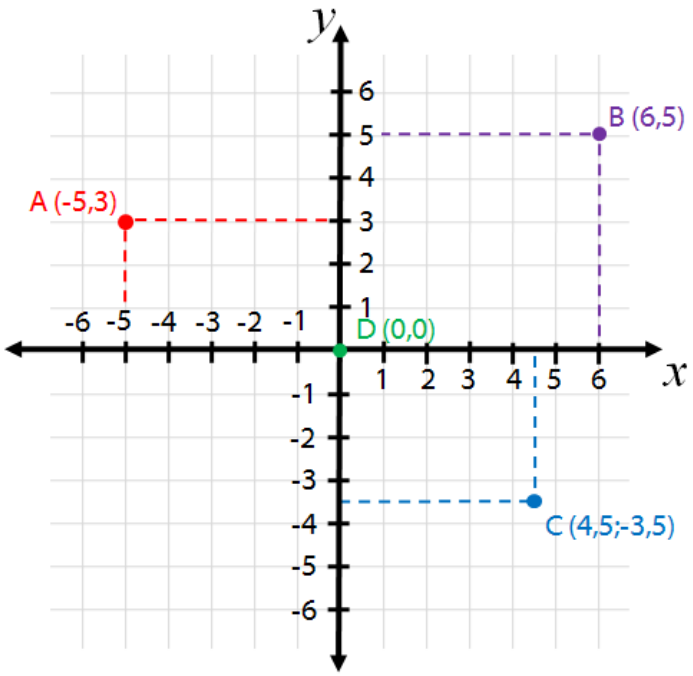
Visualização Bidimensional

Rossana Baptista Queiroz

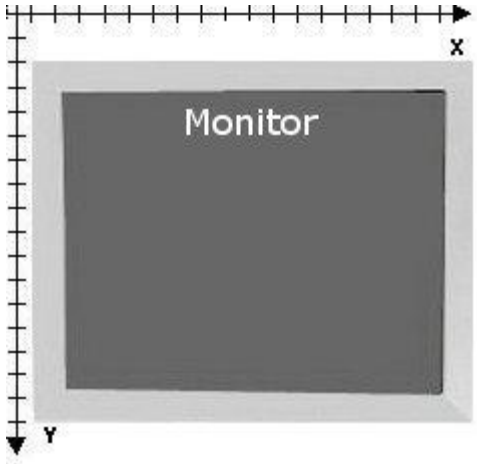
Sistema de Referência do Universo (SRU)

- Região do espaço utilizada na aplicação
- Normalmente é adotado o sistema cartesiano
  - Todos os modelos e comandos OpenGL são definidos em relação a este sistema de referência



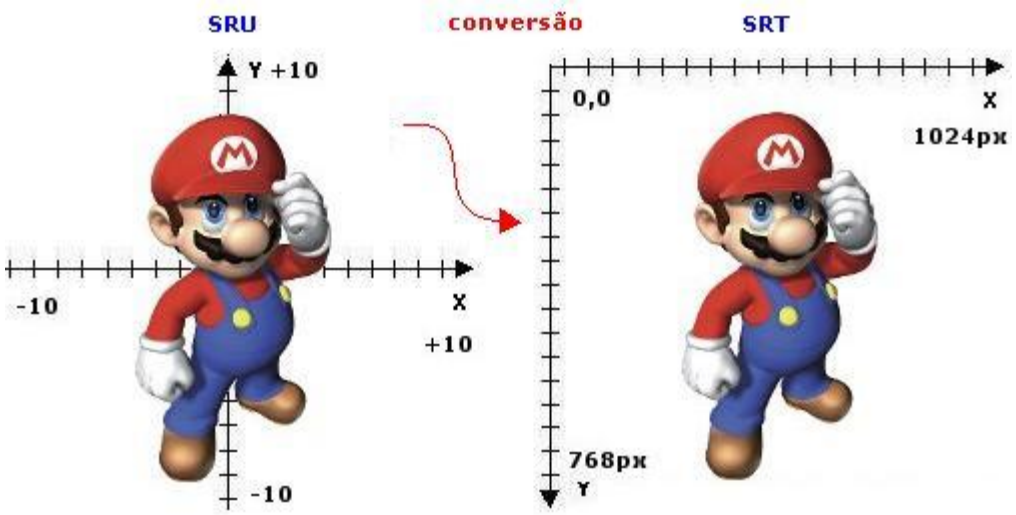
Sistema de Coordenadas de Tela (SRT)

- No monitor do computador é adotado o SRT
- No SRT, a origem fica no canto superior esquerdo do monitor



SRU vs. SRT

- Para visualizar corretamente modelos definidos no SRU, é necessário fazer uma conversão ou mapeamento

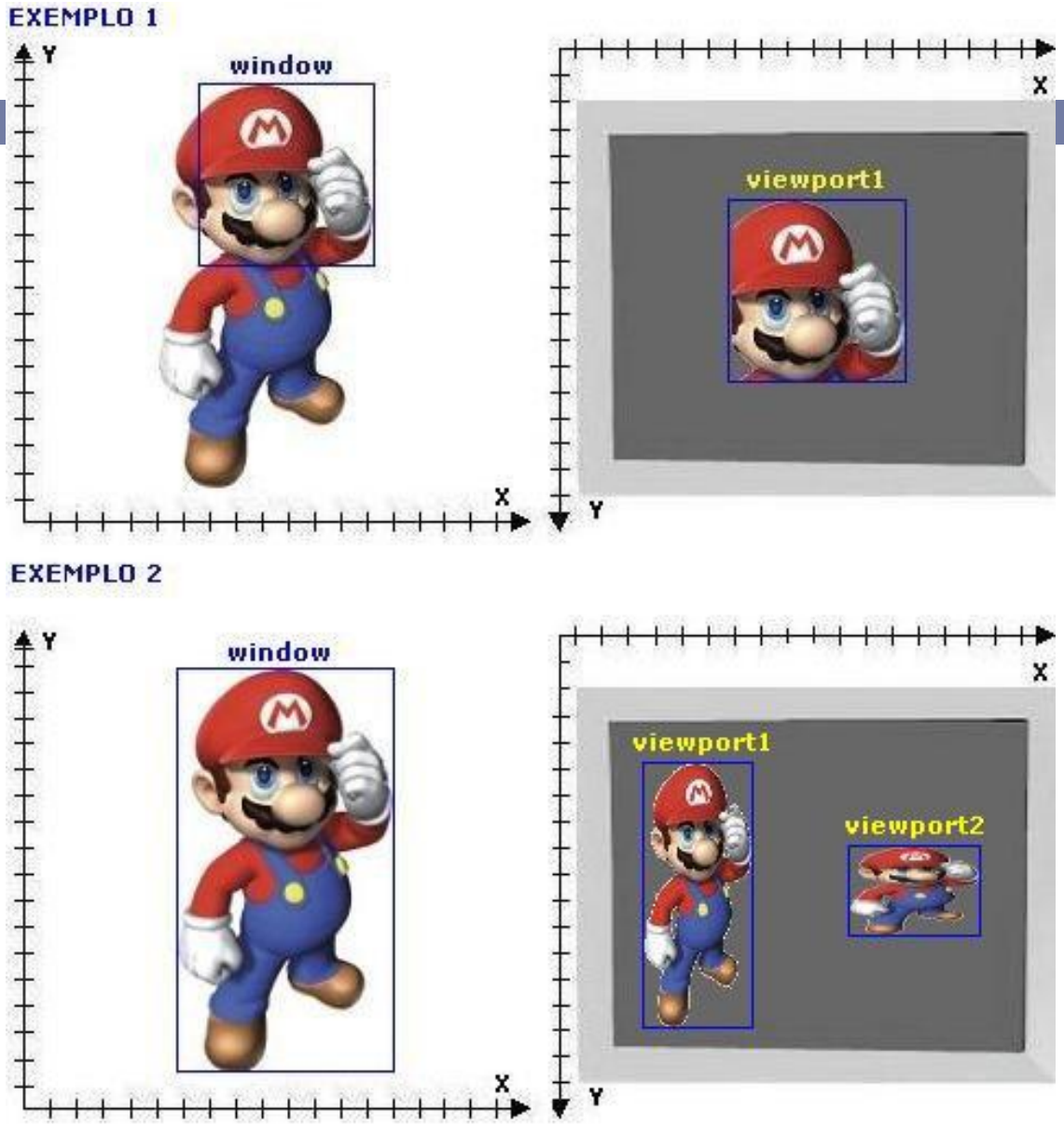


http://www.bdjogos.com/conteudo.php?link=capitulo_45.php

SRU vs. SRT

- Como o universo é infinito é preciso especificar qual porção queremos mapear na tela
- A essa área que delimita a área de interesse do usuário chamamos de ***window*** ou janela de seleção.
- Uma ***window*** é delimitada através das coordenadas de seus cantos (esquerdo, direito, superior, inferior) no **sistema SRU**
- De forma análoga também é necessário definir **em que parte desejamos exibir o conteúdo na *window* no monitor**
- Essa região é chamada de ***viewport*** ou janela de exibição, uma ***viewport***. Em OpenGL, uma *viewport* normalmente é delimitada pelo tamanho da janela GLUT correspondente ao tamanho da resolução do monitor.

SRU vs. SRT


EXEMPLO 1
Y
window
X
X
viewport1
Y
EXEMPLO 2
Y
window
X
X
viewport1
viewport2
Y

SRU vs. SRT

- Para realizar o mapeamento do SRU para o SRT em OpenGL, é necessário especificar a ***viewport*** e a ***window***.

```
gluOrtho2D (Gldouble left, Gldouble right,
Gldouble bottom, Gldouble top);
```

Essa função serve para definir a ***window*** quando se está trabalhando com desenhos 2D. Seus parâmetros correspondem especificamente cada borda da window. Isto é: x mínimo (borda esquerda - left) – x máximo (borda direita - right) – y mínimo (borda inferior - bottom) – y máximo (borda superior - top).

SRU vs. SRT

- Exemplo: casa.cpp

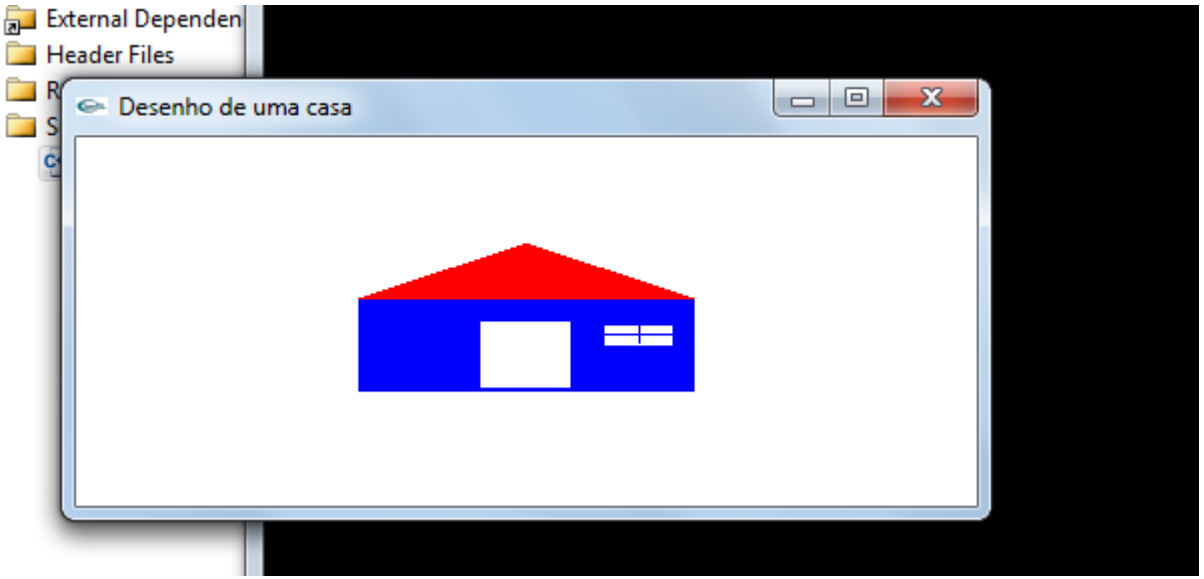
```
// Estabelece a janela de seleção (esquerda, direita, inferior,
// superior) mantendo a proporção com a janela de visualização
if (largura <= altura)
        gluOrtho2D (-40.0f, 40.0f, -40.0f*altura/largura,
40.0f*altura/largura);
else
        gluOrtho2D (-40.0f*largura/altura, 40.0f*largura/altura,
-40.0f, 40.0f);
```

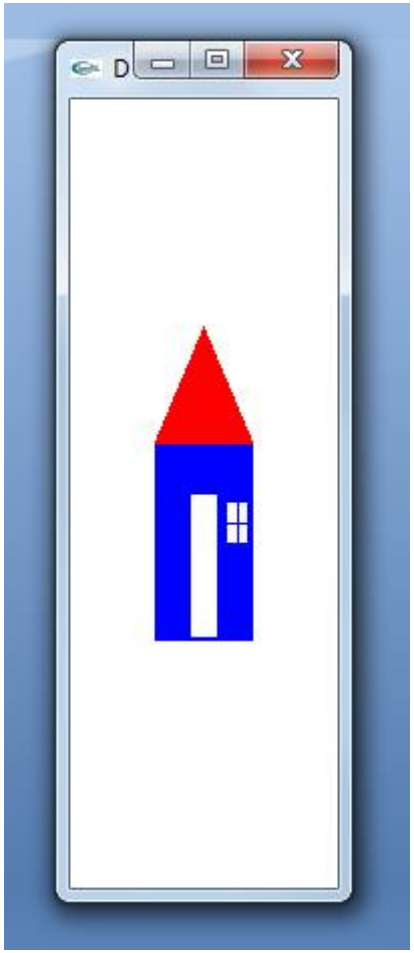
Altura e largura é o tamanho da janela da GLUT em pixels
A relação largura/altura (chamada de aspecto) garante que a *window* será definida de maneira que a imagem final não se deforme

SRU vs. SRT

- Exemplo: casa.cpp

Se não fizéssemos a correção do aspecto...





SRU vs. SRT


```
void glViewport( GLint x, GLint y, GLsizei
width, GLsizei height);
```

Serve para definir a ***viewport***. Os valores especificados nos dois primeiros parâmetros x e y definem a posição da janela na tela (canto inferior esquerdo) enquanto os dois últimos definem a largura e a altura dessa tela.

Se você não especificar viewports no seu programa, o OpenGL assume o viewport default e cria uma para trabalhar com toda a tela.

SRU vs. SRT

- Exemplo: casa.cpp

```
// Especifica as dimensões da Viewport
        glViewport(0, 0, largura, altura);
```

SRU vs. SRT

- Exemplo: DuasViewports.cpp

Em AlteraTamanhoJanela(GLsizei w, GLsizei h):

```
// Atualiza as variáveis
largura = w/2;
altura = h;
```

Em Desenha()

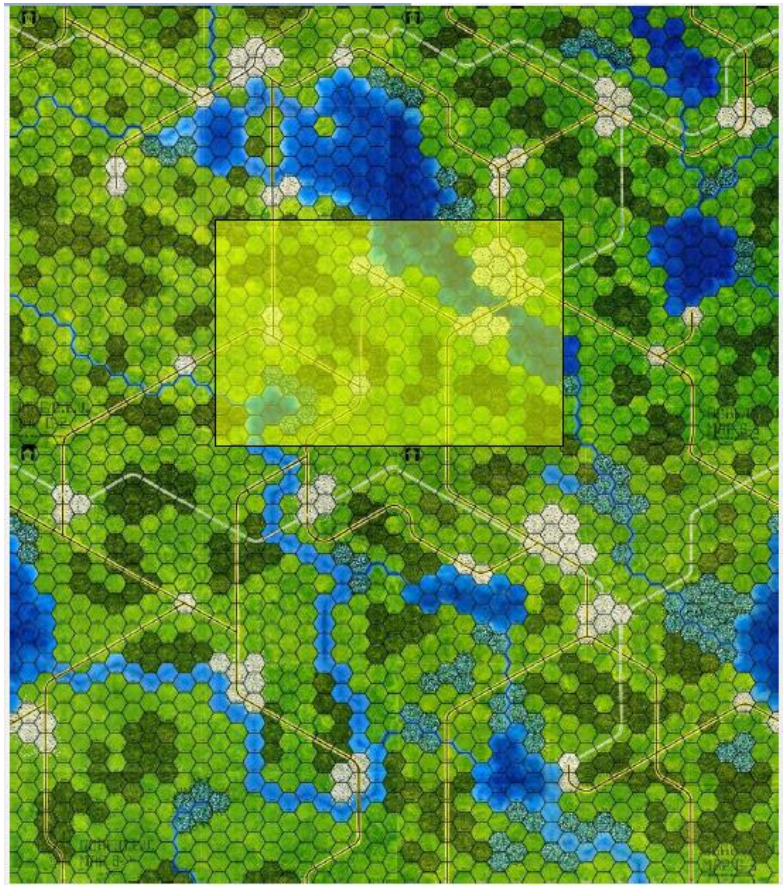
```
// Define a Viewport 1 na metade esquerda da janela
glViewport(0, 0, largura, altura);
// Desenha a casa na Viewport 1
DesenhaCasa();

// Define a Viewport 2 na metade direita da janela
glViewport(largura, 0, largura, altura);
// Desenha a casa na Viewport 2
DesenhaCasa();
```

Visão 2D

- *Scrolling*
- Tipos de visão 2D
  - *Top view*
  - *Side view*
  - *Visão isométrica*
- Efeito de *Parallax*

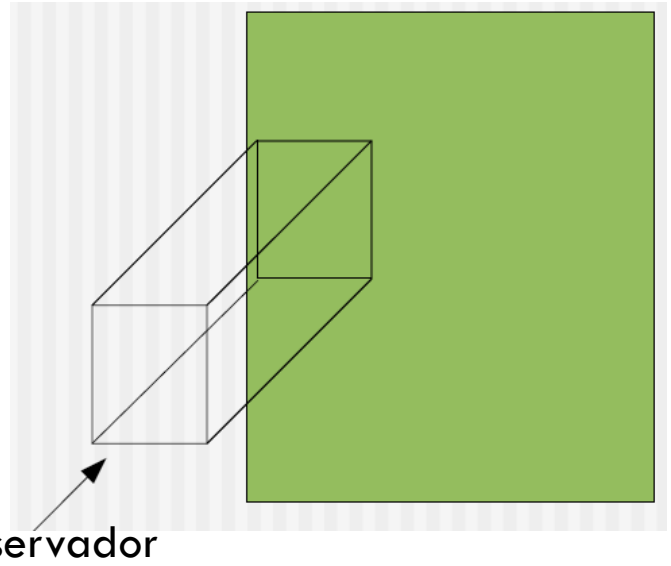
Scrolling

- Quando as dimensões do mundo são maiores que a da câmera
- A câmera (*window – ortho2D*) se movimenta com o personagem
- Isso acontece quando a *viewport* não tem tamanho adequado para mostrar todo o cenário



Scrolling

- Conceito de uma **câmera sintética**
- Em 2D, trata-se de uma **câmera ortográfica**, ou seja, sem perspectiva

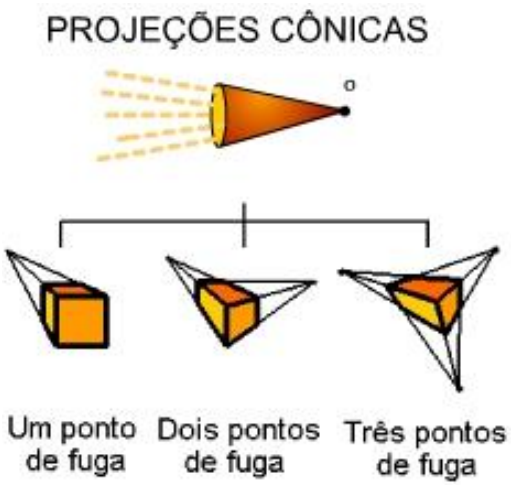


- Scrolling em OpenGL??
  - Em que comando devemos mexer??

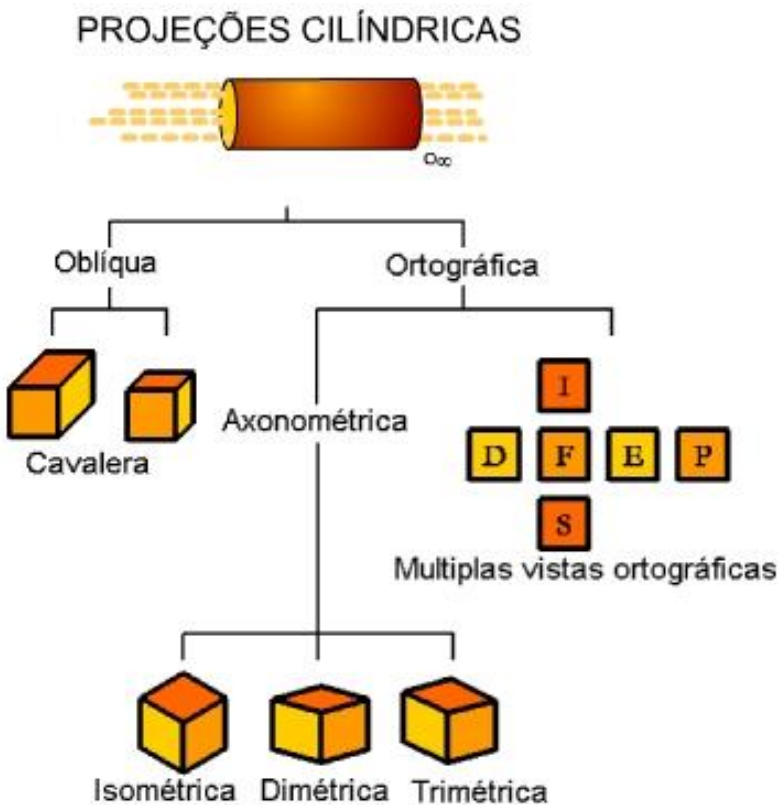
Câmera ortográfica?!?!

- Tipos de Projeção

Perspectivas



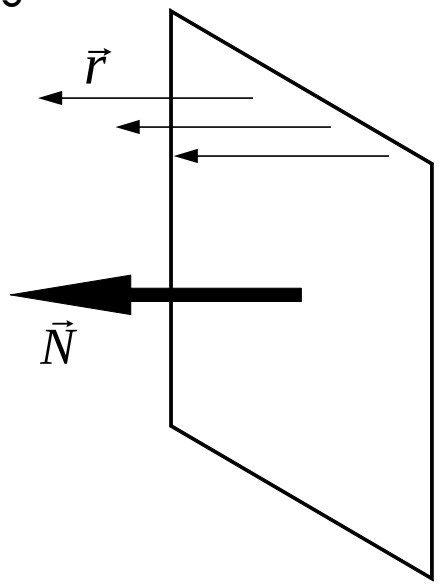
Paralelas



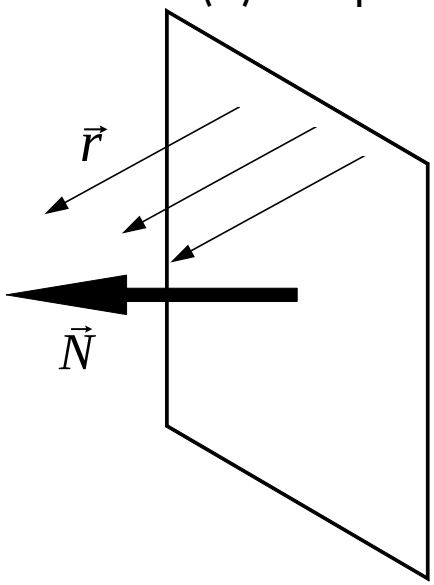
Projeções planares paralelas

- Projeções planares paralelas são subclassificadas em ortográficas e oblíquas dependendo da relação entre a direção dos **raios projetores** e a **normal** ao plano de projeção.
  - projeções ortográficas, as direções são as mesmas (raios perpendiculares ao plano de projeção).
  - projeções oblíquas, são diferentes.

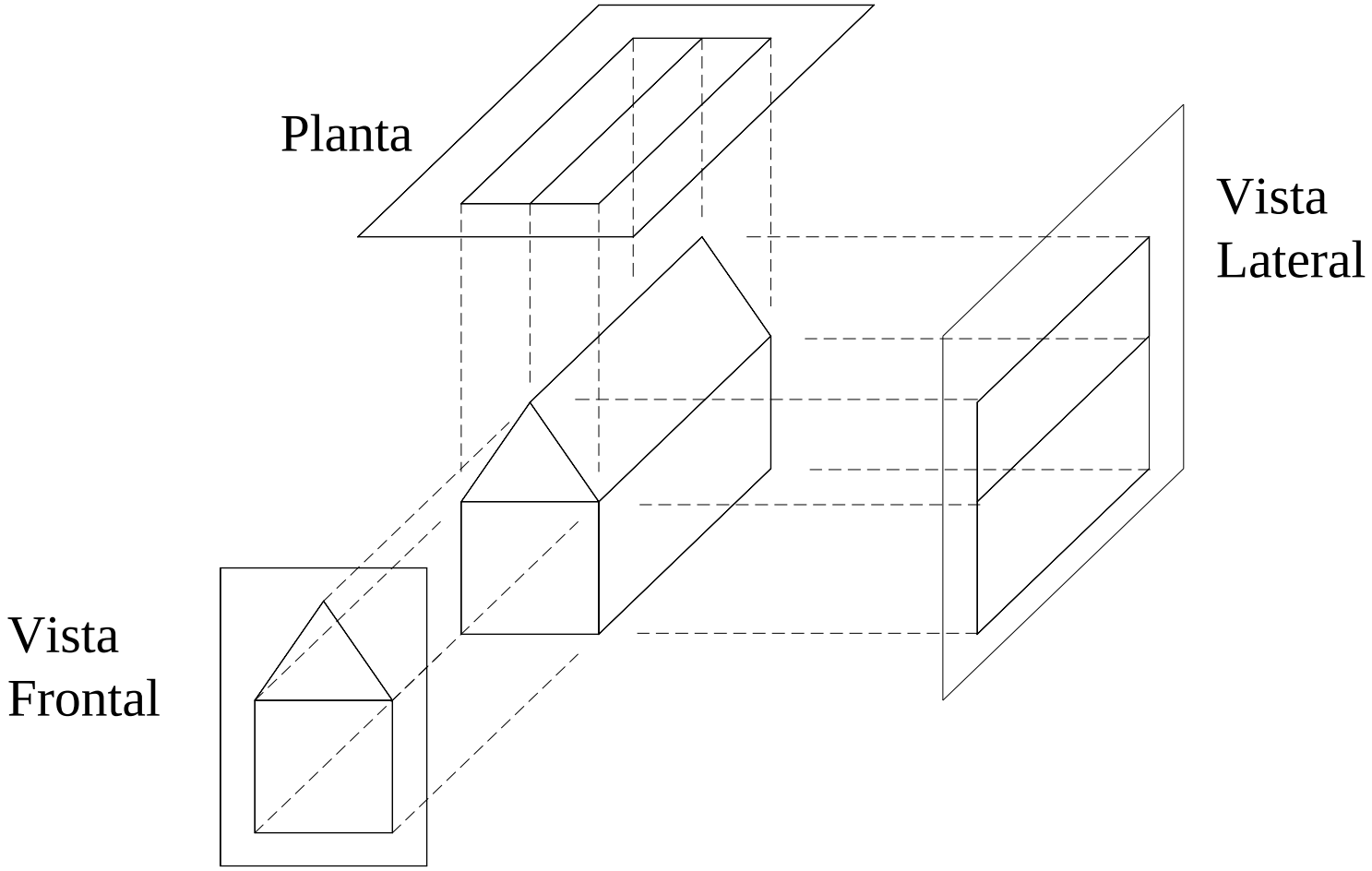
(a) Ortográfica



(b) oblíqua



Projeções ortográficas: vistas lateral, frontal e planta (de topo)



Voltando ao *Scrolling*...

- Quando definimos uma window, estamos definindo uma câmera sintética (ou observador) com tipo de projeção ortográfica.

- Movendo essa window, “navegamos” pelo mundo

Scrolling

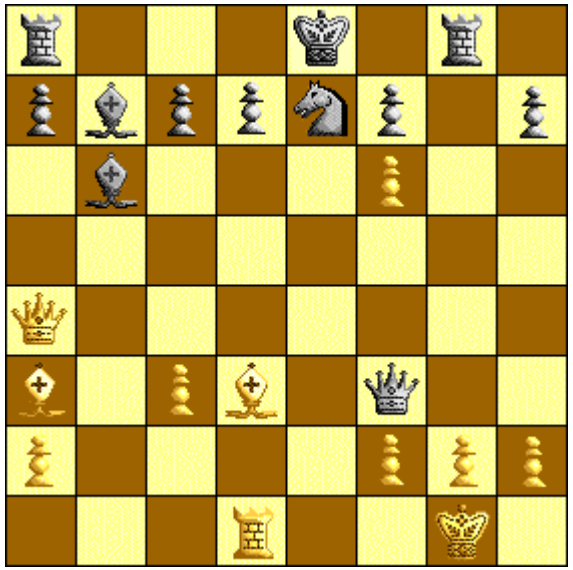
- Deve-se definir uma **taxa de *scrolling,*** que significa o quanto que a câmera se desloca (velocidade) em relação ao personagem
  - Taxa vertical
  - Taxa horizontal
- Exemplo
  - 0.5 -> cada pixel movimentado pelo personagem desloca a câmera pela metade

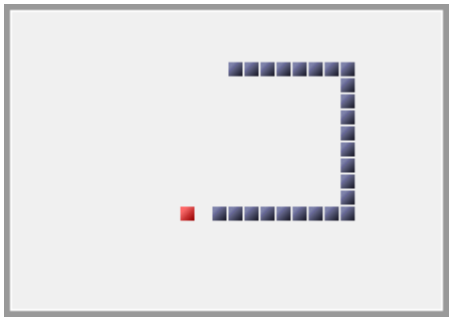
Scrolling

- Exemplo

Top View

- Ex: Jogos de naves, *board games, card games, casual games...*

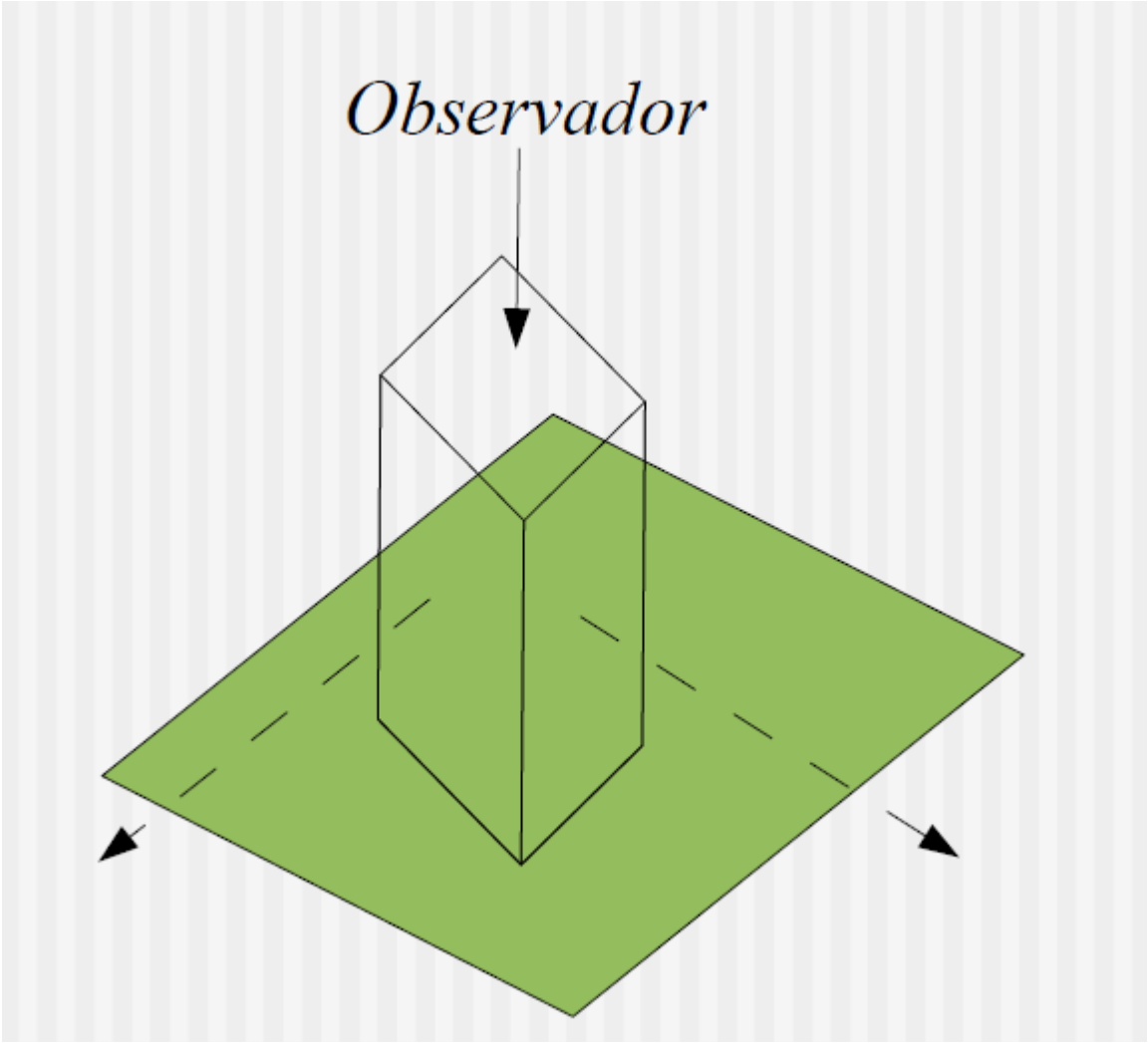




Top View

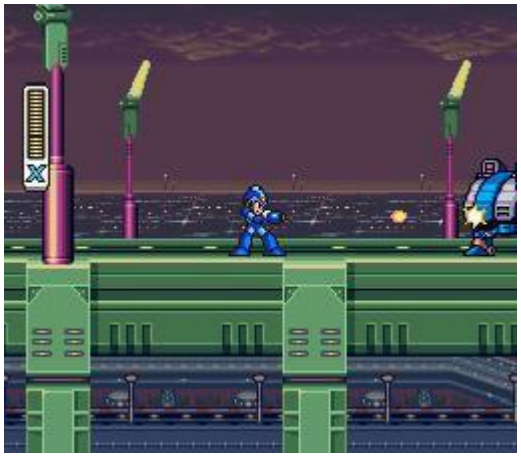
- Câmera posicionada sob a cena, sem nenhuma inclinação
- Pode ocorrer *scrolling da tela*
  - Jogos de Nave: possuem *scrolling*
  - *Casual games: sem scrolling*
    - *Viewport = tamanho do mundo*
- Uso de *tilemaps* quadriculares (caso trivial)

Top View



Side View

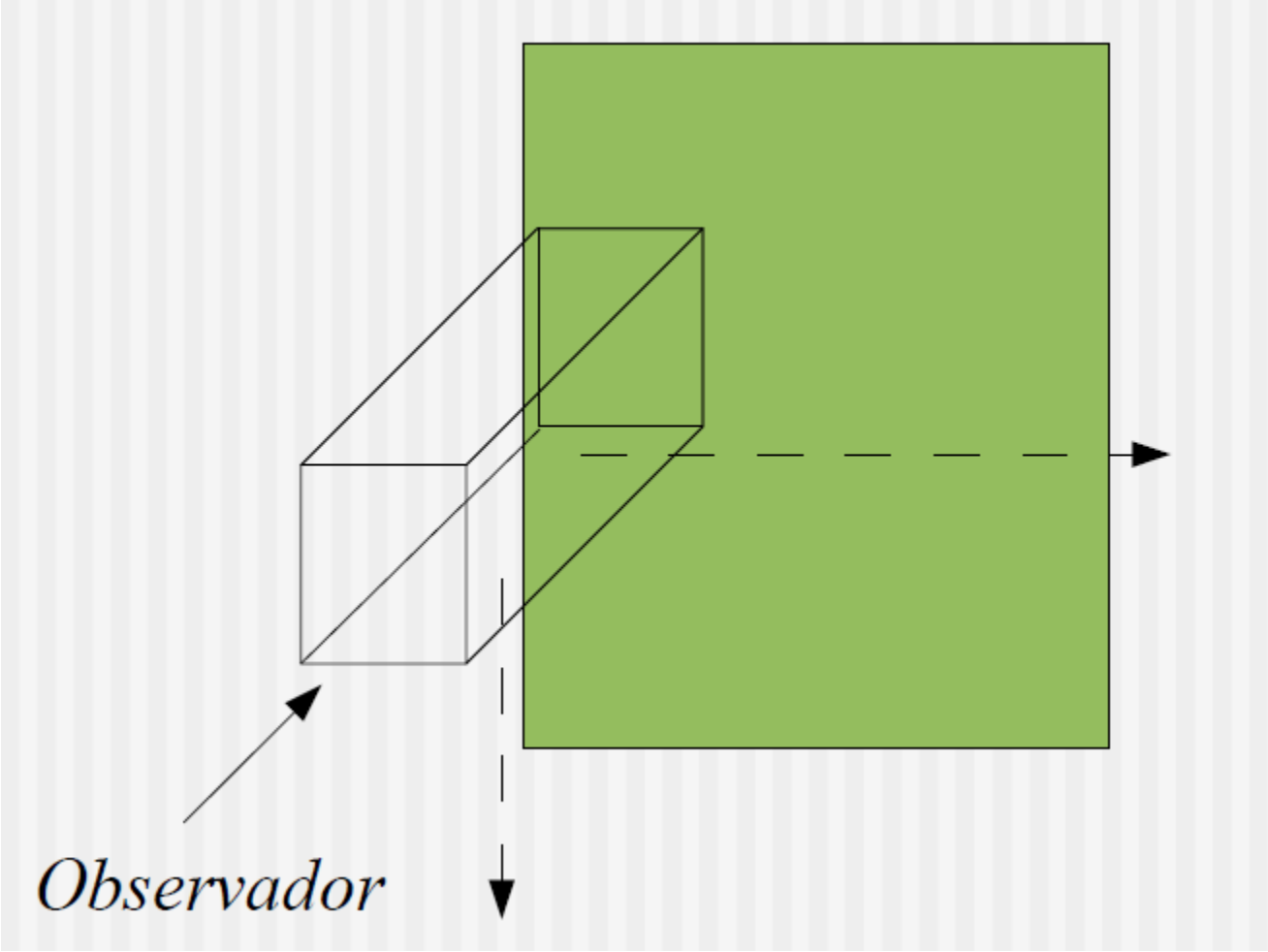
- *Platform Games, adventures*



Side View

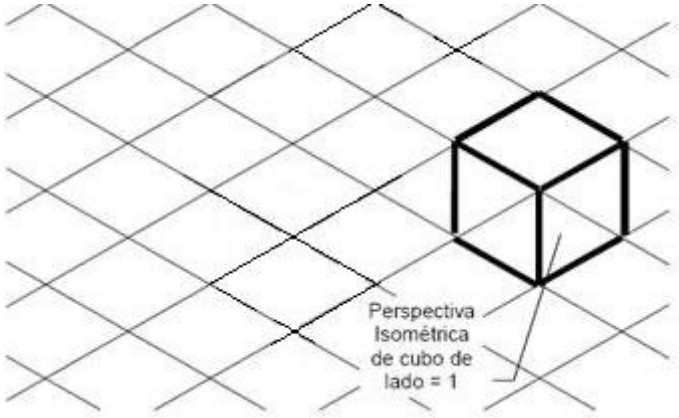
- Câmera posicionada ao lado do personagem.
- Pode ocorrer *scrolling da tela*
- Uso de *tilemaps* quadriculares (caso trivial)
- No caso dos *adventures* em geral é utilizada uma combinação de imagens (não usa *tilemaps*)

Side View



Visão Isométrica

- Isométrico: método matemático para desenhar uma figura 3D sem o uso da perspectiva.
- Todos os tamanhos são preservados e os objetos são desenhados com uma inclinação de 30°.
- Os sprites também devem ser mudados.
- Também conhecido como “2.5D” ou “falso 3D”

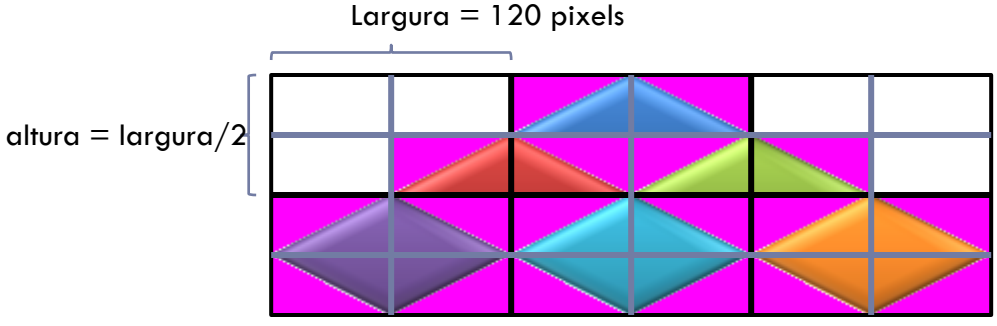


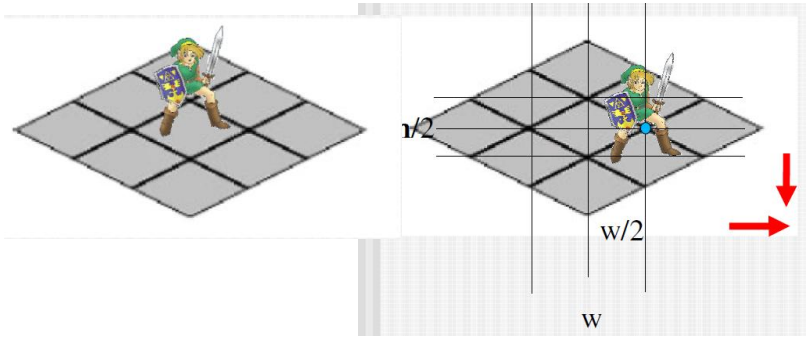
Isometria

- Caso especial em que o plano de projeção forma o mesmo ângulo com os três eixos principais. As projeções dos três vetores unitários canônicos formam ângulos de 120^o entre si.
- Isto permite que as medições feitas na projeção em cada eixo utilize a mesma escala

Visão Isométrica

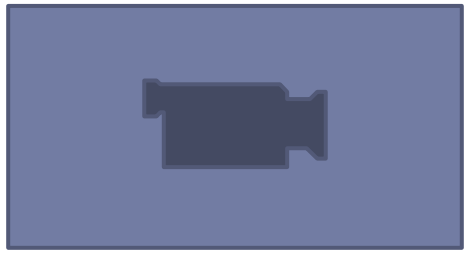
- Tiles isométricos
  - Como resolver a sobreposição?
  - Como navegar??



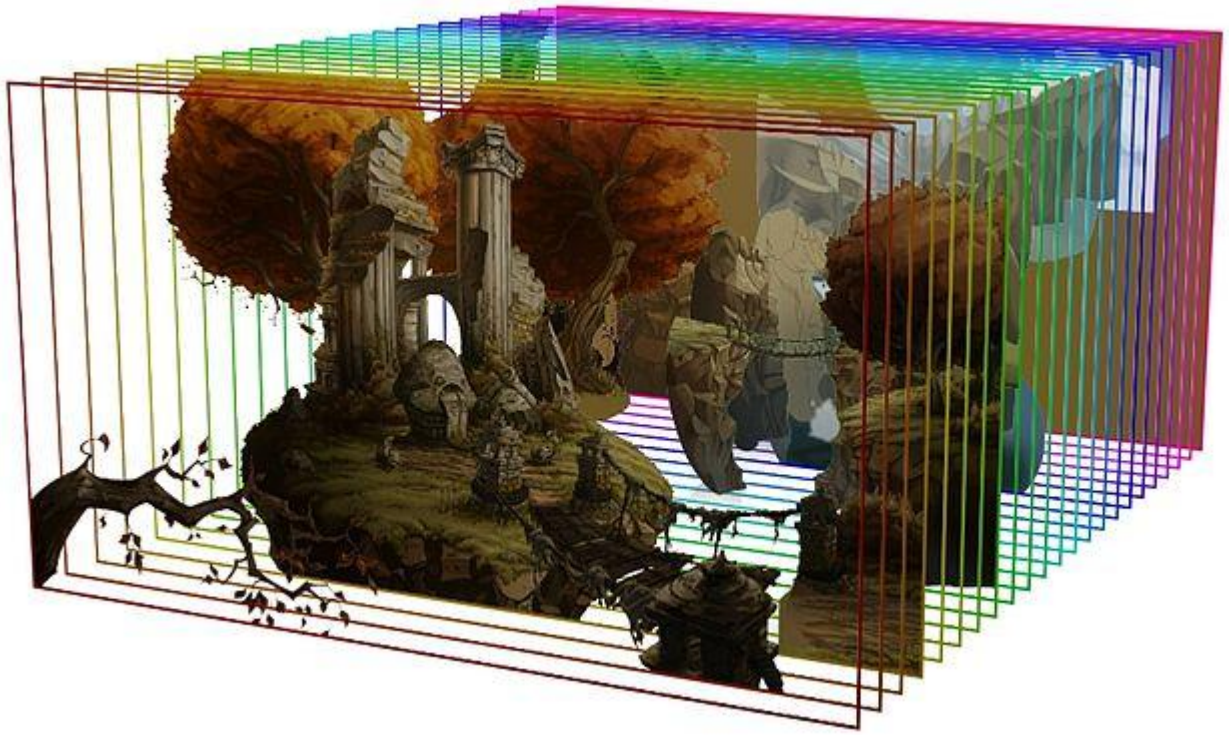


Efeito de *Parallax*

- Recurso muito interessante para aumentar o realismo de jogos 2D.
- Cria um efeito de profundidade
- Voltando no tempo...
  - Multi-plane camera, do Walt Disney

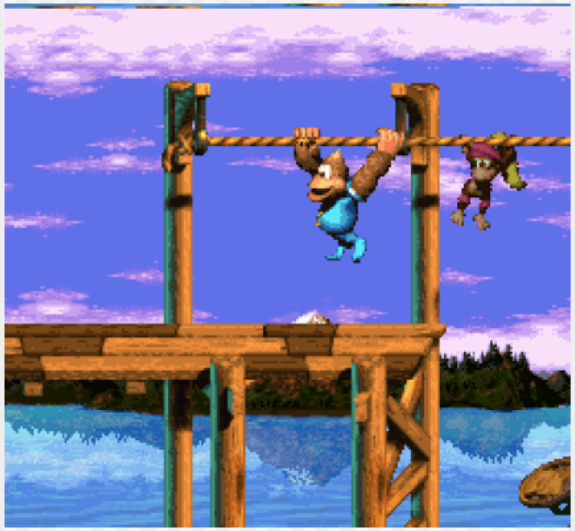


Efeito de *Parallax*



Efeito de *Parallax*

- Exemplos



Donkey Kong

Shadow of the Beast

Robot Unicorn Attack

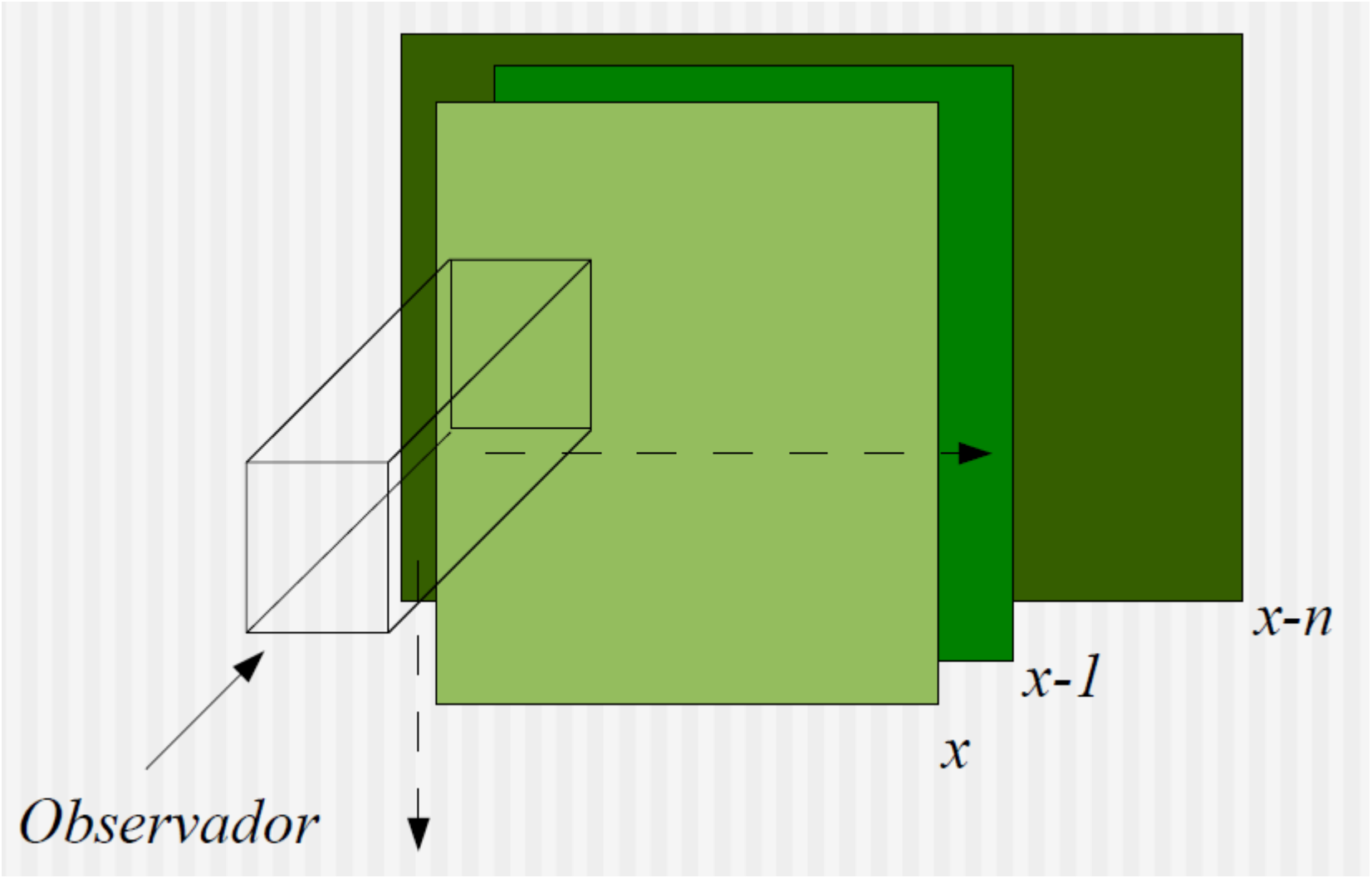
Efeito de *Parallax*

- Define-se uma imagem de fundo (pode ser feita com *tilemaps) e 'n'* camadas sob este fundo.
- Da camada do fundo para camada da frente
  - Fundo: desloca-se mais lento
  - Frente: desloca-se mais rápido

Efeito de *Parallax*

- Calcula-se a taxa de *scrolling* para cada camada em relação a camada básica (onde ocorre a ação, contém o personagem e os inimigos)
- Exemplo:
  - Camada de fundo = 600x480
  - Básica: 1000x600
  - Taxa camada de fundo:
    - taxa_x= 600/1000 = 0.6
    - taxa_y=480/600 = 0.8

Efeito *Parallax*



Referências

- COHEN, Marcelo, MANSSOUR, Isabel. OpenGL : uma abordagem prática e objetiva. São Paulo : Novatec, 2006. 478 p.
  - Capítulo 7
- Materiais do professor Leandro Tonietto e João Bittencourt