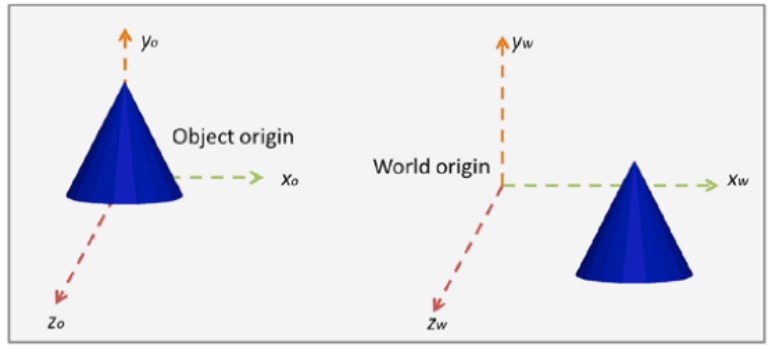
Computação Gráfica
2019/2020

Notas para a 3ª aula de WebGL

DI-FCUL

- Servidor HTTP (***python –m SimpleHTTPServer 8080***) a partir da pasta onde estão os ficheiros
- Browser (***http://localhost:8080***)

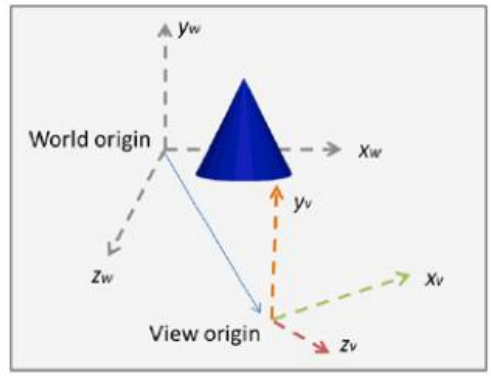
Model Transform



Object Coordinates → Model Transform → World Coordinates

The object has not moved. The model transform assigns coordinates that are shared by all objects: the world coordinates.

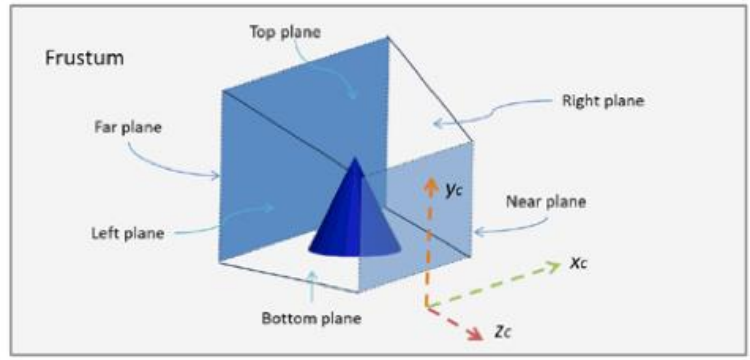
View Transform



World Coordinates → View Transform → View Coordinates

The view transform moves the origin of the world to the coordinates of the view. This is where our *camera* is located.

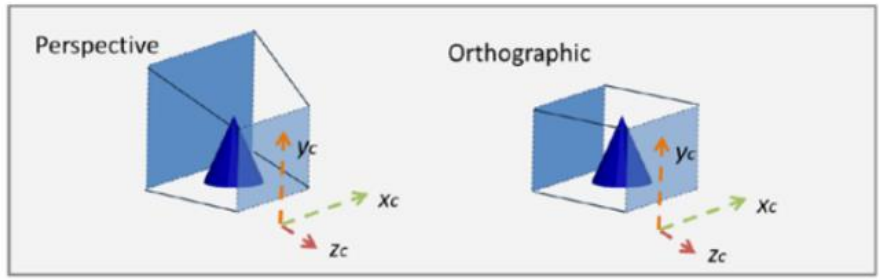
Projection Transform



View Coordinates → Projection Transform → Clip Coordinates

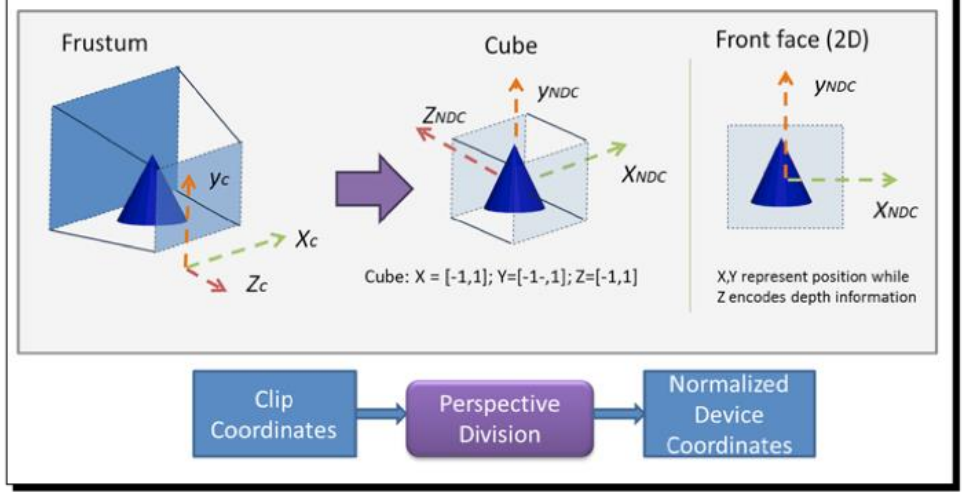
The frustum determines which objects or portion of objects will be *clipped out* and discarded.

Frustum shape



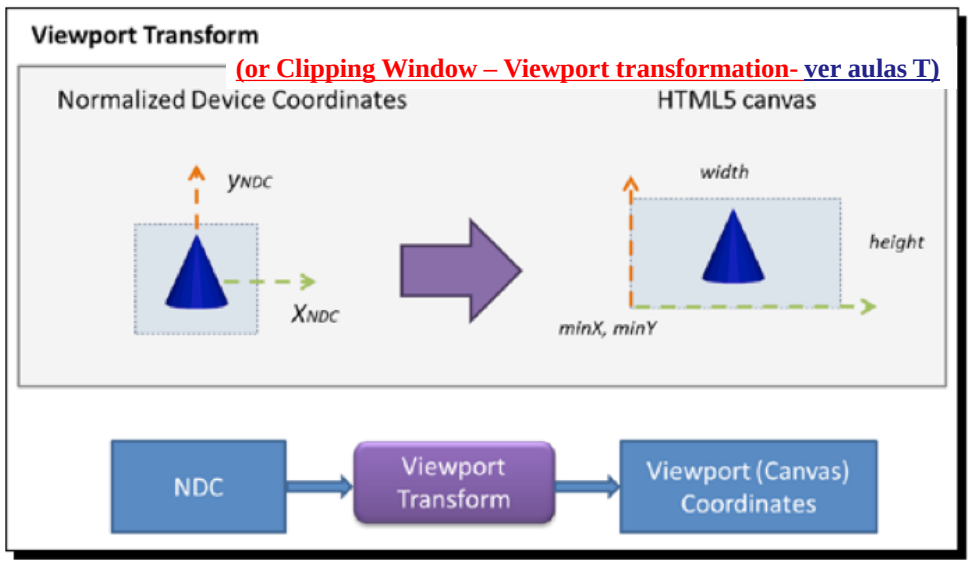
The extent and shape of the frustum determines how much of the 3D view space Is mapped to the screen and the type of 3D to 2D projection that takes place.

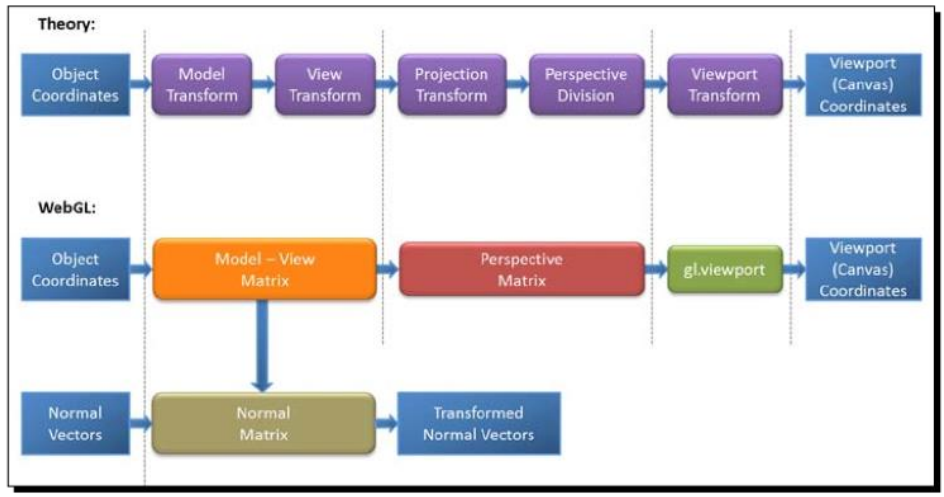
Normalized Device Coordinates



Viewport Transform

(or Clipping Window – Viewport transformation- ver aulas T)





Blender e JSON

Exportar um modelo Blender para o formato Json que pode ser usado pelo WebGL

Como o Blender não exporta diretamente para Json, temos de proceder em duas etapas:

Etapa 1: exportar no Blender para um dos formatos que esta ferramenta disponibiliza em File > Export.

Etapa 2: exportar deste formato para Json, usando outra ferramenta de software.

Nesta aula, na etapa 1, vamos exportar para o formato Wavefront (obj e mtl são formatos proprietários da empresa Wavefront). Na etapa 2 vamos usar um script em Python desenvolvido para converter para Json. Este script, que é fornecido com o material desta aula TP, está longe de ser perfeito mas permite obter o modelo definido em blender no formato Json.

Blender e JSON

Exportar um modelo Blender para o formato Json que pode ser usado pelo WebGL

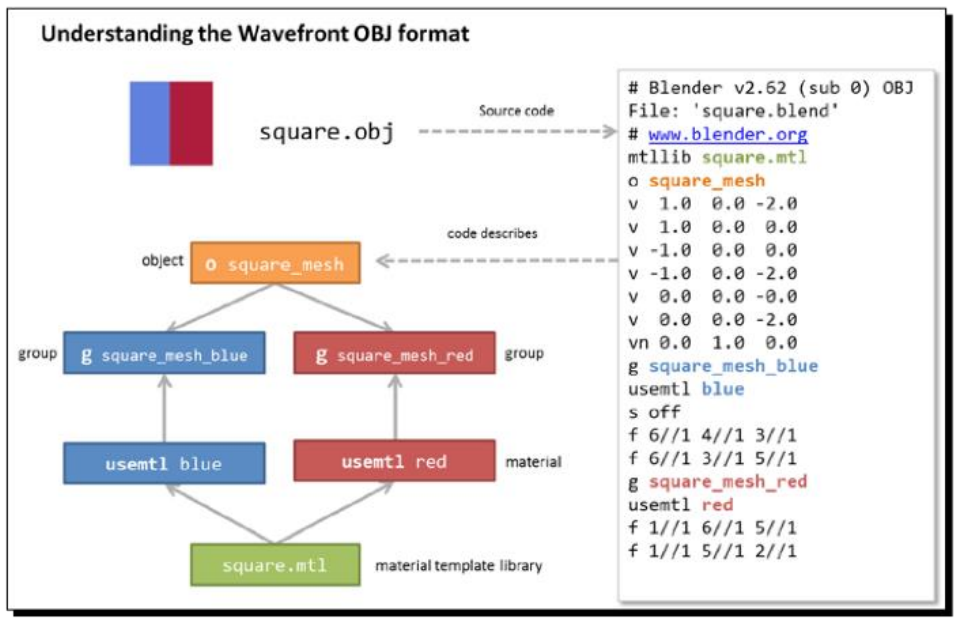
Etapa 1: exportar no Blender para o formato .obj

- **File > Export > Wavefront (.obj)**
- selecionar
  - **Apply Modifiers**
  - **Write Materials**
  - **Triangulate Faces**
  - **Objects as OBJ Objects**
  - **Material Groups**

– associar um nome

– fazer Export

O Blender gera dois ficheiros: um com extensão obj e outro com extensão mtl (*material library format*, formato também da Wavefront)



Parse do obj **para** json

Etapa 2: Usar o script python que converte de obj para Json

– python obj_parser.py myFile.obj myFile.mtl

como resultado são gerados os ficheiros json correspondentes a cada objecto da cena (part1.json, part2.json, etc.).

– O script apenas vai gerar a lista de vértices e índices, tudo o resto terá de ser adicionado manualmente.

Exercício

- Usando o Blender defina um modelo com 4 objectos de diferentes formas com dimensões de cerca de 5 unidades afastados entre si cerca de 40 unidades.
- Exporte o modelo para formato Wavefront (gera 2 ficheiros: .obj e .mtl)
- Gere os modelos .json usando o script
- Edite os ficheiros adicionado cor aos objectos, por exemplo, colocando uma linha com

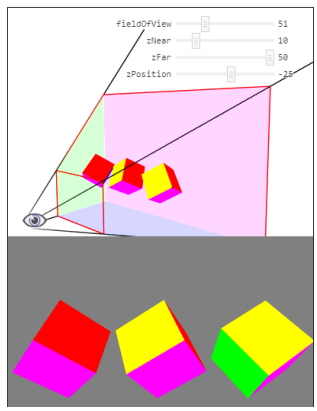
  “diffuse” : [1.00,1.00, 0.00,1.00]

Exercício

- Abrir o ficheiro *ch4_CameraTypes.html*
- Carregar os modelos para a cena corrigindo a função load().
- Manipule a matriz de projeção na função updateTransforms()

  mat4.perspective(fovy, aspect_ratio, near, far, pMatrix);
  - Que acontece quando colocamos o fovy = 140 ?
  - Que acontece quando colocamos o fovy = 3 ?
  - Mas em ambos os casos a distância à origem foi 50, que justifica isto ?
  - Coloque far = 20, que observa? Porquê ?
  - Coloque near=40 e far= 60, que mudou na imagem ?
  - Varie os valores de aspect_ratio.

https://webglfundamentals.org/webgl/frustum-diagram.html



Usem este exemplo interativo para compreender melhor como se define o volume de visualização (frustrum) numa projeção perspetiva.

Exercício

- Mude para uma projeção ortogonal,
  mat4.ortho(-c_width, c_width, -c_height, c_height, near, far, pMatrix)
  - Que acontece com a representação dos objectos? Que informação foi perdida?
  - Modifique as dimensões do paralelepípedo de projeção, o que acontece ao variar a posição da câmara?

Exercício

- Mude para uma projeção ortogonal,
  mat4.ortho(-c_width, c_width, -c_height, c_height, near, far, pMatrix)
  - Que acontece com a representação dos objectos? Que informação foi perdida?
  - Modifique as dimensões do paralelepípedo de projeção, o que acontece ao variar a posição da câmara?

- EXTRA: Modifique a interface de forma a poder fazer a câmara rodar em torno de um dos seus eixos (correspondente ao eixo zz da câmara- Roll).