Imagens Digitais

Prof. Antonio L. Apolinário Jr.

UFBA/IM/DCC/BCC - 2017.1

Roteiro

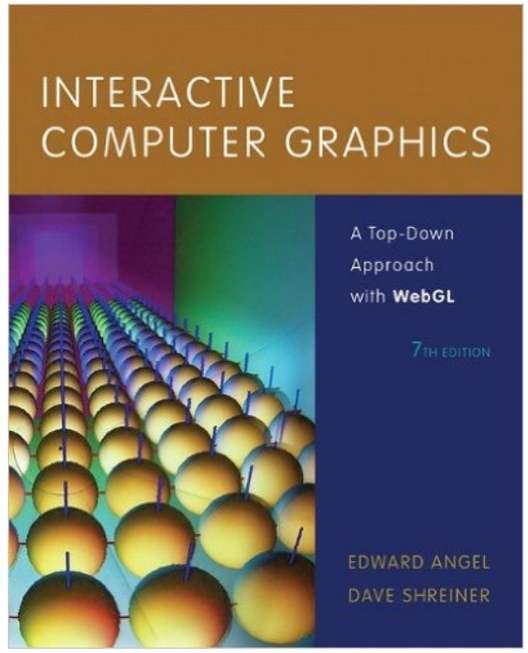
- Conceitos Básicos
- Transformações de Intensidade
- Processamento de imagem em Three.js

Leitura de referencia

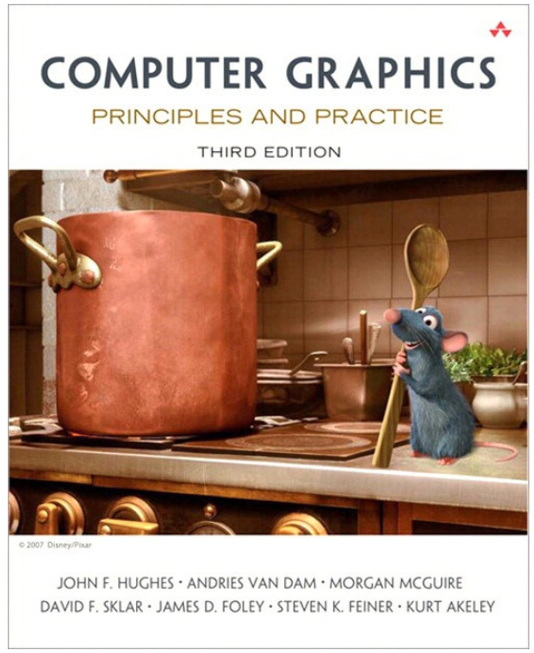
- Capitulo 3
  Processamento Digital de Imagens
  3ª. edição
  GONZALEZ, R. C.
  WOODS, R. E.
  Pearson Education do Brasil, 2010.

Leitura de referencia

- Capitulo 7
  Interactive Computer Graphics - A top-down approach with WebGL
  7th Edition
  Angel, Edward
  Shreiner, Dave
  Addison-Wesley. 2014.

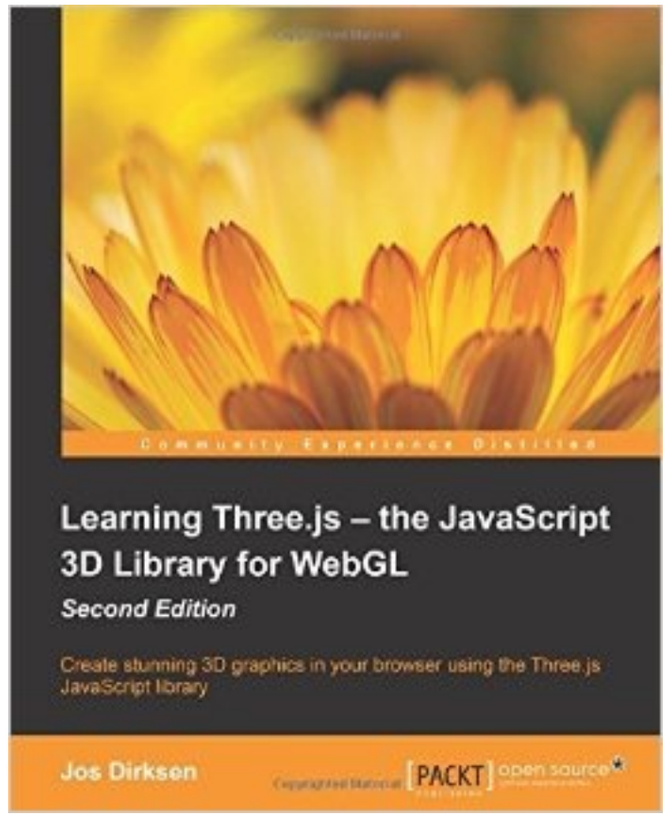


- Capitulo 17
  Computer Graphics : Principles and Practice Third Edition in C
  3rd Edition
  John F. Hughes / Andries van Dam
  Morgan McGuire / David F. Sklar
  James D. Foley / Steven K. Feiner
  Addison-Weslley. 2013.



Leitura de referencia

- Capitulo 11
  Learning Three.js: The JavaScript 3D Library for WebGL
  Jos Dirksen
  2nd Edition
  Packt Publishing - 2013.



Conceitos básicos

Imagem Digital

- Processos envolvidos
  - Discretização
  - Codificação
  - Decodificação
  - Reconstrução

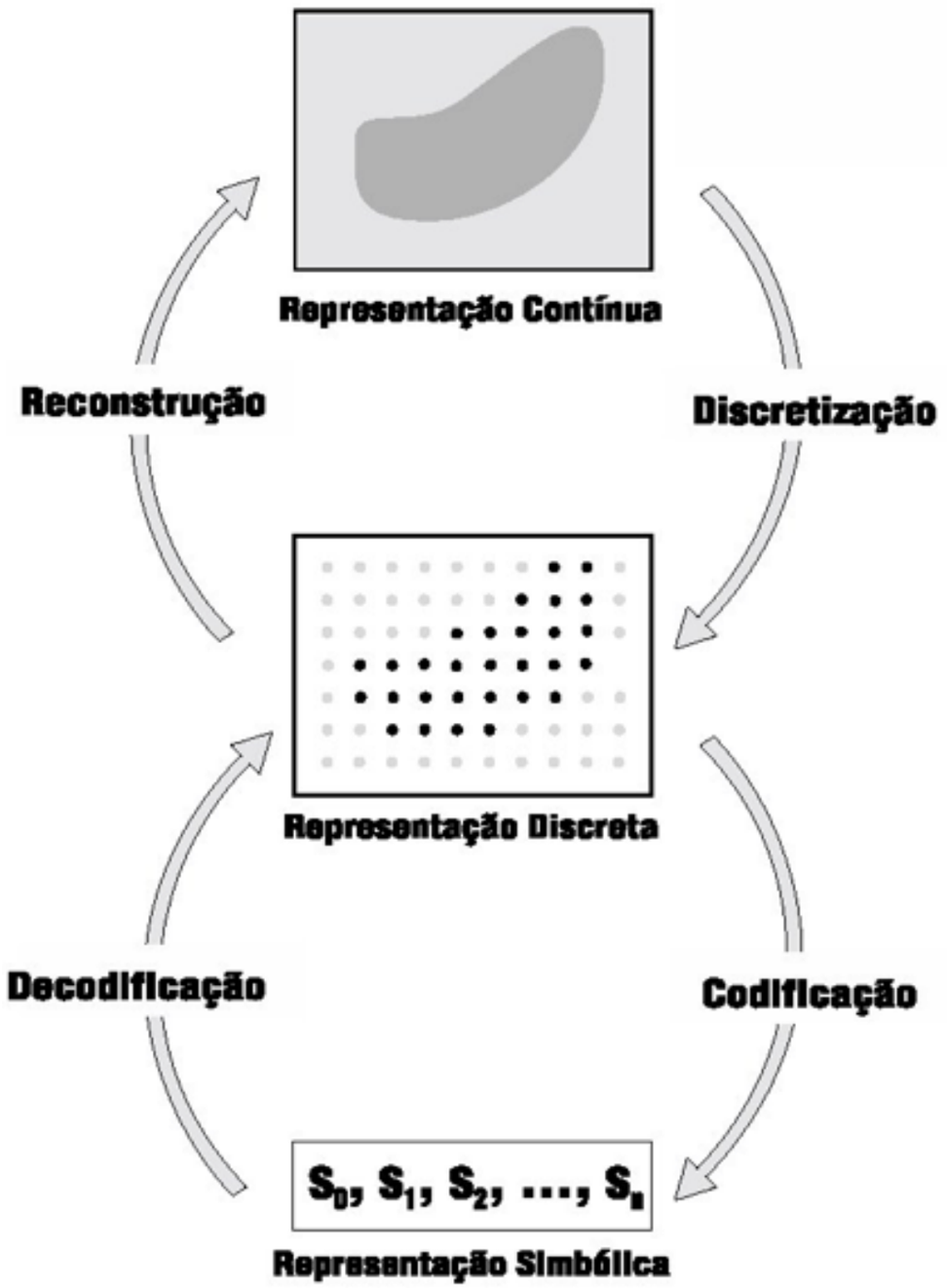


Imagem Digital

- Modelo matemático simples:
  - A intensidade luminosa é função da
  - Intensidade das fontes de luz ⇒ i(x,y)
  - Intensidade refletida pelos objetos ⇒ r(x,y)

$$f(x,y)=i(x,y)*r(x,y)$$

$$0 < i(x,y) < \infty$$

$$0 < r(x,y) < 1$$

Imagem Digital

- Amostragem espacial da função f(x,y)

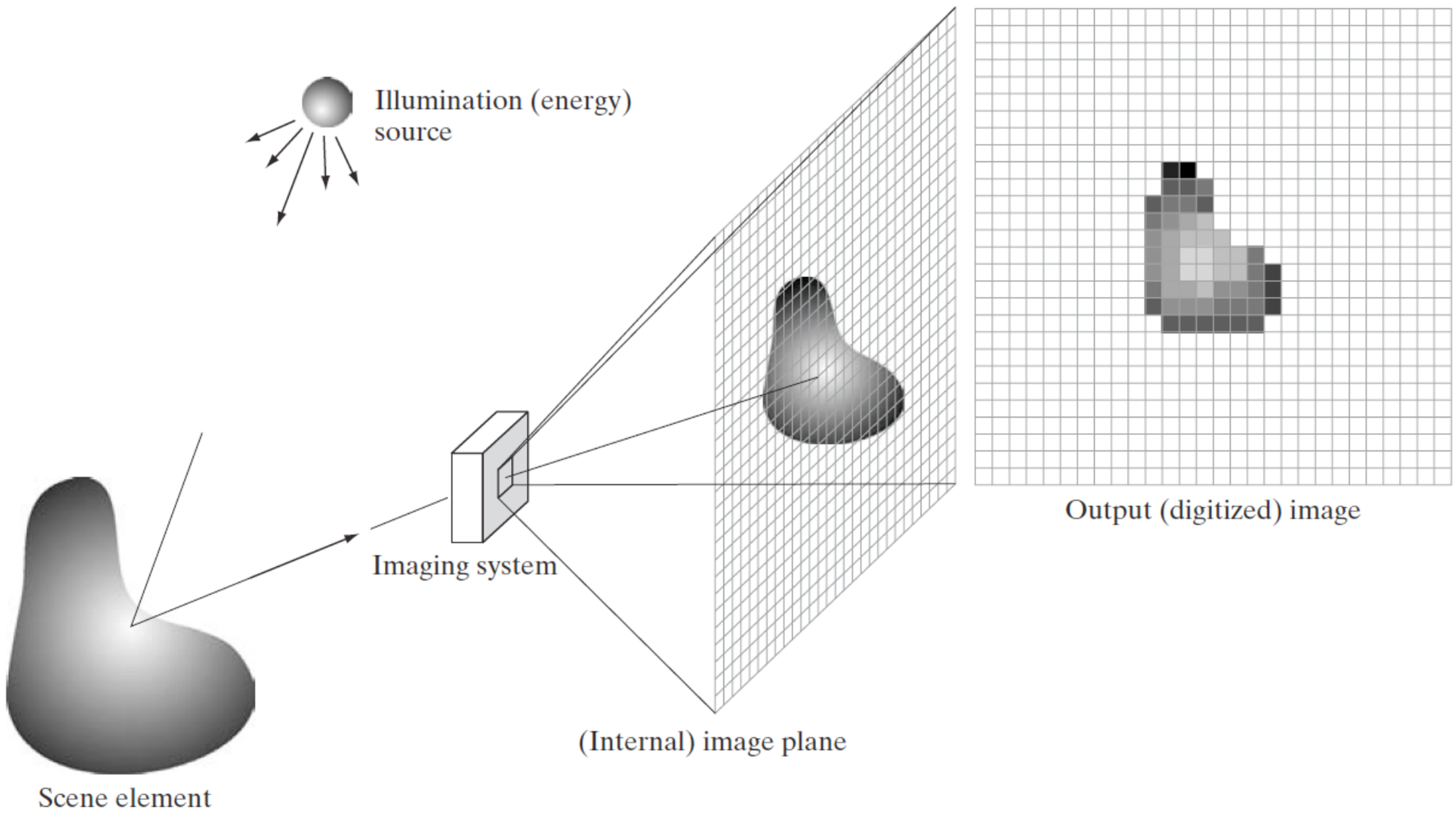
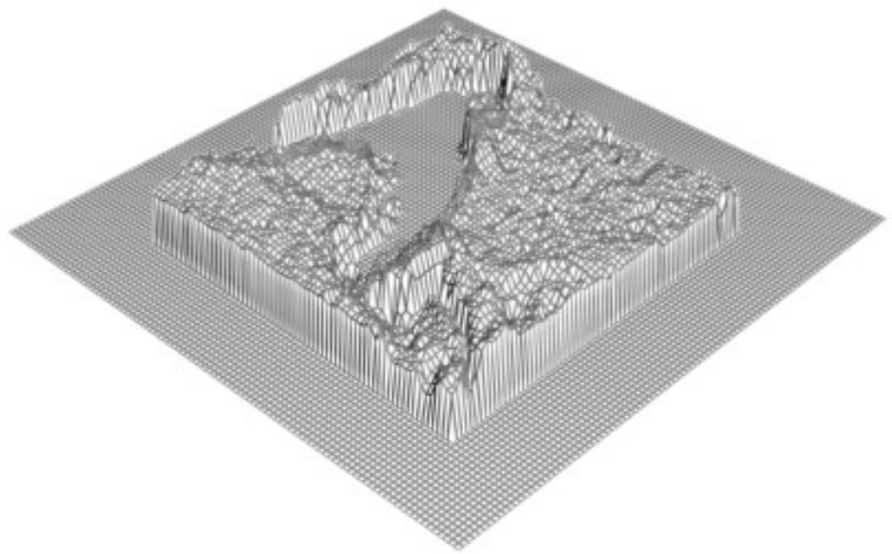


Imagem Digital

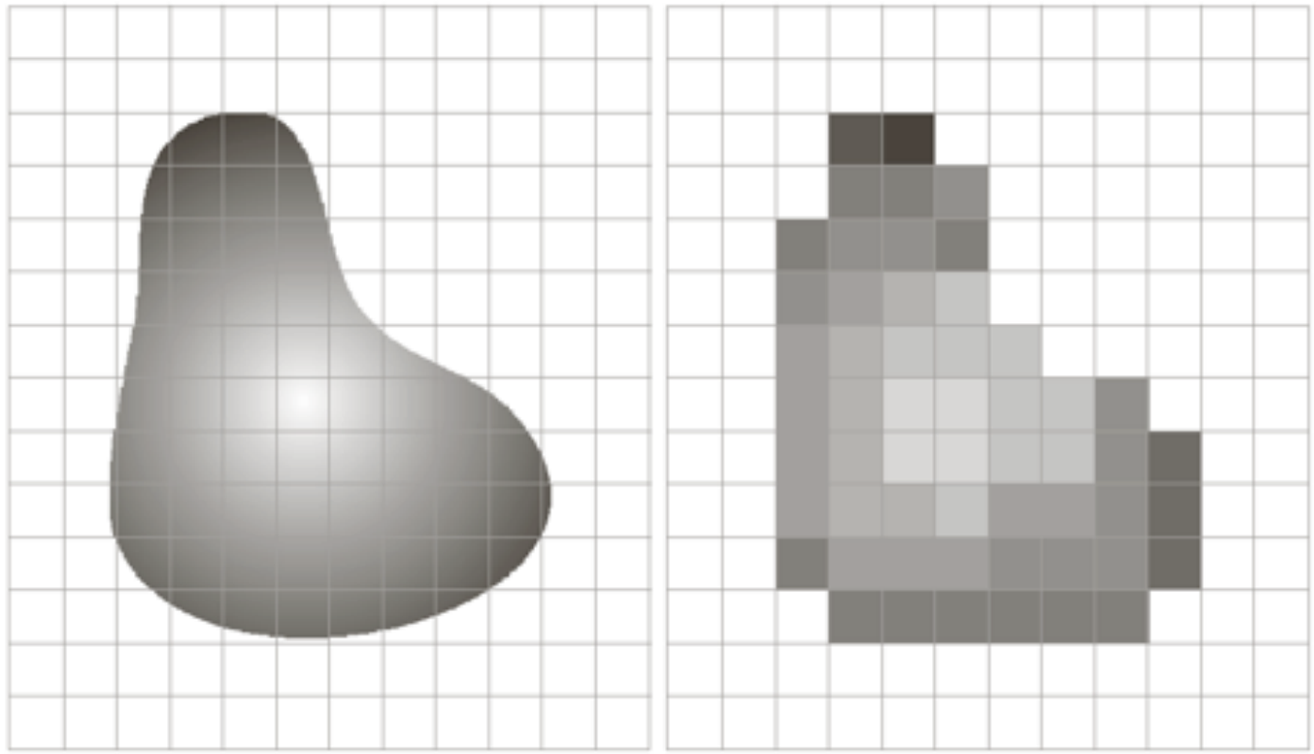
- Modelo matemático simples:
  - função bidimensional f(x,y) que mapeia a intensidade luminosa registrada na imagem

$$0 < f(x,y) < \infty$$



Amostragem

- Envolve 2 tipos de discretização
  - Espacial
    - pixels
  - Cores
    - espaço de cores



a b

FIGURE 2.17 (a) Continuous image projected onto a sensor array. (b) Result of image sampling and quantization.

Imagem Digital

- Discretização Espacial
  - Representação matricial

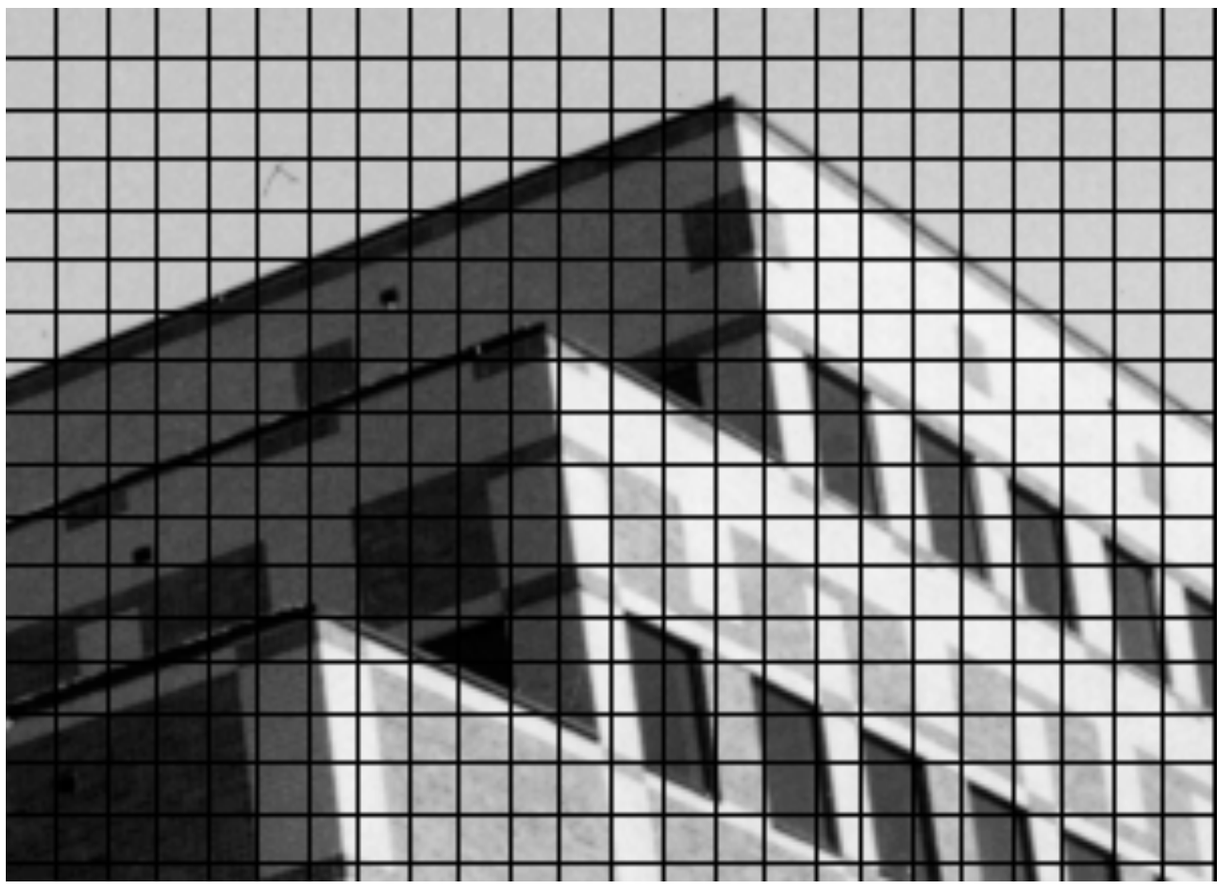
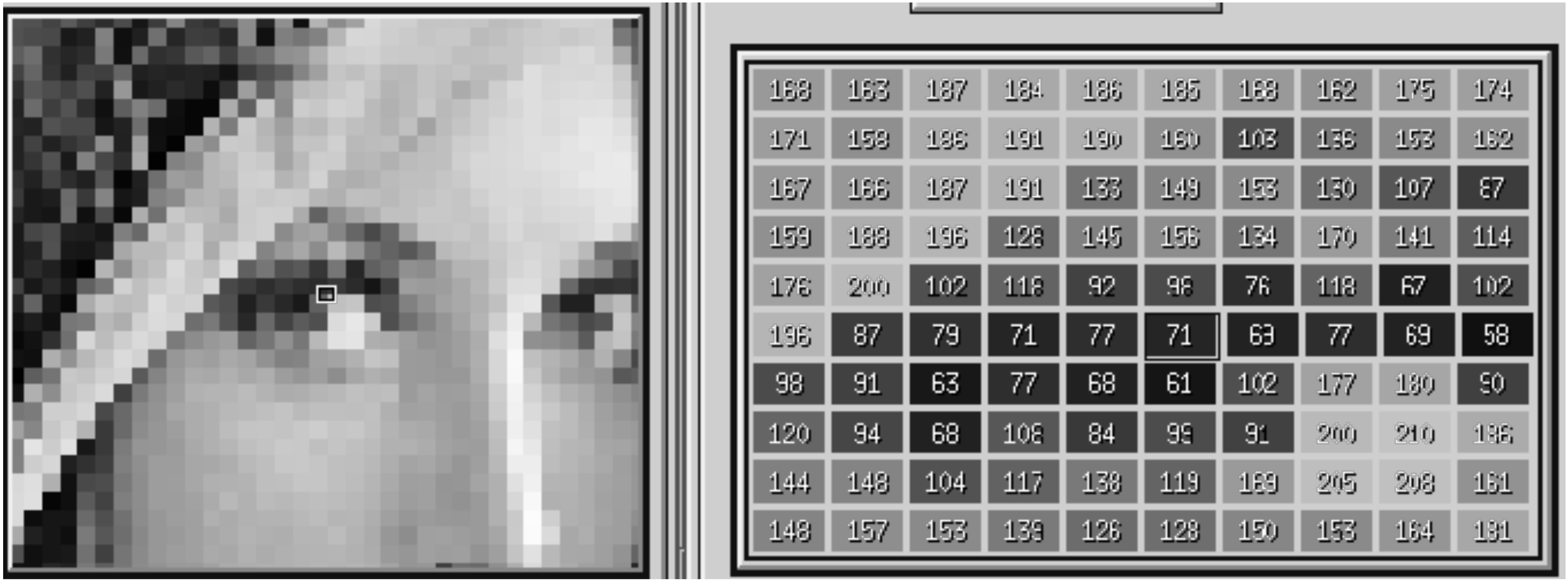


Imagem Digital

- Matriz de intensidades
  - Amostras
  - Identificação implícita dos pixels



168	163	187	184	186	185	188	162	175	174
171	158	186	191	190	180	103	136	153	162
167	166	187	191	133	149	153	130	107	87
159	188	196	128	145	156	134	170	141	114
176	200	102	118	92	98	76	118	67	102
196	87	79	71	77	71	63	77	69	58
98	91	63	77	68	61	102	177	180	90
120	94	68	108	84	99	91	200	210	186
144	148	104	117	138	119	169	205	208	161
148	157	153	139	126	128	150	153	164	181

Imagem Digital

- Representação Discreta

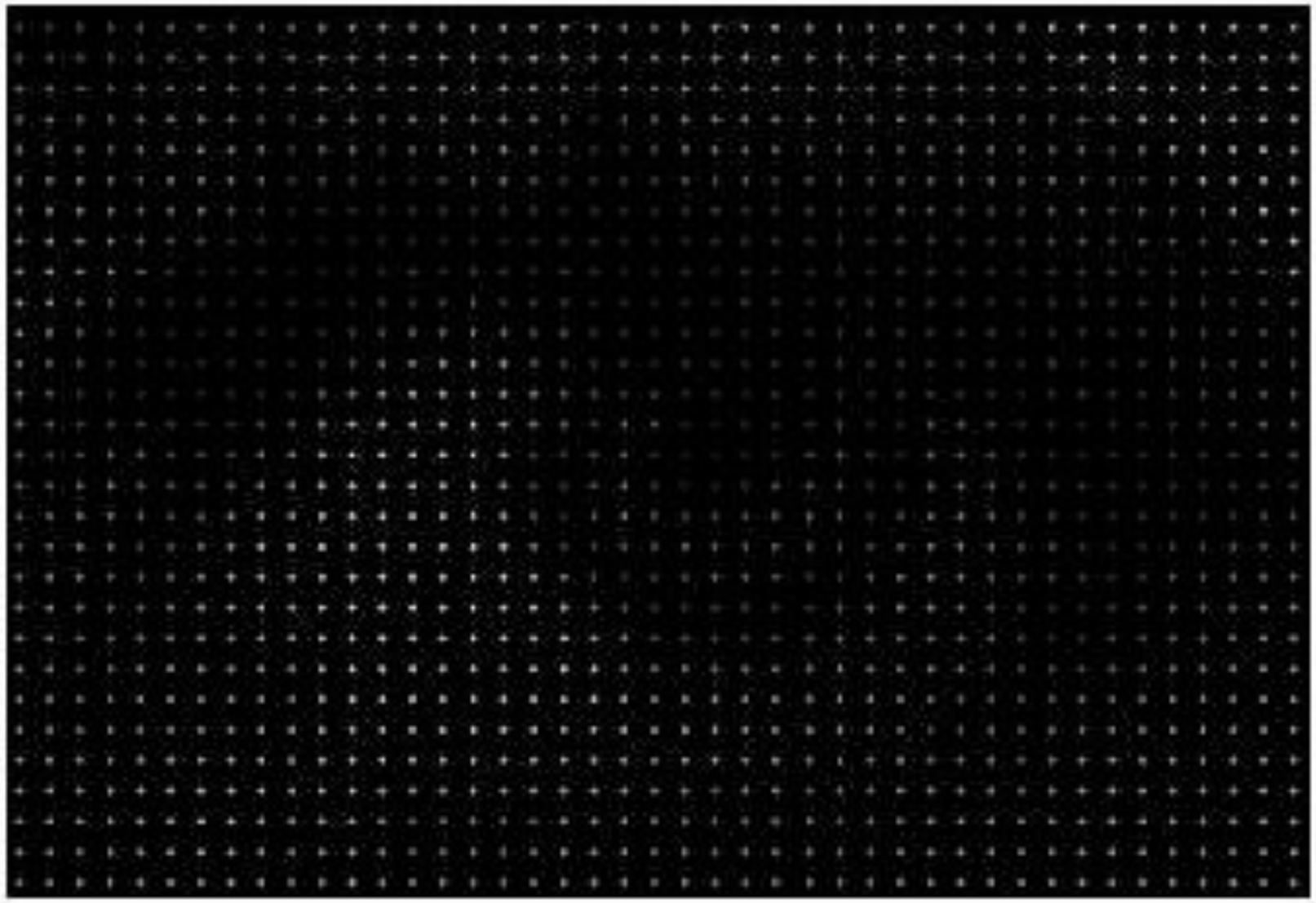


Imagem Digital

- Resolução espacial

Photographed at different Resolution settings with 2 MP camera

Photographed at different Resolution settings with 5 MP camera

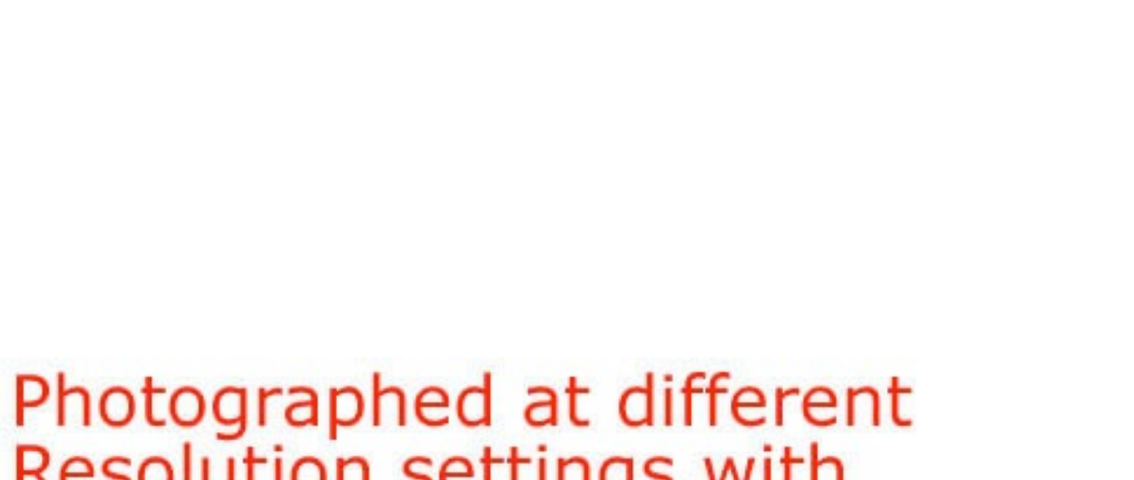


Imagem Digital

- Resolução Espacial

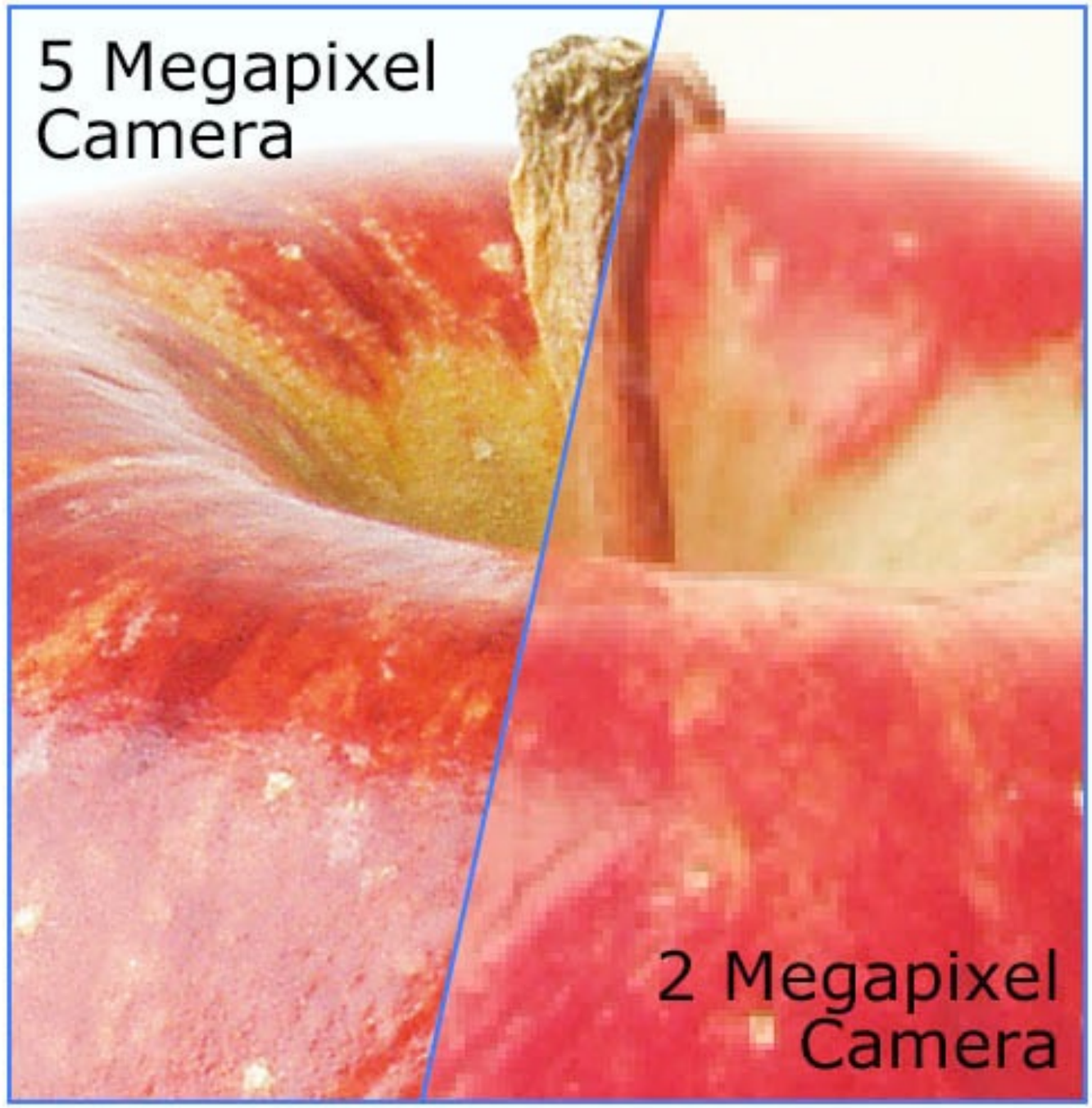


Imagem Digital

- Resolução de Cores
- Profundidade do pixel
- Quantização de cada amostra

Bits	Core	Image
24	16M	true color
8	256	gray scale
4	8	
2	4	
1	2	bitmap

Imagem Digital

- Resolução de Cores
- Profundidade do pixel
- Quantização de cada amostra

Bits	Core	Image
24	16M	true color
8	256	gray scale
4	8	
2	4	
1	2	bitmap

Imagem Digital

- Resolução de Cores
- Profundidade do pixel
- Quantização de cada amostra

Bits	Core	Image
24	16M	true color
8	256	gray scale
4	8	
2	4	
1	2	bitmap

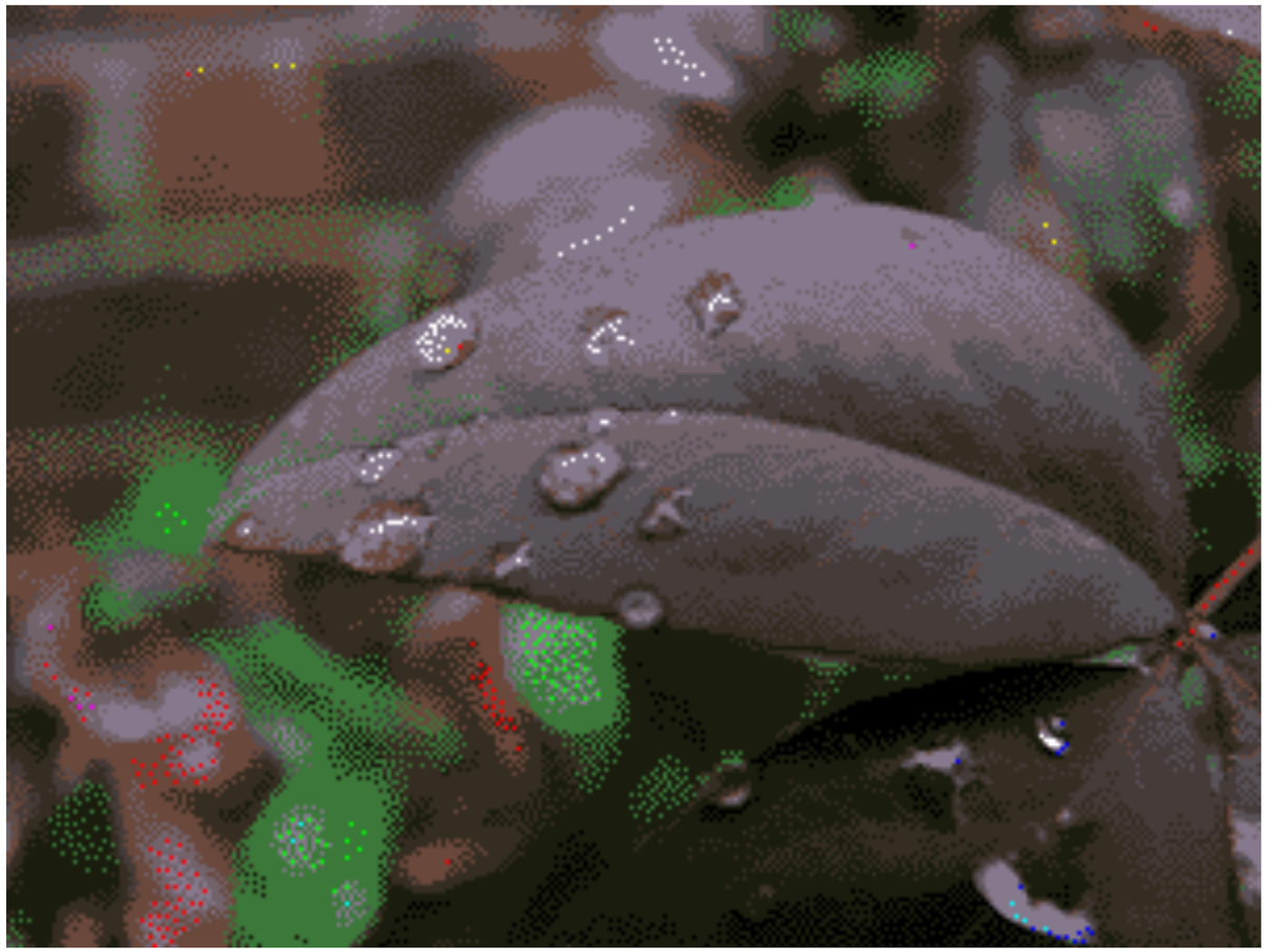


Imagem Digital

- Resolução de Cores
- Profundidade do pixel
- Quantização de cada amostra

Bits	Core	Image
24	16M	true color
8	256	gray scale
4	8	
2	4	
1	2	bitmap

Imagem Digital

- Resolução de Cores
- Profundidade do pixel
- Quantização de cada amostra

Bits	Core	Image
24	16M	true color
8	256	gray scale
4	8	
2	4	
1	2	bitmap

Imagem Digital

- Resolução de Cores



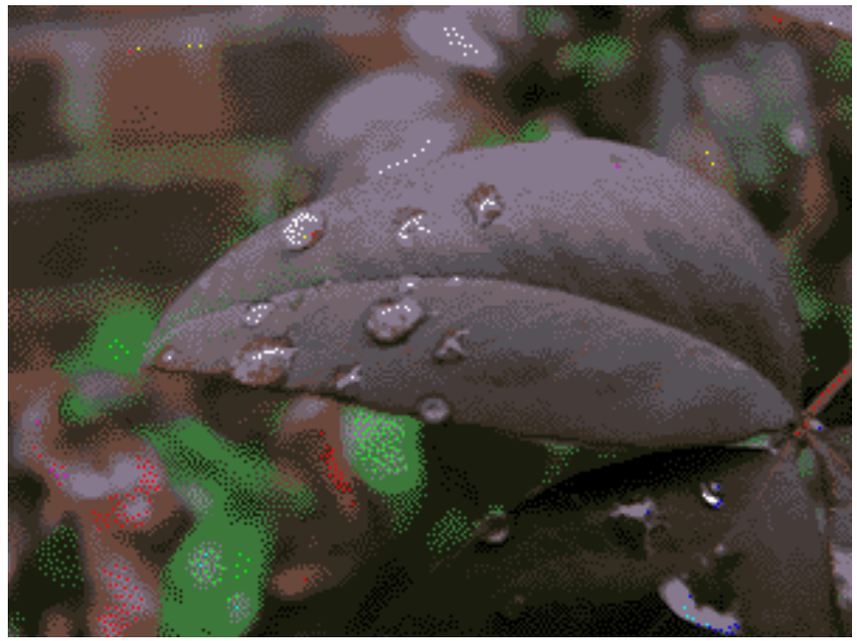


Imagem Digital

- Gerar uma representação matricial envolve:
  - Amostragem
  - Reconstrução

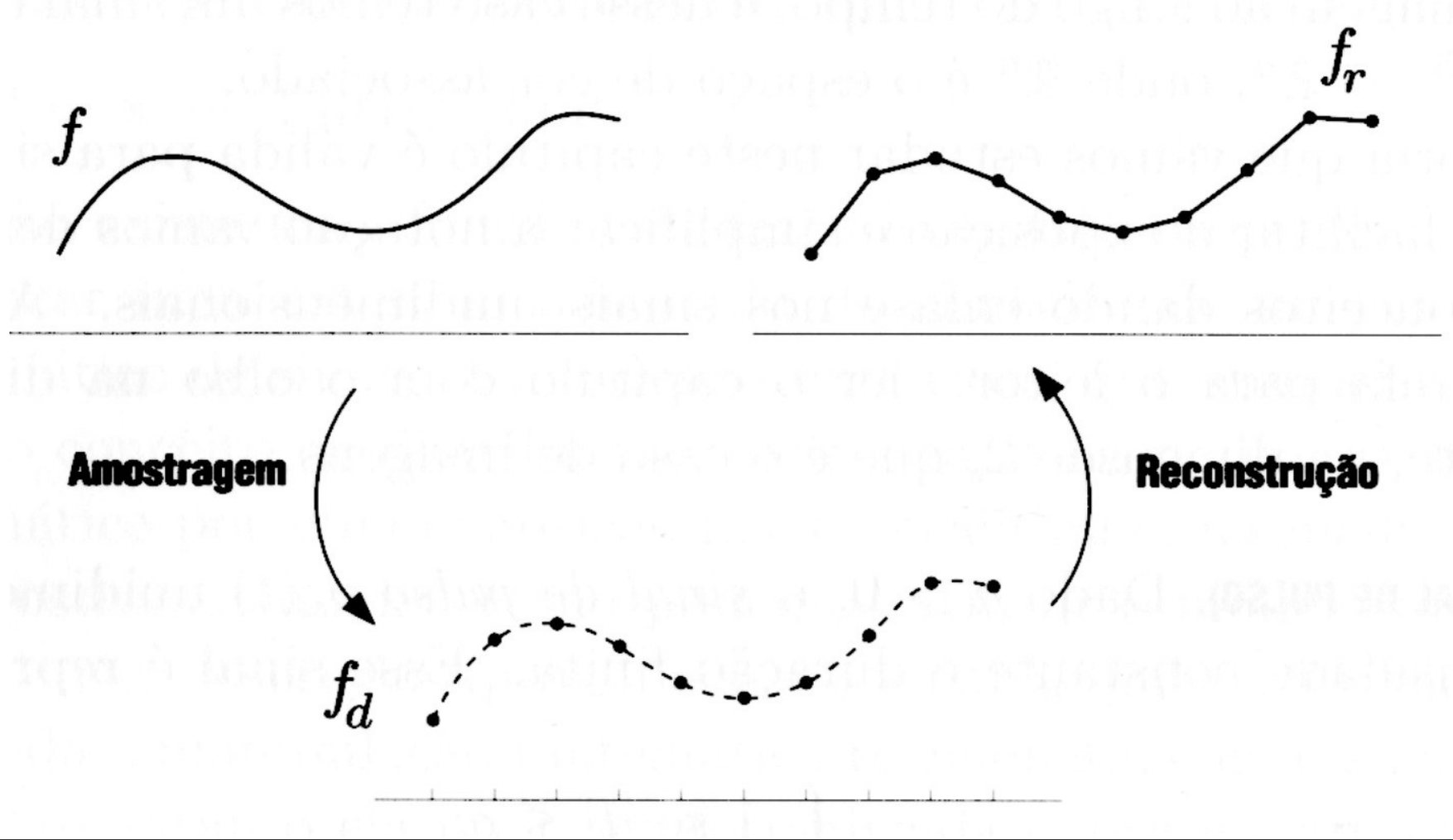
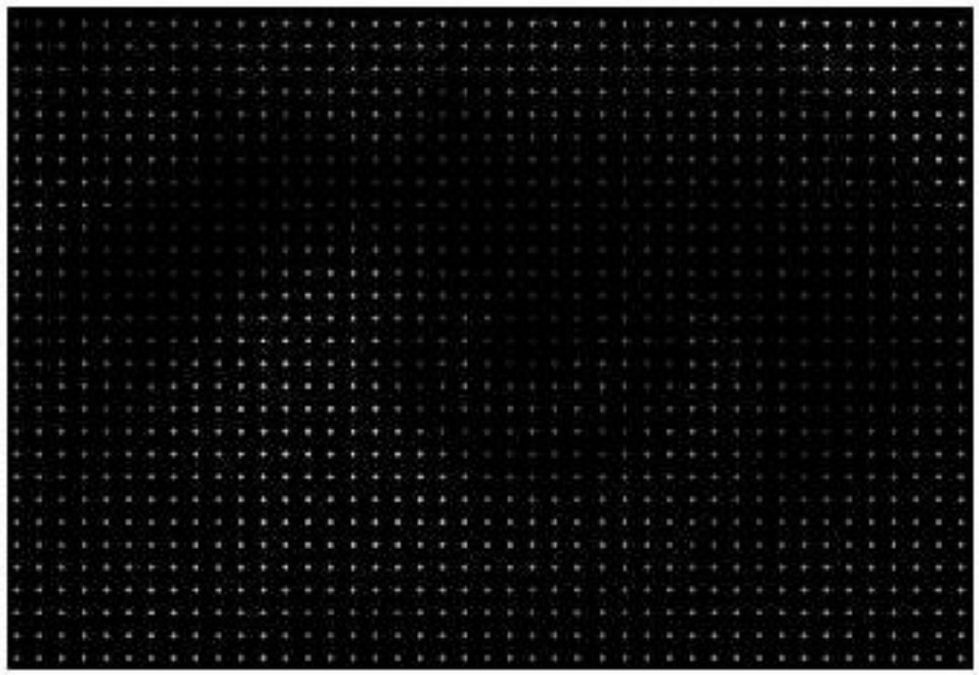
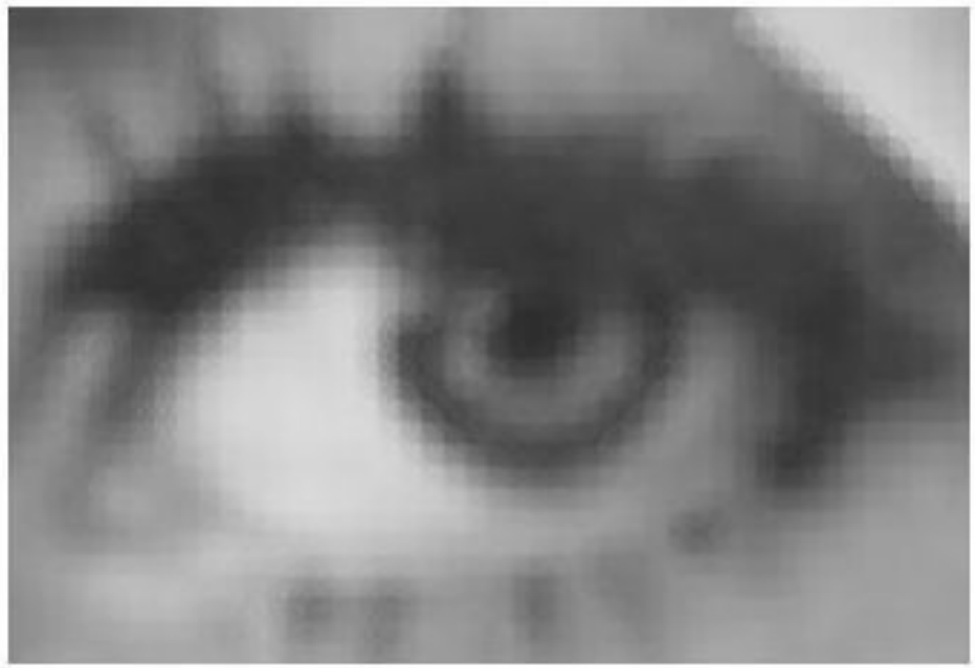
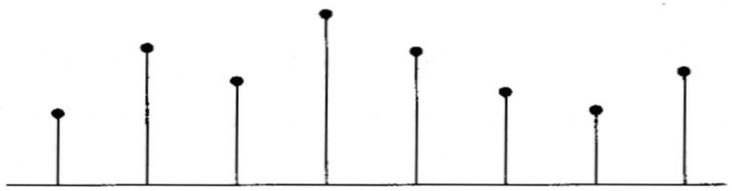


Imagem Digital

- Reconstrução
  - Interpolação







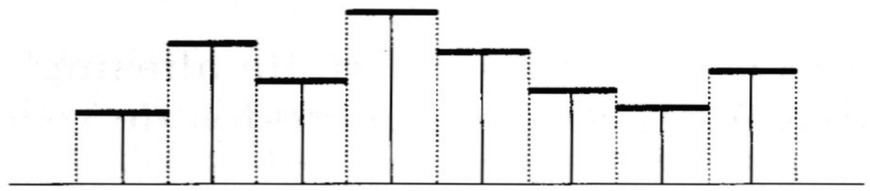
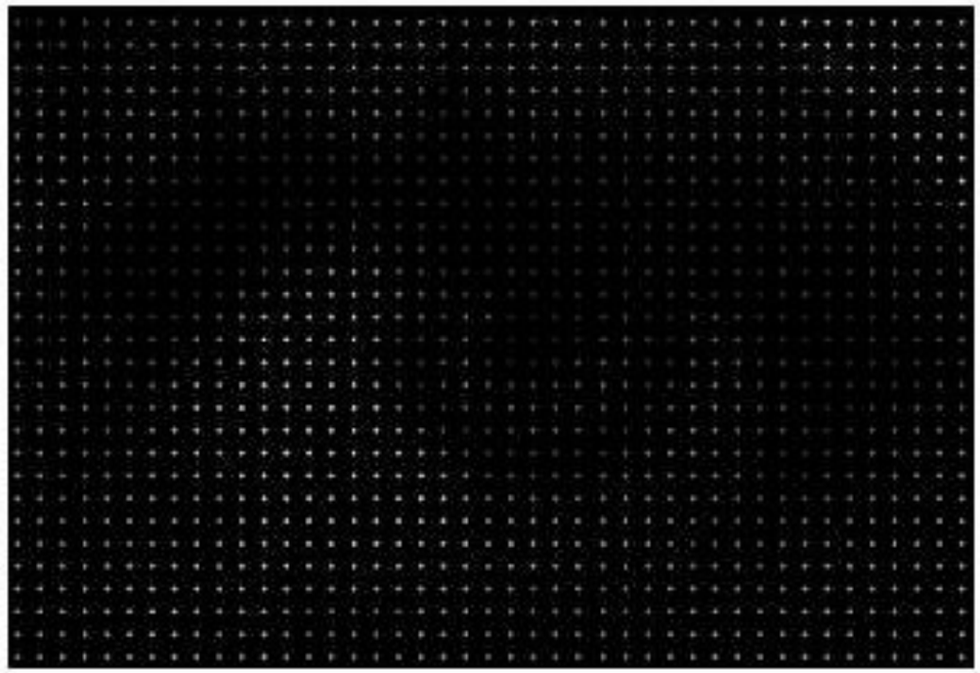
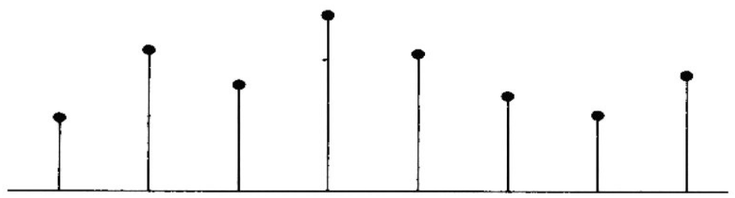
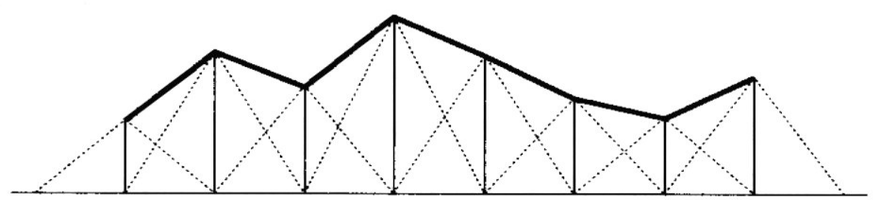


Imagem Digital

- Reconstrução
  - Interpolação







Histograma de uma Imagem

- Seja uma imagem com:
  - Resolução espacial de **M** x **N** pixels
  - Tons de cinza entre **[0,L-1]**
- Histograma é uma função discreta **h**, tal que:

$$h(r_k)=n_k$$

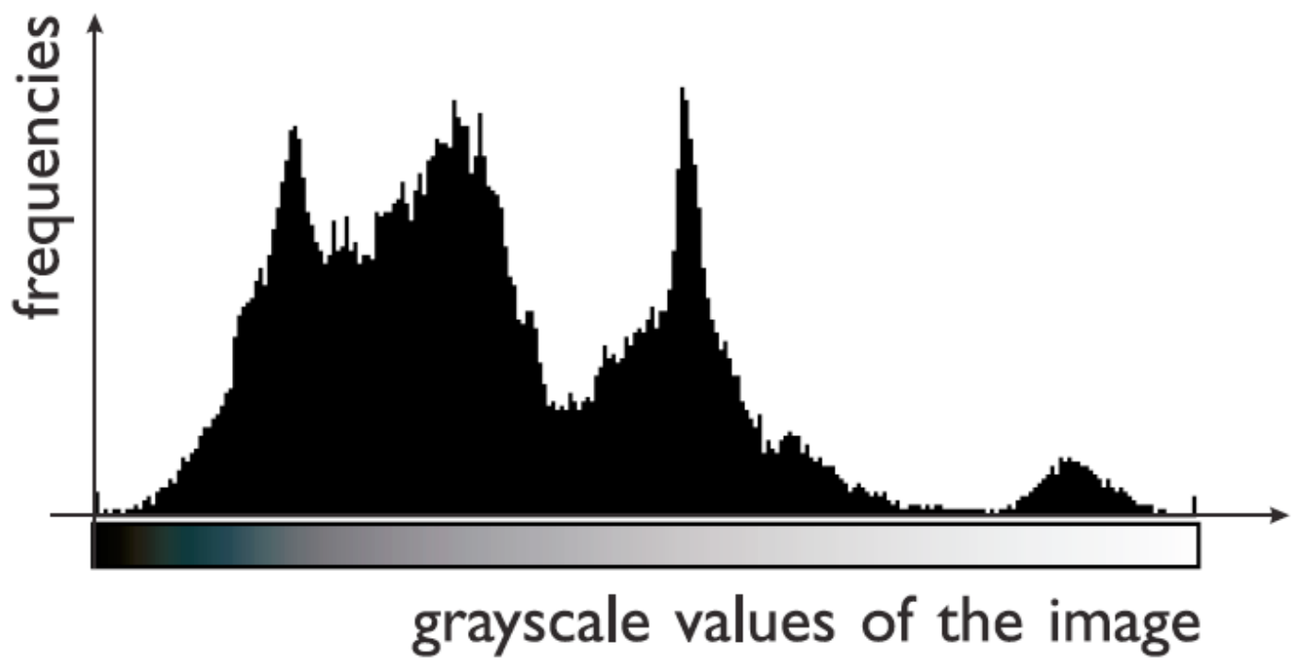
onde r_k é o k-ésimo valor de intensidade

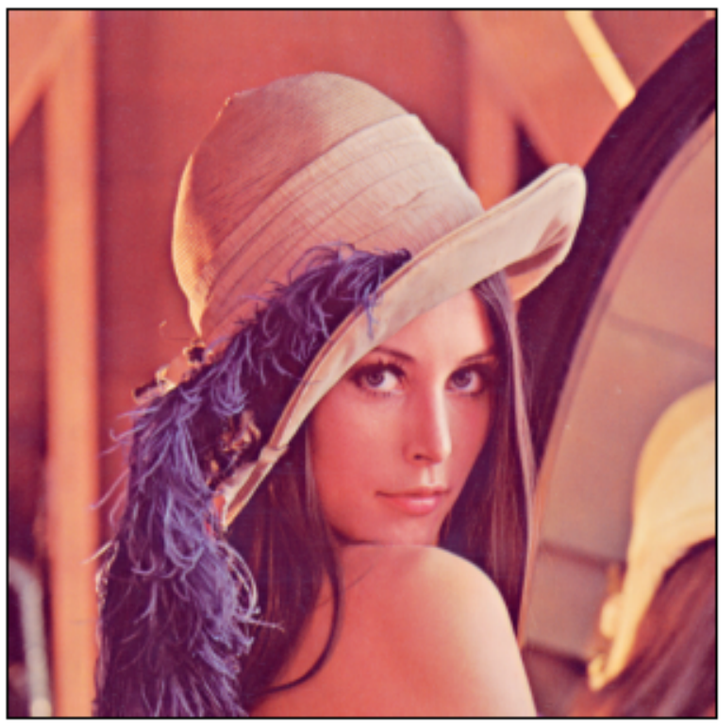
n_k é o número de pixels da imagem com intensidade r_k

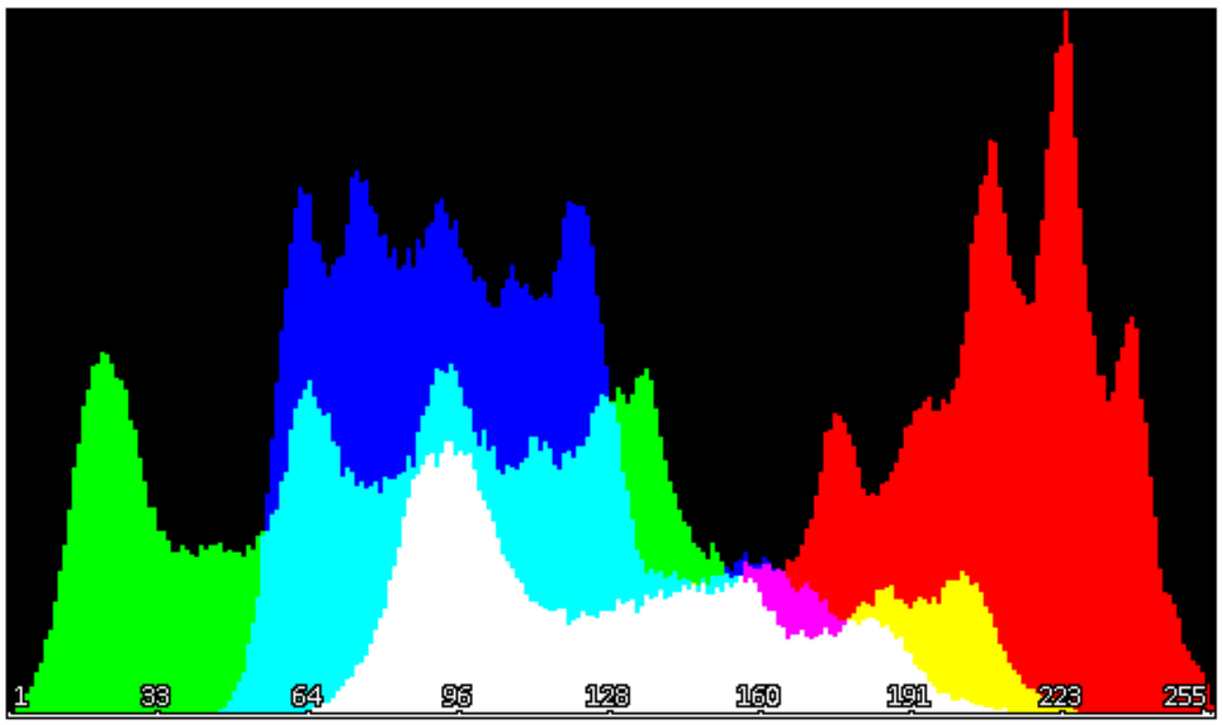
- O histograma (h) é geralmente normalizado
  - Corresponde a probabilidade de ocorrência do valor r_k na imagem dado por:

$$p(r_k) = h(r_k) / M.N$$

Histograma de uma Imagem





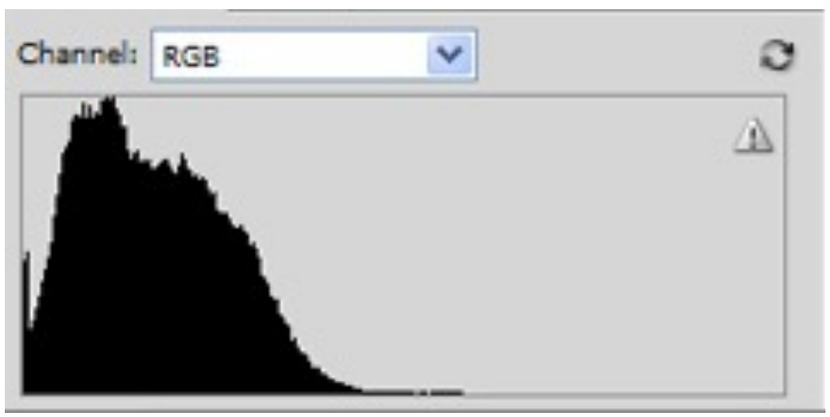


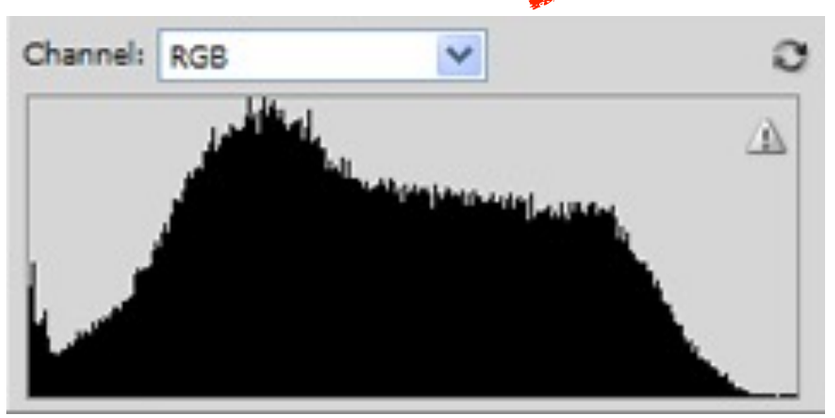
Histograma de uma Imagem

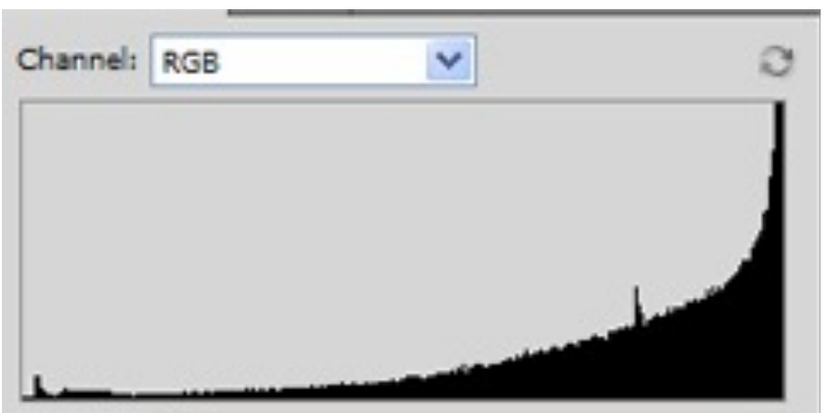
- Ferramenta comum para analise de imagens

Histograma de uma Imagem

- Exemplos:



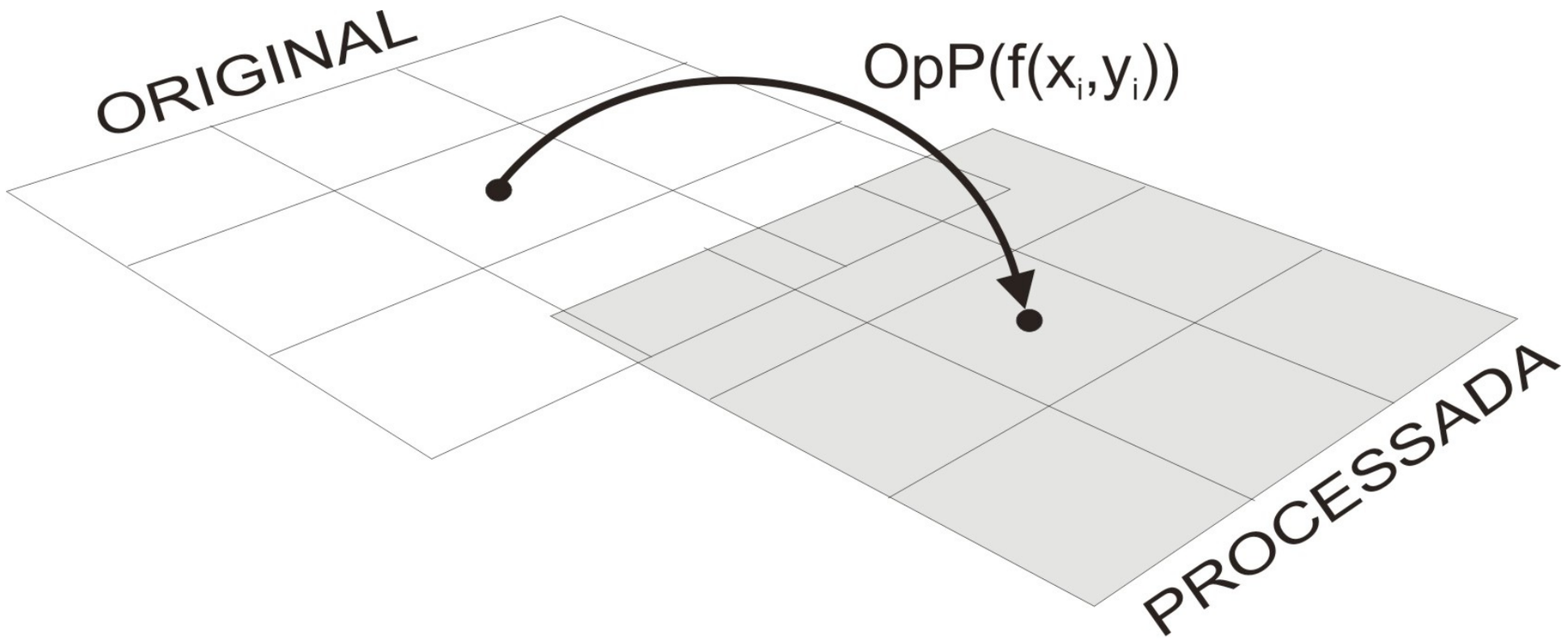




Transformações de Intensidade

Transformações de Intensidade

- Classe mais simples de transformações de imagens digitais
  - Modificam individualmente a intensidade dos *pixels*



Transformações de Intensidade

- Funções de mapeamento da forma:
- De forma simples:

$$g(x,y) = T[f(x,y)]$$

onde

$f(x,y)$ é a imagem de entrada

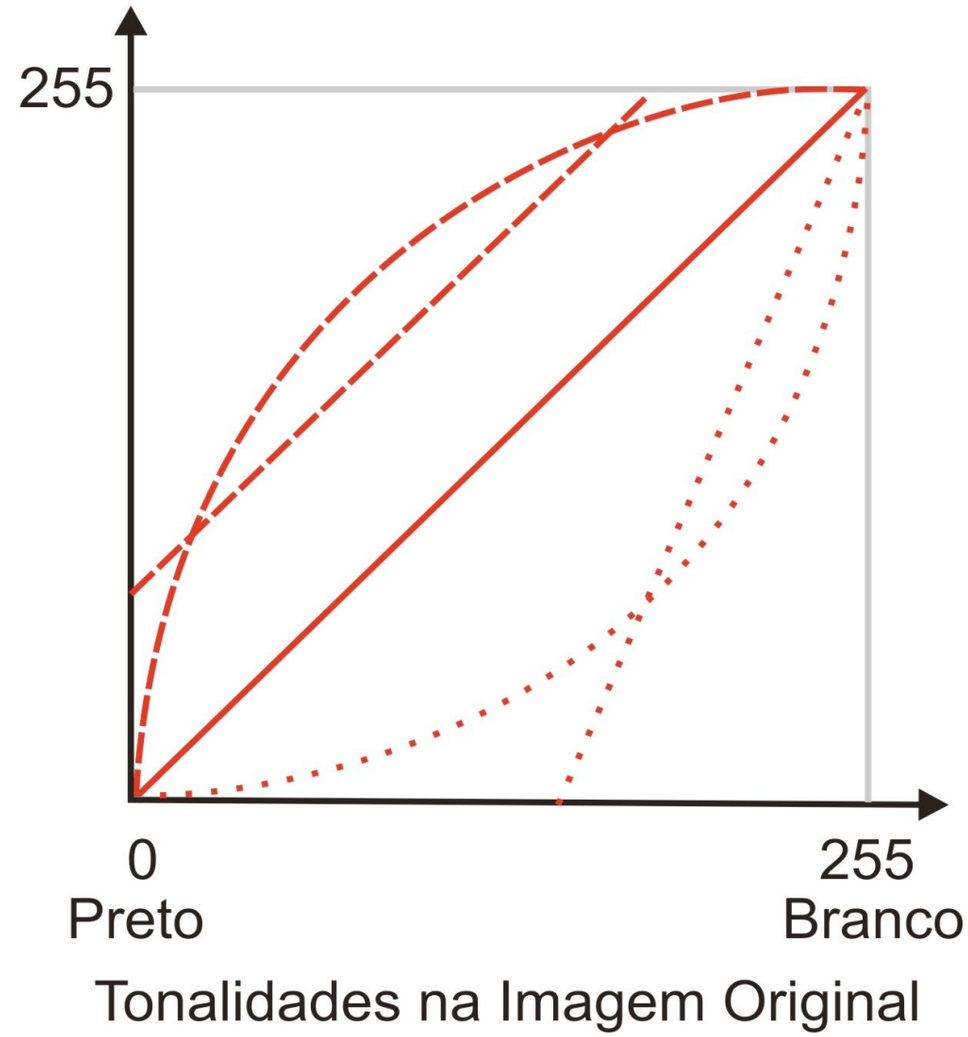
$g(x,y)$ é a imagem de saída

$$s = T[r]$$

onde

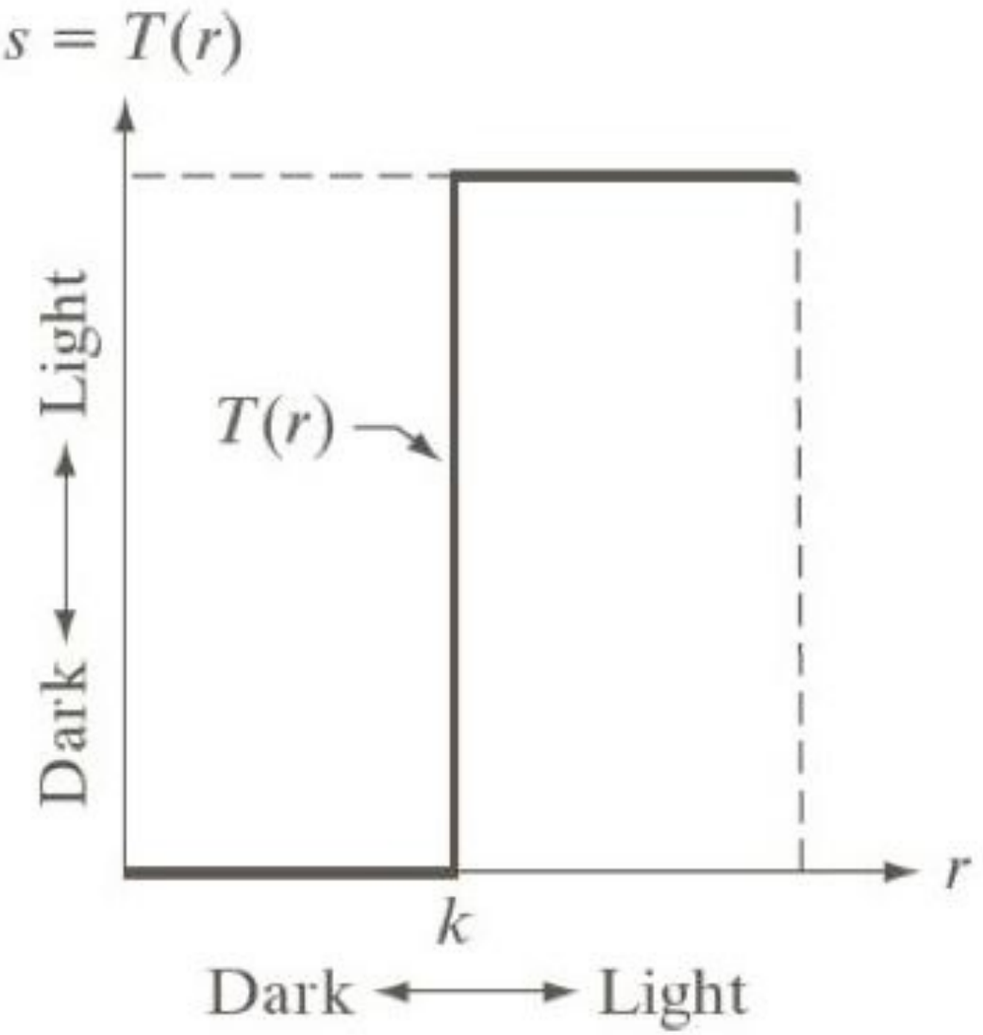
$r = f(x,y)$

$s = g(x,y)$



Transformações de Intensidade

- Binarização



Transformações de Intensidade

- Alargamento de contraste

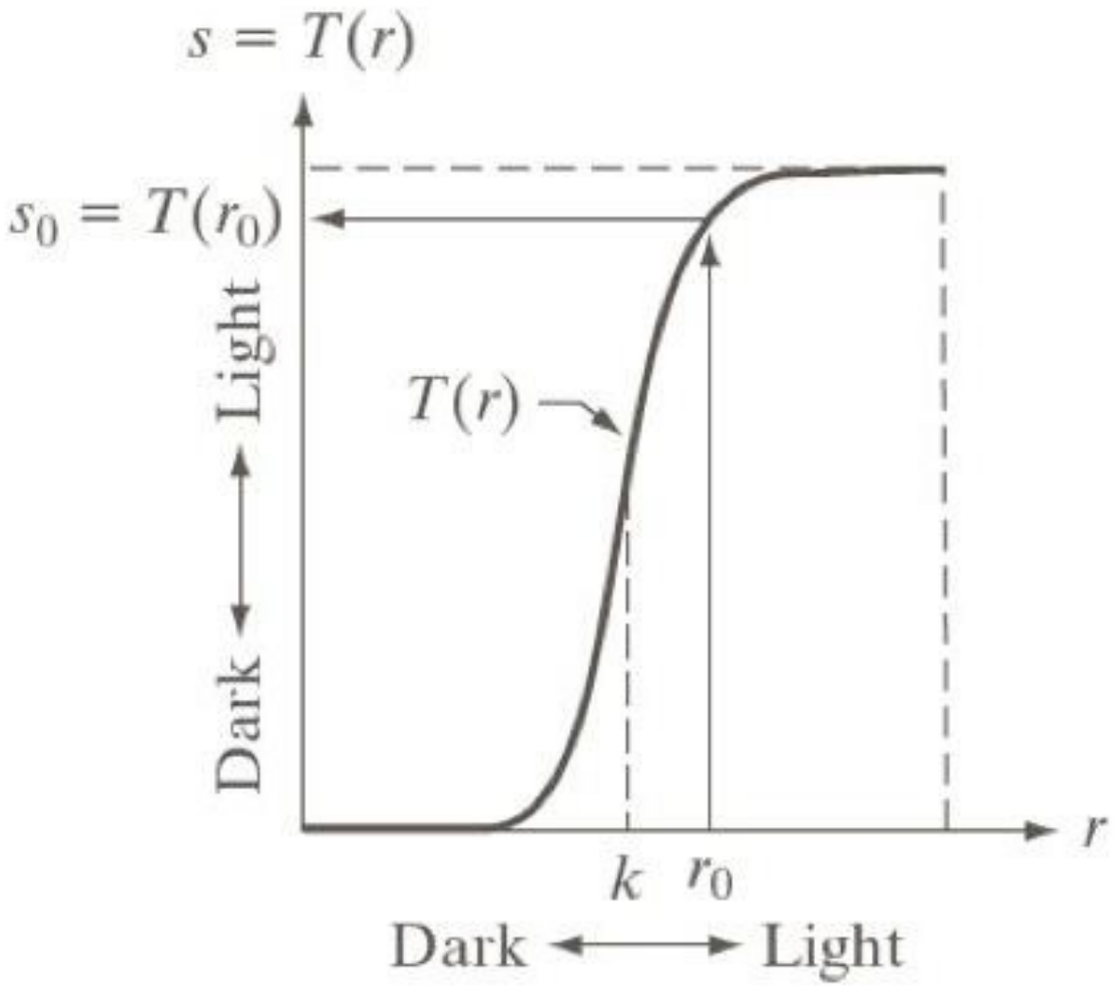


Imagem Negativa

- Transformação “negativo”

$$s = L - 1 - r \quad \text{sendo} \quad r \in [0, L-1]$$

Imagem Negativa

- Uso: facilitar a interpretação de certos elementos na imagem
  - Exemplo:

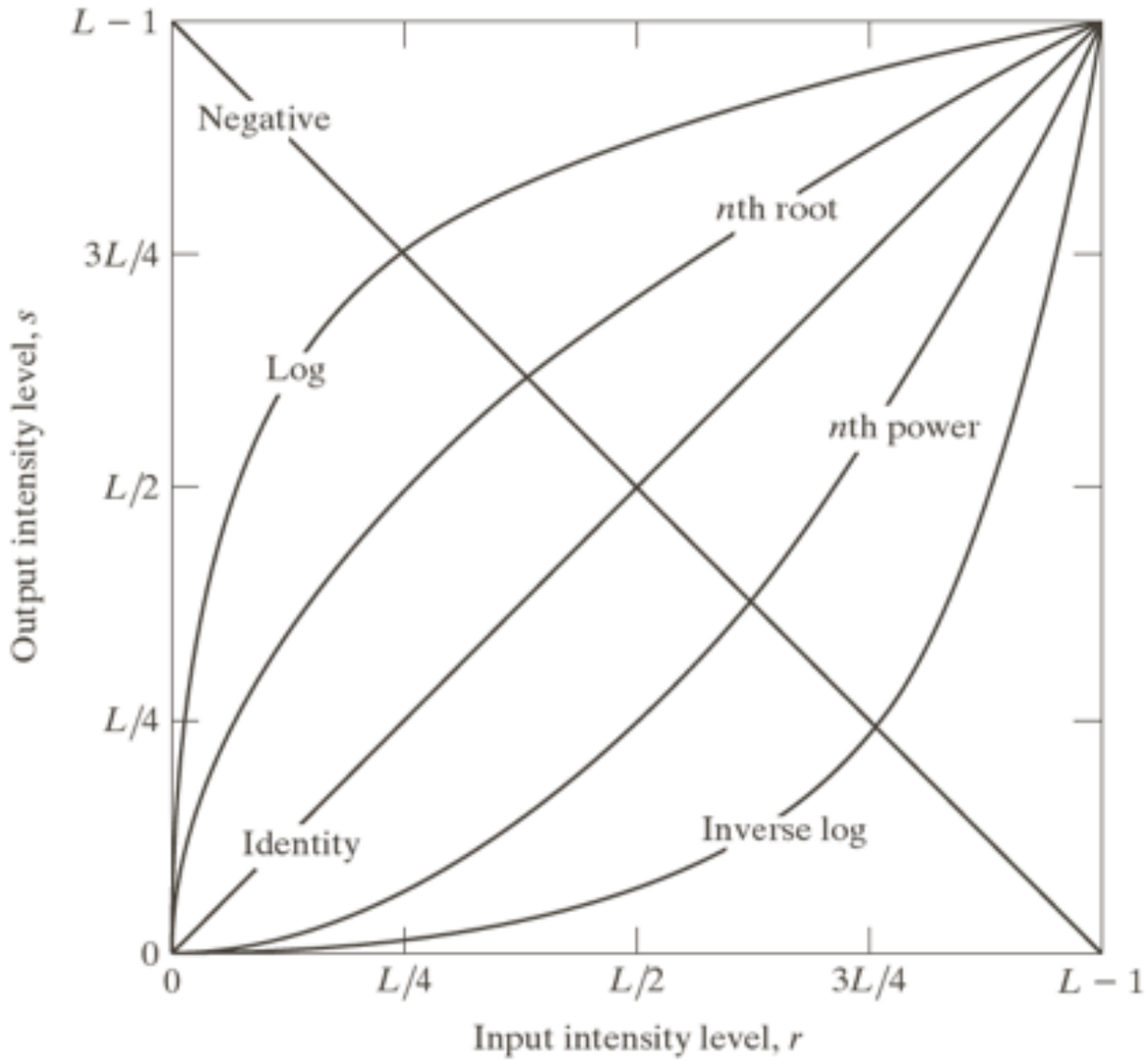
a b

FIGURE 3.4
(a) Original digital mammogram. (b) Negative image obtained using the negative transformation in Eq. (3.2-1). (Courtesy of G.E. Medical Systems.)

Transformações Logarítmicas

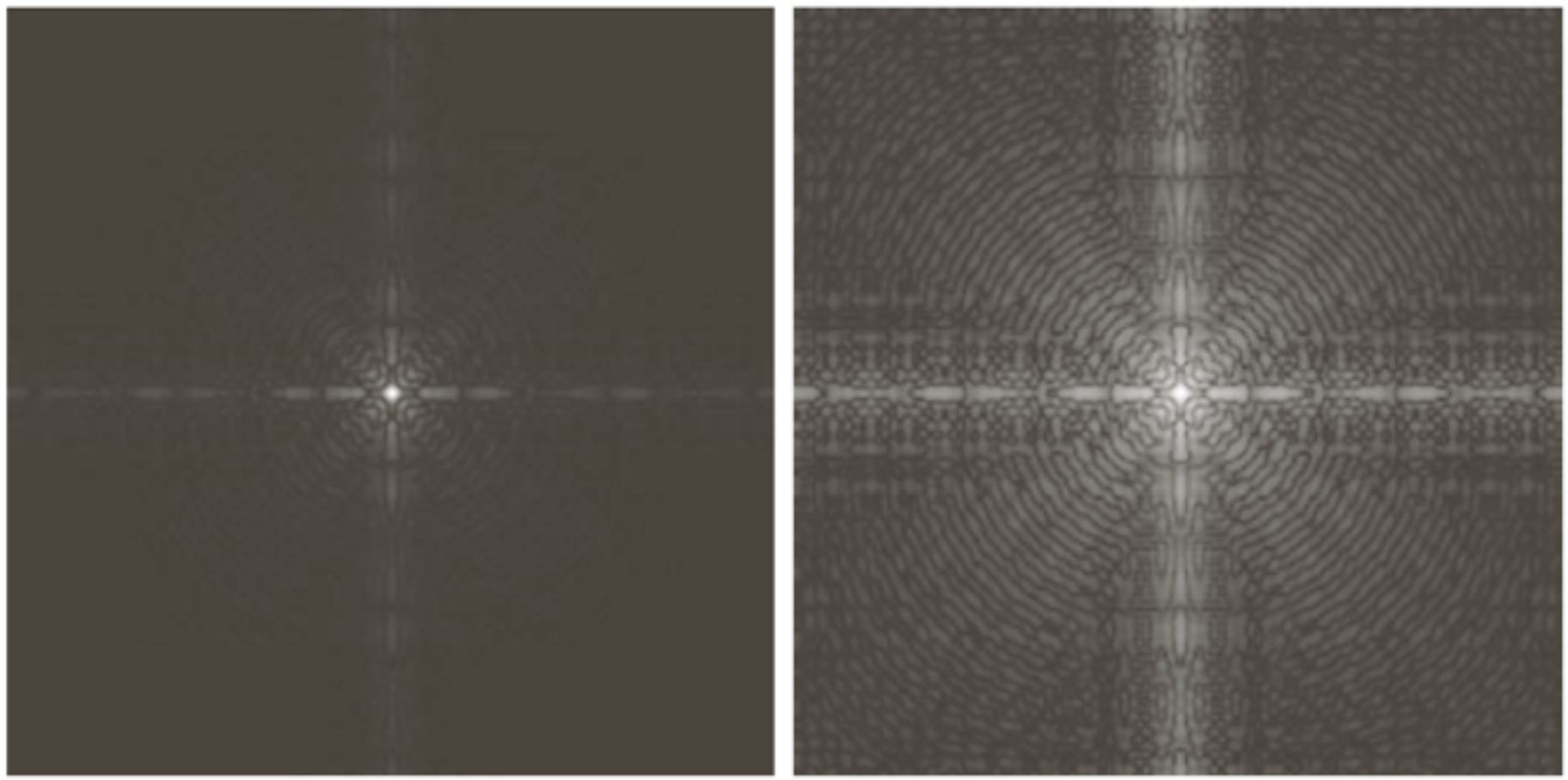
- Permitem a compressão ou expansão dos níveis de intensidade de uma imagem

$$s = c.\log(1 + r)$$



Transformações Logarítmicas

- Amplitude do sinal é maior que a representação discreta utilizada
  - Exemplo: $r \in [0, 1{,}5x10^6]$



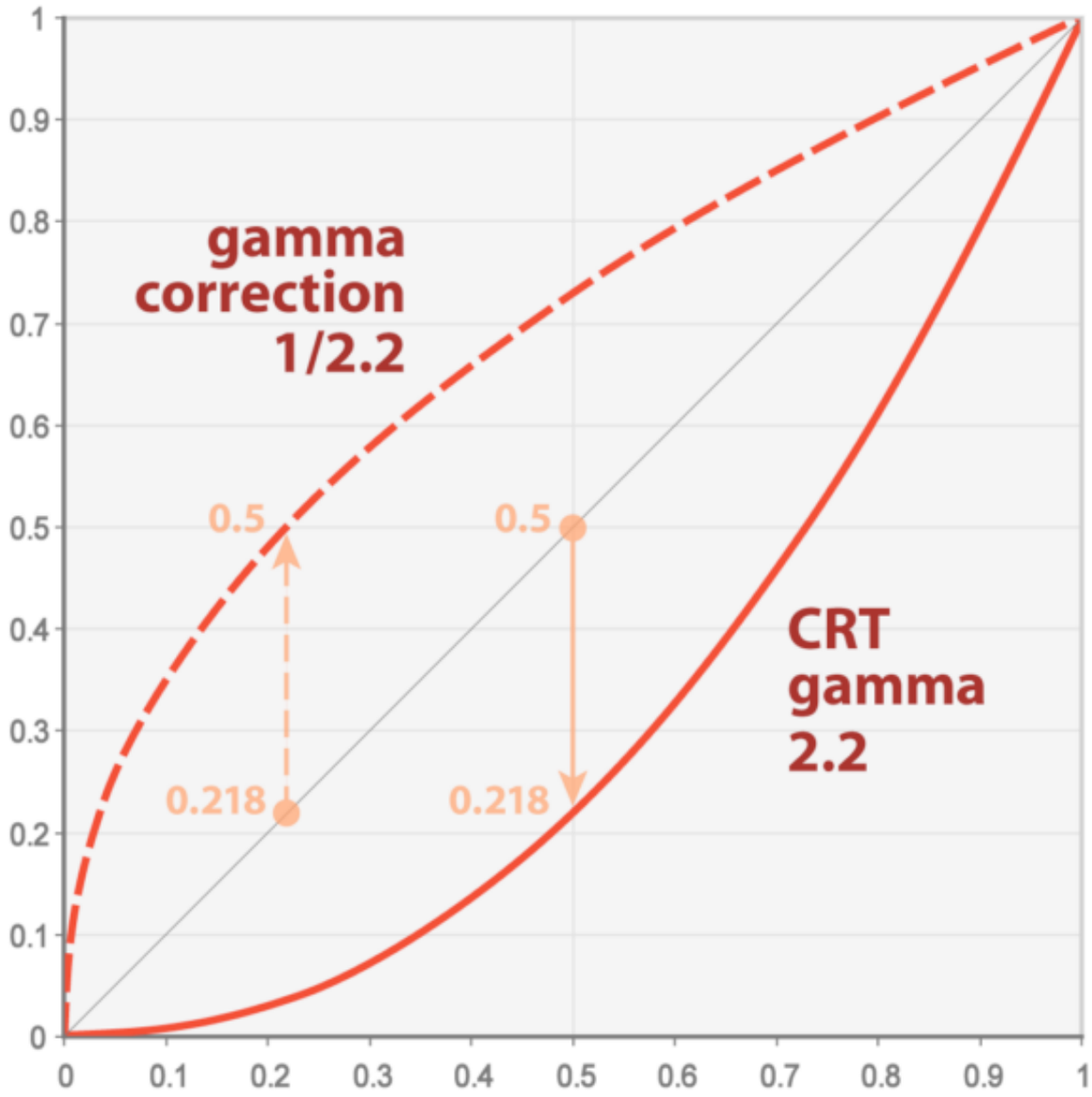
Transformações de Potência

- Ou transformação gama

$$s = c.r^{\gamma}$$

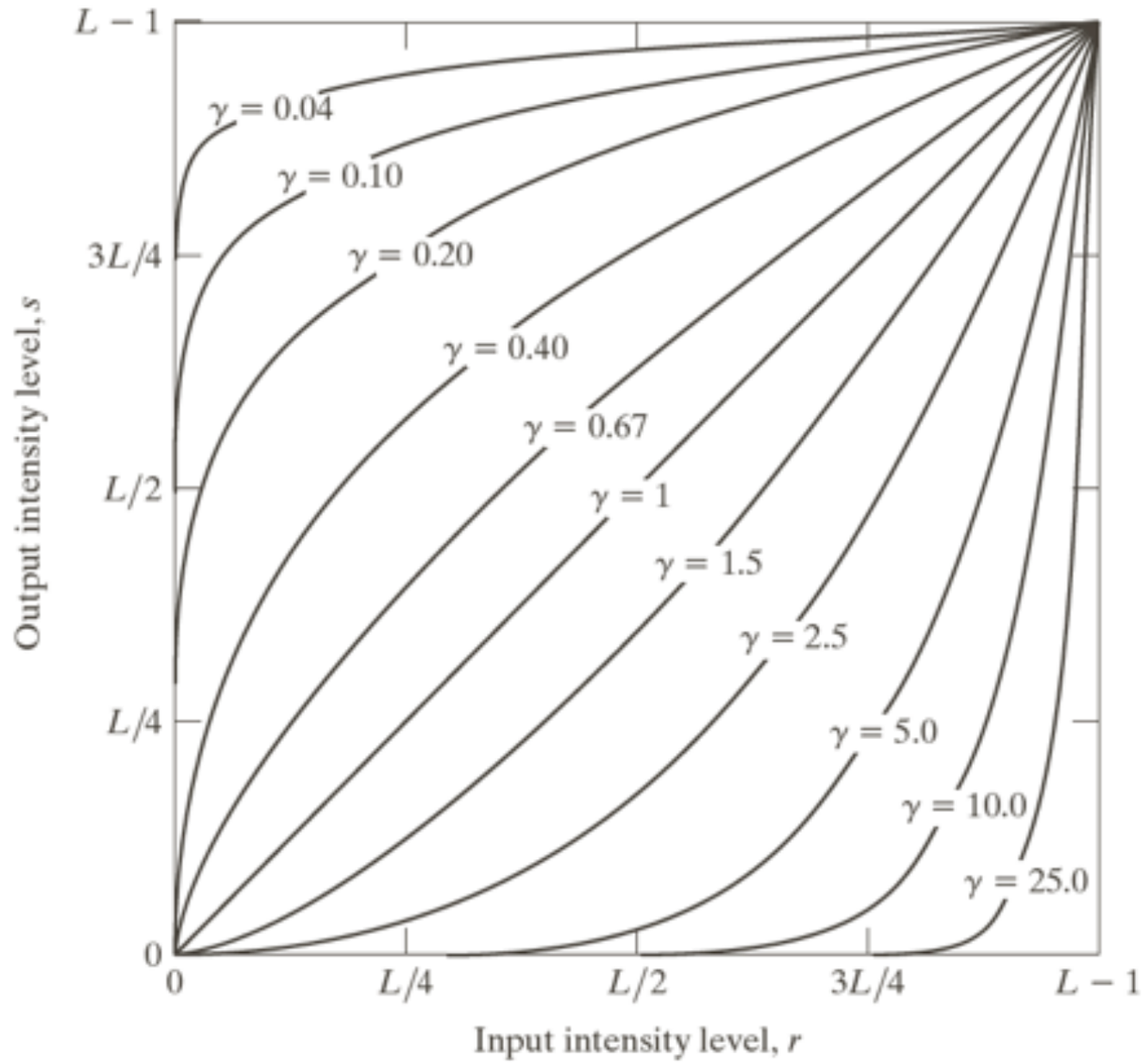
ou

$$s = (c + \varepsilon)^{\gamma}$$



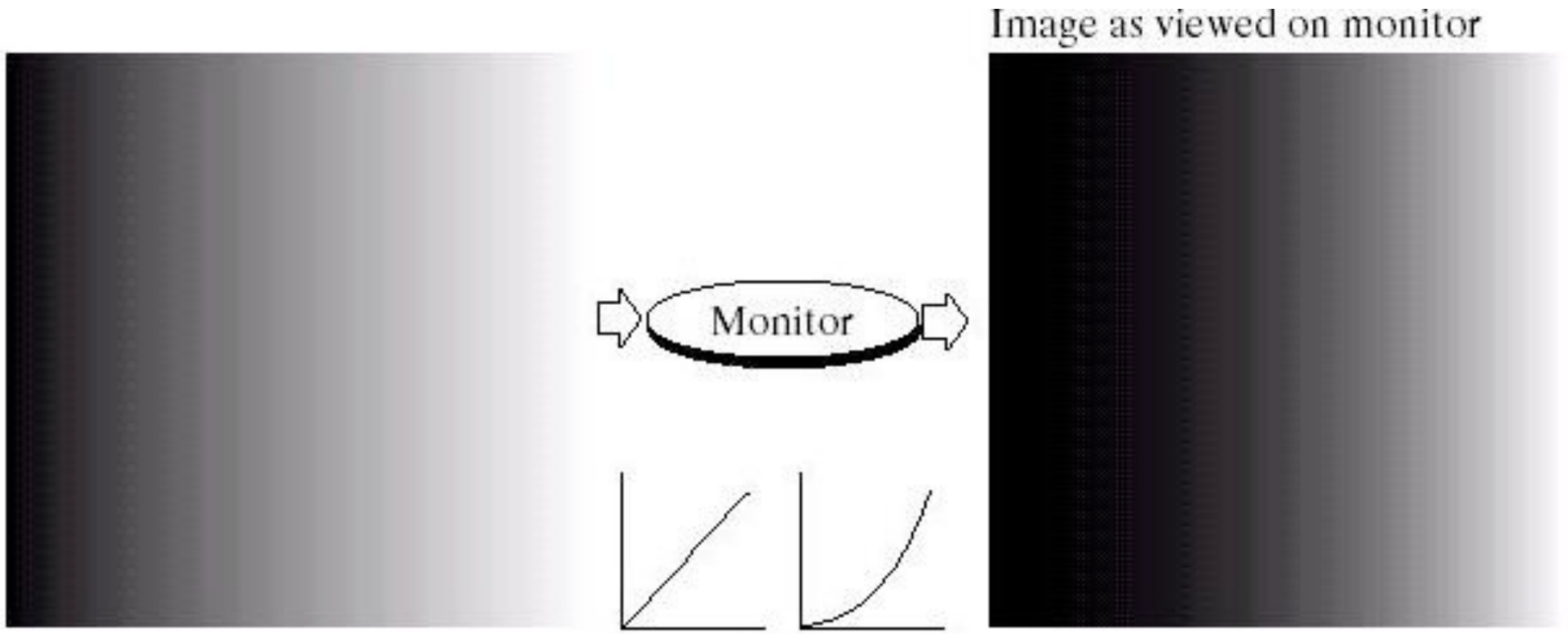
Transformações de Potência

- Correção Gama



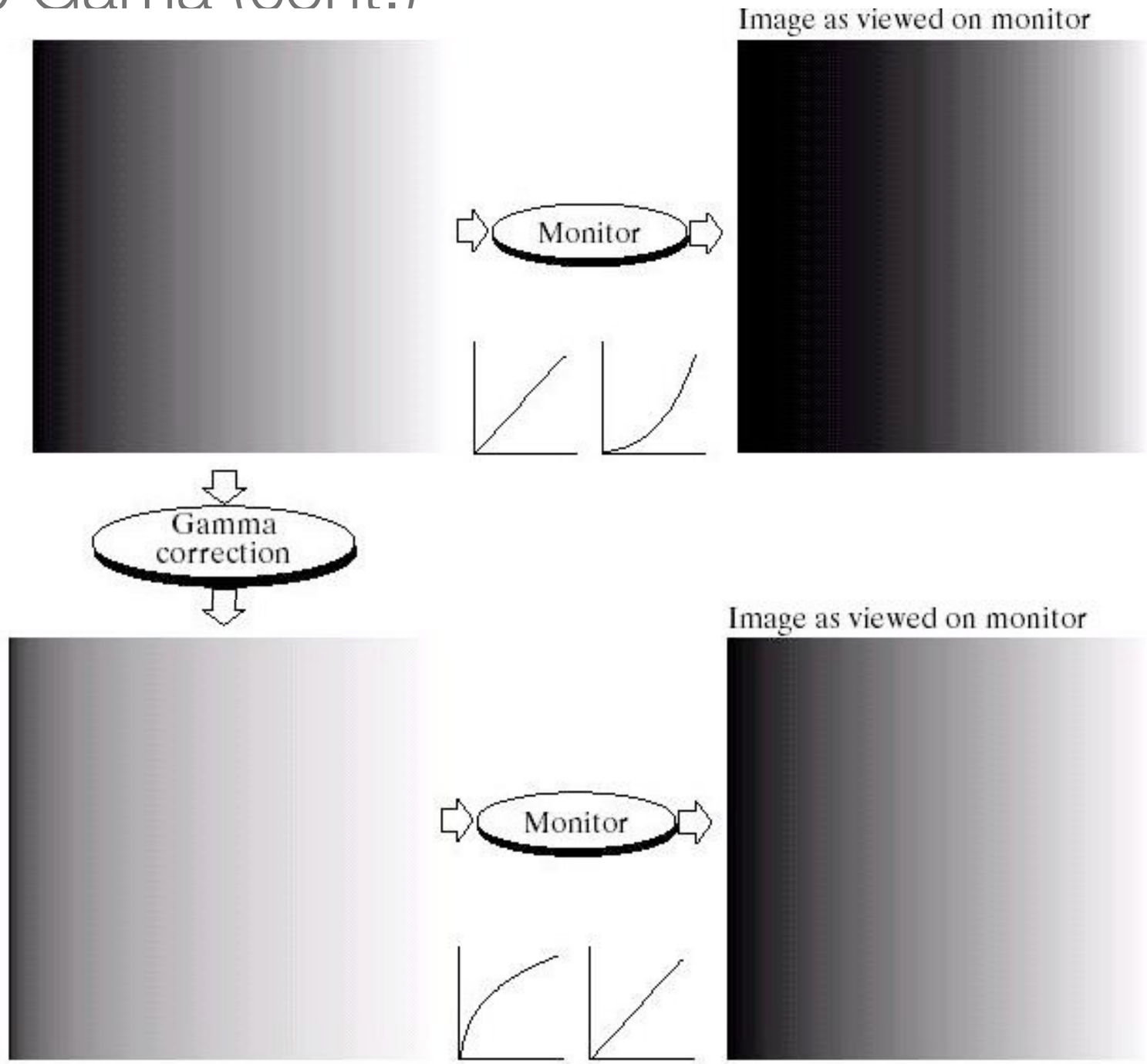
Transformações de Potência

- Permite com corrigir a resposta de um dispositivo
  - Correção Gama

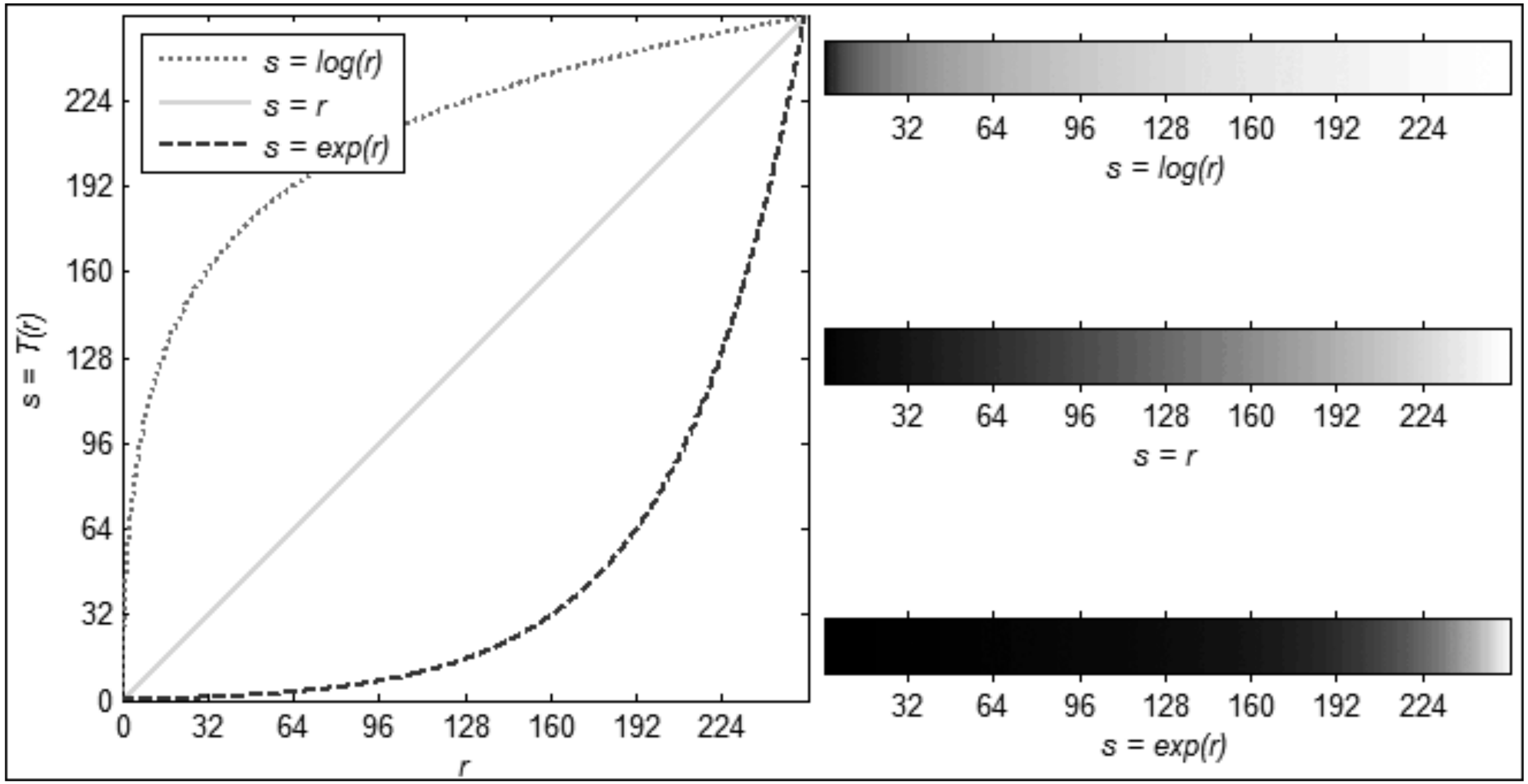


Correção Gama

- Correção Gama (cont.)



Transformações de Log e Exp

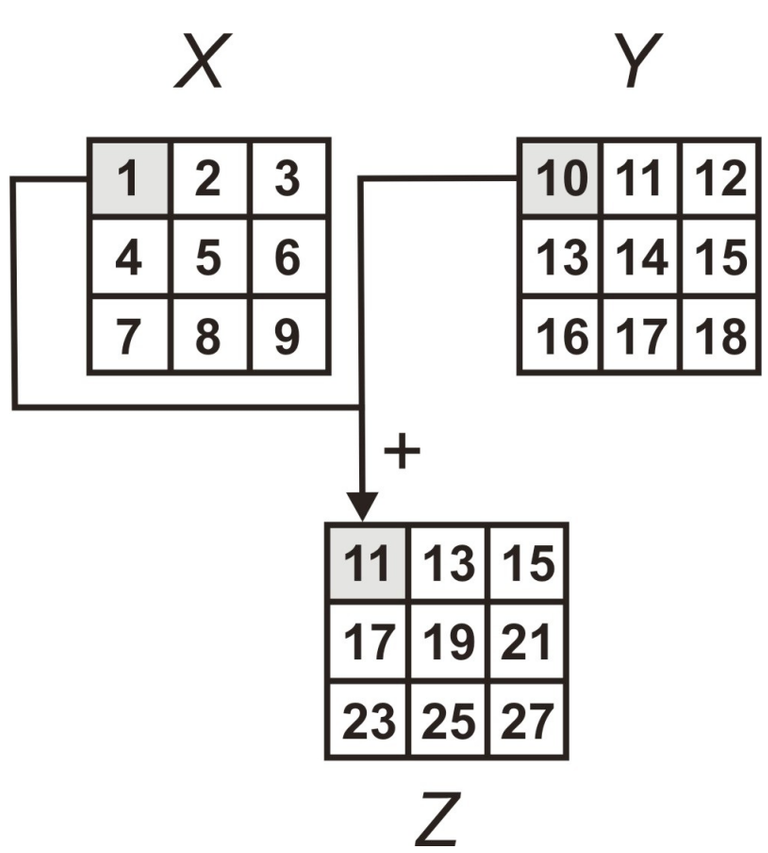


Transformações Lineares Por Partes

- Ajustes mais complexos necessitam de funções de comportamento mais sofisticado
  - Custo de avaliação
  - Controle preciso do comportamento local
- Aproximação Linear por partes

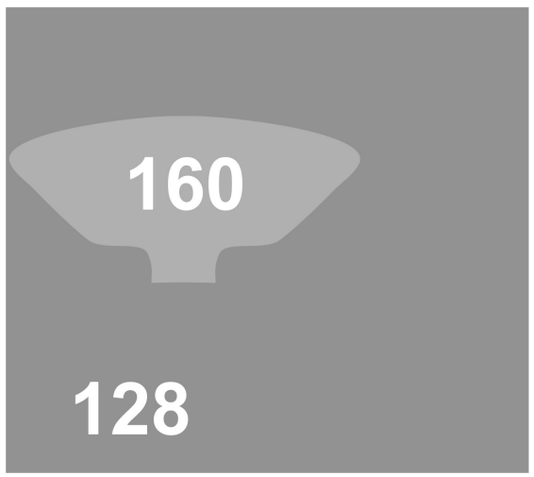
Operações Aritméticas

- Composição de duas imagens
  - Aplicação do operador pixel a pixel

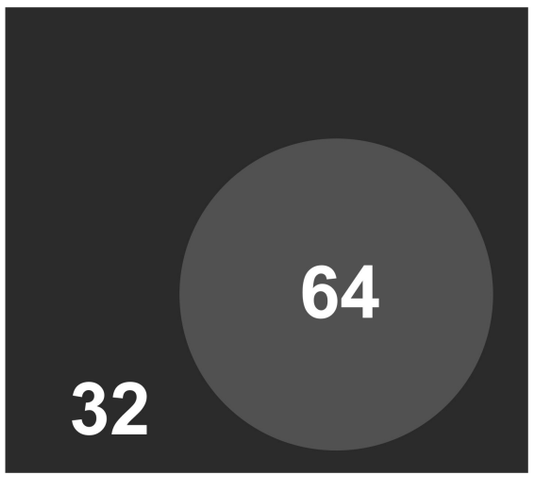


Operações Aritméticas

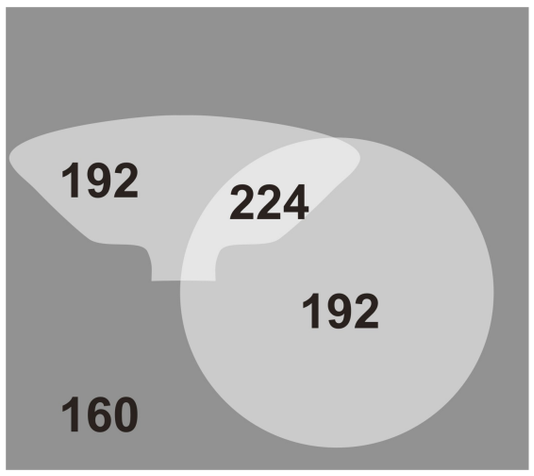
- Exemplos:



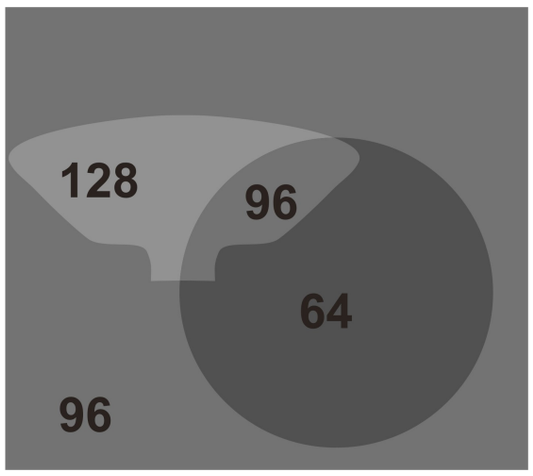
A



B



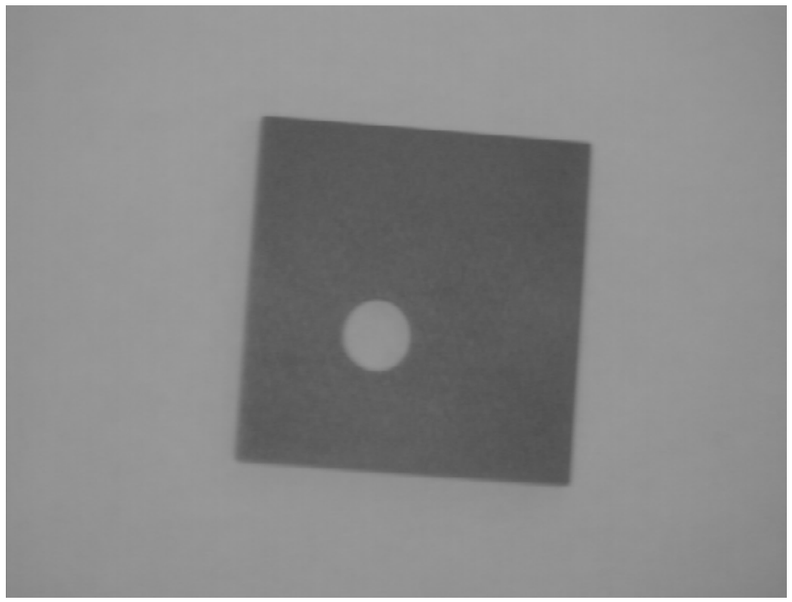
A+B

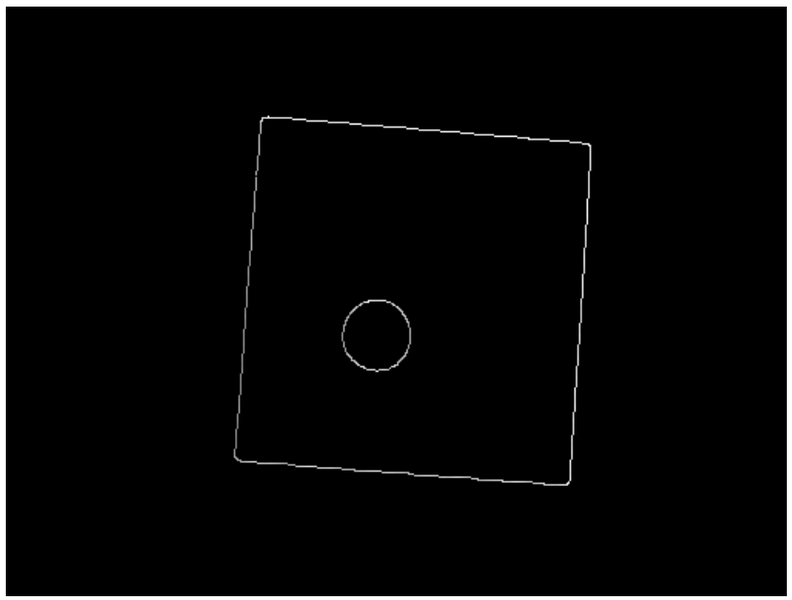


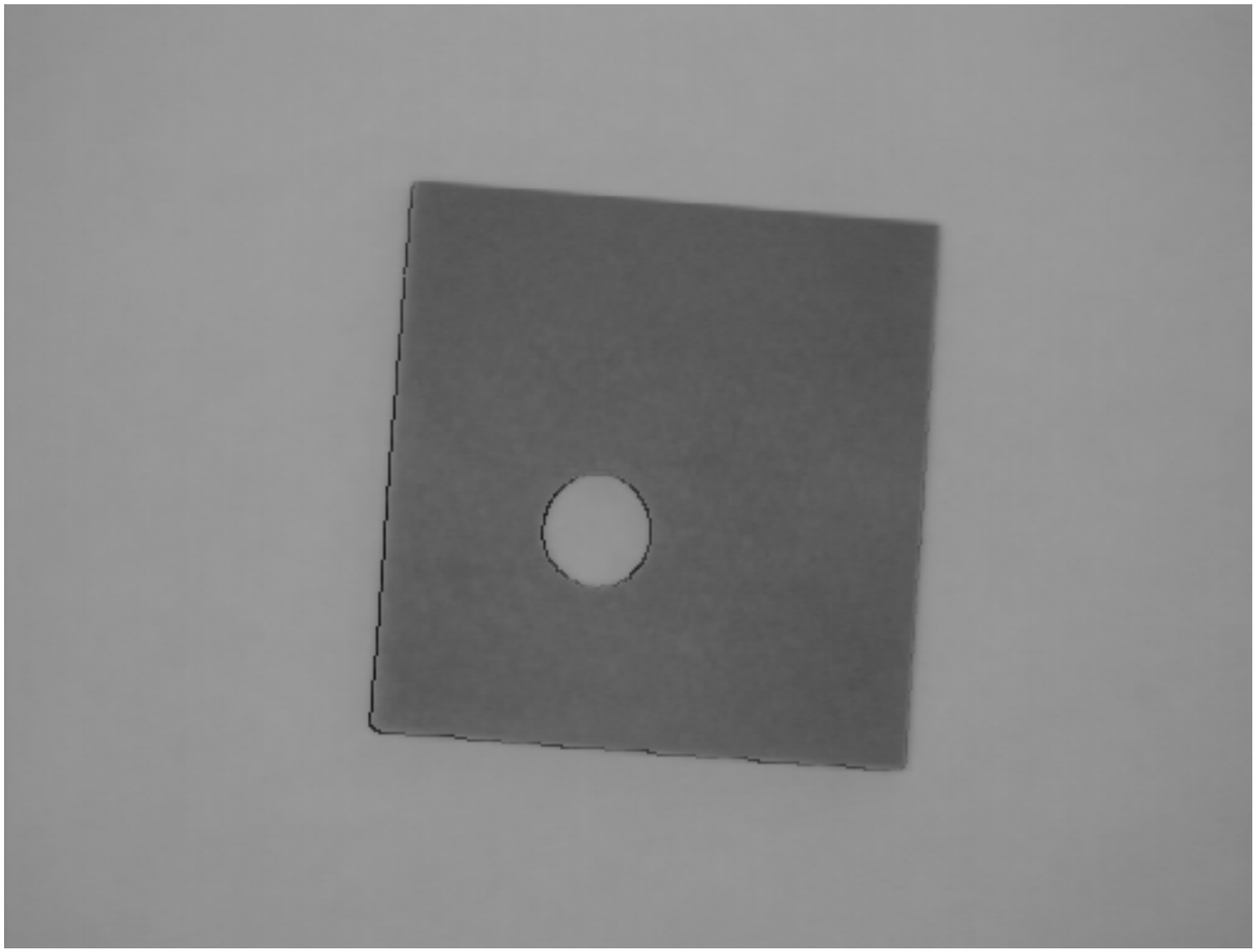
A-B

Operações Aritméticas

- Exemplos: A + B

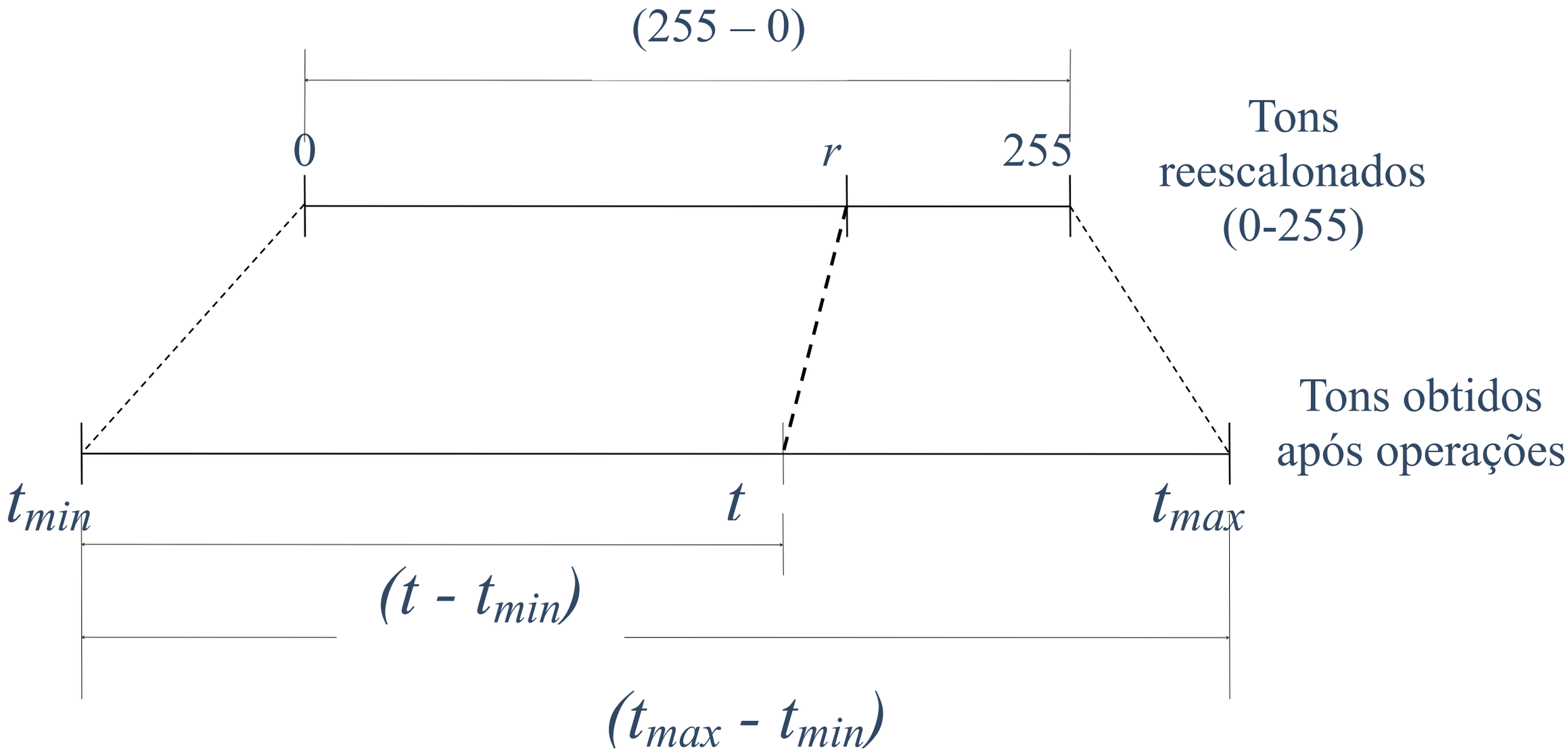






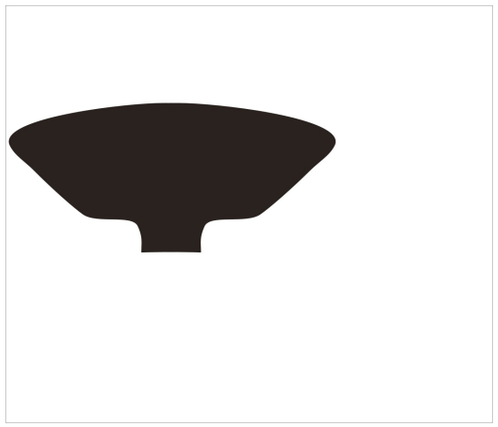
Operações Aritméticas

- Necessidade de re-escalonamento:

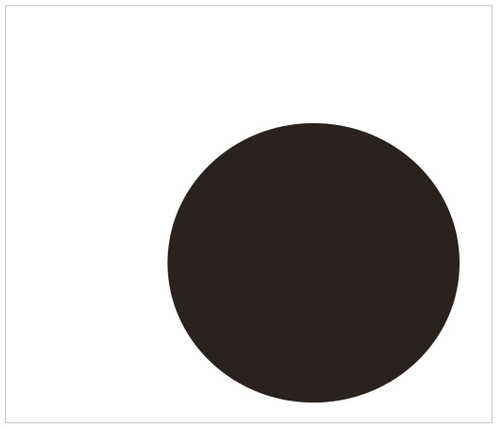


Operações Lógicas

- Aplicação de operadores lógicos pixel a pixel

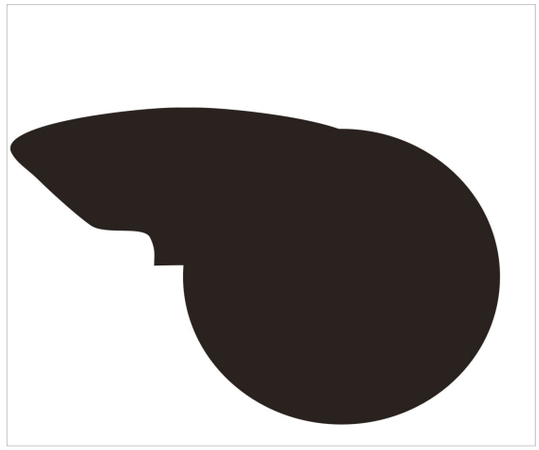


(b) Y

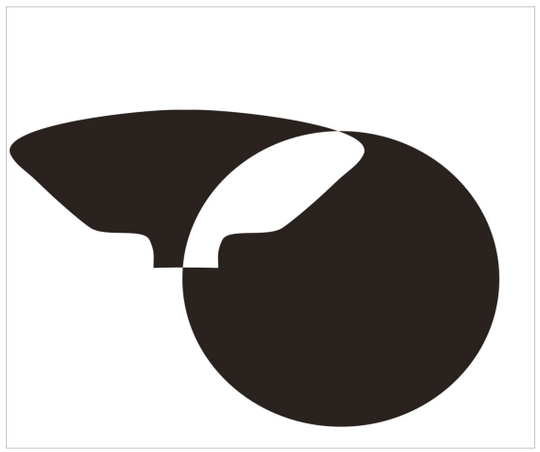


(a) X

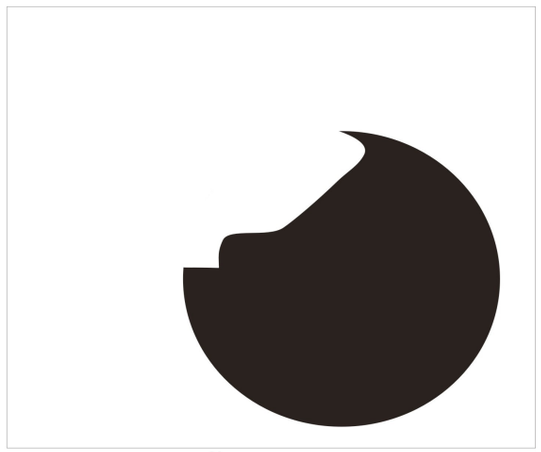
(a) (X) E (Y)



(b) (X) OU (Y)



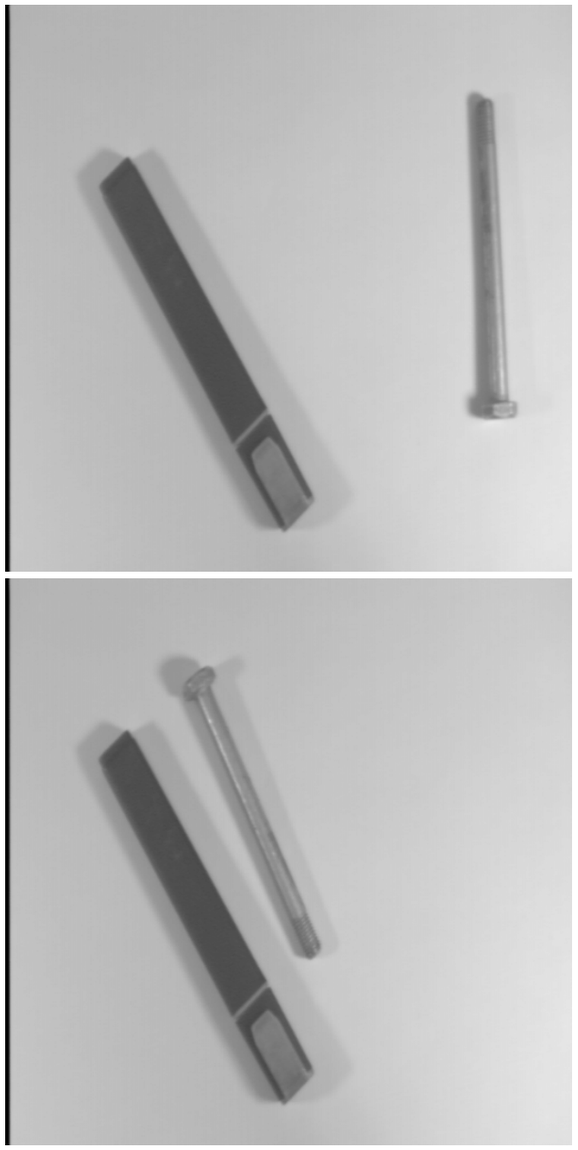
(c) (X) XOU (Y)



(b) (NÃO X) OU (Y)

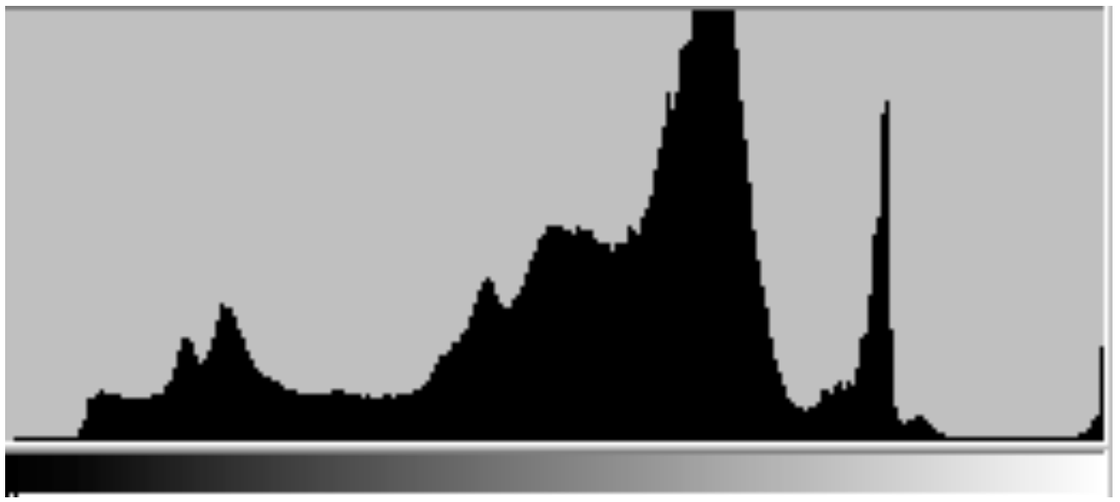
Operações Lógicas

- Exemplos: A AND B



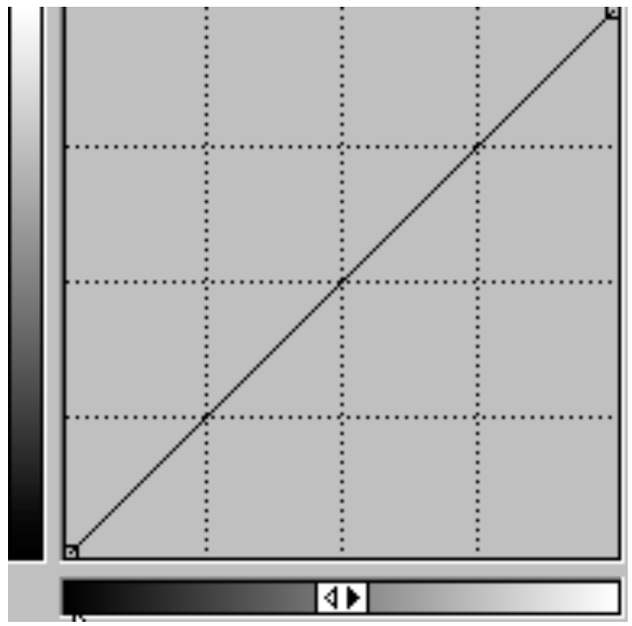
Processamento de Histograma

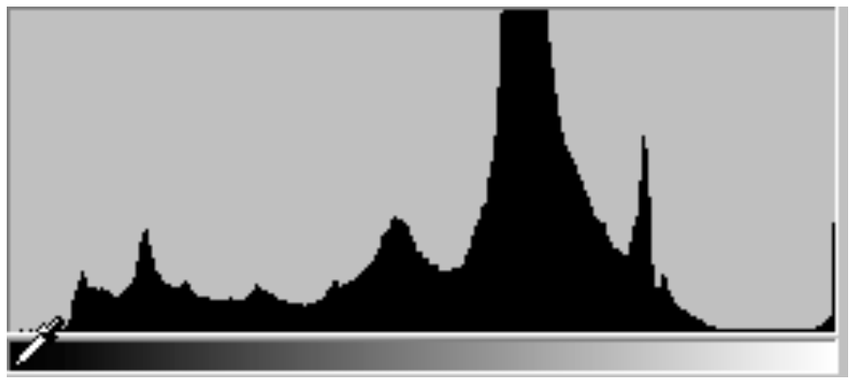
- Histograma:
  - Relaciona intensidade e sua ocorrência na imagem



Processamento de Histograma

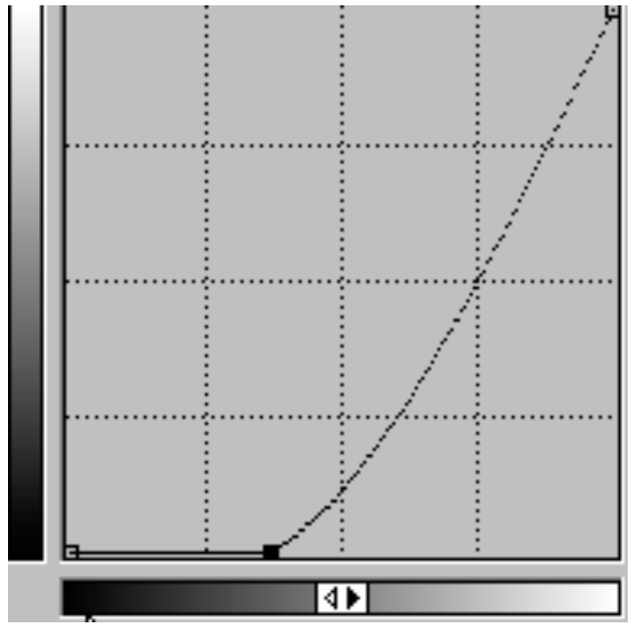
- Operações sobre os *pixels* tem reflexo no histograma da imagem

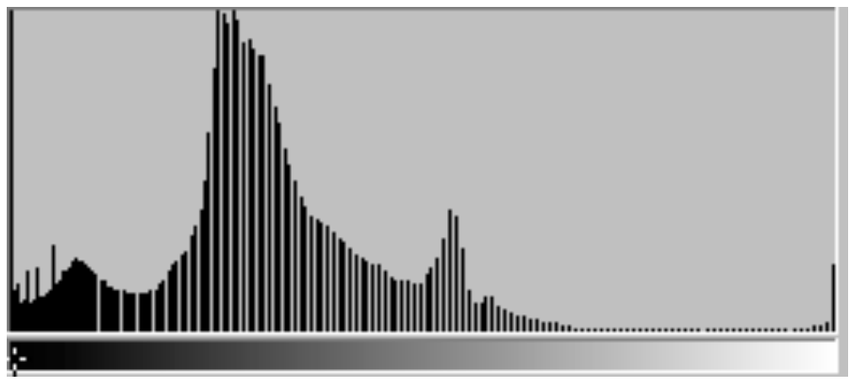




Processamento de Histograma

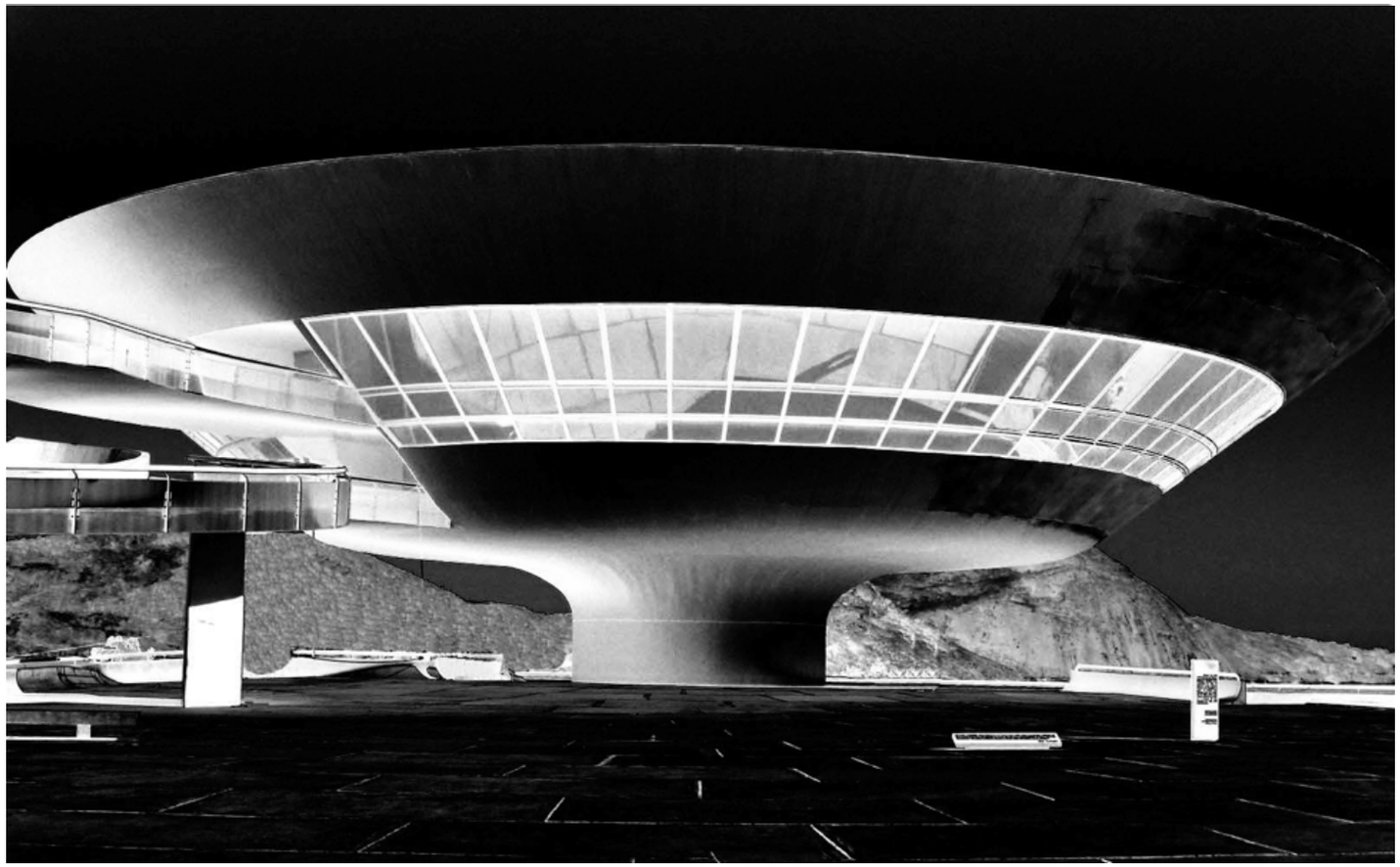
- Operações sobre os *pixels* tem reflexo no histograma da imagem

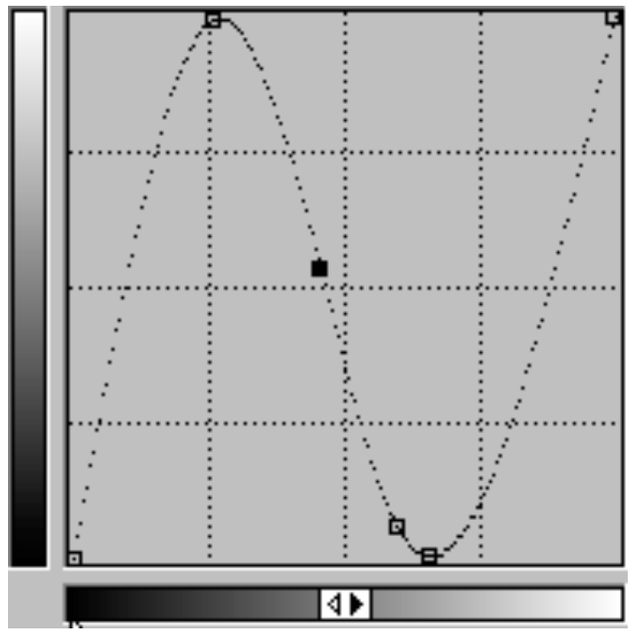




Processamento de Histograma

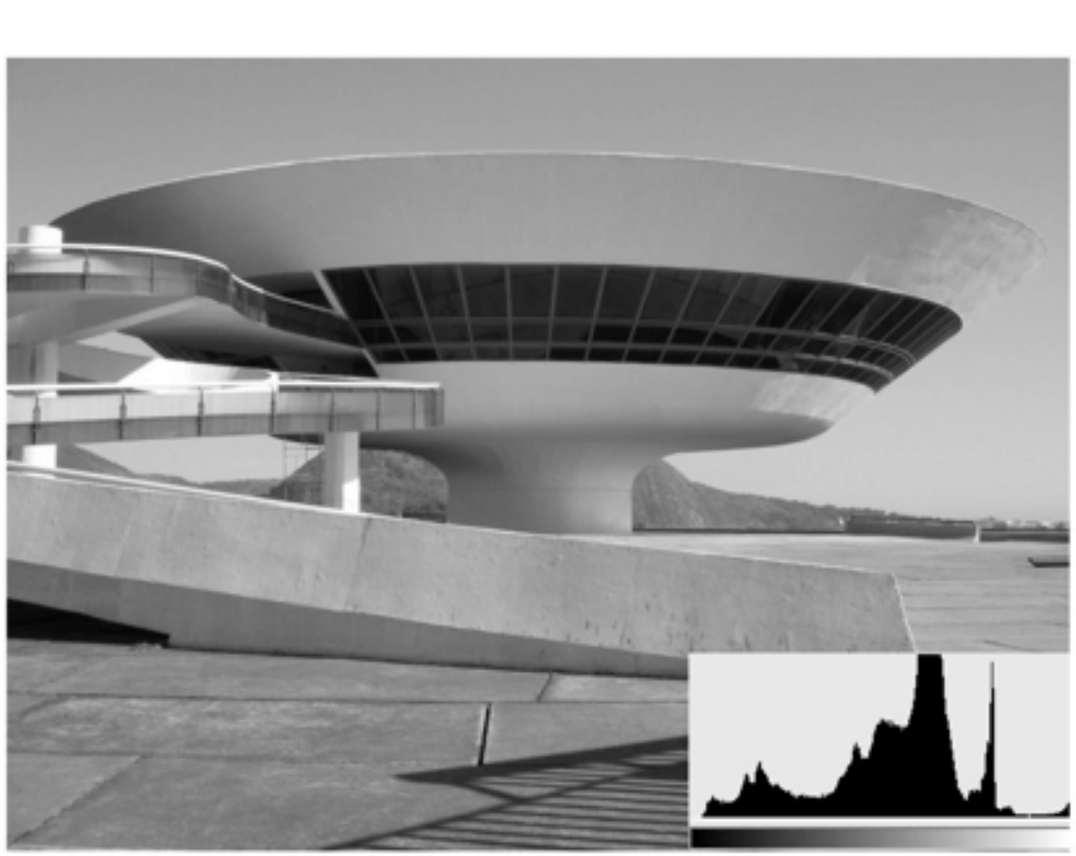
- Operações sobre os *pixels* tem reflexo no histograma da imagem

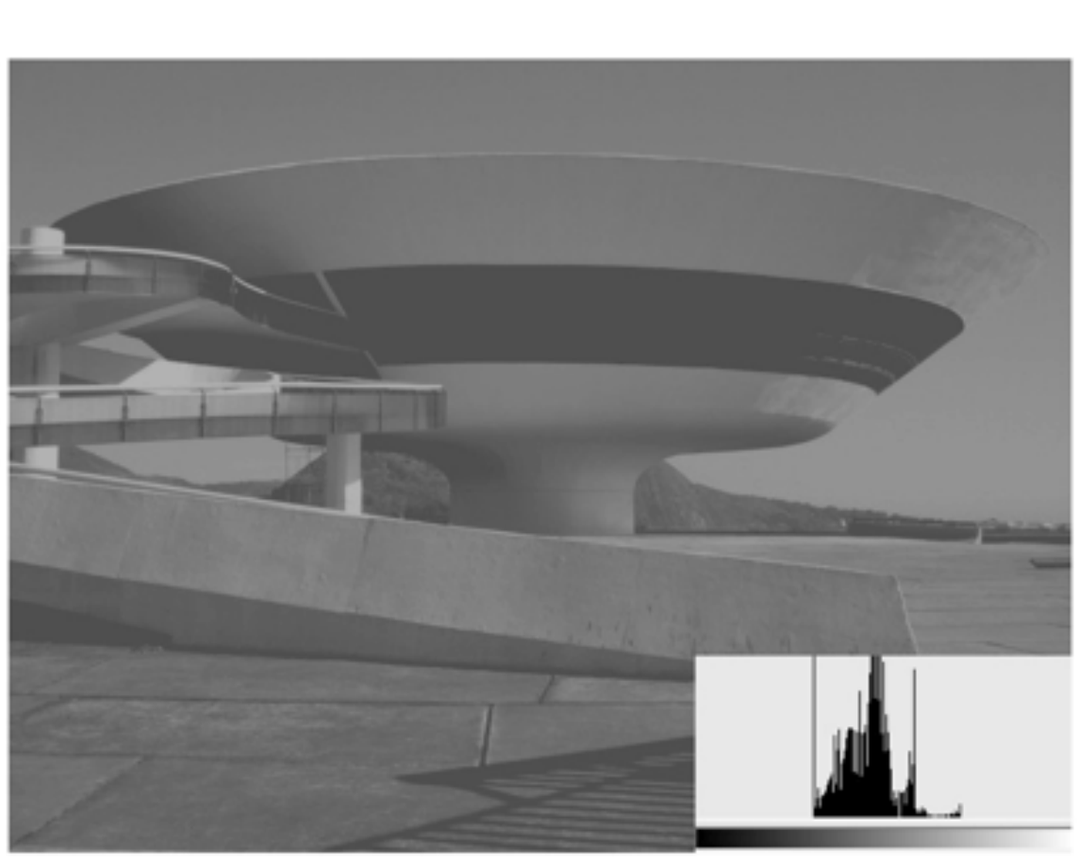




Processamento de Histograma

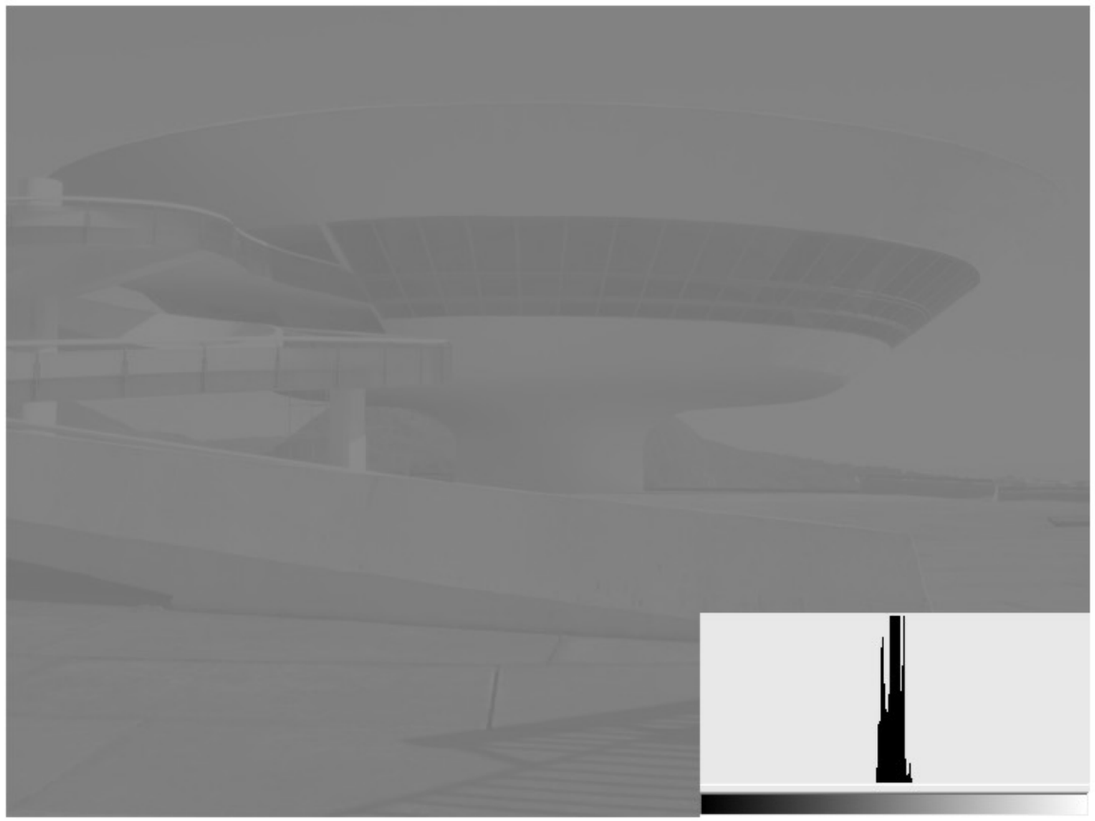
- Compressão do histograma





Processamento de Histograma

- Expansão do histograma



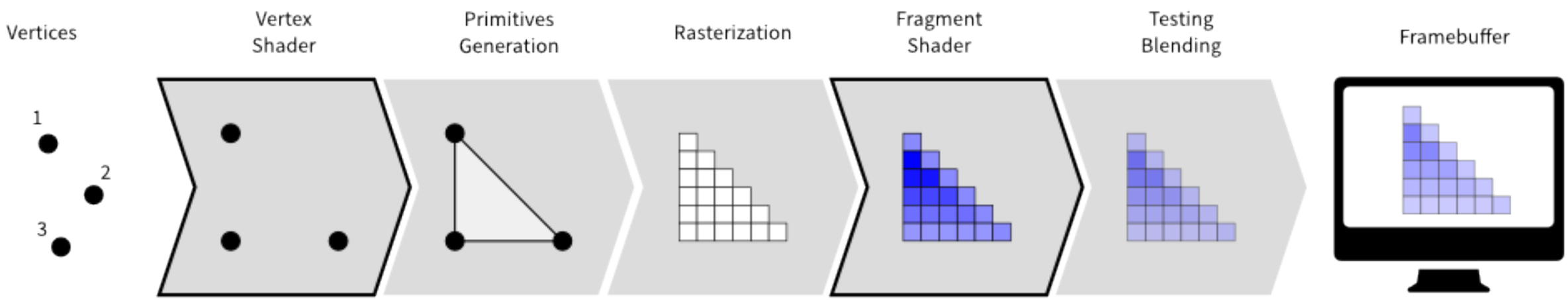
Transformações de Intensidade em WebGL/ Three.js

Imagens Digitais em WebGL/Three.js

- Duas possibilidades
  - Carregamento direto na página via objeto Image do HTML5
    - Definição da imagem a priori
  - Utilizar o Three.js/WebGL para promover o carregamento
    - Dinamico

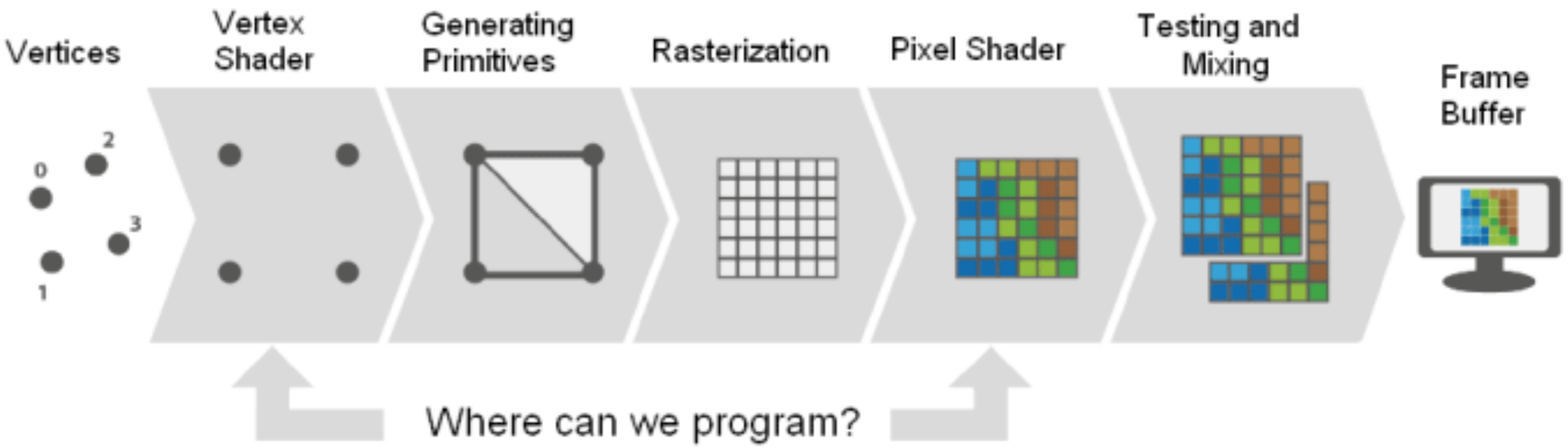
Imagens Digitais em WebGL/Three.js

- Problema:
  - Three.js/WebGL desenha primitivas geométricas
    - Etapa de rasterização
- Como associar uma imagem aos fragmentos de uma primitiva?



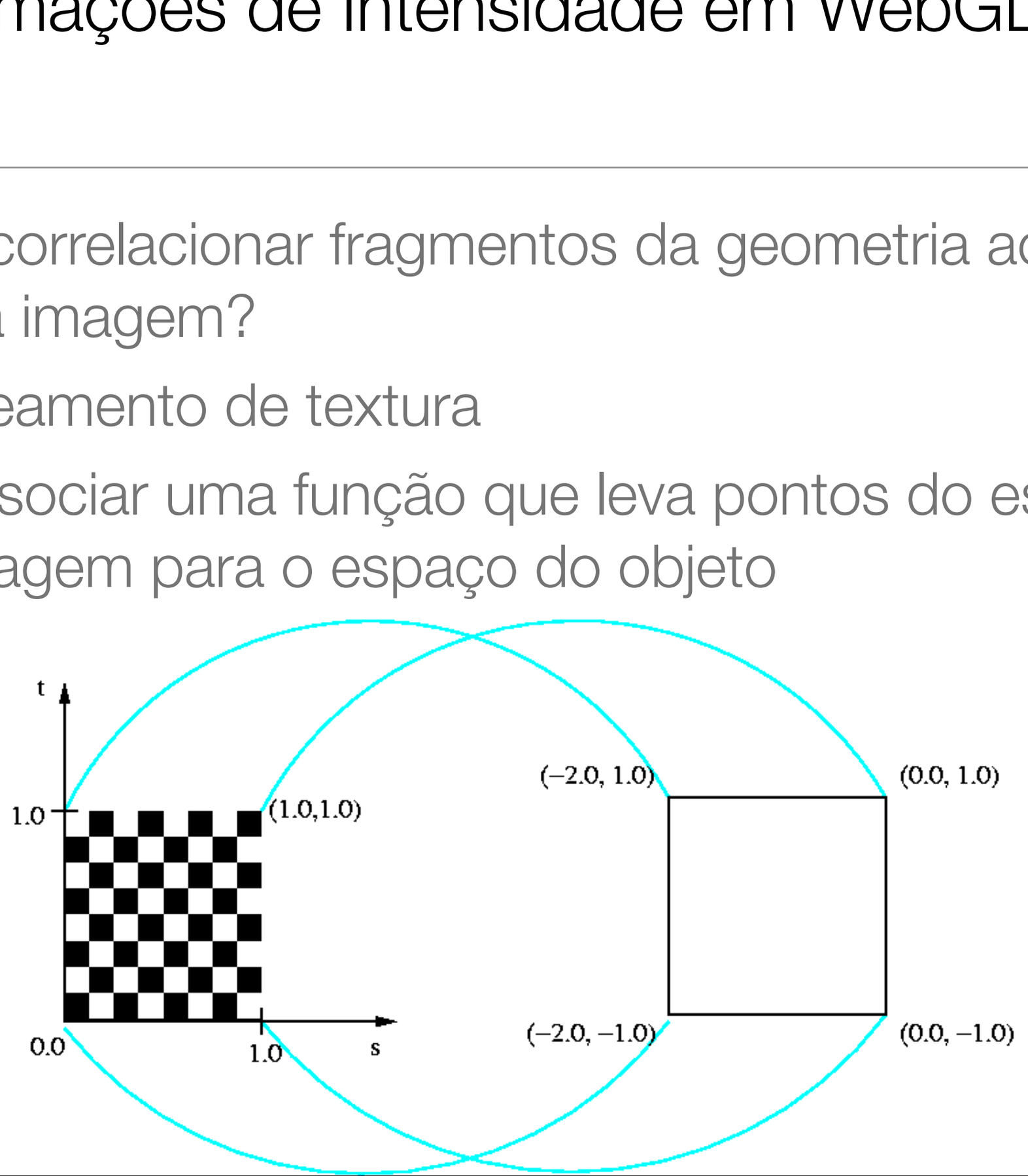
Imagens Digitais em WebGL/Three.js

- Como associar uma imagem aos fragmentos de uma primitiva?
  - Primitiva que mais se aproxima de ima imagem é um quadrado
- Como correlacionar fragmentos da geometria aos pixels de uma imagem?



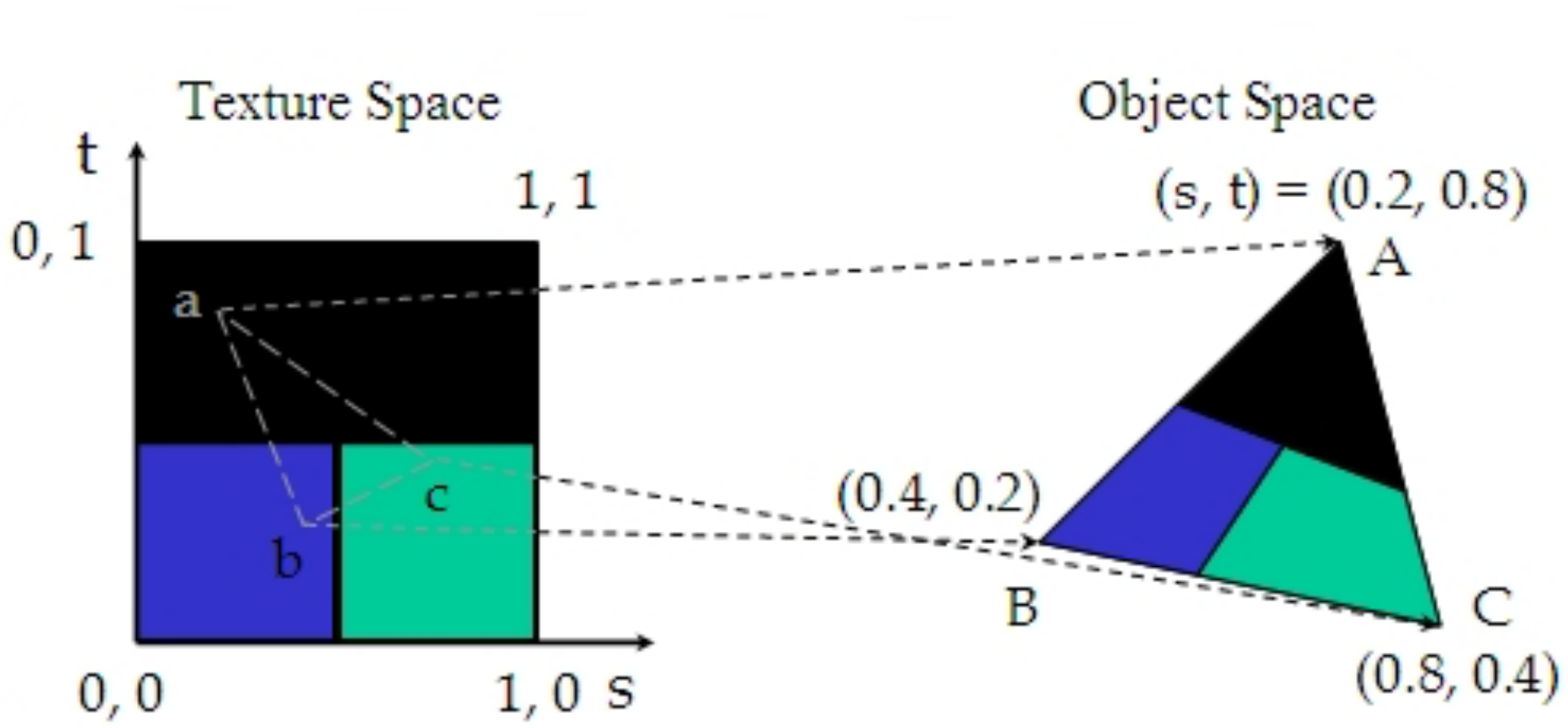
Transformações de Intensidade em WebGL/ Three.js

- Como correlacionar fragmentos da geometria aos pixels de uma imagem?
  - Mapeamento de textura
    - Associar uma função que leva pontos do espaço da imagem para o espaço do objeto



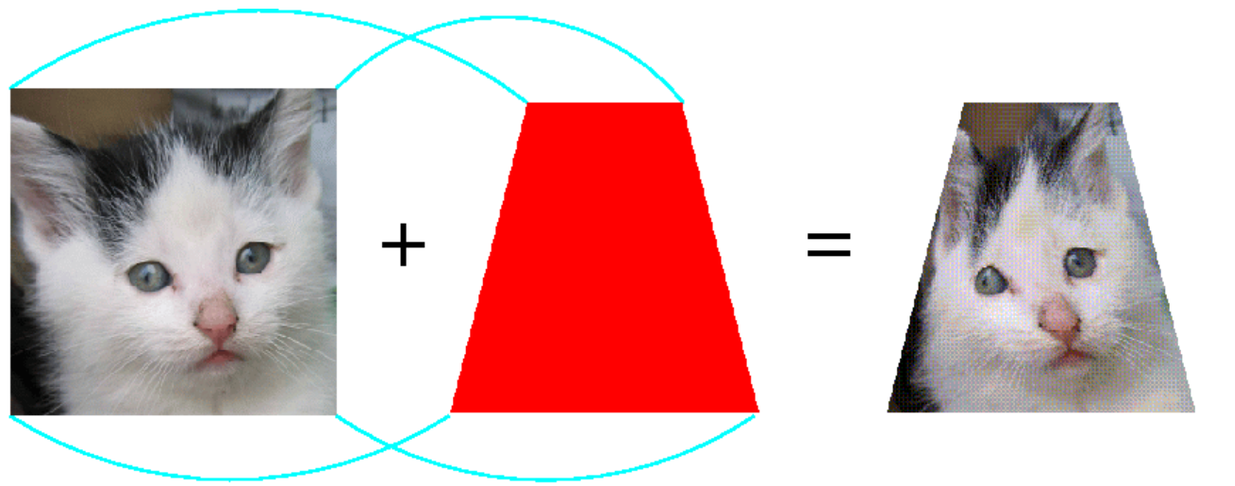
Imagens Digitais em WebGL/Three.js

- Mapeamento de textura
  - A função de mapeamento define como a imagem será projetada no objeto



Imagens Digitais em WebGL/Three.js

- Como?
  - Definir uma geometria que permita uma associação 1:1 entre pixels da imagem e fragmentos
    - Canvas da mesma dimensão da imagem
    - Quadrado que ocupe o canvas completamente



Imagens Digitais em WebGL/Three.js

- Mapeando uma imagem em um quadrado

```
var texture;
var renderer;
var scene;
var camera;
function init() {
   scene = new THREE.Scene();
   renderer = new THREE.WebGLRenderer();
   texture =
THREE.ImageUtils.loadTexture("../../
Assets/Images/lena.png");
   renderer.setClearColor(new
THREE.Color(0.0, 0.0, 0.0));
   camera = new THREE.OrthographicCamera(
-0.5, 0.5, 0.5, -0.5, -1.0, 1.0 );
   scene.add( camera );
   var planeGeometry = new
THREE.PlaneBufferGeometry(1.0, 1.0, 20,
20);
   var planeMat = new
THREE.MeshBasicMaterial( {map:texture,
wireframe:false, side:
THREE.DoubleSide } );
   var plane = new
THREE.Mesh( planeGeometry, planeMat );
   plane.position.set(0.0, 0.0, -0.5);
   scene.add( plane );
   document.getElementById("WebGL-
output").appendChild(renderer.domElement
);
   renderer.clear();
   requestAnimationFrame(render);
   renderer.render(scene, camera);
};
```

Imagens Digitais em WebGL/Three.js

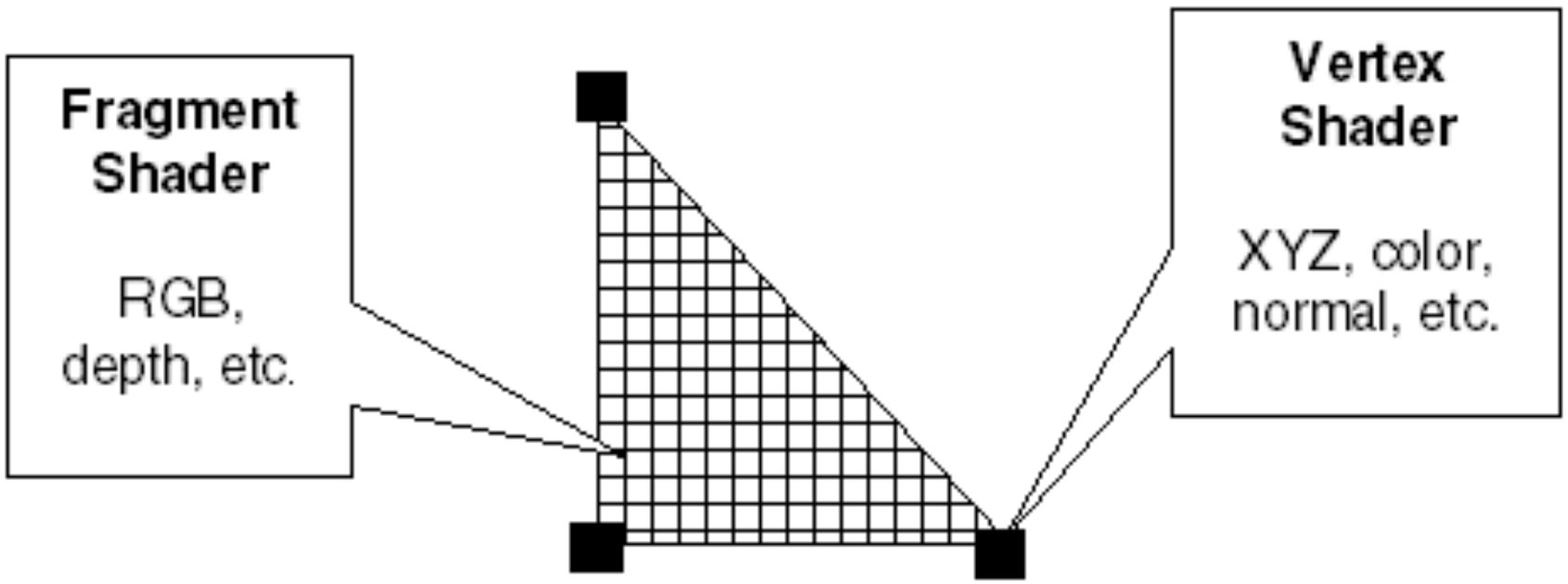
- Mapeando uma imagem em um quadrado (cont.)

```
function render() {

    if (!texture.image) {
        requestAnimationFrame(render);
        }
    else {
        console.log("****" + texture.image.width);
        console.log("****" + texture.image.height);
        renderer.setSize(texture.image.width, texture.image.height);
        renderer.render(scene, camera);
        }
}
```

Transformações de Intensidade em WebGL/ Three.js

- Como processar os dados geométricos e da imagem?
  - Segundo Passo: Construir os *shaders* que processem os vertice e os fragmentos/pixels da imagem



Shaders em WebGL/Three.js

- *Vertex Shader* Minimo

```
<script id="base-vs" type="x-shader/x-vertex">
    varying vec3 vColor;

    void main(){
        vColor = color;
        gl_Position = projectionMatrix *
                      modelViewMatrix * vec4(position, 1.0);
    }
</script>
```

Shaders em WebGL/Three.js

- *Fragment Shader* Minimo

```
<script id="base-fs" type="x-shader/x-fragment">
    varying vec3 vColor;
    void main() {
        gl_FragColor = vec4( vColor.rgb, 1.0 );
        }
</script>
```

Shaders em WebGL/Three.js

- Chamando os shaders dentro do Threejs
  - Material vinculado ao *shader* (**`THREE.ShaderMaterial`**)

```
var matShader = new THREE.ShaderMaterial( {
    vertexColors: THREE.VertexColors,
vertexShader: document.getElementById( 'base-vs' ).textContent,
fragmentShader: document.getElementById( 'base-fs' ).textContent
} );
```

Shaders em WebGL/Three.js

- Chamando os shaders dentro do Threejs
  - Associar o material (`THREE.ShaderMaterial`) a uma geometria

```
// Plane
var plane = new THREE.Geometry();
plane.vertices.push(   new THREE.Vector3( -0.75, -0.75, 0.0 ),
                       new THREE.Vector3(  0.75, -0.75, 0.0 ),
                       new THREE.Vector3(  0.75,  0.75, 0.0 ),
                       new THREE.Vector3( -0.75,  0.75, 0.0 )    );
plane.faces.push( new THREE.Face3(0, 1, 2));
plane.faces.push( new THREE.Face3(0, 2, 3));
plane.faces[0].vertexColors[0] = new THREE.Color("rgb(255,0,0)");
plane.faces[0].vertexColors[1] = new THREE.Color("rgb(0,255,0)");
plane.faces[0].vertexColors[2] = new THREE.Color("rgb(0,0,255)");
plane.faces[1].vertexColors[0] = new THREE.Color("rgb(255,0,0)");
plane.faces[1].vertexColors[1] = new THREE.Color("rgb(0,255,0)");
plane.faces[1].vertexColors[2] = new THREE.Color("rgb(0,0,255)");
var planeMesh = new THREE.Mesh( plane, matShader );
scene.add( planeMesh );
```

Transformações de Intensidade em WebGL/ Three.js com *Shaders*

- Como passar a imagem como fonte de cor de cada pixel?
  - Passar ao *shader* a textura (`Sampler2D`)

```
<script id="base-fs" type="x-shader/x-fragment">
    precision mediump float;

    uniform sampler2D texture;
    varying vec2 vUv;

    void main(void) {
        vec4 c = texture2D(texture, vUv);
        gl_FragColor = vec4(c.rgb, 1.0);
    }
</script>
```

Transformações de Intensidade em WebGL/ Three.js com *Shaders*

- Como passar a imagem como fonte de cor de cada pixel?
  - Definir a textura como parte do Material (`THREE.ShaderMaterial`)

```
function render() {
if (!texture.image)
     requestAnimationFrame(render);
else {
     uniforms = {
          texture: { type: "t", value:texture }
     };
     var matShader = new THREE.ShaderMaterial( {
               uniforms: uniforms,
               vertexShader: document.getElementById( 'base-vs' ).textContent,
               fragmentShader: document.getElementById( 'base-fs' ).textContent
          } );
     }
```

Transformações de Intensidade em WebGL/ Three.js com *Shaders*

- Como passar a imagem como fonte de cor de cada pixel?
  - Definir a textura como parte do Material (`THREE.ShaderMaterial`)

```
        // Plane
        var planeGeometry = new THREE.PlaneBufferGeometry(1.0, 1.0,
20, 20);
        var plane = new THREE.Mesh( planeGeometry, matShader );
        plane.position.set(0.0, 0.0, -0.5);
        scene.add( plane );
        renderer.setSize(texture.image.width, texture.image.height);
        renderer.render(scene, camera);
        }
}
```

Transformações de Intensidade em WebGL

- Imagem é mapeada em uma forma geométrica (quadrado) na forma de uma textura
  - Quadrado do mesmo tamanho do canvas
  - Cada fragmento corresponde a um fragmento
  - Fragment Shaders processam os fragmentos/pixels da imagem

A seguir…

Filtragem Espacial