

Sistema de Visualização

Prof. Antonio L. Apolinário Junior
Estagiária Docente: Rafaela Alcantara

UFBA/IM/DCC/BCC - 2018.1

1

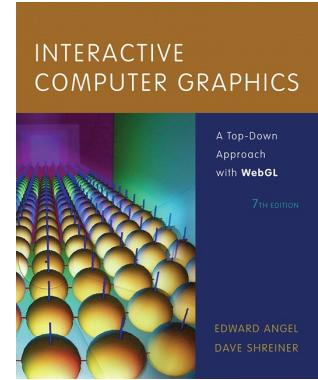
Roteiro

- Sistema de visualização
- Processo de formação de imagem
- Modelo de Câmera Virtual
 - Controles
 - Mudança de sistema de coordenadas
- Aplicações utilizando Three.JS/WebGL

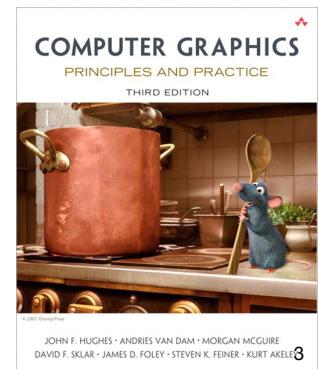
2

Leitura de referencia

- Capítulo 5
Interactive Computer Graphics - A top-down approach with OpenGL
7th Edition
Angel, Edward.
Addison-Wesley. 2014.

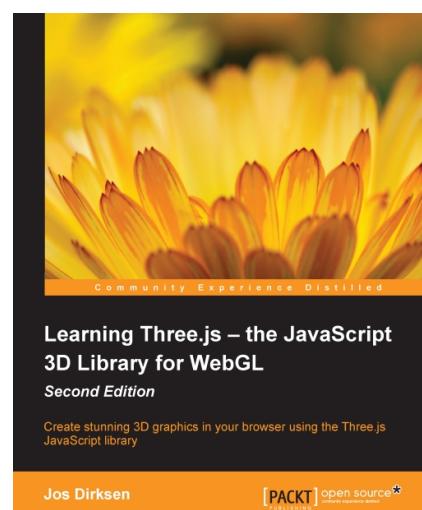


- Capítulos 13
Computer Graphics : Principles and Practice Third Edition in C
John F. Hughes / Andries van Dam
Morgan McGuire / David F. Sklar
James D. Foley / Steven K. Feiner
Addison-Wesley. 2013.



Leitura de referencia

- Capítulo 2 e 9
Learning Three.js: The JavaScript 3D Library for WebGL
Jos Dirksen
2nd Edition.
Packt Publishing - 2015.

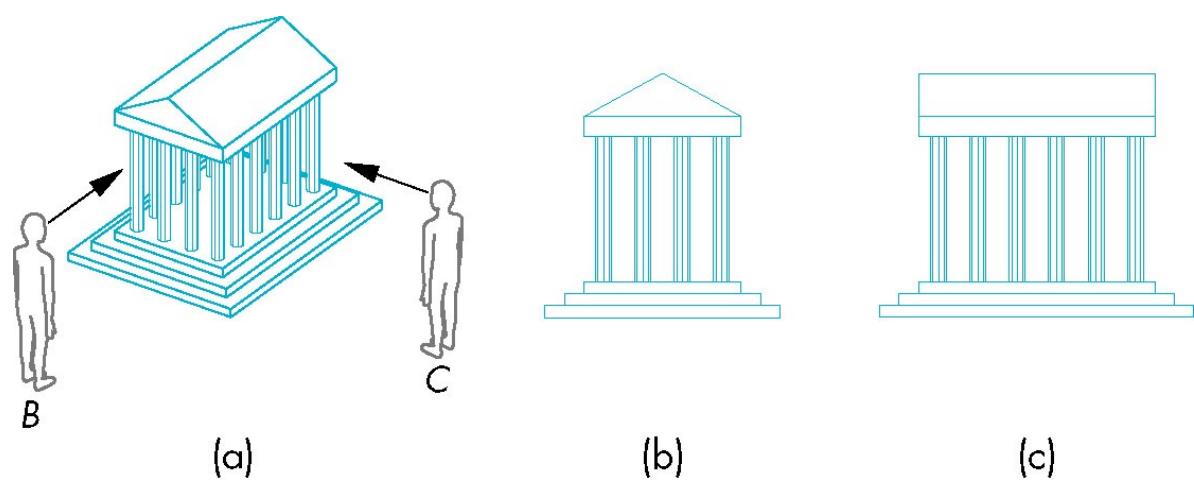


Sistema de visualização

5

Sistema de visualização

- Problema básico:
 - Qualquer imagem 2D do mundo 3D depende de um referencial



6

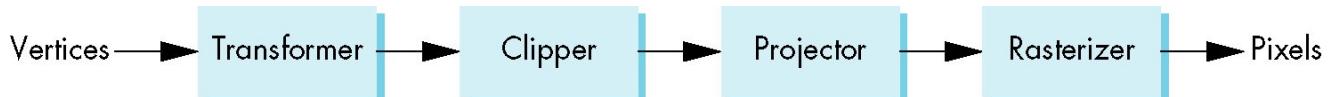
Sistema de visualização

- Problema básico:
 - Qualquer imagem 2D do mundo 3D depende de um referencial

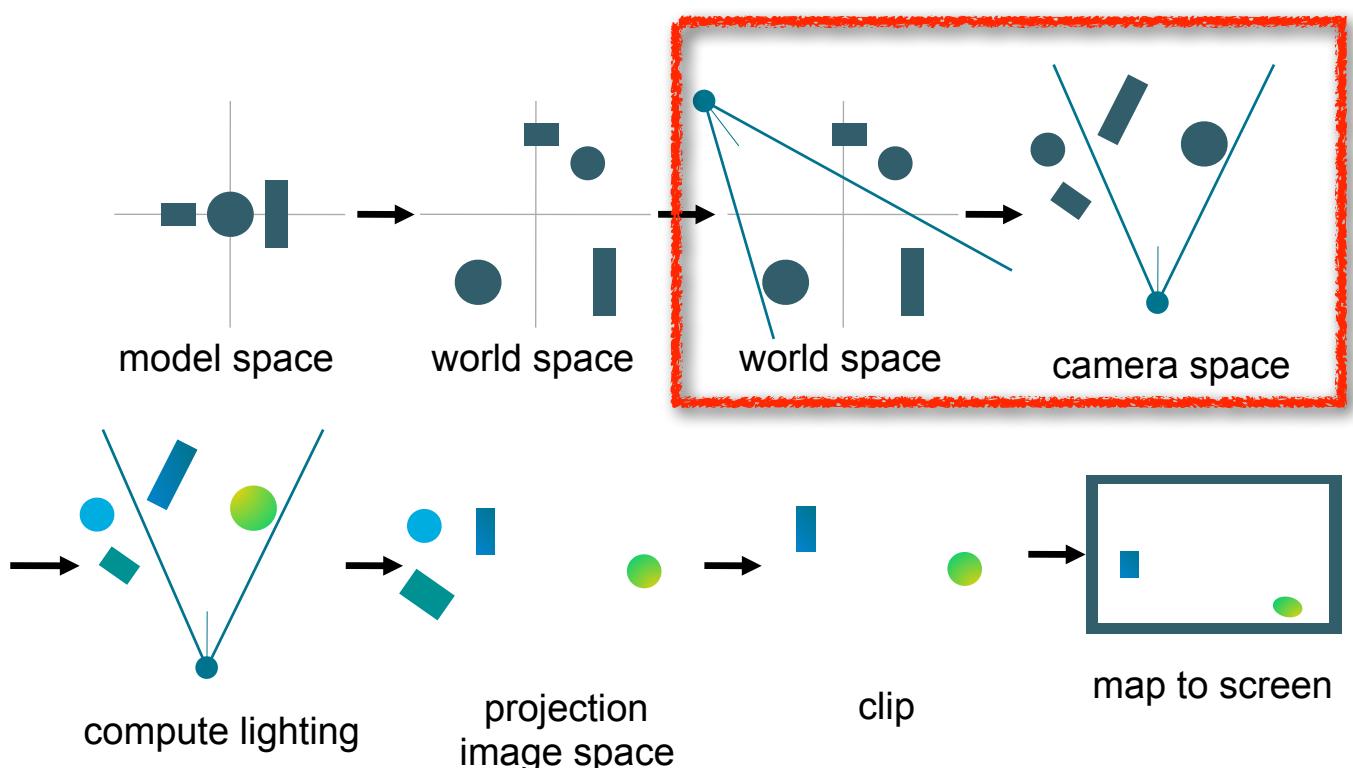


Sistema de visualização

- No contexto do *pipeline* gráfico:
 - Exerce influencia nas etapas de **recorte (clipping)** e **projeção**
 - Envolve a definição de:
 - parâmetros da camera/observador
 - mudança de sistema de referencia para a projeção



Pipeline Gráfico



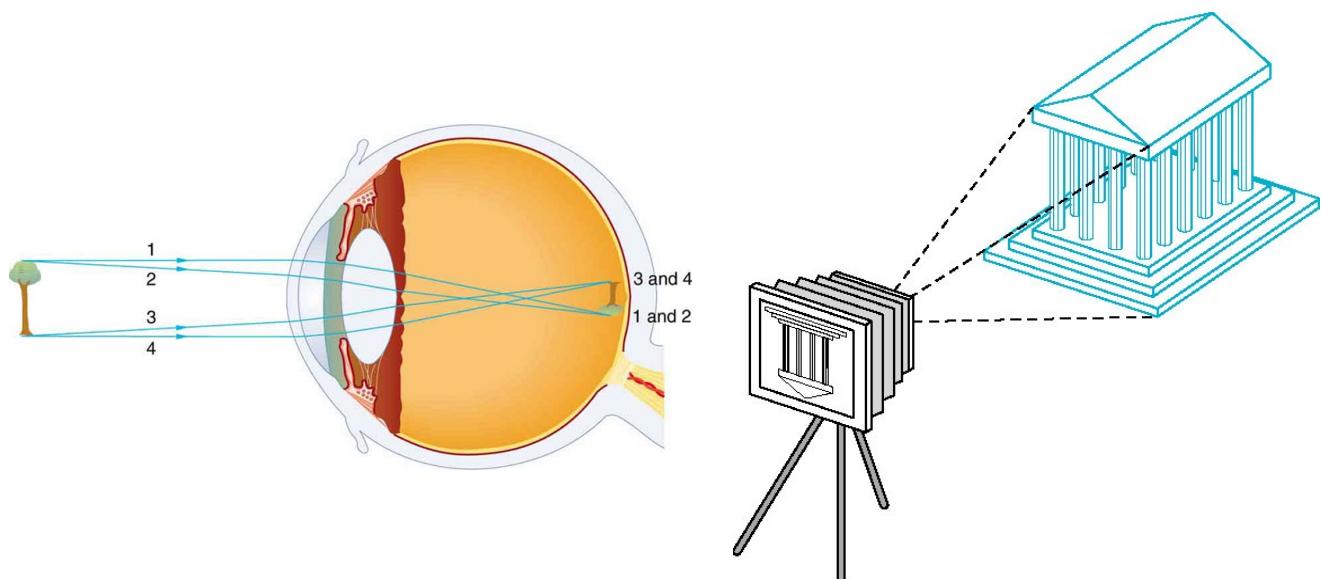
9

Processo de formação de imagem

10

Processo de formação de imagem

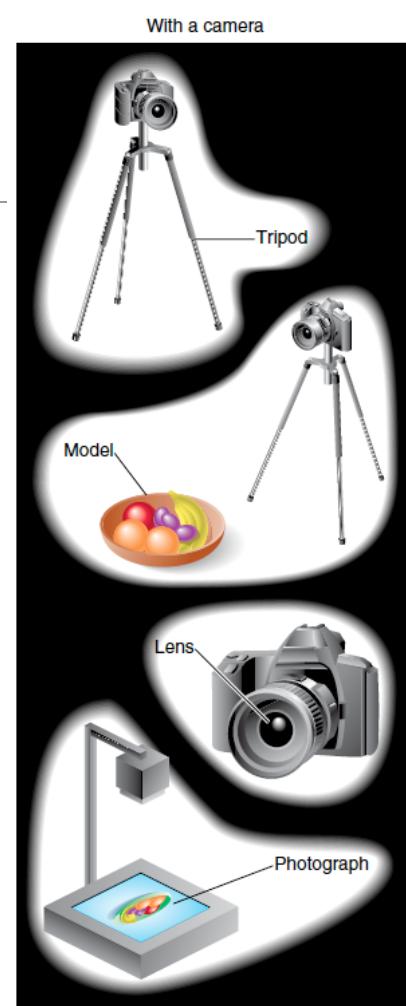
- Analogia com o processo naturais e artificiais



11

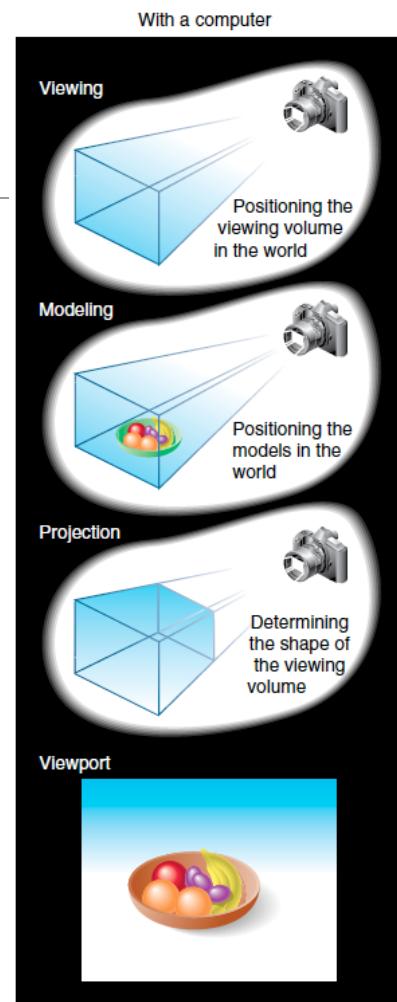
Processo de formação de imagem

- Analogia com o processo da fotografia
 - Posicionamento da câmera
 - Enquadramento
 - Escolha da lente
 - Reveleção



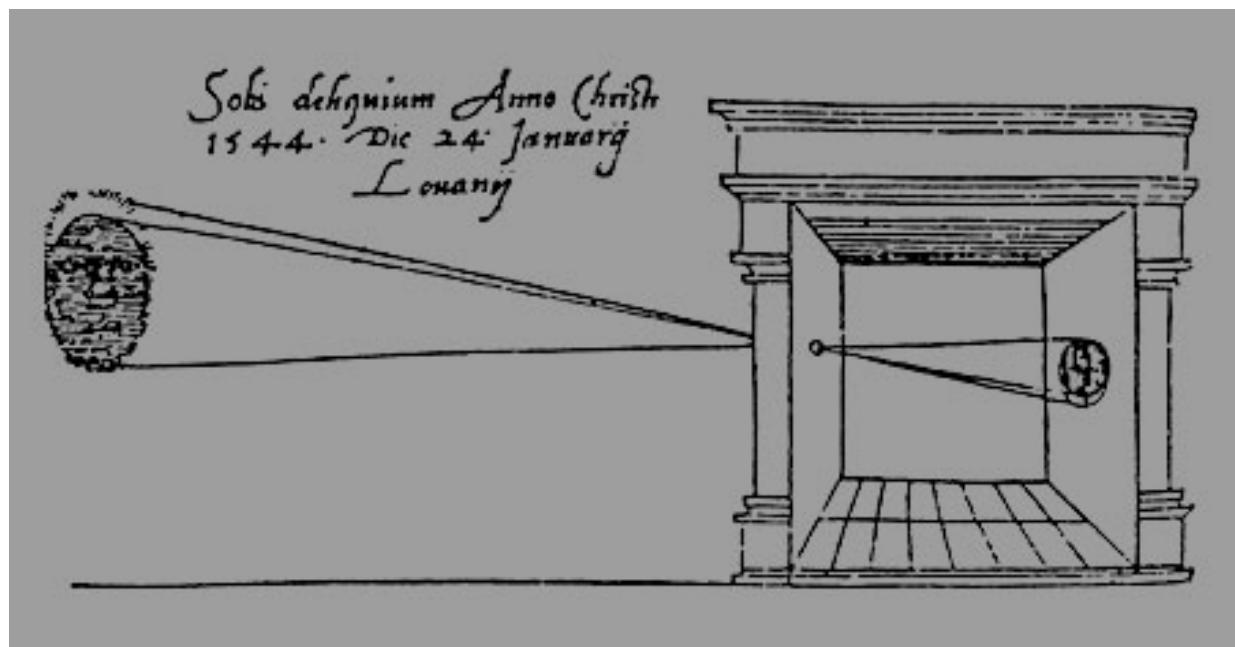
Processo de formação de imagem

- Processo virtual
 - Posicionamento da câmera
 - Posicionamento dos objetos na cena virtual
 - Projeção
 - *Viewport*



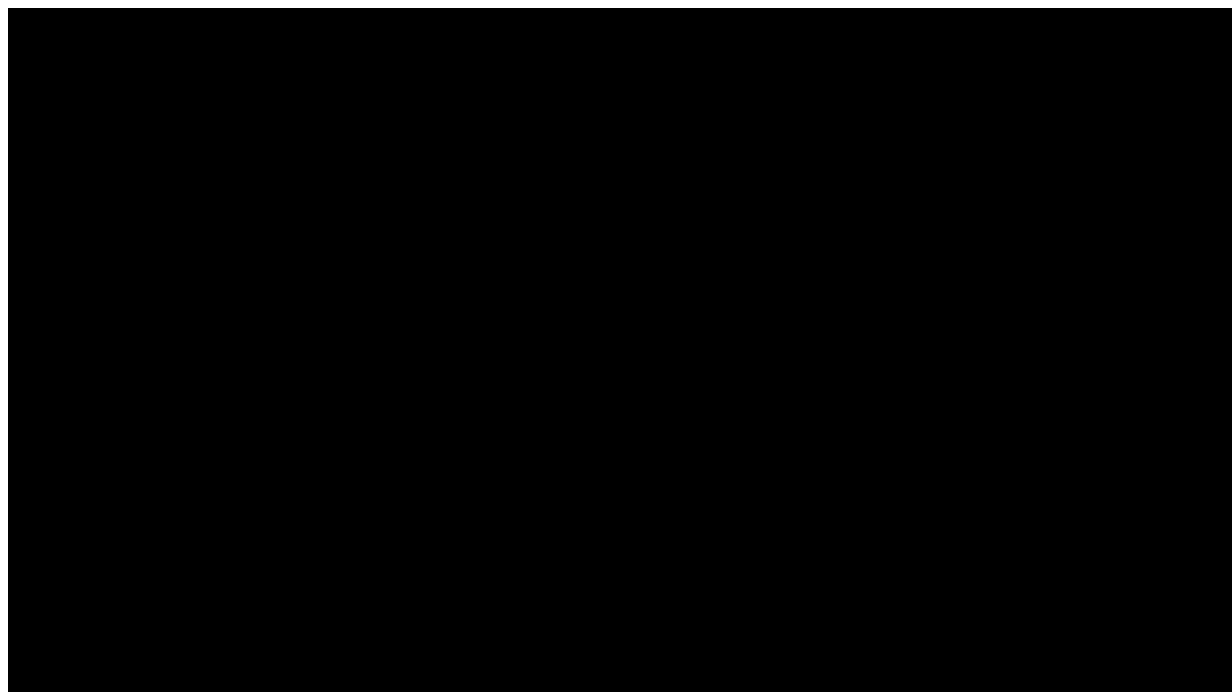
Modelo de Câmera

- *Pinhole Camera*



Modelo de Câmera

- *Pinhole*



Modelo de Câmera

- *Pinhole Camera*

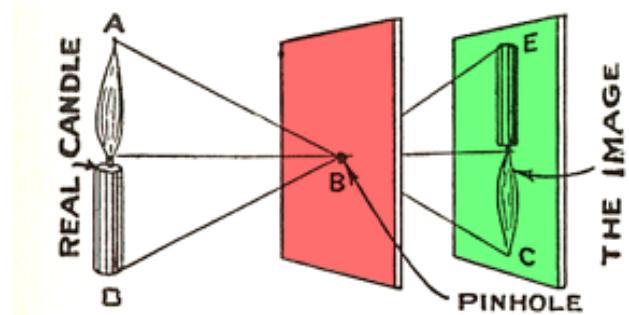
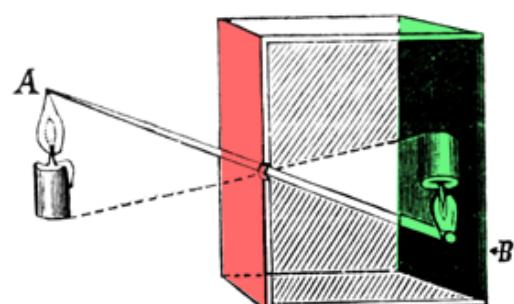


FIG. 131.—How Light and a Pinhole Form an Image.

Modelo de Câmera Pinhole

- Utilizado na prática



17

Modelo de Câmera Pinhole

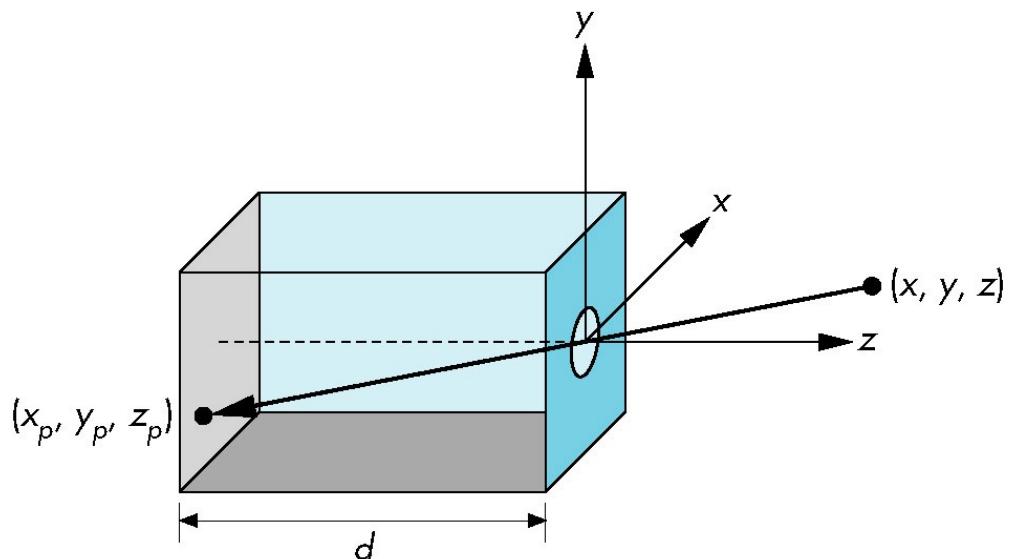
- Utilizado na prática



18

Modelo de Câmera Pinhole

- Define um sistema de coordenadas próprio (**camera space**)



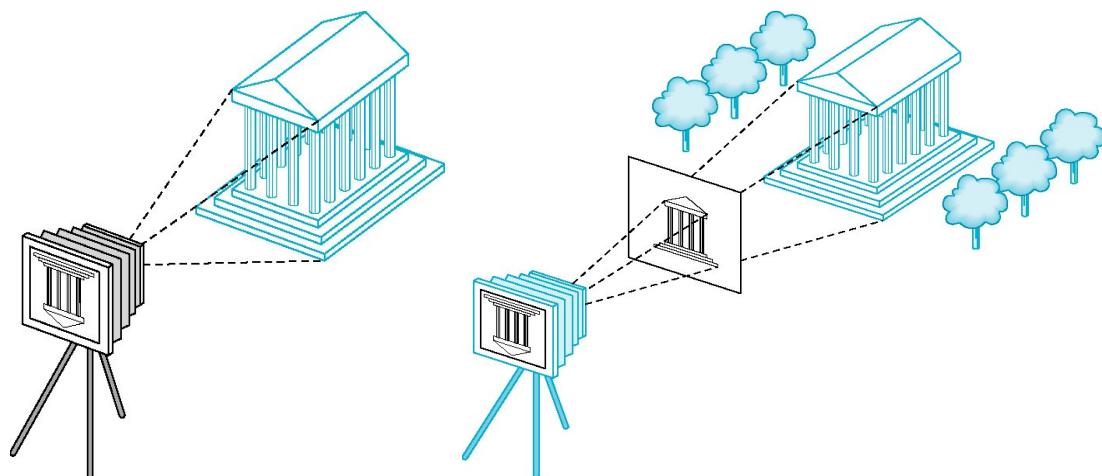
19

Modelo de Câmera Virtual

20

Modelo de Câmera Virtual

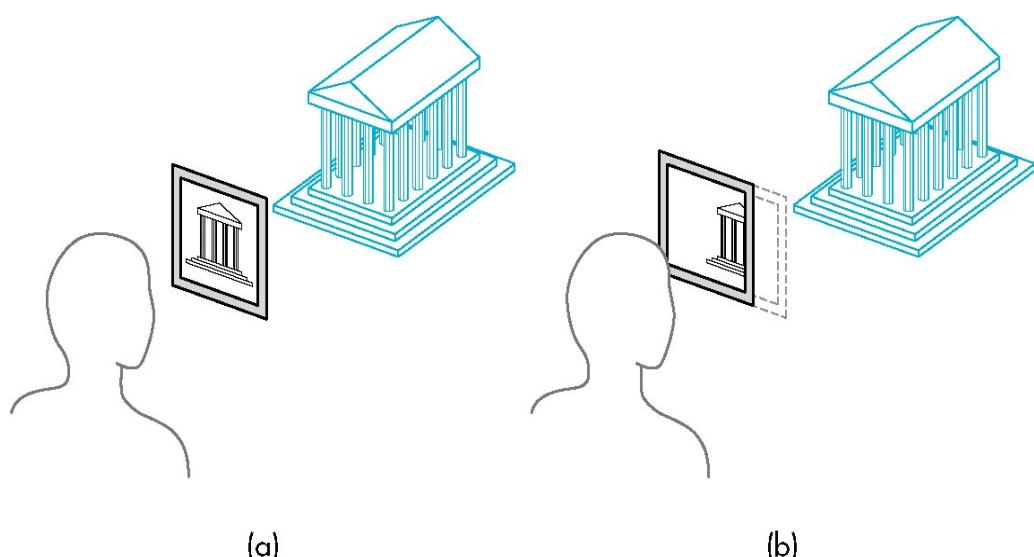
- Formação da imagem invertida:
 - Utilizar um plano simétrico ao plano de projeção da câmera
 - Geração de uma imagem direita



21

Modelo de Câmera Virtual

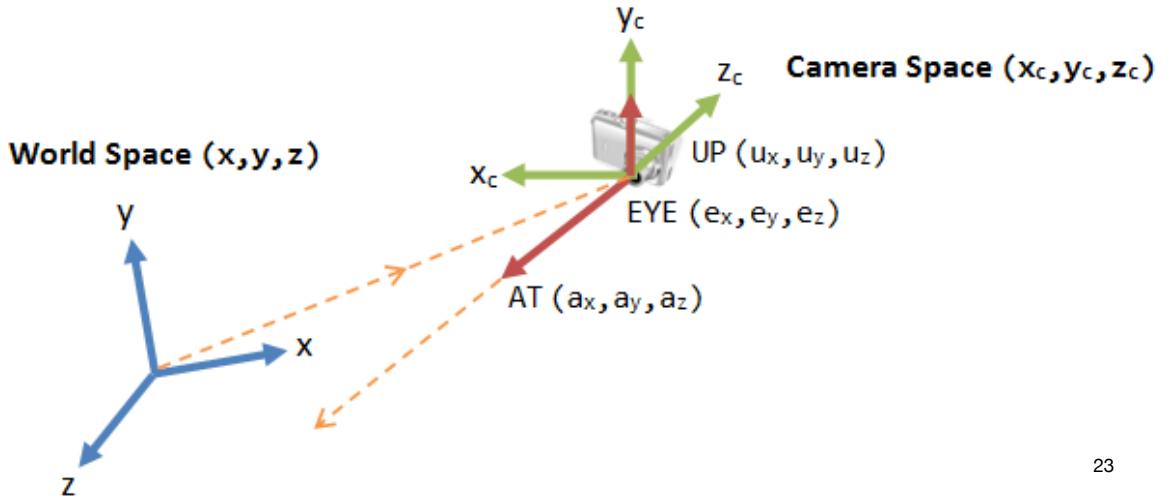
- Posicionamento da câmera interfere no enquadramento
 - Recorte (*clipping*)



22

Modelo de Câmera Virtual

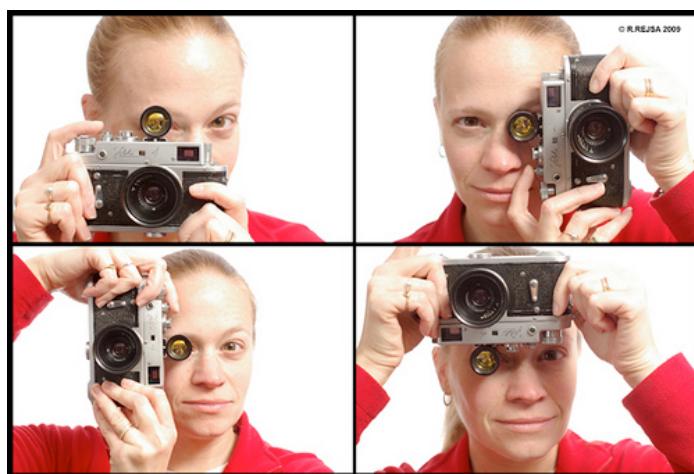
- Posicionamento no espaço do mundo (**world space**)
 - Ponto no espaço 3D (*Eye*)
 - Direção de Observação (*At* - *Eye*)
 - Orientação (*Up* - *Eye*)



23

Modelo de Câmera Virtual

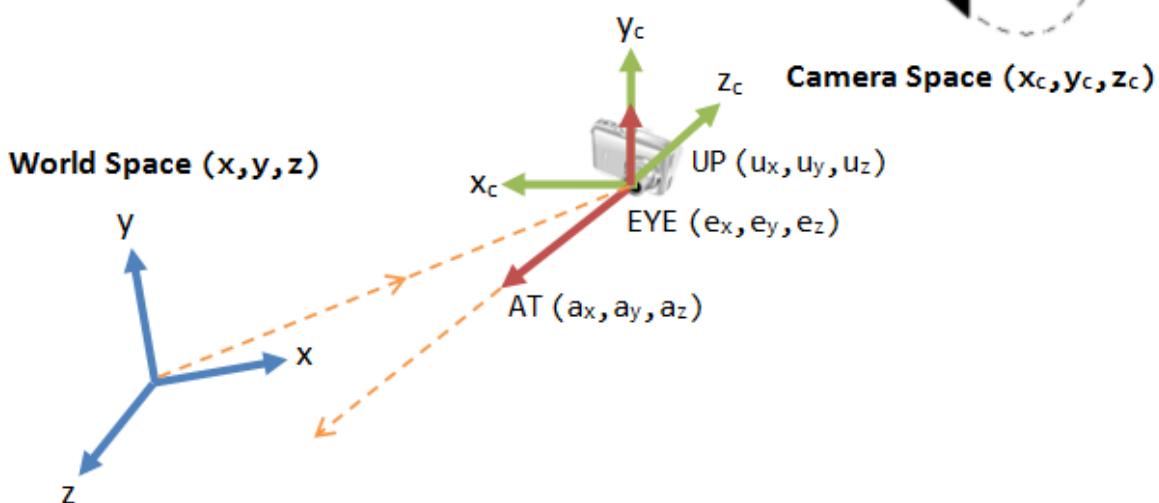
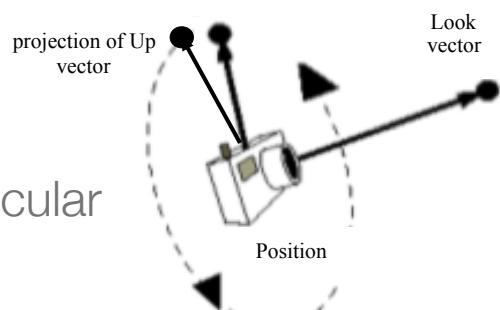
- Orientação da câmera
 - Retrato (*Portrait*)
 - Paisagem (*Landscape*)



24

Modelo de Câmera Virtual

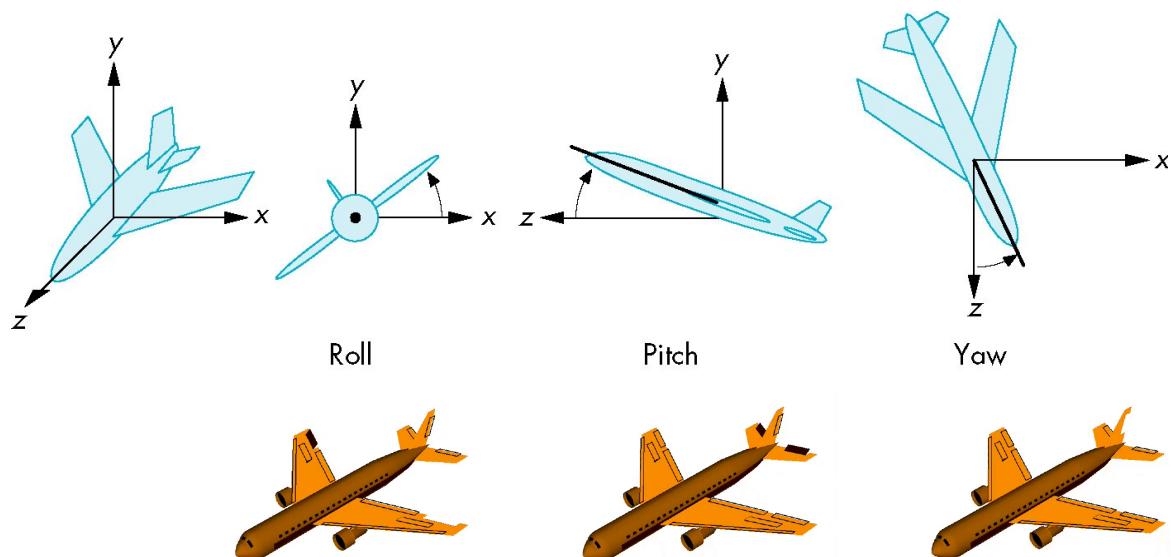
- Orientação da câmera
 - Vetor “para cima” (up)
 - Pertence ao plano perpendicular a direção de observação



25

Modelo de Câmera Virtual

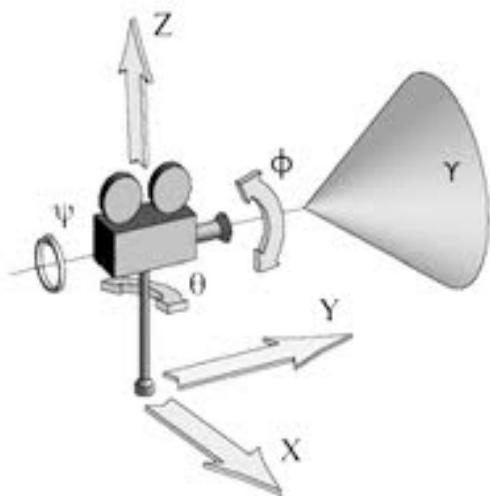
- Controles relativos ao próprio sistema de coordenadas da camera virtual



26

Modelo de Câmera Virtual

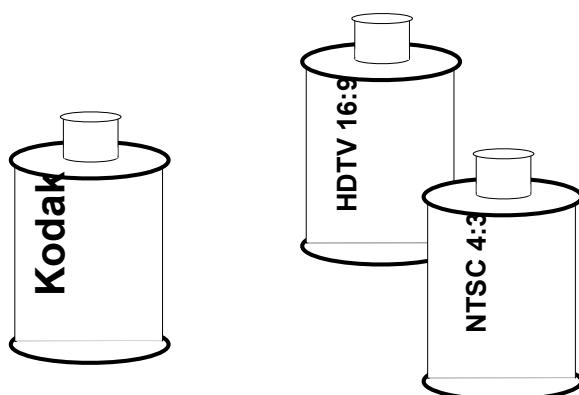
- Controles da câmera virtual
 - Resumindo:
 - Posição
 - Direção de observação
 - Orientação



27

Modelo de Câmera Virtual

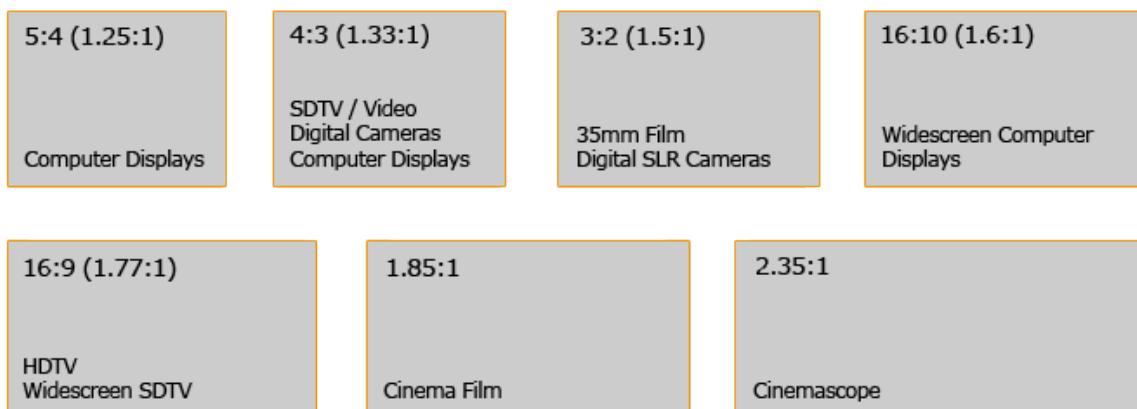
- Razão de Aspecto
 - Conceito análogo ao tamanho do filme utilizado em uma câmera real
 - Proporção entre a altura e a largura da imagem a ser gerada



28

Modelo de Câmera Virtual

- Razão de Aspecto
 - Exemplos



29

Modelo de Câmera Virtual

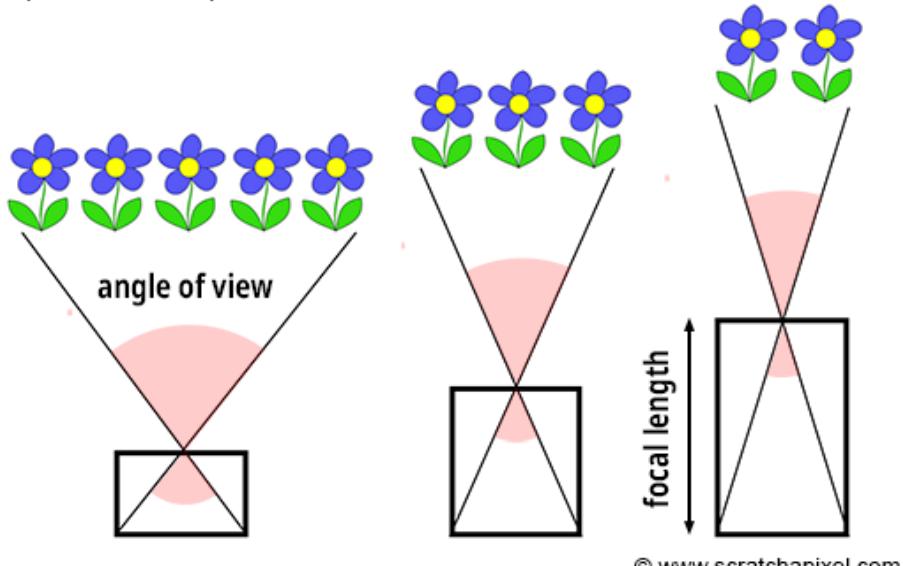
- Razão de Aspecto
 - Relação com as lentes utilizadas



30

Modelo de Câmera Virtual

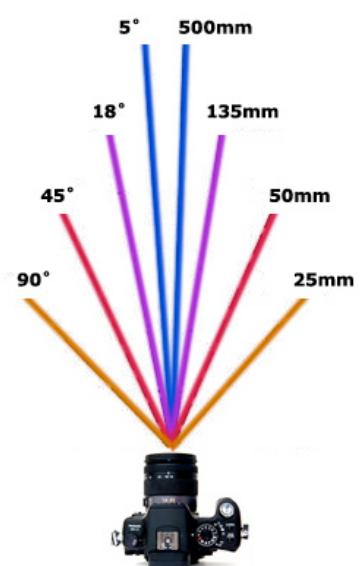
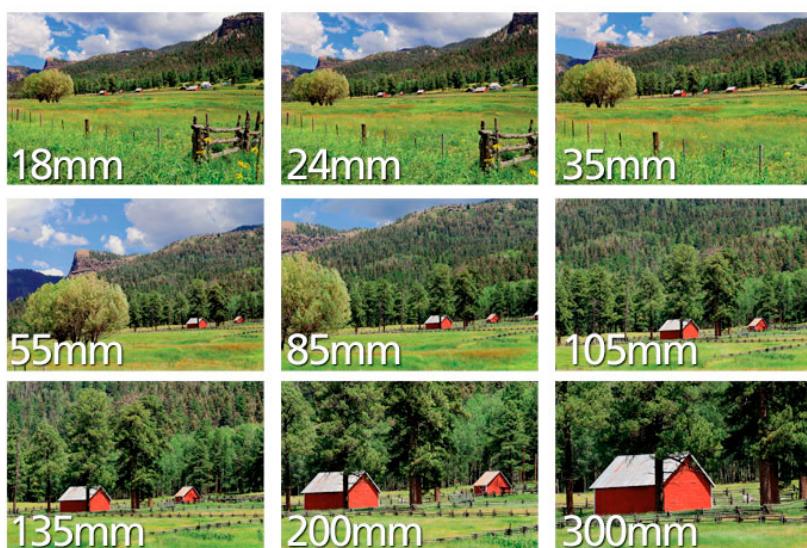
- Campo de visão
 - O que é possível ver a frente da camera
 - Na *pinhole*: profundidade da camera



31

Modelo de Câmera Virtual

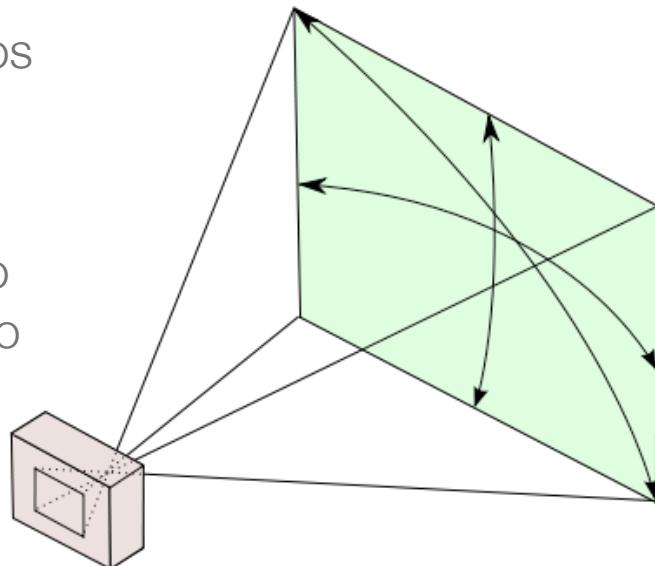
- Campo de visão
 - O que é possível ver a frente da camera
 - Com lente: distância focal da lente



32

Modelo de Câmera Virtual

- Controles da câmera virtual
 - Campo de visão
 - Dado por 2 ângulos
 - Horizontal
 - Vertical
 - Diretamente ligado a razão de aspecto



33

Modelo de Câmera Virtual

- Controles da câmera virtual
 - Campo de visão
 - acentua ou reduz deformações na imagem



34

Modelo de Câmera Virtual

- Controles da câmera virtual
 - Campo de visão
 - acentua ou reduz deformações na imagem



35

Modelo de Câmera Virtual

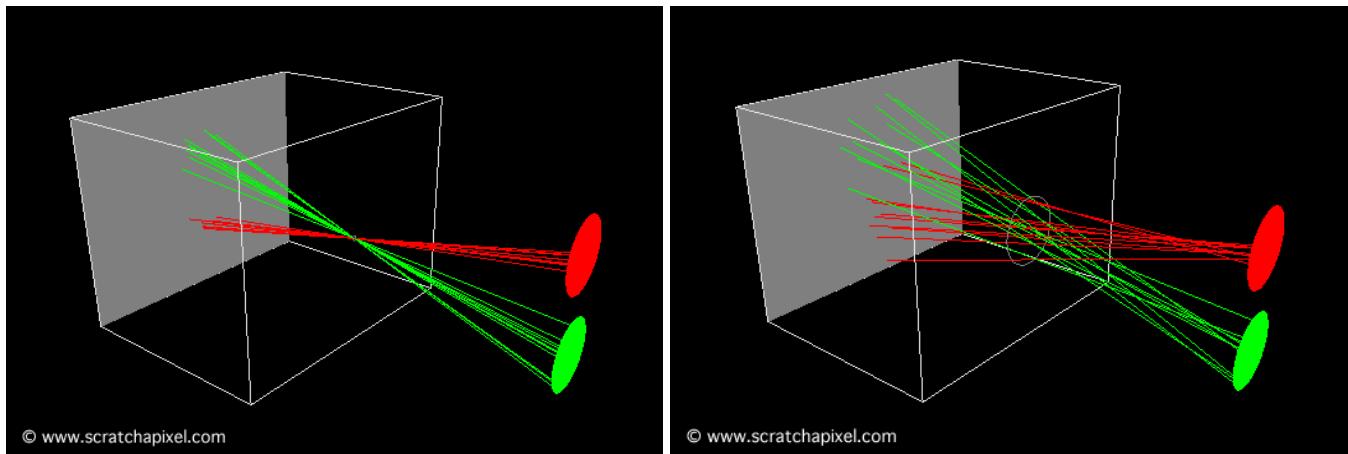
- Controles da câmera virtual
 - Campo de visão



36

Modelo de Câmera Virtual

- Profundidade de Campo
 - Capacidade da lente registrar objetos em foco
 - Na pinhole: tamanho do orifício



37

Modelo de Câmera Virtual

- Profundidade de Campo
 - Capacidade da lente registrar objetos em foco
 - Camera com lente: abertura do diafragma da lente



38

Modelo de Câmera Virtual

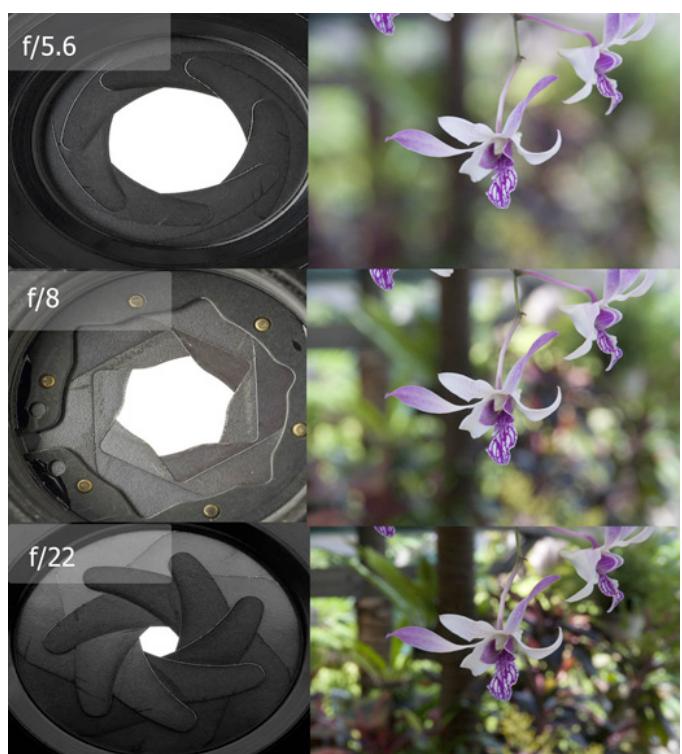
- Profundidade de Campo
 - Capacidade da lente registrar objetos em foco



39

Modelo de Câmera Virtual

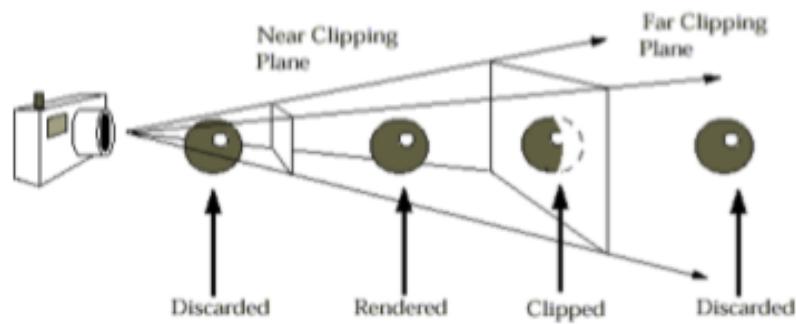
- Profundidade de Campo
 - Relacionada com a abertura do diafragma da lente



40

Modelo de Câmera Virtual

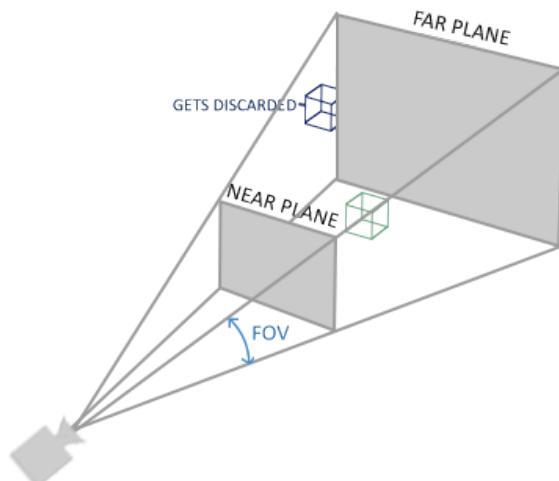
- Controles da câmera virtual
 - Profundidade de Campo => Volume de visão
 - Define a faixa de objetos que estarão em foco na imagem
 - Simula a profundidade de campo, sem o desfoque.



41

Modelo de Câmera Virtual

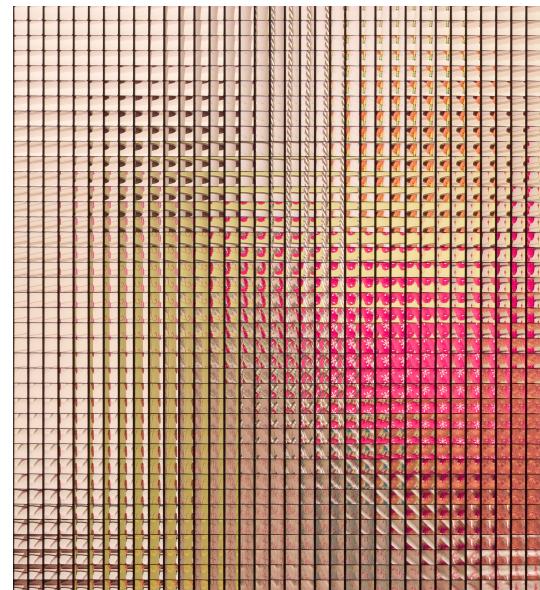
- Volume de visão
 - Definido a partir de 2 planos
 - paralelos entre si
 - distancias z_{near} e z_{far} a partir do espaço da camera



42

Modelo de Câmera Virtual

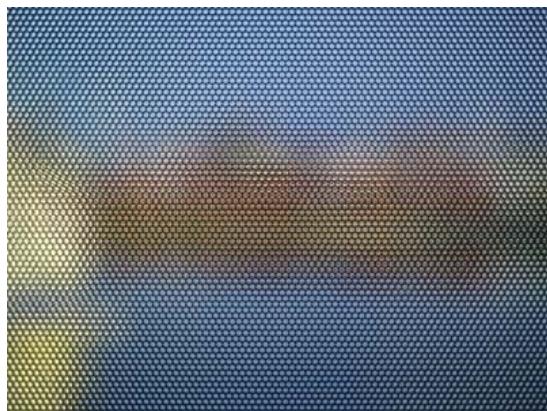
- Profundidade de Campo
 - Em breve será "coisa do passado"?
 - Cameras Plenópticas



43

Modelo de Câmera Virtual

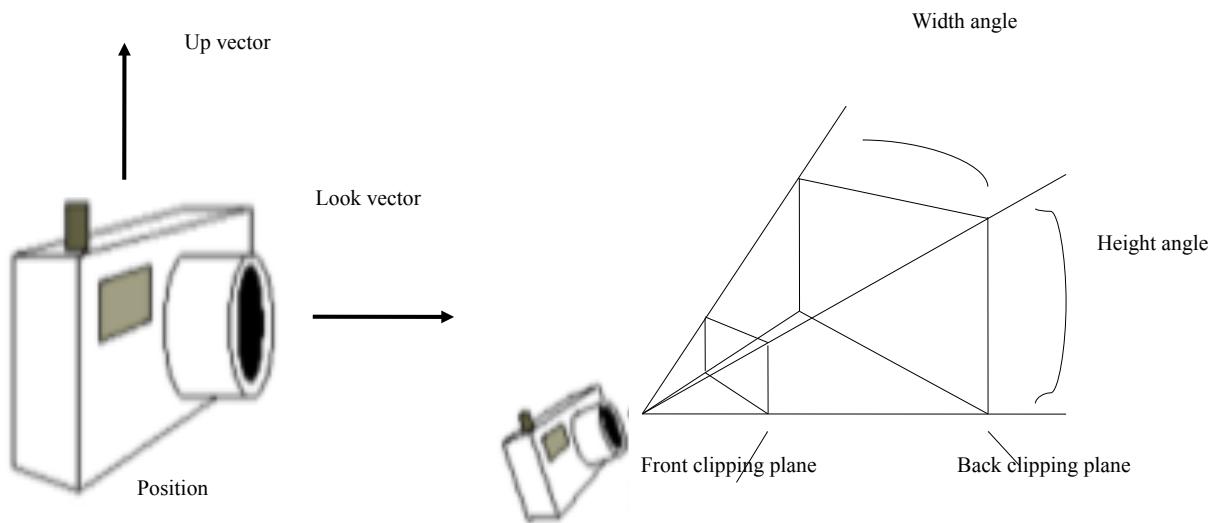
- Profundidade de Campo
 - Em breve será "coisa do passado"?
 - Cameras Plenópticas



44

Modelo de Câmera Virtual

- Resumindo...



45

Aplicações com *Three.JS/WebGL*

46

Controle de camera virtual na Three.JS

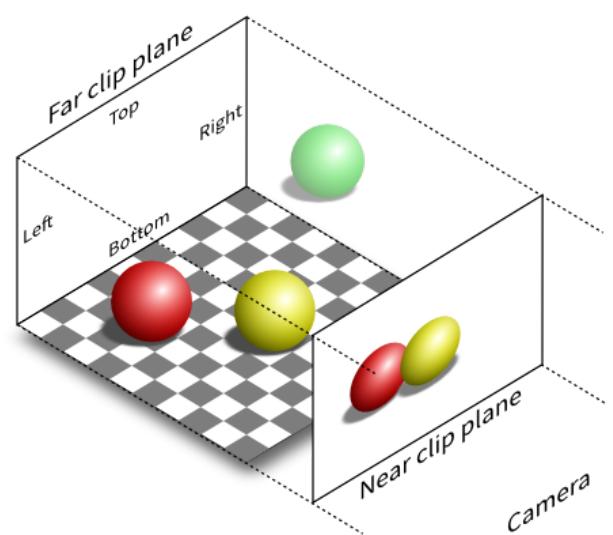
- Baseado no objeto **Camera**
 - Encapsula propriedades básicas da camera
 - **matrixWorldInverse**
 - **projectionMatrix**
 - **lookAt(vector)**

47

Tipos de camera virtual na Three.JS

• **OrthographicCamera**

- Propriedades
 - **left**
 - **right**
 - **top**
 - **bottom**
 - **near**
 - **far**



Orthographic projection (O)

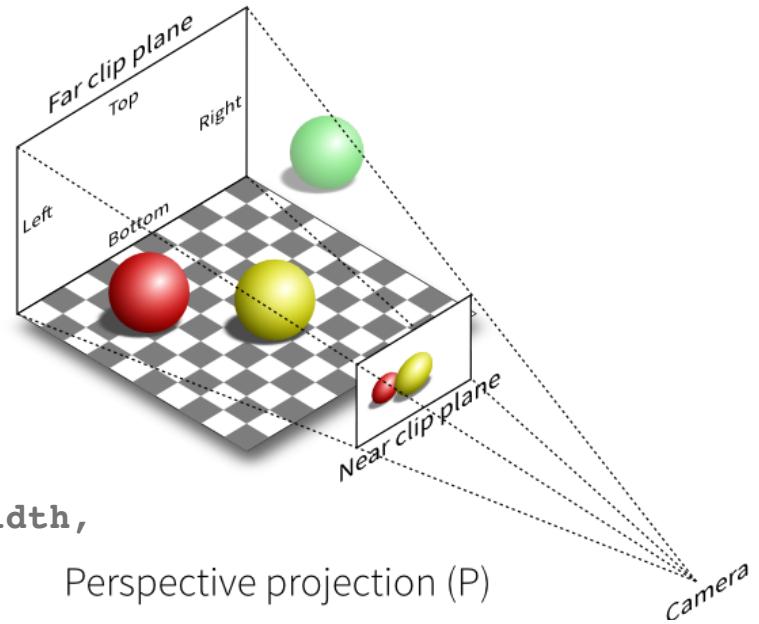
- Metodos
 - **updateProjectionMatrix()**

48

Tipos de camera virtual na Three.JS

• PerspectiveCamera

- Propriedades
 - **fov**
 - **aspect**
 - **near**
 - **far**
- Métodos
 - **setLens(focalLength, frameSize)**
 - **setViewOffset(fullWidth, fullHeight, x, y, width, height)**
 - **updateProjectionMatrix()**



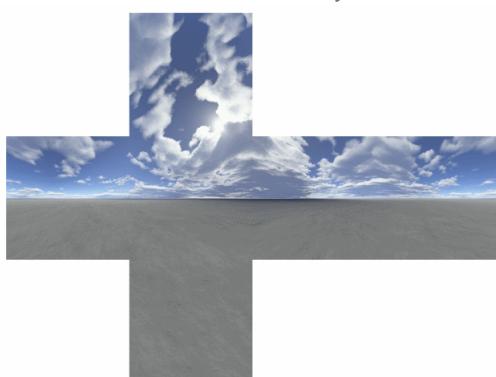
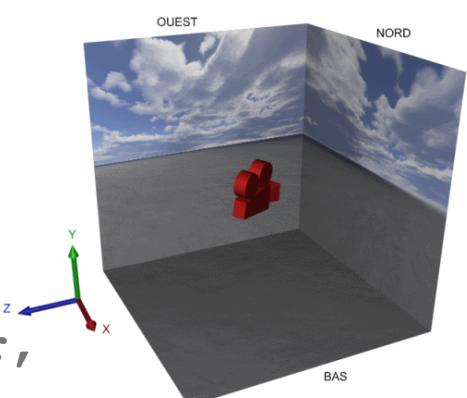
Perspective projection (P)

49

Tipos de camera virtual na Three.JS

• CubeCamera

- Propriedades
 - **renderTarget**
- Metodos
 - **updateCubeMap(renderer, scene)**



Exemplos de código em Three.JS

```
function init() {  
    scene = new THREE.Scene();  
    renderer = new THREE.WebGLRenderer();  
    renderer.setClearColor(new THREE.Color(0.0, 0.0, 0.0));  
    renderer.setSize(window.innerWidth*0.7,  
                    window.innerHeight*0.7);  
    document.getElementById("WebGL-output").appendChild(renderer.domElement);  
    camera = new THREE.PerspectiveCamera( 60.0, 1.0, 0.1, 10.0 );  
    scene.add( camera );  
    buildScene();  
    renderer.clear();  
    render();  
};
```

51

Exemplos de código em Three.JS

```
(...)  
camera = new THREE.PerspectiveCamera();  
camera.fov      = 60.0;  
camera.aspect   = 1.0;  
camera.near     = 1.7;  
camera.far      = 3.0;  
camera.position.x = 1.0;  
camera.position.y = 1.0;  
camera.position.z = 1.5;  
camera.lookAt(new THREE.Vector3(0, 0, 0));  
  
scene.add( camera );  
(...)
```

52

Controles de camera

- Objetos para controle de cameras podem ser encontrados nos exemplos que acompanham a biblioteca *Three.JS*
 - **TrackballControls**
 - **FirstPersonControls**
 - **FlyControls**
 - **OrbitControls**
- Cada controle possui um conjunto particular de parâmetros.
 - Ajustam automaticamente os parametros básicos da camera.

53

Controles de camera

```
• TrackballControls
  // Controle de Camera Trackball
  camera = new THREE.PerspectiveCamera( 45.0, 1.0, 0.1, 1000.0 );
  camera.lookAt(new THREE.Vector3(0, 0, 0));

  trackballControls = new THREE.TrackballControls(camera);
  trackballControls.rotateSpeed = 1.0;
  trackballControls.zoomSpeed = 1.0;
  trackballControls.panSpeed = 1.0;
  trackballControls.staticMoving = true;
  trackballControls.dynamicDampingFactor=0.3;

  (...)

  function render() {
    var delta = clock.getDelta();
    trackballControls.update(delta);
    renderer.render(scene, camera);
    requestAnimationFrame(render);
  }
```

54

Controles de camera

- **FirstPersonControls**
(...)

```
// Controle de Camera de Primeira Pessoa
camera = new THREE.PerspectiveCamera( 45.0, 1.0, 0.1, 1000.0 );
camera.lookAt(new THREE.Vector3(0, 0, 0));

fpControl = new THREE.FirstPersonControls(camera);
fpControl.lookSpeed = 0.4;
fpControl.movementSpeed = 20;
fpControl.noFly = true;
fpControl.lookVertical = true;
fpControl.constrainVertical = true;
fpControl.verticalMin = 1.0;
fpControl.verticalMax = 2.0;
fpControl.lon = -150;
fpControl.lat = 120;
```

55

Controles de camera

- **FlyControls**

- **FlyControls**
(...)

```
// Controle de Camera de Primeira Pessoa
camera = new THREE.PerspectiveCamera( 45.0, 1.0, 0.1, 1000.0 );

flyControls = new THREE.FlyControls(camera);
flyControls.movementSpeed = 25;
flyControls.domElement = document.querySelector("#WebGL-output");
flyControls.rollSpeed = Math.PI / 24;
flyControls.autoForward = true;
flyControls.dragToLook = false;
```

56

Controles de camera

• OrbitControls

(...)

```
// Controle de Camera Orbital  
camera = new THREE.PerspectiveCamera(45.0, 1.0, 0.1, 1000.0);  
camera.lookAt(new THREE.Vector3(0, 0, 0));  
  
orbitControls = new THREE.OrbitControls(camera);  
orbitControls.autoRotate = true;
```

57

Controles de camera

• RollControls

(...)

```
// Controle de Camera de Voo com Roll  
camera = new THREE.PerspectiveCamera( 45.0, 1.0, 0.1, 1000.0 );  
camera.lookAt(new THREE.Vector3(0, 0, 0));  
  
rollControls = new THREE.RollControls(camera);  
rollControls.movementSpeed = 25;  
rollControls.lookSpeed = 3;
```

58

A seguir... Transformações Projetivas