



Computação
Gráfica

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

```
class ponto:  
    def __init__(self, x, y, z):  
        self.x = x  
        self.y = y  
        self.z = z
```

```
class aresta:  
    def __init__(self, pontoA, pontoB):  
        self.pontoA = pontoA  
        self.pontoB = pontoB
```

```
class face:  
    def __init__(self, vertices, arestas):  
        self.vertices = vertices  
        self.arestas = arestas
```

```
class objeto:  
    def __init__(self, vertices, arestas, faces):  
        self.vertices = vertices  
        self.arestas = arestas  
        self.faces = faces
```

Cada objeto é representado por uma instância da classe chamada objeto, que em sua inicialização recebe uma lista de pontos (vértices do objeto), uma lista de arestas (linhas que compõem o objeto) e uma lista de faces (conjunto de vértices e arestas que compõem a face)

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

```
#Vértices Frontais
pontoA = ponto(0, 0, 0)
pontoB = ponto(0, 100, 0)
pontoC = ponto(100, 100, 0)
pontoD = ponto(100, 0, 0)

#Vértices Posteriores
pontoW = ponto(0, 0, 100)
pontoX = ponto(0, 100, 100)
pontoY = ponto(100, 100, 100)
pontoZ = ponto(100, 0, 100)
```

Cada ponto do objeto é representado por uma instância da classe chamada ponto, que em sua inicialização recebe os valores de x, y e z

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

```
# Arestas Frontais
arestaAB = aresta(pontoA, pontoB)
arestaBC = aresta(pontoB, pontoC)
arestaCD = aresta(pontoC, pontoD)
arestaDA = aresta(pontoD, pontoA)

# Arestas Posteriores
arestaWX = aresta(pontoW, pontoX)
arestaXY = aresta(pontoX, pontoY)
arestaYZ = aresta(pontoY, pontoZ)
arestaZW = aresta(pontoZ, pontoW)

# Arestas Laterais Esquerdas
arestaAW = aresta(pontoA, pontoW)
arestaBX = aresta(pontoB, pontoX)

# Arestas Laterais Direitas
arestaCY = aresta(pontoC, pontoY)
arestaDZ = aresta(pontoD, pontoZ)
```

Cada aresta do objeto é representada por uma instância da classe chamada aresta, que em sua inicialização recebe dois pontos que juntos compõem a aresta.

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

Cada face do objeto é representada por uma instância da classe chamada face, que em sua inicialização recebe uma lista de pontos e de arestas que compõem a face.

```
# Face Frontal
faceABCD = face([pontoA, pontoB, pontoC, pontoD], [arestaAB, arestaBC, arestaCD, arestaDA])

# Face Posterior
faceWXYZ = face([pontoW, pontoX, pontoY, pontoZ], [arestaWX, arestaXY, arestaYZ, arestaZW])

# Face Lateral Esquerda
faceABWX = face([pontoA, pontoB, pontoW, pontoX], [arestaAB, arestaWX, arestaAW, arestaBX])

# Face Lateral Direita
faceCDYZ = face([pontoC, pontoD, pontoY, pontoZ], [arestaCD, arestaYZ, arestaCY, arestaDZ])

# Face Superior
faceADWZ = face([pontoA, pontoD, pontoW, pontoZ], [arestaDA, arestaZW, arestaAW, arestaDZ])

# Face Inferior
faceBCXY = face([pontoB, pontoC, pontoX, pontoY], [arestaBC, arestaXY, arestaBX, arestaCY])
```

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

Os vértices, as arestas e as faces são colocados em listas, que então são passadas como parâmetro para a instância do objeto.

```
# Parâmetros do objeto Cubo
vertices = [pontoA, pontoB, pontoC, pontoD, pontoW, pontoX, pontoY, pontoZ]
arestas = [arestaAB, arestaBC, arestaCD, arestaDA, arestaWX, arestaXY, arestaYZ, arestaZW, arestaAW, arestaBX, arestaCY, arestaDZ]
faces = [faceABCD, faceWXYZ, faceABWX, faceCDYZ, faceADWZ, faceBCXY]

cubo = objeto(vertices, arestas, faces)
```

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

Na **Projeção Axiométrica Isométrica**, assim como na Dimétrica e na Trimétrica, é necessário realizar uma rotação em torno do eixo y seguida de uma rotação em torno do eixo x. A ordem de rotação importa, então é necessário segui-la para obter a transformação correta desejada. Na projeção isométrica, a rotação em y é de 45° e a rotação em x é de $35,26^\circ$.

$$\begin{vmatrix} \cos\beta & 0 & -\sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha & 0 \\ 0 & -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = M(\beta_{\text{em } y} \ \alpha_{\text{em } x})$$

$$\begin{vmatrix} \cos\beta & \sin\beta\sin\alpha & -\sin\beta\cos\alpha & 0 \\ 0 & \cos\alpha & \sin\alpha & 0 \\ \sin\alpha & -\sin\alpha\cos\beta & \cos\alpha\cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = M(\beta_{\text{em } y} \ \alpha_{\text{em } x})$$

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

Depois de realizar as rotações, multiplica-se a Matriz resultante pela matriz abaixo, que resulta em uma projeção em $z = 0$.

$$M'(\beta_{em\ y} \ \alpha_{em\ x}) = M(\beta_{em\ y} \ \alpha_{em\ x}) \cdot \begin{vmatrix} 1 & & & \\ & 1 & & \\ & & 0 & \\ & & & 1 \end{vmatrix}$$

$$M'(\beta_{em\ y} \ \alpha_{em\ x}) = \begin{vmatrix} \cos\beta & \sin\beta\sin\alpha & 0 & 0 \\ 0 & \cos\alpha & 0 & 0 \\ \sin\alpha & -\sin\alpha\cos\beta & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

As transformações anteriores resultarão nos vetores unitários de x e de y a seguir, e o vetor unitário z será obtido a partir da matriz sem a projeção $z = 0$:

$$U'_x = |1, 0, 0, 1| \cdot M'(\beta_{em\ y} \ \alpha_{em\ x}) = |\cos\beta, \ \text{sen}\beta\text{sen}\alpha, \ 0, \ 1,|$$

$$U'_y = |0, 1, 0, 1,| \cdot M'(\beta_{em\ y} \ \alpha_{em\ x}) = |0, \ \cos\alpha, \ 0, \ 1,|$$

$$U'_z = |0, 0, 1, 1|.M(\beta_{em\ y} \ \alpha_{em\ x}) = |\text{sen}\beta - \text{sen}\alpha, \cos\beta, \ 0, \ 1|$$

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

Pela projeção isométrica, possuir a mesma redução em x, em y e em z, os vetores transformados devem possuir a mesma norma:

$$|U'_x| = |U'_y| = |U'_z|$$

$$|U'_x| = |U'_y| \rightarrow \cos^2\beta + \sin^2\beta\sin^2\alpha = \cos^2\alpha$$

$$|U'_z| = |U'_y| \rightarrow \sin^2\beta + \sin^2\alpha\cos^2\beta = \cos^2\alpha$$

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

Manipulando as duas equações, tem-se que beta é igual a 45° e alfa é aproximadamente 35,26°. Com isso, a matriz de transformação para a projeção isométrica ficará da seguinte forma:

$$M(\text{isometrica}) = \begin{vmatrix} \sqrt{2}/2 & \sqrt{6}/6 & 0 & 0 \\ 0 & \sqrt{6}/3 & 0 & 0 \\ \sqrt{2}/2 & -\sqrt{6}/6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} 0.707 & 0.408 & 0 & 0 \\ 0 & 0.817 & 0 & 0 \\ 0.707 & -0.408 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

Voltando ao objeto, para representar o objeto 3D em uma perspectiva isométrica, considerando as transformações citadas anteriormente, teremos as seguintes estruturas de rotação:

```
def girarX(self, graus):  
    self.centrarNaOrigem()  
    rad = 2 * math.pi * graus/360  
    cos = math.cos(rad)  
    sen = math.sin(rad)  
    for vertice in self.vertices:  
        verticeZ = vertice.z * cos - vertice.y * sen  
        verticeY = vertice.z * sen + vertice.y * cos  
        vertice.z = verticeZ  
        vertice.y = verticeY  
    self.descentrarDaOrigem()
```

```
def girarY(self, graus):  
    self.centrarNaOrigem()  
    rad = 2 * math.pi * graus/360  
    cos = math.cos(rad)  
    sen = math.sin(rad)  
    for vertice in self.vertices:  
        verticeX = vertice.x * cos - vertice.z * sen  
        verticeZ = vertice.x * sen + vertice.z * cos  
        vertice.x = verticeX  
        vertice.z = verticeZ  
    self.descentrarDaOrigem()
```

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

Para mostrar o objeto na forma isométrico, utilizam-se os seguintes métodos:

```
def mostrarIsometrico(self):  
    obj = self  
    obj.girarY(45)  
    obj.girarX(35.26)  
    for aresta in obj.arestas:  
        pygame.draw.lines(screen,BLACK,False, aresta.get2D(), 1)  
    obj.girarX(-35.26)  
    obj.girarY(-45)
```

```
def mostrarIsometricoCentralizado(self):  
    obj = self  
    obj.girarY(45)  
    obj.girarX(35.26)  
    obj.transladar(590, 260, 0)  
    for aresta in obj.arestas:  
        pygame.draw.lines(screen,BLACK,False, aresta.get2D(), 1)  
    obj.transladar(-590, -260, 0)  
    obj.girarX(-35.26)  
    obj.girarY(-45)
```

```
def mostrarFacesPrincipaisIsometricoCentralizado(self):  
    obj = self  
    obj.girarY(45)  
    obj.girarX(35.26)  
    obj.transladar(590, 260, 0)  
  
    # Desenha face posterior  
    for aresta in obj.faces[1].arestas:  
        pygame.draw.lines(screen,BLACK,False, aresta.get2D(), 1)  
  
    # Desenha face lateral direita  
    for aresta in obj.faces[3].arestas:  
        pygame.draw.lines(screen,BLACK,False, aresta.get2D(), 1)  
  
    # Desenha face superior  
    for aresta in obj.faces[4].arestas:  
        pygame.draw.lines(screen,BLACK,False, aresta.get2D(), 1)  
    obj.transladar(-590, -260, 0)  
    obj.girarX(-35.26)  
    obj.girarY(-45)
```

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

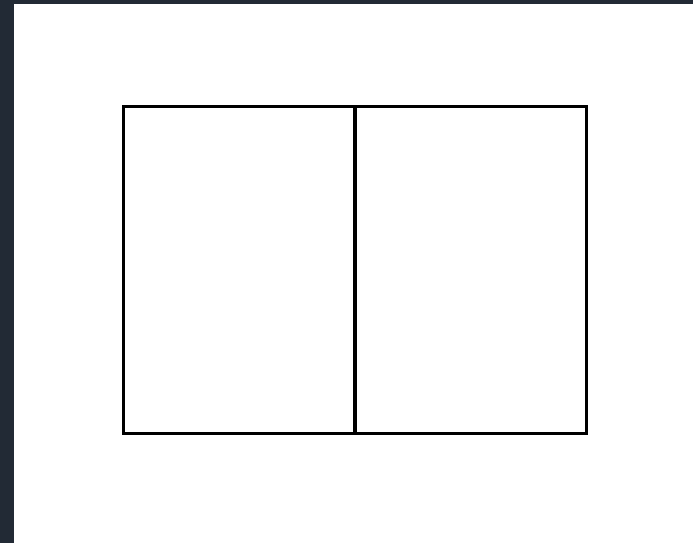
Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

A rotação de 45° em y resultará na seguinte representação isométrica:



Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

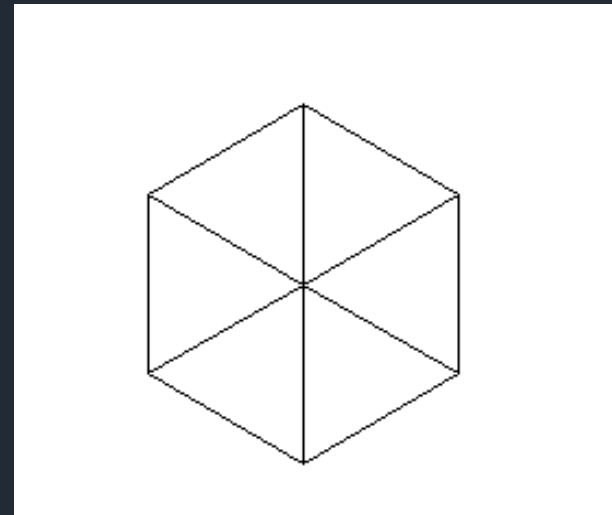
Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

A rotação de 45° em y, seguida da rotação de $35,26^\circ$ em x, resultará na seguinte representação isométrica:



Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

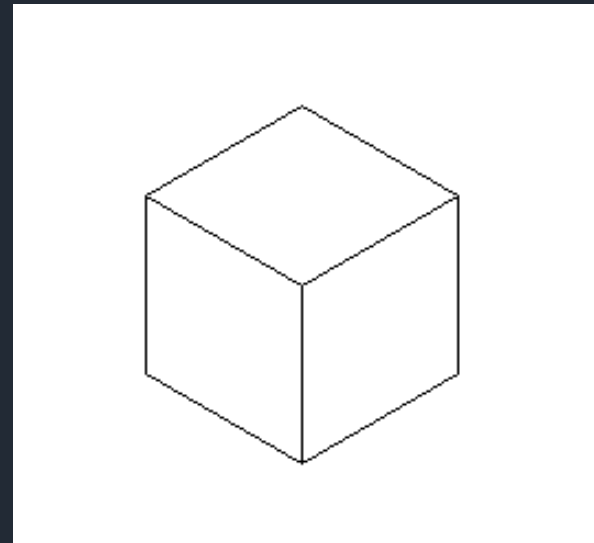
Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

A representação das faces principais da rotação de 45° em y, seguida da rotação de $35,26^\circ$ em x, resultará na seguinte imagem:



Computação Gráfica

Objeto 3D

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

Referências

Vídeo - Projeções.mp4, Aura Conci

<https://drive.google.com/file/d/1u8anY0Tn6NzJipldYc1UyNLKeTuJyWSu/view?usp=sharing>

PDF - Transformações chomog.pdf

<https://drive.google.com/file/d/1TGWMGuca4p3nZe8t7I6oacLMnXzBffop/view?usp=sharing>

Computação Gráfica

Objeto 3D

Classes da Estrutura

Declaração de pontos

Declaração de Arestas

Declaração de Faces

Criação do Objeto

Projeção Isométrica

Referências

Fim da Apresentação

Aluno

Pedro Henrique Mendes Pereira

Disciplina

Computação Gráfica - 2021.1

Professora

Aura Conci

Curso

Ciência da Computação

UFF – Niterói