Insper

Computação Gráfica

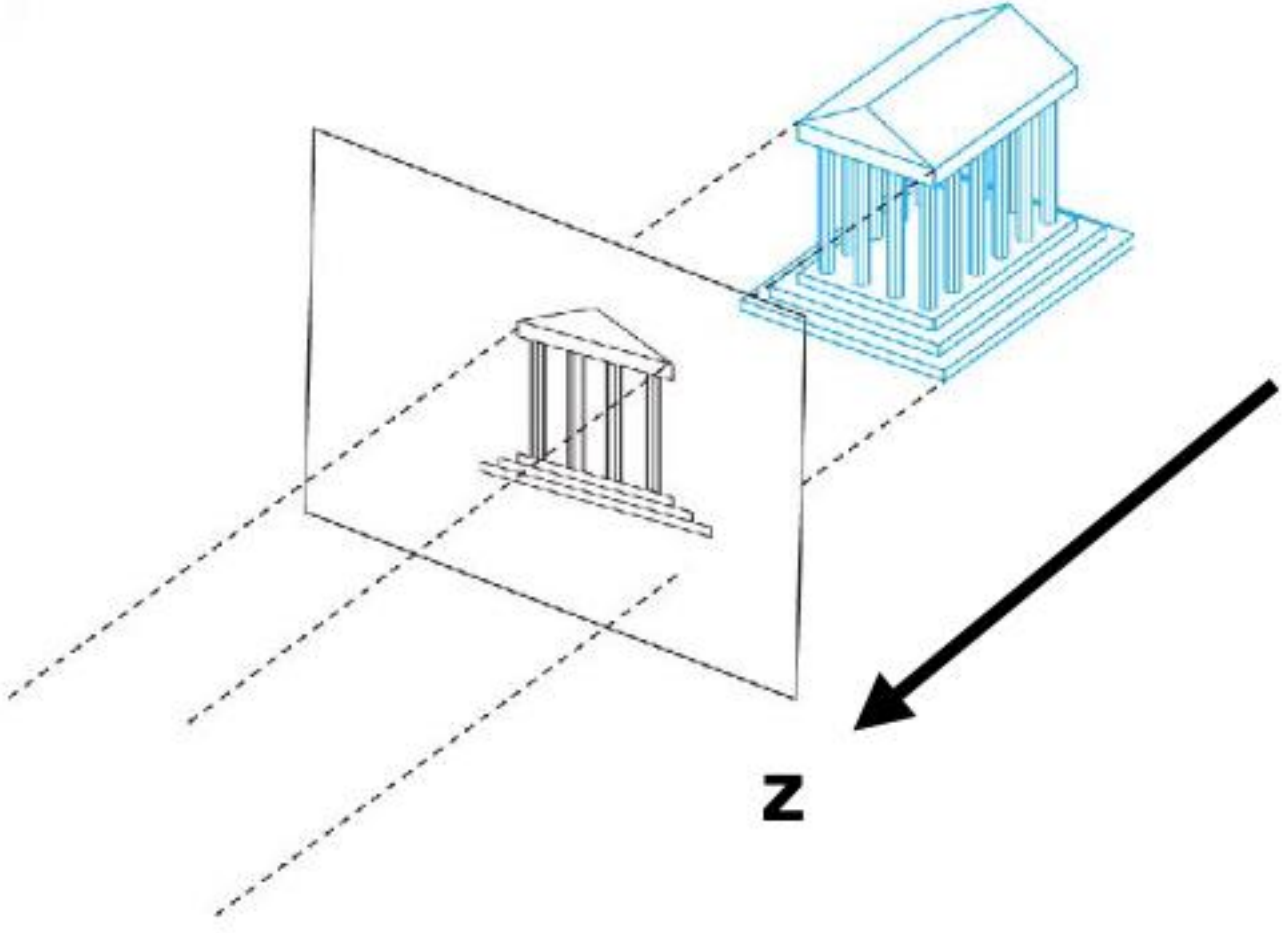
Aula 6: Projeções

Kahoot

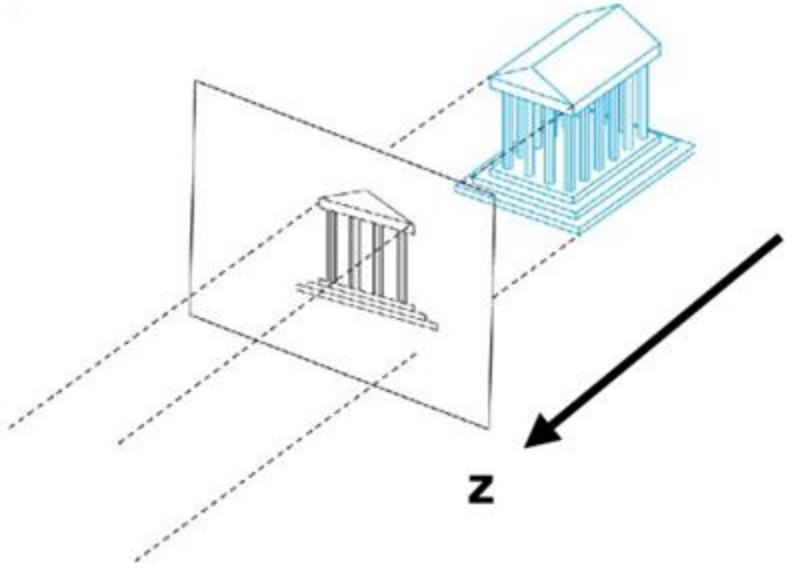
Entrar em Kahoot.it : https://kahoot.it/

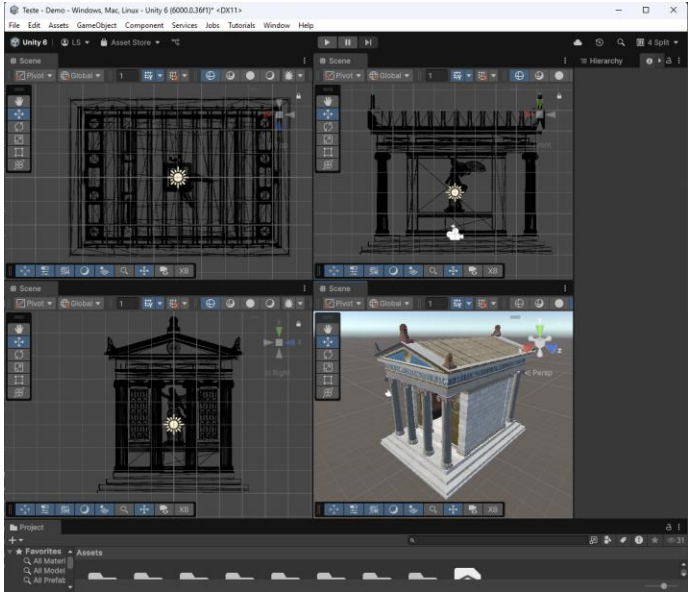
Perspectiva

Do espaço da câmera (3D) para Plano de Imagem (2D)



Do espaço da câmera (3D) para Plano de Imagem (2D)





Como transformar do espaço da câmera 3D em um plano de imagem 2D?

- Uma opção: projeção ortográfica (basta excluir z)
- Útil, para, por exemplo, para desenhos de engenharia
- <u>Mas será que é conveniente para tudo?</u>

Perspectiva na Arte

Paradiesgärtlein; c. 1410

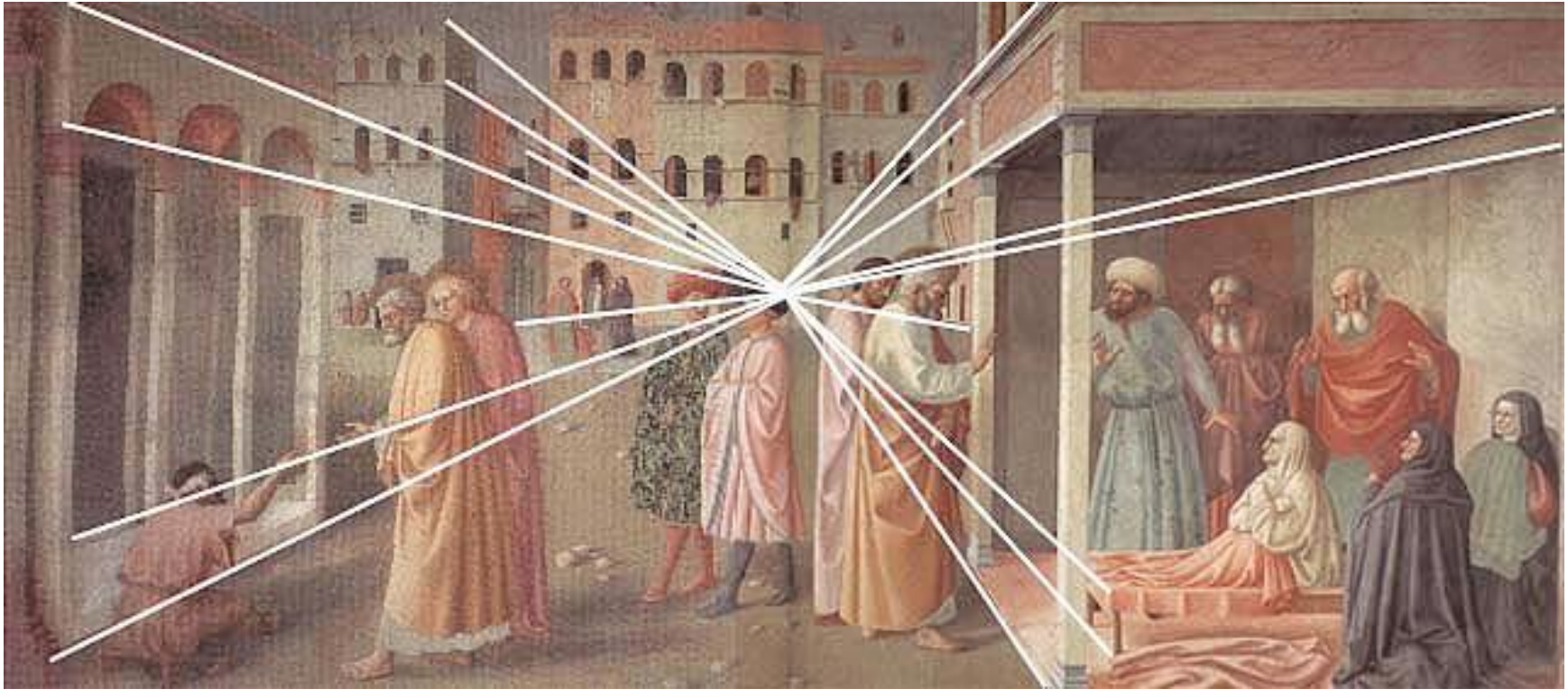
Perspectiva da Idade Média para Renascentismo

Cristo davanti a Caifa; por Giotto; c.1305

"As vigas do teto convergem de maneira convincente, mas a projeção geométrica expõe a falha em convergir com precisão."

Tyler, Christopher. (2000). Perspective as a geometric tool that launched the Renaissance. Proceedings of SPIE - The International Society for Optical Engineering.

Perspectiva no Renascentismo

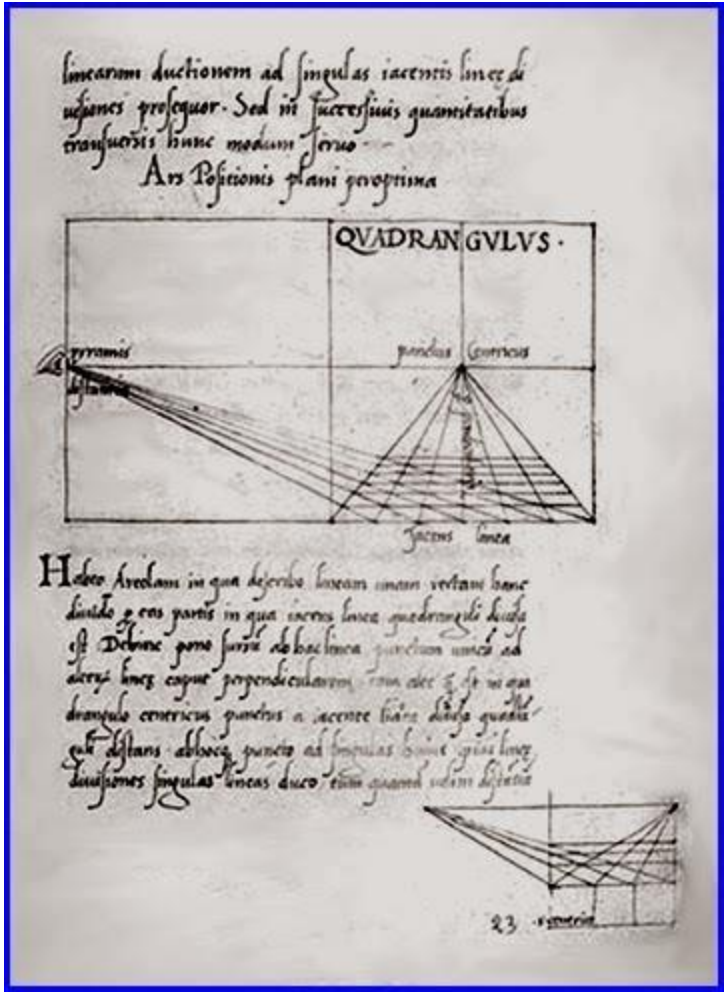


Guarigione dello storpio e resurrezione di Tabita; por Masolino; 1424-1425

O primeiro uso conhecido de convergência central precisa. Ele fez uso consciente da perspectiva linear, como demonstrado pelo prego usado na parede para traçar linhas de fuga.

Tyler, Christopher. (2000). Perspective as a geometric tool that launched the Renaissance. Proceedings of SPIE - The International Society for Optical Engineering.

De pictura



Leon Battista Alberti escreveu De pictura pela primeira vez em 1435, no qual descreveu, em palavras e ilustrações, os princípios geométricos da perspectiva na pintura. No ano seguinte, ele escreveu uma versão no dialeto toscano, Della pittura.

Hoje, apenas 20 manuscritos latinos e três manuscritos toscanos sobrevivem. Foi publicado pela primeira vez na imprensa em Basel em 15XX.

Fonte: https://euclidinflorence.wordpress.com/exhibition-catalog/albertis-de-pittura/

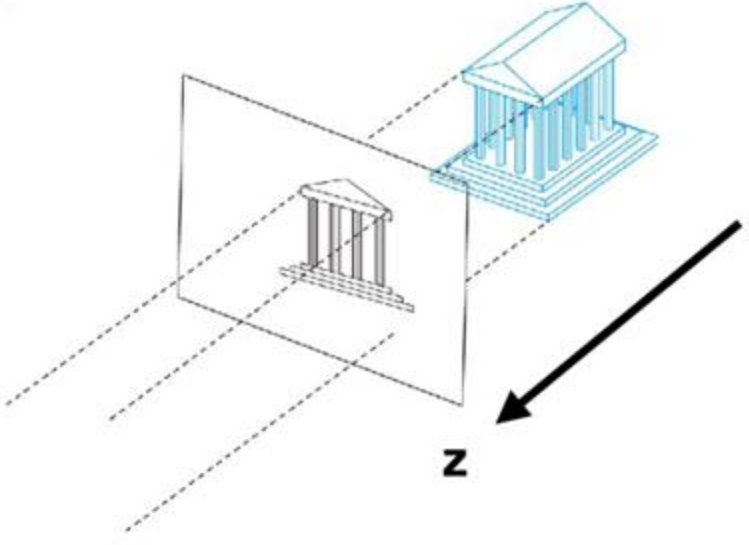
Insper

Perspectiva na Arte

A Última Ceia; por Leonardo da Vinci; 1499

Perspectiva na Arte

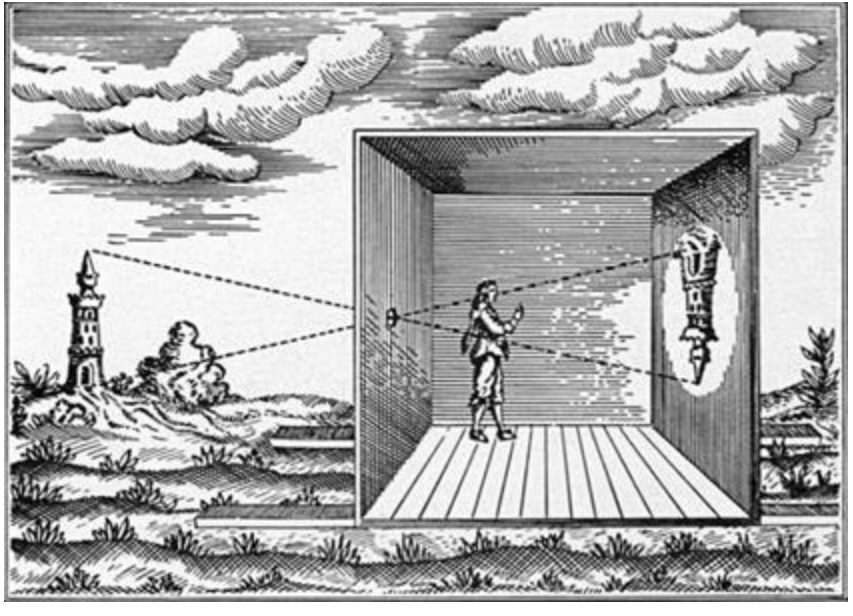
Uma projeção ortográfica teria o mesmo resultado na obra da última ceia de Leonardo da Vinci?



NÃO

Câmara Estenopeica

Camera Obscūra (latim) em geral um sala onde existe somente um orifício que permite a luz entrar e numa parede reversa a esse furo se encontra uma parede ou tela onde se pode ver a imagem do exterior de forma invertida.



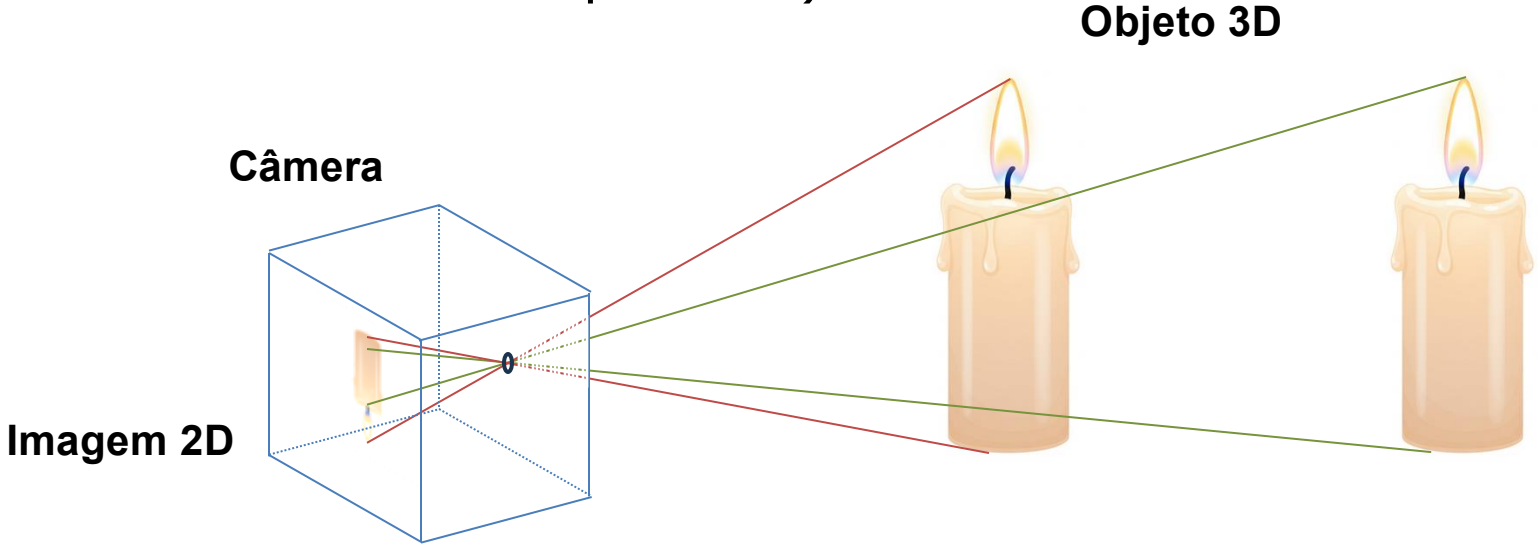
Uma propriedade interessante dessas câmeras é que as imagens geradas tem um foco em todos os planos da cena.

Projeção Perspectiva

Objetos parecem menores conforme ficam mais distantes.

Por que isso acontece?

Considere o modelo simplificado de câmera "pinhole"
(câmeras reais são mais complicadas)

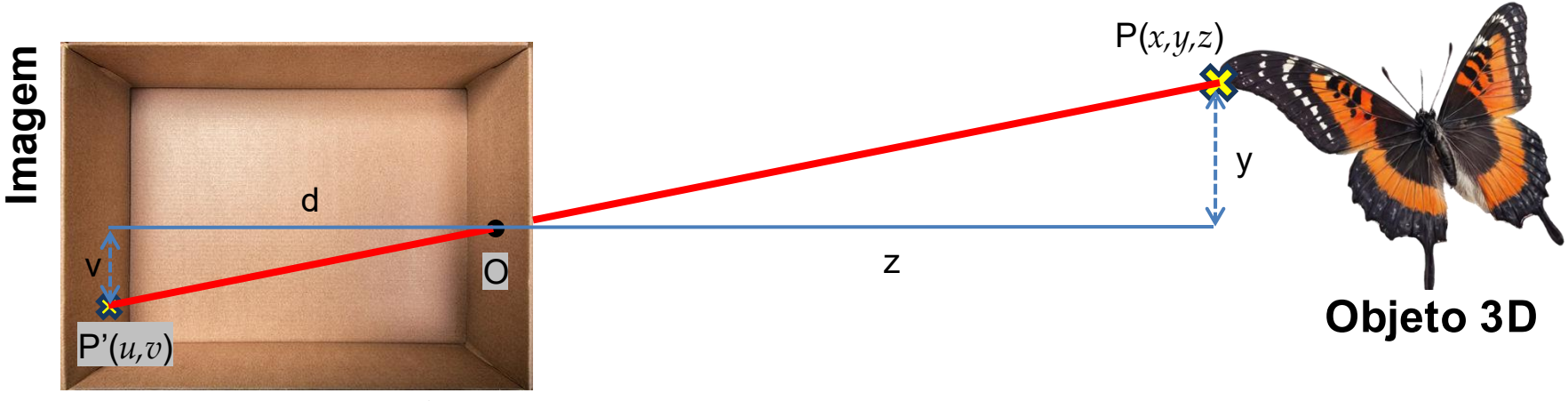


A altura e largura do objeto projetado depende da distância dele.

Insper

Projeção Perspectiva : Visão Lateral

Onde exatamente um ponto P(x,y,z) vai aparecer na imagem?
Vamos chamar o ponto na imagem como P' (u,v)



visão perpendicular para simplificar análise

Perceba a semelhança de triângulos:

- Assumindo a câmera estar alinhada ao eixo Z
- Logo v/d = y/z, assim o deslocamento vertical da imagem (v) é igual a d·y/z
- Se simplificarmos d=1, então v = y/z
- Da mesma forma na horizontal (u) a coordenada vale x/z

Atividade: Modelar e Desenhar um Cubo

Objetivo:

- gerar a imagem de um cubo

Perguntas:

- Modelagem: Como descrever um cubo?
- Renderização: Como visualizar esse modelo?

Atividade: Modelando o Cubo

Suponha que:

- Nosso cubo está centralizado na origem (0,0,0)
- Está alinhado aos eixos do sistema de coordenadas
- Tem dimensões 2 x 2 x 2 (a unidade não é relevante agora)
- A abertura da câmera está posicionada no ponto: O(2,-3,5)
- O plano de projeção da câmera está a uma distância 1 em Z

Questão:

Quais são as coordenadas dos vértices do cubo?

A:(-1,-1,-1)	E:(-1,1,-1)
B:(-1,-1,1)	F:(-1,1,1)
C:(1,-1,1)	G:(1,1,1)
D: (1,-1,-1)	H: (1,1,-1)

Questão:

Como são as arestas desse cubo?

AB, BC, CD, DA, EF, FG, GH, HE, BF, CG, DH, AE

Atividade: Desenhando o Cubo

Temos uma descrição do Cubo:

Vértices (VERTICES)

A:(-1,-1,-1) E:(-1,1,-1)
B:(-1,-1,1) F:(-1,1,1)
C:(1,-1,1) G:(1,1,1)
D: (1,-1,-1) H: (1,1,-1)

Arestas (EDGES)

AB, BC, CD, DA, EF, FG,
GH, HE, BF, CG, DH, AE

De que forma desenhar esse cubo como uma imagem 2D:

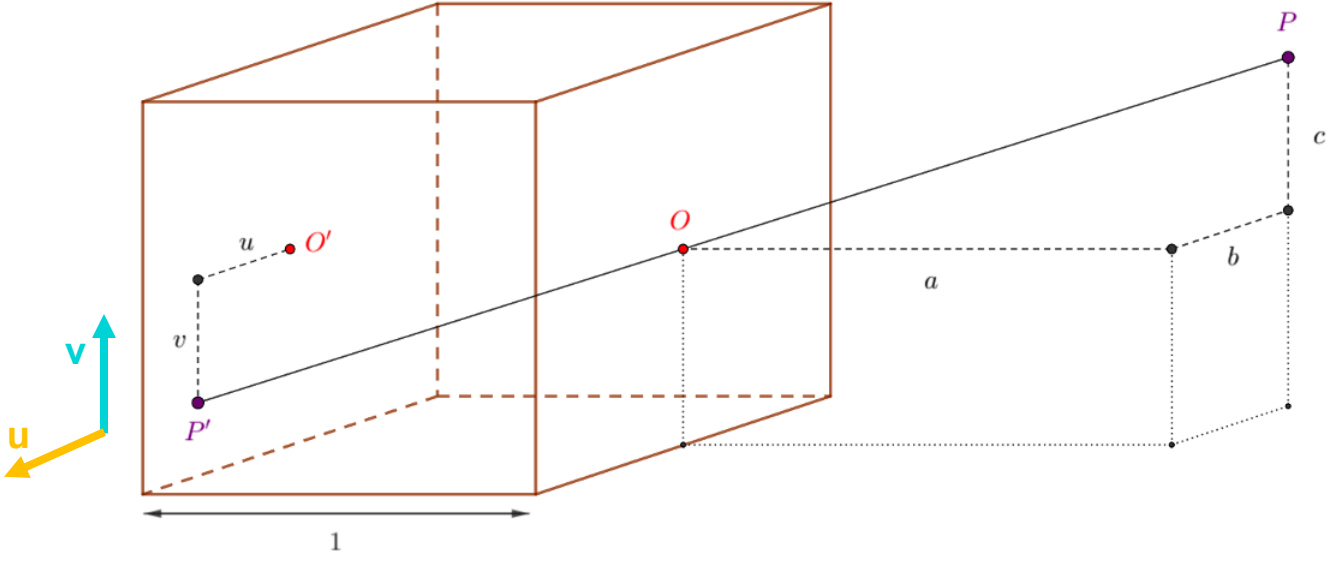
- Projetar os vértices 3D como pontos em um plano 2D.
- Conectar os pontos 2D com linhas retas.

ATIVIDADE 1: DESENHANDO UM CUBO

Acesse o documento no site da disciplina.

Crie uma cópia para você e realize todos os exercícios.

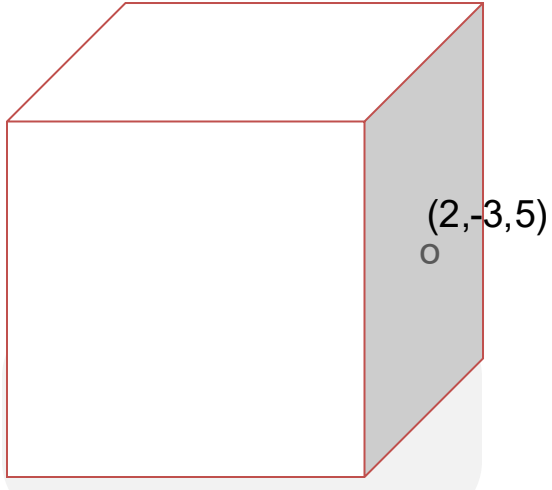
Voltamos em 30 minutos?

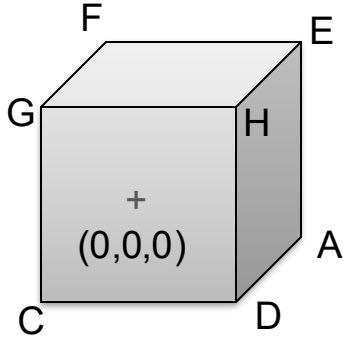


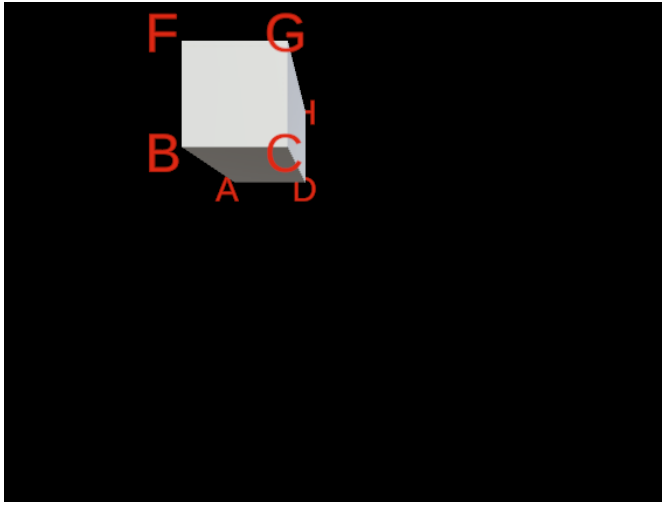
Insper

Atividade: Pensando na Solução

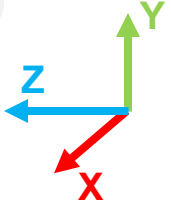
Visualize primeiramente o posicionamento do cubo e da câmera. Identifique os pontos e como se organizam.







Renderizado no Unity



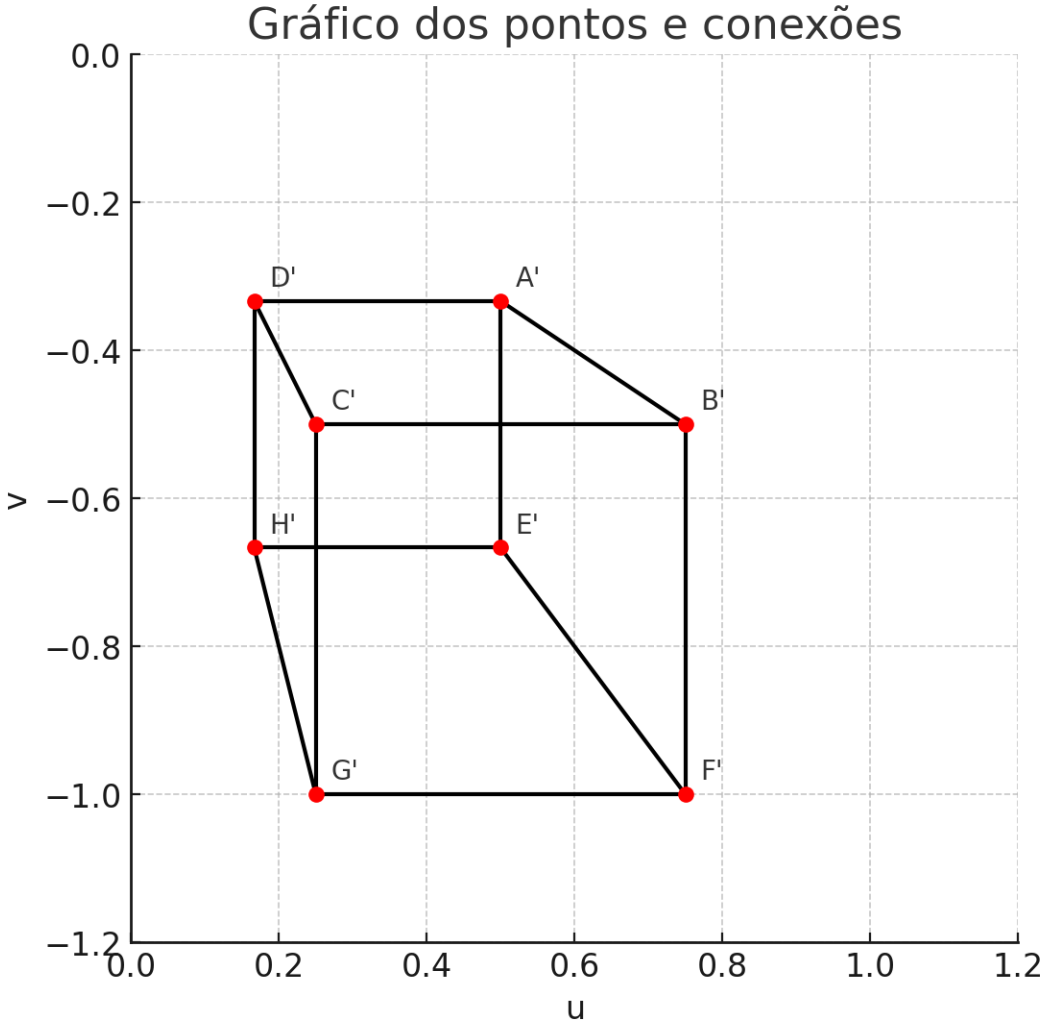
Insper

Atividade: Realize os Cálculos

A:(-1,-1,-1)
B:(-1,-1,1)
C:(1,-1,1)
D: (1,-1,-1)
E:(-1,1,-1)
F:(-1,1,1)
G:(1,1,1)
H: (1,1,-1)

	A	B	C	D	E	F	G	H
$u = \frac{x-2}{z-5}$	$\frac{-3}{-6}$	$\frac{-3}{-4}$	$\frac{-1}{-4}$	$\frac{-1}{-6}$	$\frac{-3}{-6}$	$\frac{-3}{-4}$	$\frac{-1}{-4}$	$\frac{-1}{-6}$
$v = \frac{y-(-3)}{z-5}$	$\frac{2}{-6}$	$\frac{2}{-4}$	$\frac{2}{-4}$	$\frac{2}{-6}$	$\frac{4}{-6}$	$\frac{4}{-4}$	$\frac{4}{-4}$	$\frac{4}{-6}$

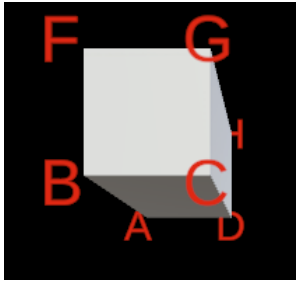
Atividade: Resultado Esperado



Coordenadas:

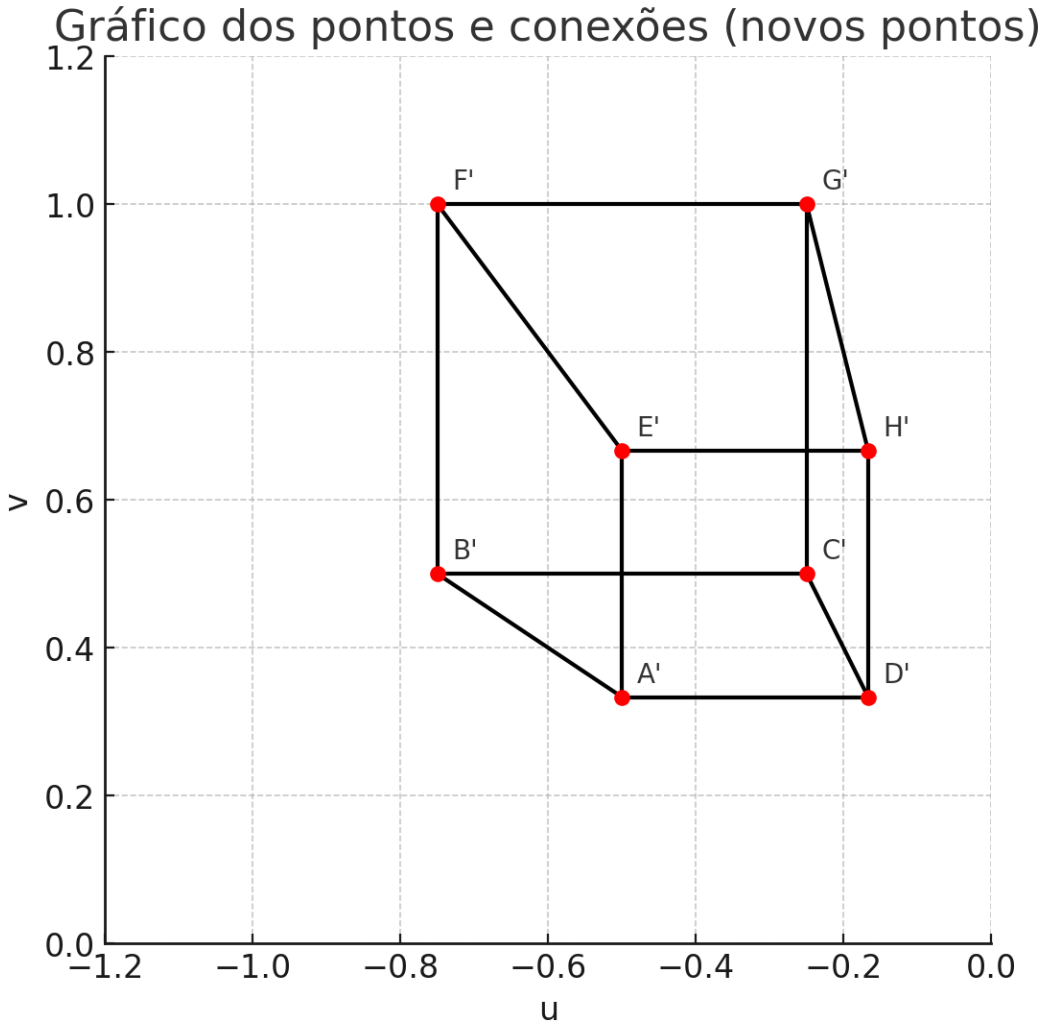
A’: (1/2, -1/3)
B’: (3/4, -1/2)
C’: (1/4, -1/2)
D’: (1/6, -1/3)
E’: (1/2, -2/3)
F’: (3/4, -1)
G’: (1/4, -1)
H’: (1/6, -2/3)

O que aconteceu?



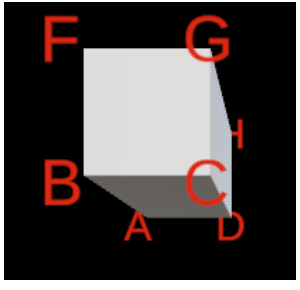
Insper

Atividade: Resultado Final



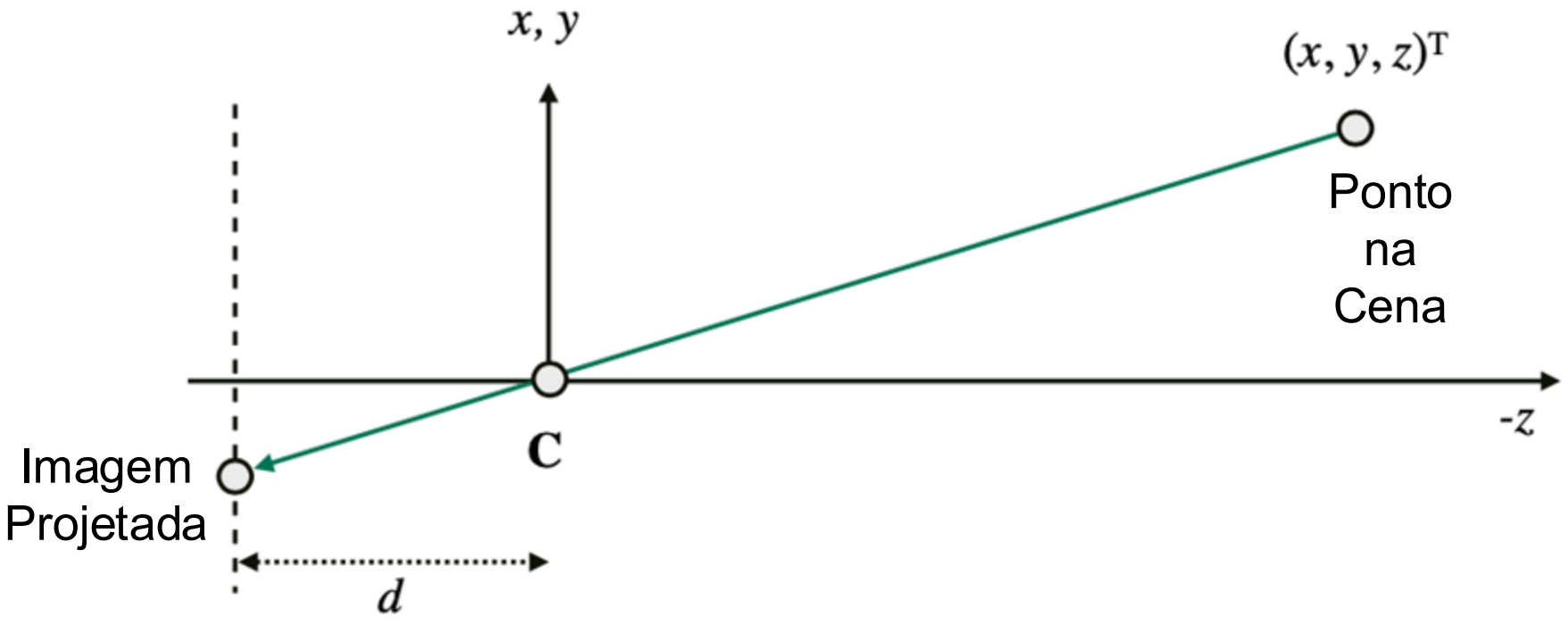
Coordenadas:

A’: (-1/2, 1/3)
B’: (-3/4, 1/2)
C’: (-1/4, 1/2)
D’: (-1/6, 1/3)
E’: (-1/2, 2/3)
F’: (-3/4, 1)
G’: (-1/4, 1)
H’: (-1/6, 2/3)



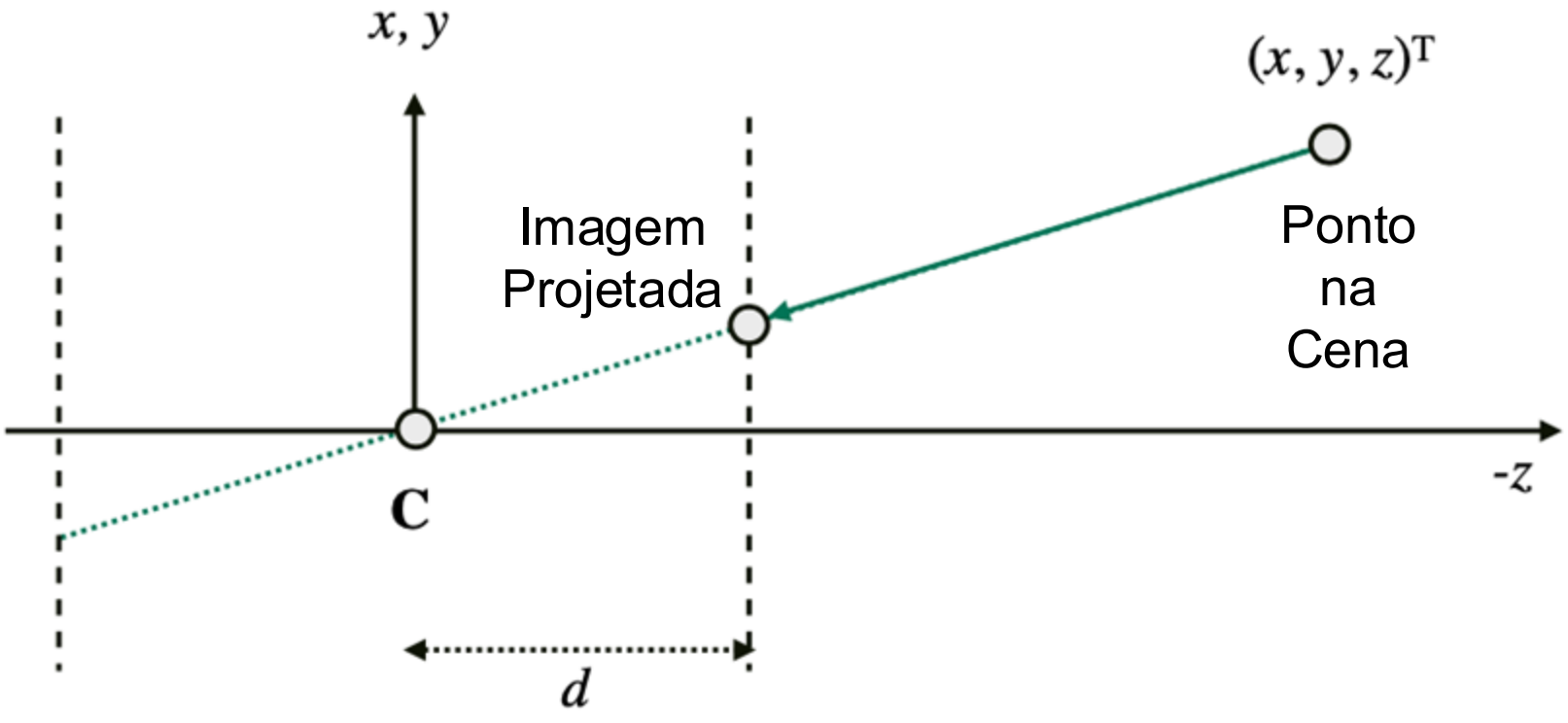
Insper

Calculando Câmera Pinhole



A imagem vai estar invertida (como numa câmera pinhole real)

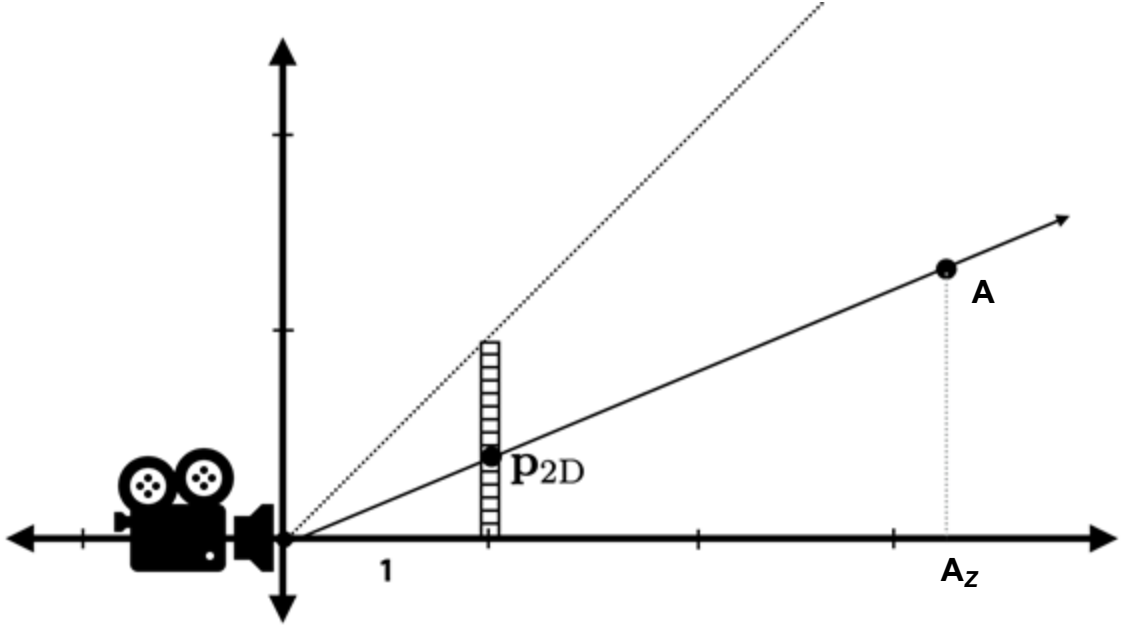
Transformação Projetiva de Câmera Pinhole



Não está mais invertida

Insper

Projeção Perspectiva Básica (2D)



Resultado da projeção perspectiva desejada (ponto 2D)

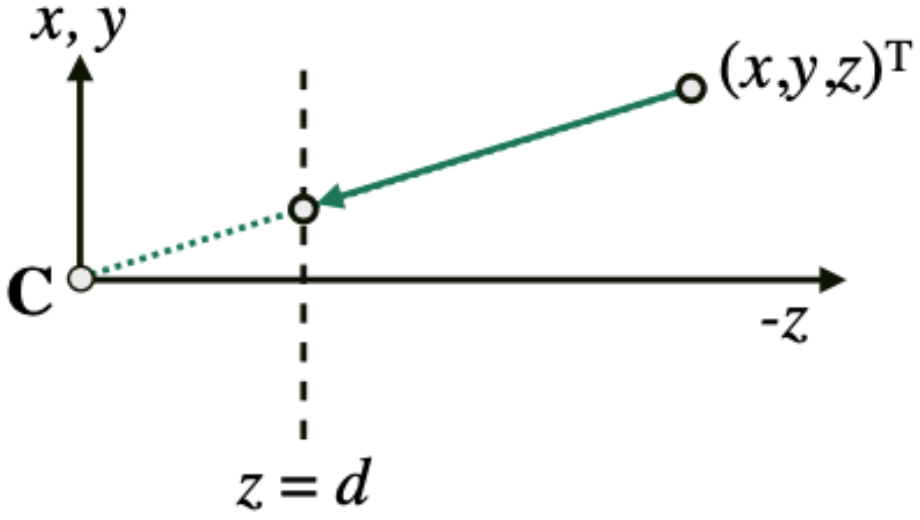
$$P_{2D} = \begin{bmatrix} \frac{A_x}{A_z} \\ \frac{A_y}{A_z} \end{bmatrix}$$

Insper

Transformações Projetivas

Projeção em perspectiva padrão:

- Centro de projeção: $(0, 0, 0)^T$
- Plano de imagem em z = d



$$proj\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} d \cdot \frac{x}{z} \\ d \cdot \frac{y}{z} \end{pmatrix}$$

Transformações Projetivas

Projeção em perspectiva padrão:

- Centro de projeção: $(0, 0, 0)^T$
- Plano de imagem em z = d

Projeção perspectiva:

- Precisa de divisão por z
- Como representar por matriz?

$$proj\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} d \cdot \frac{x}{z} \\ d \cdot \frac{y}{z} \end{pmatrix}$$

Coordenadas homogêneas!

Projeção em Coordenadas Homogêneas (3D)

$$\mathbf{p} = \begin{pmatrix} wx \\ wy \\ wz \\ w \end{pmatrix} \longleftrightarrow \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

$$\mathbf{M} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{pmatrix}$$

$$\mathbf{q} = \mathbf{M} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \\ z/d \end{pmatrix} \longleftrightarrow \begin{pmatrix} xd/z \\ yd/z \\ d \\ 1 \end{pmatrix}$$

Termo diferente de zero na linha final.

termo final que não é mais 1 (um)

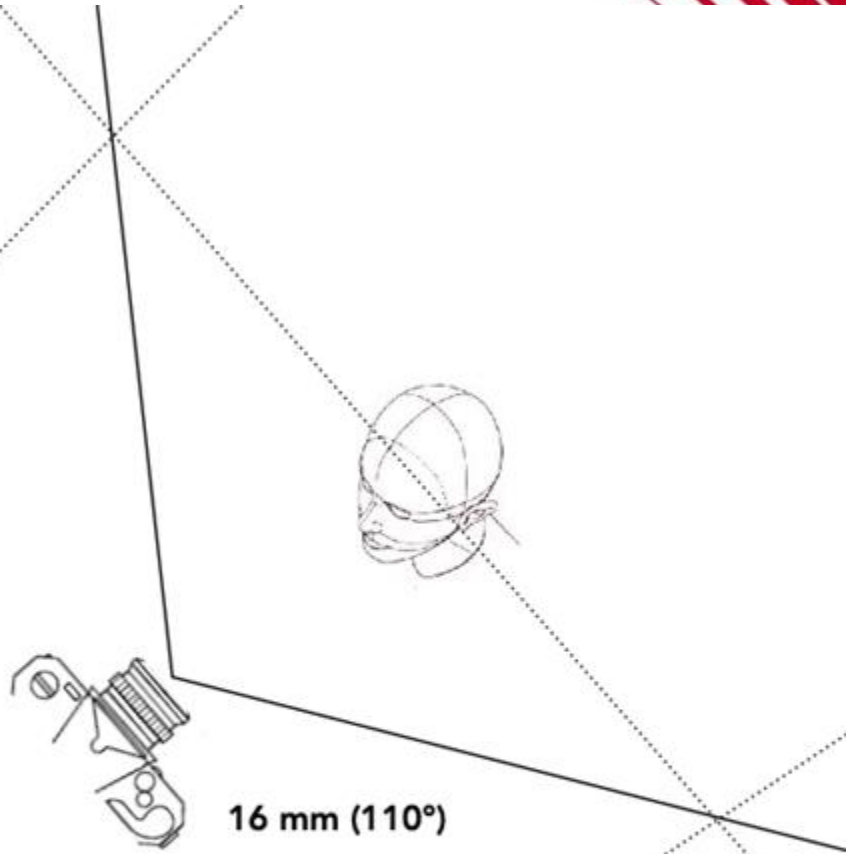
Perspectiva: posição da câmera + ângulo de visão

Nesta sequência, o ângulo de visão diminui à medida que a distância da pessoa aumenta, para assim manter o tamanho da pessoa na imagem.

Observe a mudança dramática na perspectiva do fundo.

Insper

Composição Perspectiva

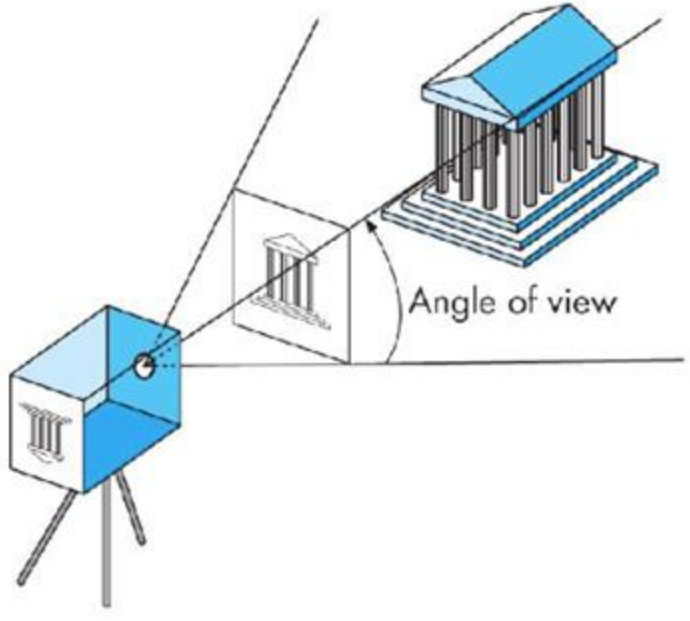


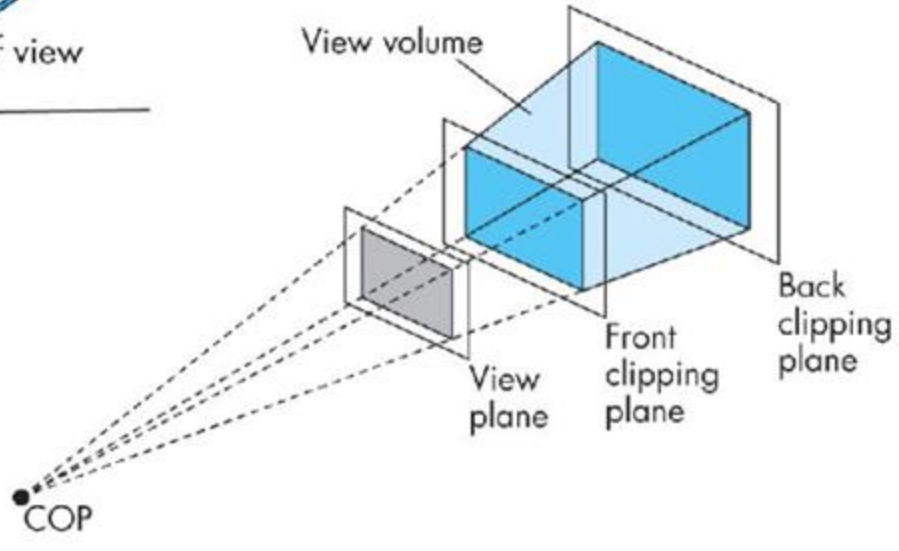
Próximo e com um pequeno zoom

Composição Perspectiva

Afastado e com um grande zoom

Especificando a Projeção Perspectiva



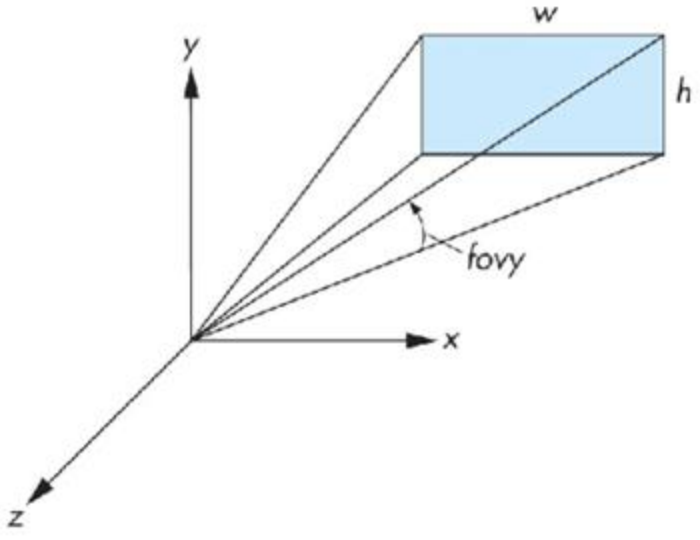


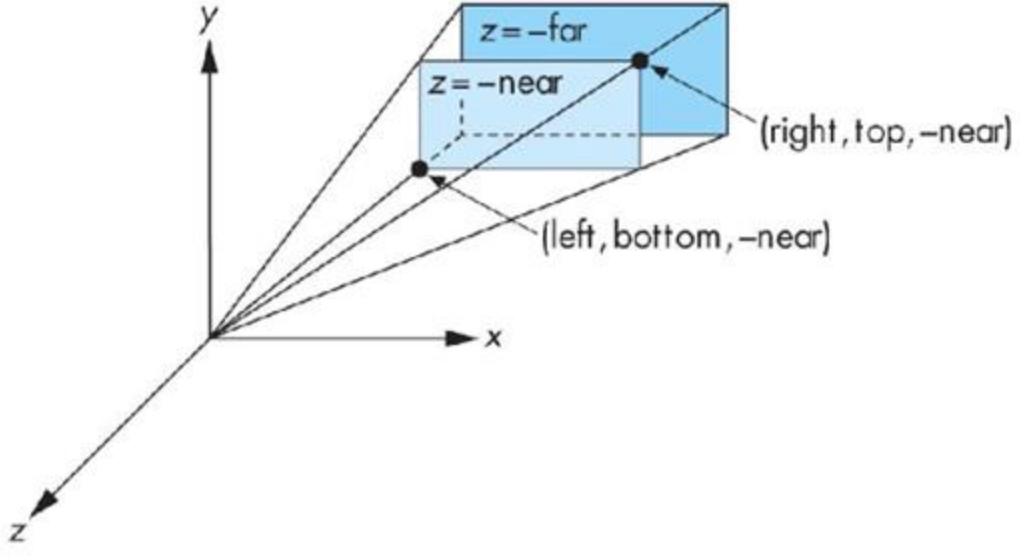
COP = Center Of Projection

do livro: Angel and Shreiner, Interactive Computer Graphics

Insper

Especificando o Volume de Visão Perspectiva





do livro: Angel and Shreiner, Interactive Computer Graphics

Insper

Especificando o Volume de Visão Perspectiva

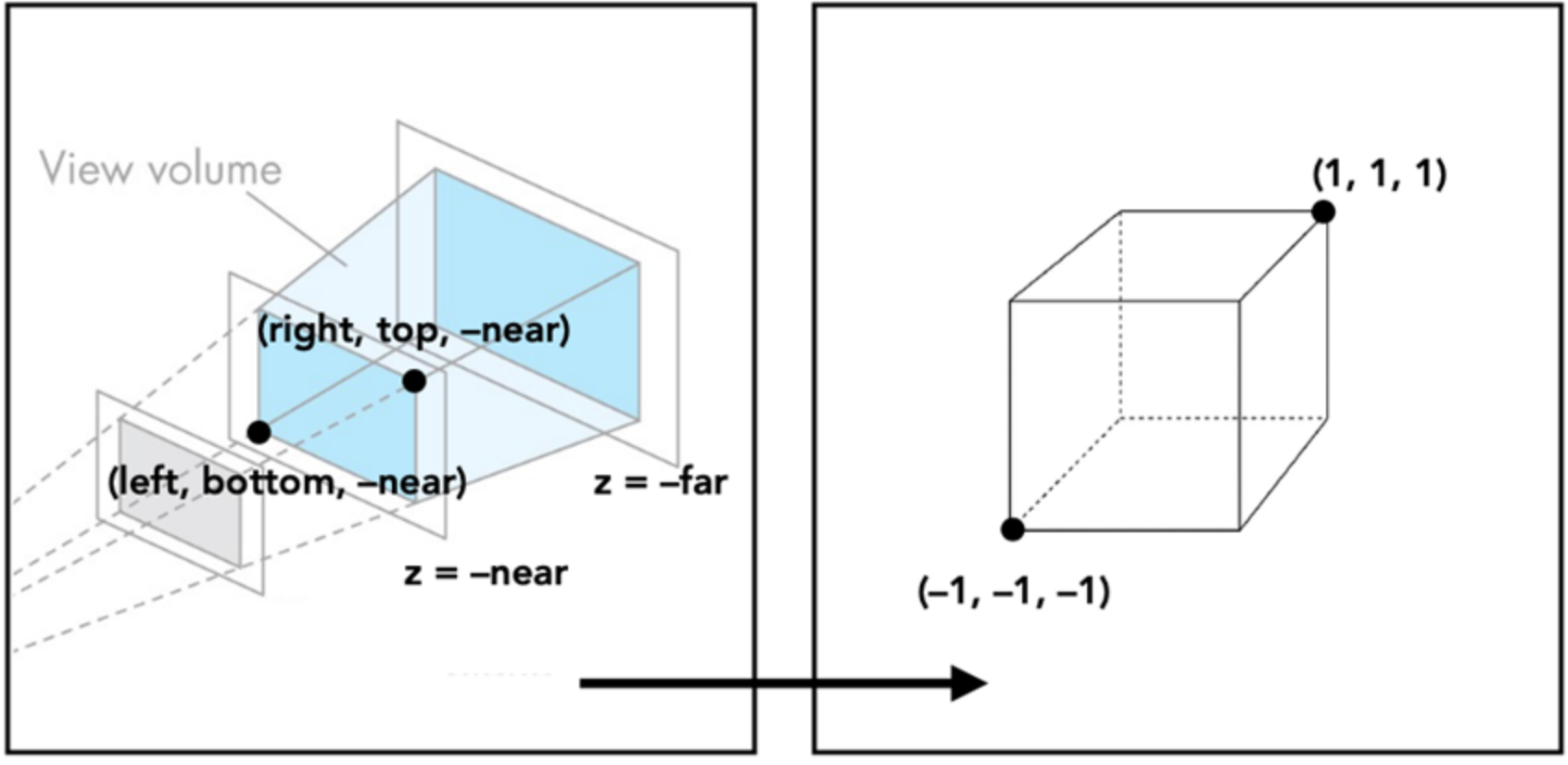
Parametrizado por

- fovy: campo de visão angular vertical (eixo y)
- razão de aspecto: largura / altura do campo de visão
- near: profundidade do plano de corte próximo
- far: profundidade do plano de recorte distante

Quantidades derivadas

- top = near * tan (fovy)
- bottom = –top
- right = top * aspect
- left = –right

Implementação de Projeção Perspectiva



Coordenadas de câmera (view)

Normalized Device Coords (NDC)

já iremos para as coordenadas da tela

Transformações Perspectivas

- Não precisa ser simétrica em relação ao eixo z, porém, para simplificar, assumimos por enquanto que é simétrica

- A transformação preservará as informações de profundidade

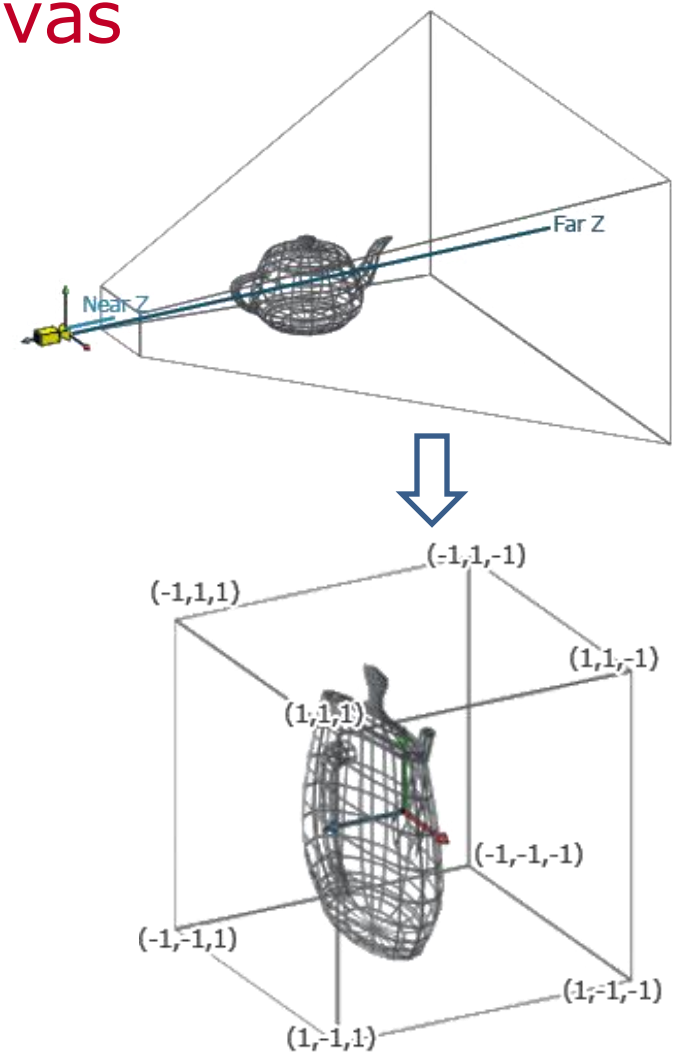


Imagem: http://www.codinglabs.net/article_world_view_projection_matrix.aspx

Insper

Matriz de Transformação Perspectiva

$$\mathbf{P} = \begin{bmatrix} \frac{near}{right} & 0 & 0 & 0 \\ 0 & \frac{near}{top} & 0 & 0 \\ 0 & 0 & -\frac{far+near}{far-near} & \frac{-2far*near}{far-near} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

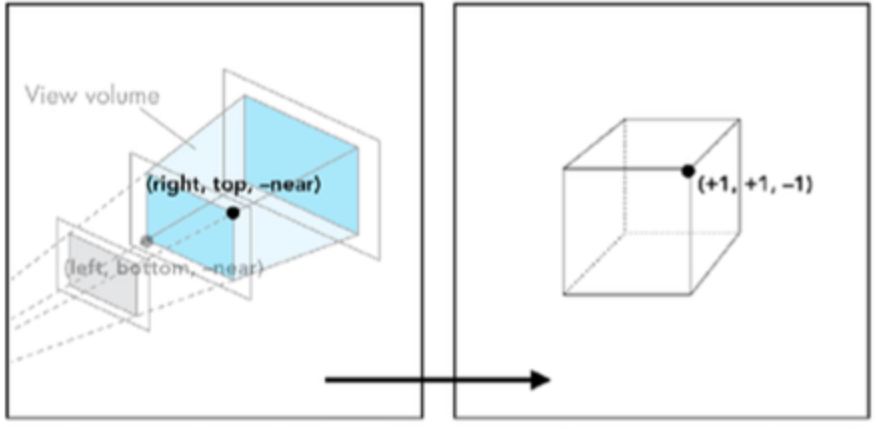
Exemplo com a Matriz Perspectiva

$$\mathbf{P} = \begin{bmatrix} \frac{near}{right} & 0 & 0 & 0 \\ 0 & \frac{near}{top} & 0 & 0 \\ 0 & 0 & -\frac{far+near}{far-near} & \frac{-2far*near}{far-near} \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} right \\ top \\ -near \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} near \\ near \\ near * \frac{far+near}{far-near} - \frac{2far*near}{far-near} \\ near \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 1 \\ \frac{-far+near}{far-near} \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$



Insper

Divisão Homogênea
(Perspective Divide ou Homogeneous Divide)

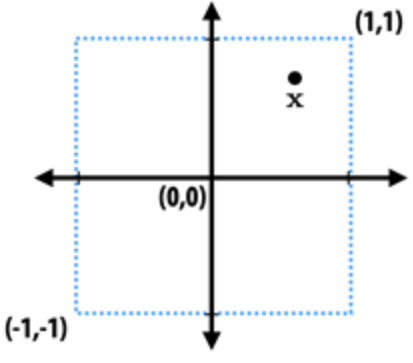
Após a multiplicação pela matriz de projeção em coordenadas homogêneas, o quarto componente (w) geralmente deixa de ser igual a 1. Para normalizar o vetor, divide-se todos as dimensões por w, fazendo com que o quarto valor volte a ser 1.

$$\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} \frac{x}{w} \\ \frac{y}{w} \\ \frac{z}{w} \\ 1 \end{bmatrix}$$

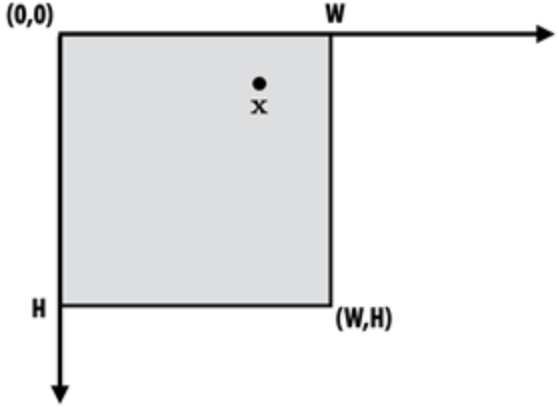
Transformações para Tela
Screen Transformation

Converter pontos no espaço de coordenadas normalizado da câmera para coordenadas de pixel da tela

sistema normalizado da câmera



Espaço de coordenadas da tela (imagem de W x H)



Todos os pontos dentro da região (-1,-1) a (1,1) estão na tela

(1, 1) no espaço da câmera mapeia para (W, 0) no espaço da tela

(-1, -1) no espaço da câmera mapeia para (0, H) no espaço de tela

Perceba que é preciso espelhar em Y, transladar por (1, 1) e escalar para (W/2, H/2)

Transformações para Tela
Screen Transformation

Qual matriz espelha em Y, transladar por (1, 1) e escalar para (W/2, H/2) ?

Escala para (W/2, H/2) Translate por (1,1) Espelha em Y

$$\begin{bmatrix} \frac{W}{2} & 0 & 0 & 0 \\ 0 & \frac{H}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{W}{2} & 0 & 0 & \frac{W}{2} \\ 0 & -\frac{H}{2} & 0 & \frac{H}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Uma projeção ortogonal do objeto no plano x-y do espaço da câmera (Camera Space) resultará na imagem esperada.

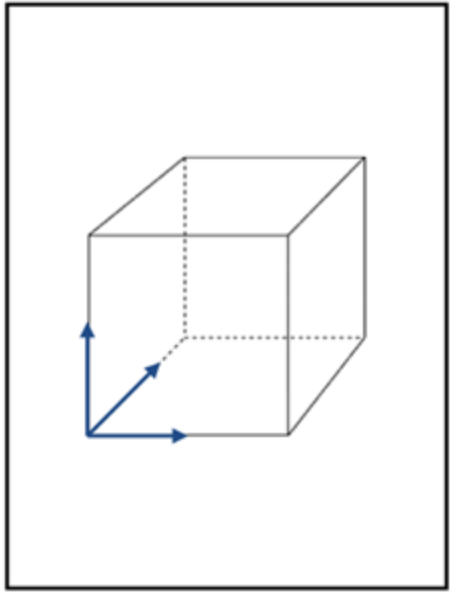
Contudo perceba que quando vamos do espaço da câmera (Camera Space) para o espaço da tela (Screen Space) ainda temos o valor do Z (profundidade), ou seja, podemos dizer o que espaço da tela é um espaço 3D. Essa informação de profundidade será usada no futuro.

Recapitulando Transformações

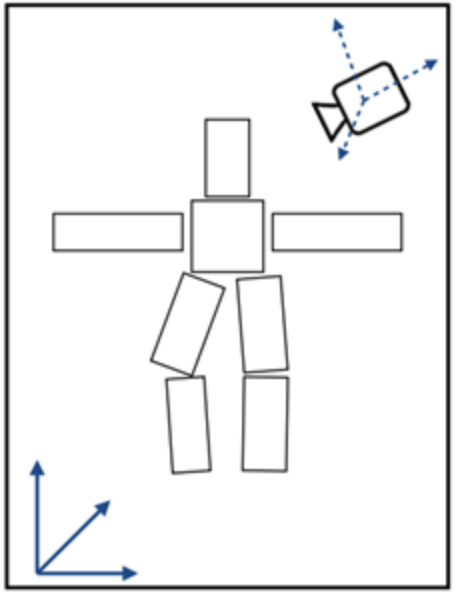
Sistemas coordenadas:

- Coordenadas do objeto (model 3D)
  - Aplicar transformações do modelo ...
- Coordenadas do mundo (world/cena 3D)
  - Aplicar transformação de visualização ...
- Coordenadas da câmera (view/visão 3D)
  - Aplicar transformação de projeção perspectiva
- Coordenadas normalizadas (NDC 3D)
  - Aplicar transformação de tela 2D ...
- Coordenadas da tela (screen 2D)

Recapitulando Transformações



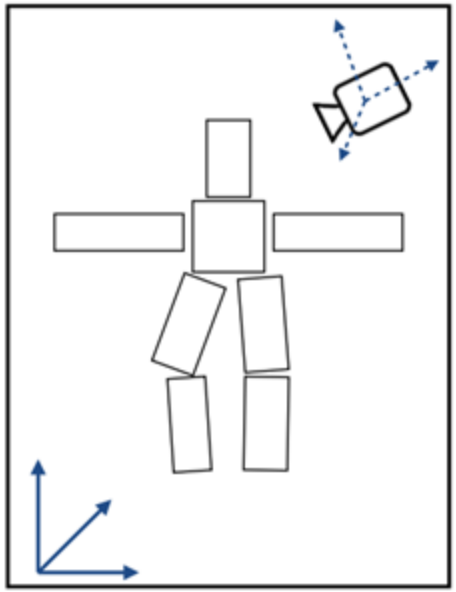
Modeling transforms



Coordenadas do Objeto

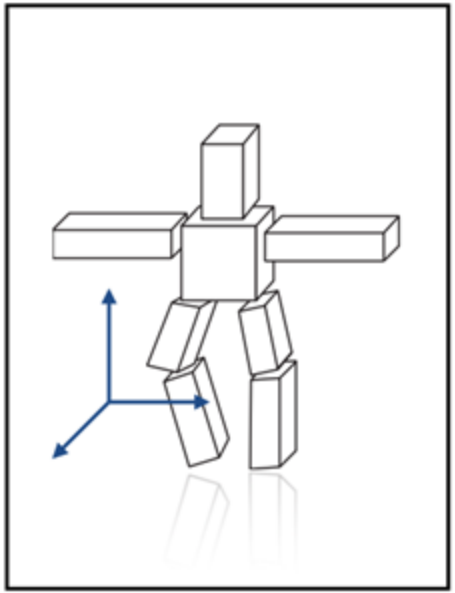
Coordenadas do Mundo

Recapitulando Transformações



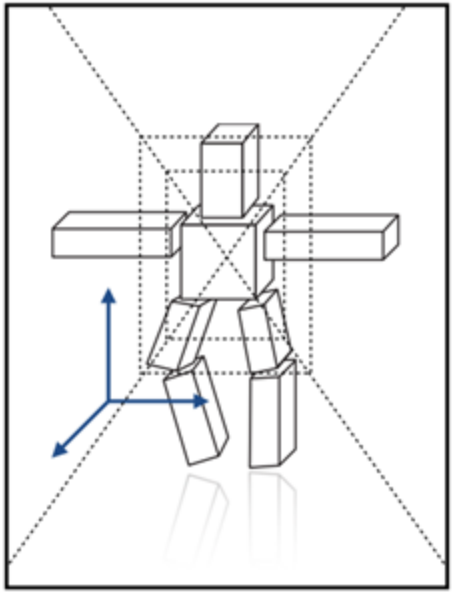
Coordenadas do Mundo

Viewing transform



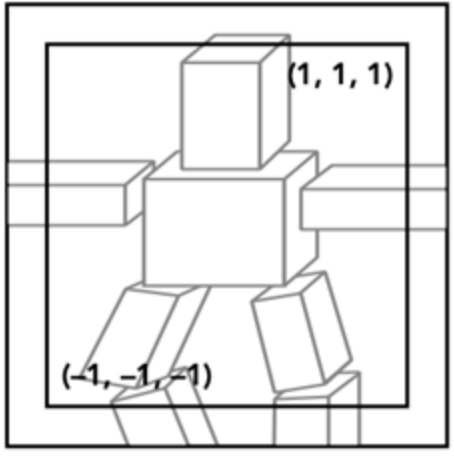
Coordenadas da Câmera

Recapitulando Transformações



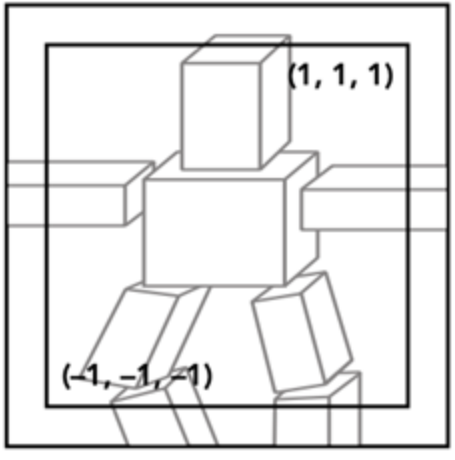
Coordenadas da Câmera

Perspective projection and homogeneous divide



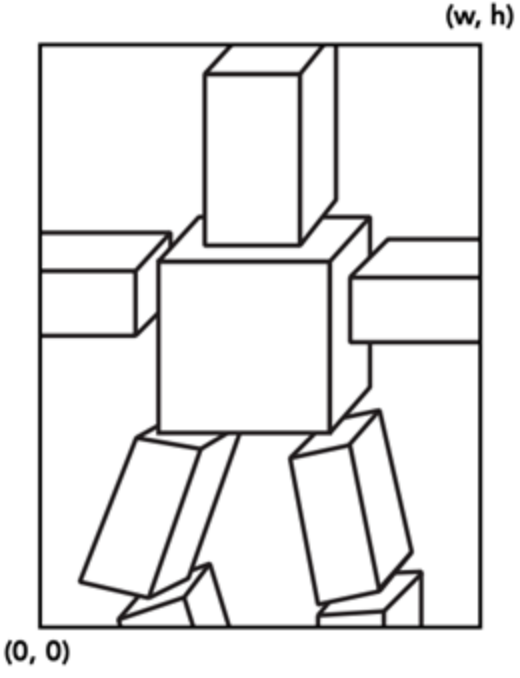
Coordenadas Normalizadas de Câmera "Device" (NDC)

Recapitulando Transformações



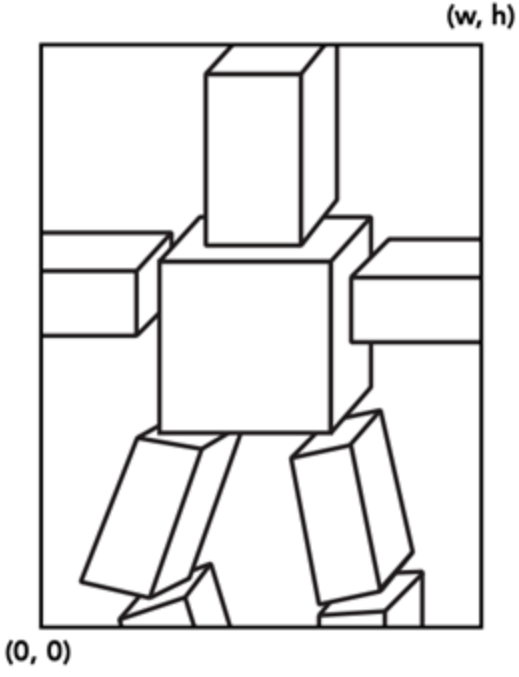
Coordenadas Normalizadas de Câmera "Device" (NDC)

Screen transform



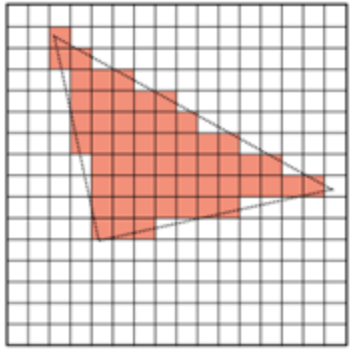
Coordenadas da Tela

Recapitulando Transformações



Coordenadas da Tela

Rasterização



Imagem

X3D Continuação

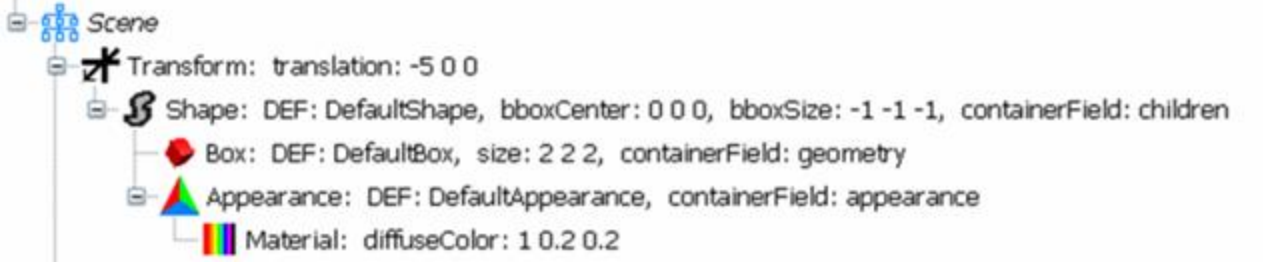
- Transform (mas somente um por vez)
- Viewpoint
- Coordinate
- TriangleSet
- IndexedTriangleStripSet
- TriangleStripSet
- Box

Transform

Transform : X3DGroupingNode {

MFNode	[in]	addChildren			[X3DChildNode]
MFNode	[in]	removeChildren			[X3DChildNode]
SFVec3f	[in,out]	center		0 0 0	(-∞,∞)
MFNode	**[in,out]**	**children**		**[]**	**[X3DChildNode]**
SFNode	[in,out]	metadata		NULL	[X3DMetadataObject]
SFRotation	**[in,out]**	**rotation**		**0 0 1 0**	**[-1,1] or (-∞,∞)**
SFVec3f	**[in,out]**	**scale**		**1 1 1**	**(0,∞)**
SFRotation	[in,out]	scaleOrientation	0 0 1 0	[-1,1] or (-∞,∞)	
SFVec3f	**[in,out]**	**translation**		**0 0 0**	**(-∞,∞)**
SFVec3f	[]	bboxCenter		0 0 0	(-∞,∞)
SFVec3f	[]	bboxSize		-1 -1 -1	[0,∞) or −1 −1 −1

}



https://www.web3d.org/documents/specifications/19775-1/V3.0/Part01/components/group.html#Transform

Transform

Ordem de execução das operações de Transformação quando mais de uma definida no mesmo nó Transform:

1. Escala
2. Rotação
3. Translação

```
<Transform scale="2 2 2" rotation="0 1 0 1.5708" translation="1 1 0">
```

=

```
<Transform translation="1 1 0">
        <Transform rotation="0 1 0 1.5708">
                <Transform scale="2 2 2">
```

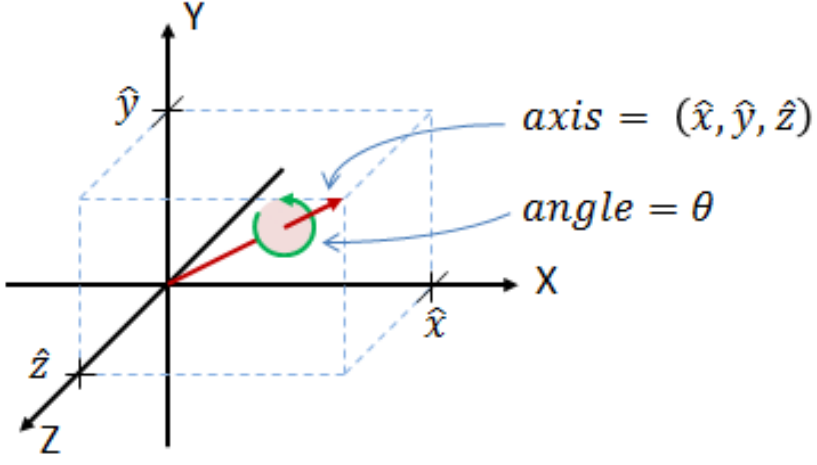
Obs: Não necessário implementar hierarquia de Transforms para essa fase do projeto

Em especial: SFRotation

O **SFRotation** é um vetor usado para definir rotações dos objetos.

Os três primeiros valores especificam um vetor de eixo normalizado sobre o qual a rotação ocorre, de modo que os três primeiros valores devem estar dentro do intervalo [-1 .. +1] para representar um vetor de unidade normalizado.

O quarto valor especifica a quantidade de rotação pela regra da mão direita em torno desse eixo em radianos.



Insper

Viewpoint

O nó **Viewpoint** define um ponto de vista que fornece uma câmera virtual em perspectiva da cena. O valor de **fieldOfView** representa o ângulo de visão mínimo em radianos em qualquer eixo de direção perpendicular à visão deste ponto de vista. O campo de visão deve ser maior que zero e menor que π.

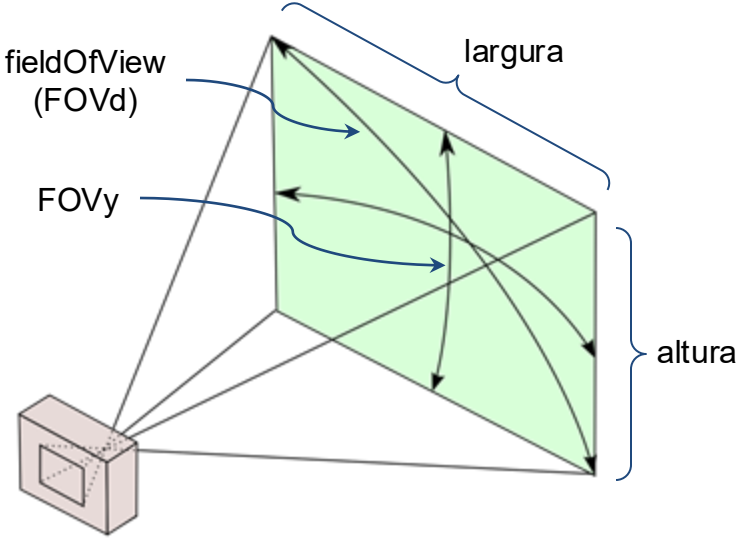
Viewpoint : X3DViewpointNode {				
SFBool	[in]	set_bind		
SFVec3f	[in,out]	centerOfRotation	0 0 0	(-∞,∞)
SFString	[in,out]	description	""	
SFFloat	**[in,out]**	**fieldOfView**	**π/4**	**(0,π)**
SFBool	[in,out]	jump	TRUE	
SFNode	[in,out]	metadata	NULL	[X3DMetadataObject]
SFRotation	**[in,out]**	**orientation**	**0 0 1 0**	**[-1,1],(-∞,∞)**
SFVec3f	**[in,out]**	**position**	**0 0 10**	**(-∞,∞)**
SFBool	[in,out]	retainUserOffsets	FALSE	
SFTime	[out]	bindTime		
SFBool	[out]	isBound		
}				

https://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/navigation.html#X3DViewpointNode
https://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/navigation.html#Viewpoint

FOVY (opcional para projeto)

No X3D o fieldOfView é determinado pelo ângulo de visão mínimo de qualquer eixo de direção perpendicular à visão.

$$\frac{\text{display width}}{\text{display height}} = \frac{\tan\left(\text{FOV}_{horizontal}/2\right)}{\tan\left(\text{FOV}_{vertical}/2\right)}$$



$$\text{FOV}y = 2 \cdot \arctan\left(\tan\left(\frac{FOVd}{2}\right) \cdot \frac{\text{Altura}}{\sqrt{\text{Altura}^2 + \text{Largura}^2}}\right)$$

Insper

Coordinate

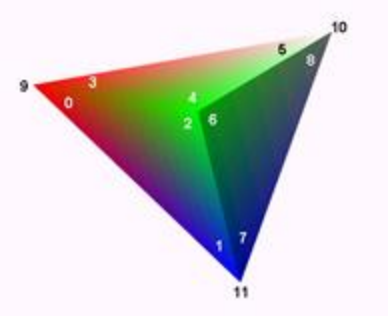
O nó **Coordinate** define um conjunto de coordenadas 3D a serem usadas no campo de coordenadas de nós de geometria baseada em vértices.

```
Coordinate : X3DCoordinateNode {
  SFNode          [in,out]    metadata NULL     [X3DMetadataObject]
  MFVec3f         [in,out]    point    []       (-∞,∞)
}
```

https://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/rendering.html#Coordinate

TriangleSet

O nó **TriangleSet** representa uma geometria 3D que representa uma coleção de triângulos individuais. Cada triângulo é formado por um conjunto consecutivo de três vértices do nó **Coordinate**. Se o nó de coordenadas não contém um múltiplo de três valores de coordenadas, os vértices restantes devem ser ignorados.



TriangleSet : X3DComposedGeometryNode {

MFNode	[in,out]	attrib	[]	[X3DVertexAttributeNode]
SFNode	[in,out]	color	NULL	[X3DColorNode]
SFNode	**[in,out]**	**coord**	**NULL**	**[X3DCoordinateNode]**
SFNode	[in,out]	fogCoord	NULL	[FogCoordinate]
SFNode	[in,out]	metadata	NULL	[X3DMetadataObject]
SFNode	[in,out]	normal	NULL	[X3DNormalNode]
SFNode	[in,out]	texCoord	NULL	[X3DTextureCoordinateNode]
SFBool	[]	ccw	TRUE	
SFBool	[]	colorPerVertex	TRUE	
SFBool	[]	normalPerVertex	TRUE	
SFBool	[]	solid	TRUE	

}

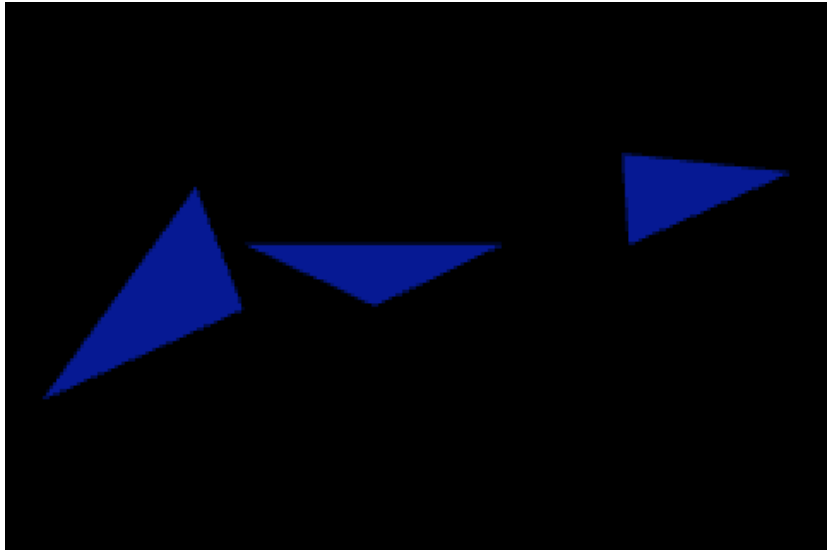
https://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/rendering.html#TriangleSet

Exemplo de TriangleSet

```
<X3D>
 <Scene>
  <NavigationInfo headlight="false"/>
  <Viewpoint orientation='0 -1 0 0.05' position='0.13 2.51 11.24'/>
  <Transform>
   <Shape>
    <Appearance>
     <Material emissiveColor='0 .1 0.6'/>
    </Appearance>
    <TriangleSet>
     <Coordinate point='
                -4  1  3
                -2  2  1
                -3  4  0

                 0  2  0
                 2  3  1
                -2  3  1

                 5  5 -2
                 4  3  1
                 6  4  2
               '/>
    </TriangleSet>
   </Shape>
  </Transform>
 </Scene>
</X3D>
```



https://x3dgraphics.com/examples/X3dForWebAuthors/Chapter13GeometryTrianglesQuadrilaterals/TriangleSetExampleIndex.html

Projeto 1 : Segunda Parte

Continuar renderizador:

Tarefa 2: fazer as transformadas no modelo
Tarefa 3: fazer as transformadas de câmera
Tarefa 4: fazer as transformadas de projeção perspectiva
Tarefa 5: fazer as transformações para coordenadas da tela

TAREFA: Desenhar triângulos (Triangle Set)

Base:

https://github.com/lpsoares/Renderizador

Data de Entrega (máximo):

7/9/2024 às 23:59 (DOMINGO) [Supondo mesmo repositório da fase 1.]

Detalhes:

Página da Disciplina (https://lpsoares.github.io/ComputacaoGrafica/)

Insper

Computação Gráfica

Luciano Soares
<lpsoares@insper.edu.br>

Fabio Orfali
<fabioO1@insper.edu.br>