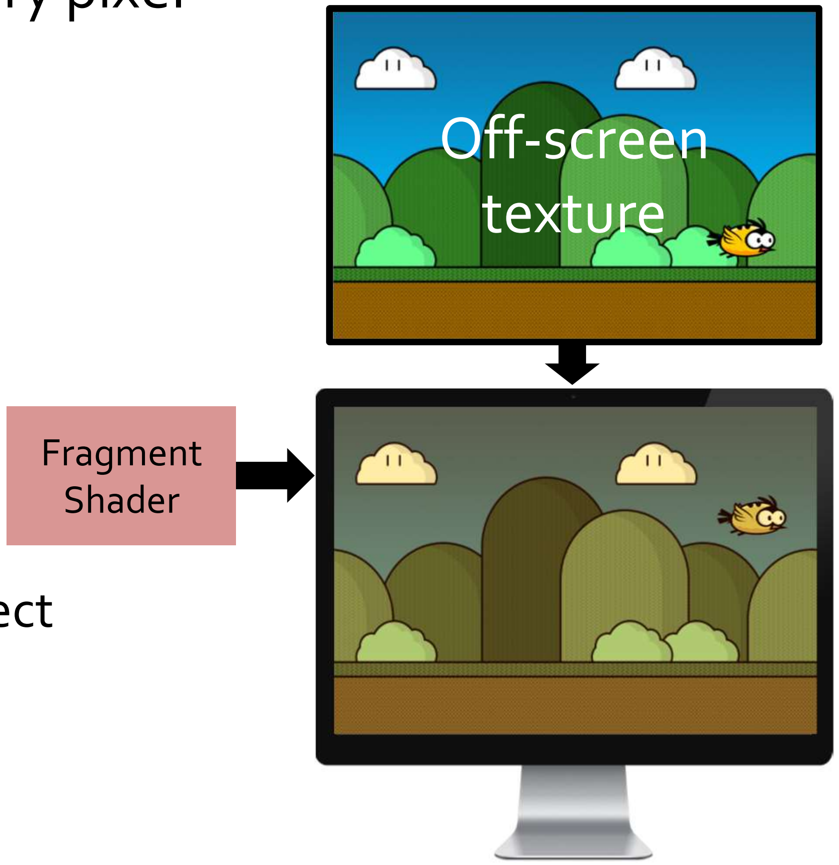
Fragment shader



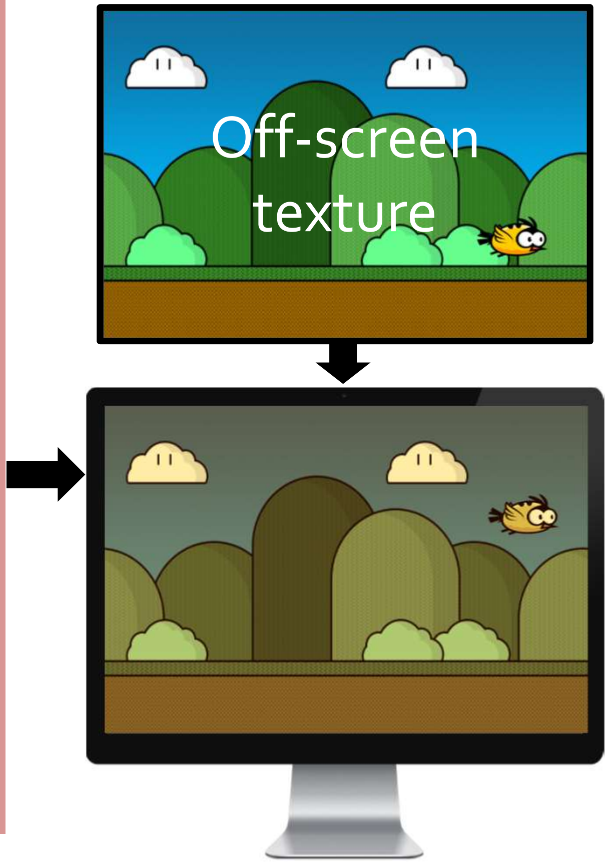
Each fragment is calculated individually

Post-Processing Shader

- Intend: shader is executed once for every pixel
- Two steps
  1. Render scene into off-screen texture
  2. Render window filling quad with
     - Off-screen texture enabled
     - Shader enabled that implements an effect



```
uniform sampler2D offScreenTexture;
in vec2 uv;
float grayScale(vec3 color) {
  vec3 weight = vec3(0.2126, 0.7152, 0.0722);
  return dot(color, weight);
}
out vec4 color;
void main() {
  vec3 tex = texture(offScreenTexture, uv).rgb;
  vec3 grayColor = vec3(grayScale(tex));
  color.rgb = grayColor;
  color.a = 1.0; }
```


Off-screen
texture