Computação Gráfica

Aula 08 (parte 1) – *Model*

Prof. Jean R. Ponciano

Motivação

Jupyter Notebook

Carregando Modelos - Wavefront

Motivação

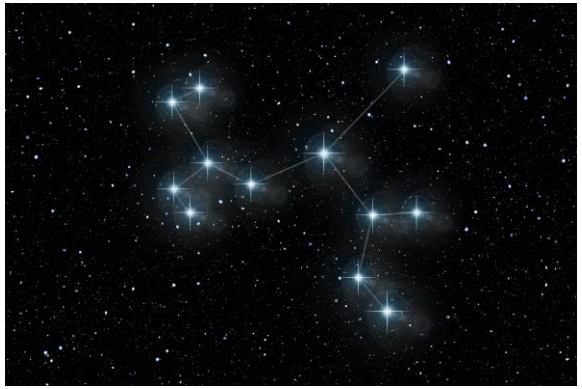
- Até agora, vimos:
  - Primitivas para desenhar os objetos/modelos
  - Transformações geométricas 2D e 3D
  - Textura e Importação de modelos

- Tudo que fizemos está representado em um **espaço de coordenadas local**

  - Só que isso não é suficiente!

Por que mudanças de coordenadas?

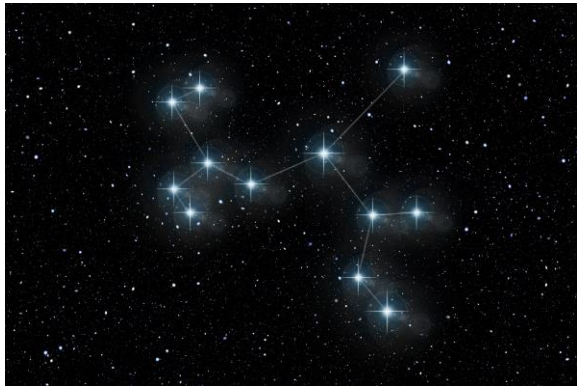
- Cenas e objetos do mundo real habitam um sistema de coordenadas inerente ao seu domínio, com suas características e grandezas.



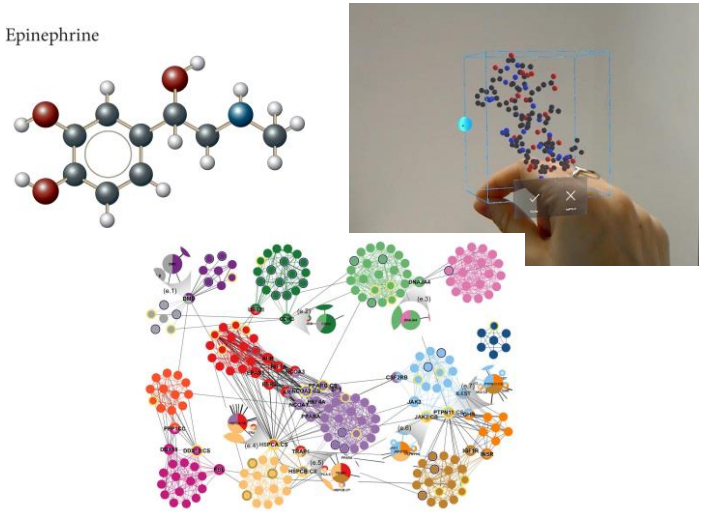
- Constelações
- Telescópios
- Medição em anos-luz

Por que mudanças de coordenadas?

- Cenas e objetos do mundo real habitam um sistema de coordenadas inerente ao seu domínio, com suas características e grandezas.



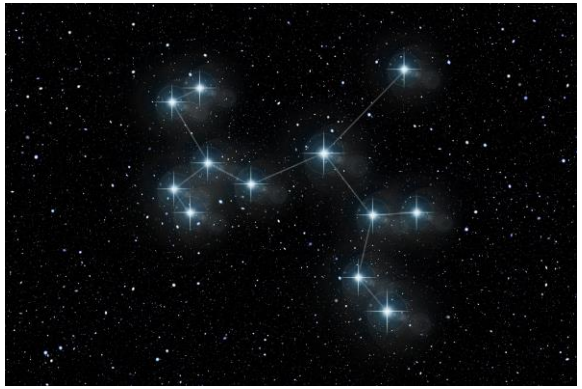
- Constelações
- Telescópios
- Medição em anos-luz



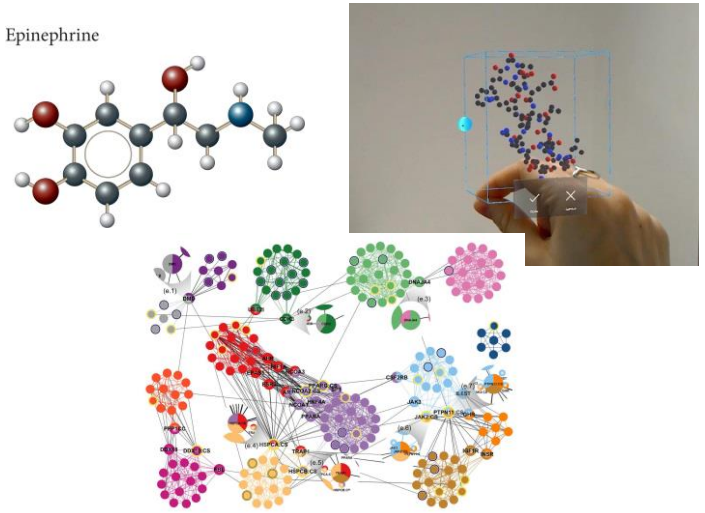
- Ambientes moleculares
- Microscópios
- Medição em microns (por ex)

Por que mudanças de coordenadas?

- Cenas e objetos do mundo real habitam um sistema de coordenadas inerente ao seu domínio, com suas características e grandezas.



- Constelações
- Telescópios
- Medição em anos-luz

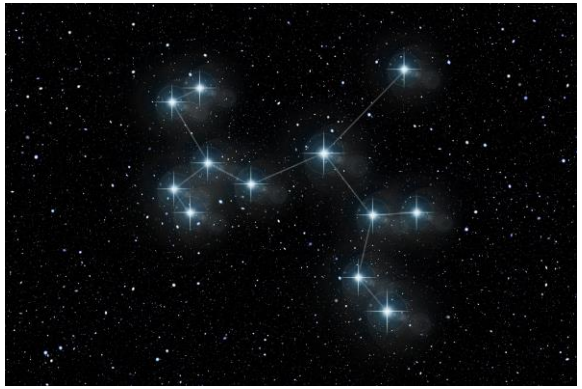


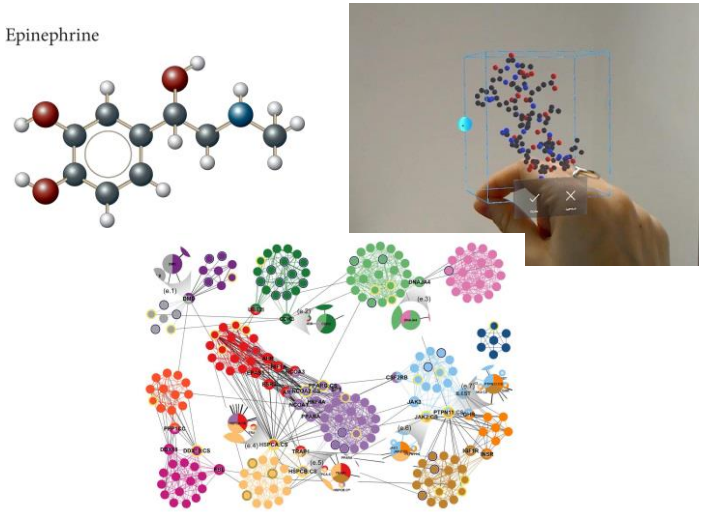
- Ambientes moleculares
- Microscópios
- Medição em microns (por ex)

- Cena na praia
- Câmeras fotográficas
- Medição em metros ou centímetros

Por que mudanças de coordenadas?

- Cenas e objetos do mundo real habitam um sistema de coordenadas inerente ao seu domínio, com suas características e grandezas.





Em qualquer situação, a apresentação no sistema gráfico exige **alterar as coordenadas** dos objetos para as coordenadas usadas pelo sistema de exibição/visualização. E isso é feito para cada objeto da cena.

Espaços de Coordenadas

- Até a exibição, vários espaços de coordenadas são necessários:

- **Espaço Local:**
  - É onde cada objeto foi modelado ("**minimundo**" do objeto).
  - Espaço definido pelo sistema de coordenadas do objeto

  - É preciso agora posicionar diferentes objetos em uma mesma cena, isto é, em um mesmo "**mundo**".

Espaços de Coordenadas

- Até a exibição, vários espaços de coordenadas são necessários:

- **Espaço Mundo:**
  - É onde os objetos estão posicionados de forma relativa uns aos outros, na mesma cena, no mesmo “mundo”.
  - Sistema de coordenadas do mundo.

  - Como posicioná-los assim? Transformações geométricas (escala, translação, rotação...)!

Espaços de Coordenadas

- **Espaço Visão:**
  - Transformação nos vértices (mundo) para que os objetos sejam visualizados a partir de um ponto de referência (observador ou câmera)
  - Sistema de coordenadas da câmera.

Espaços de Coordenadas

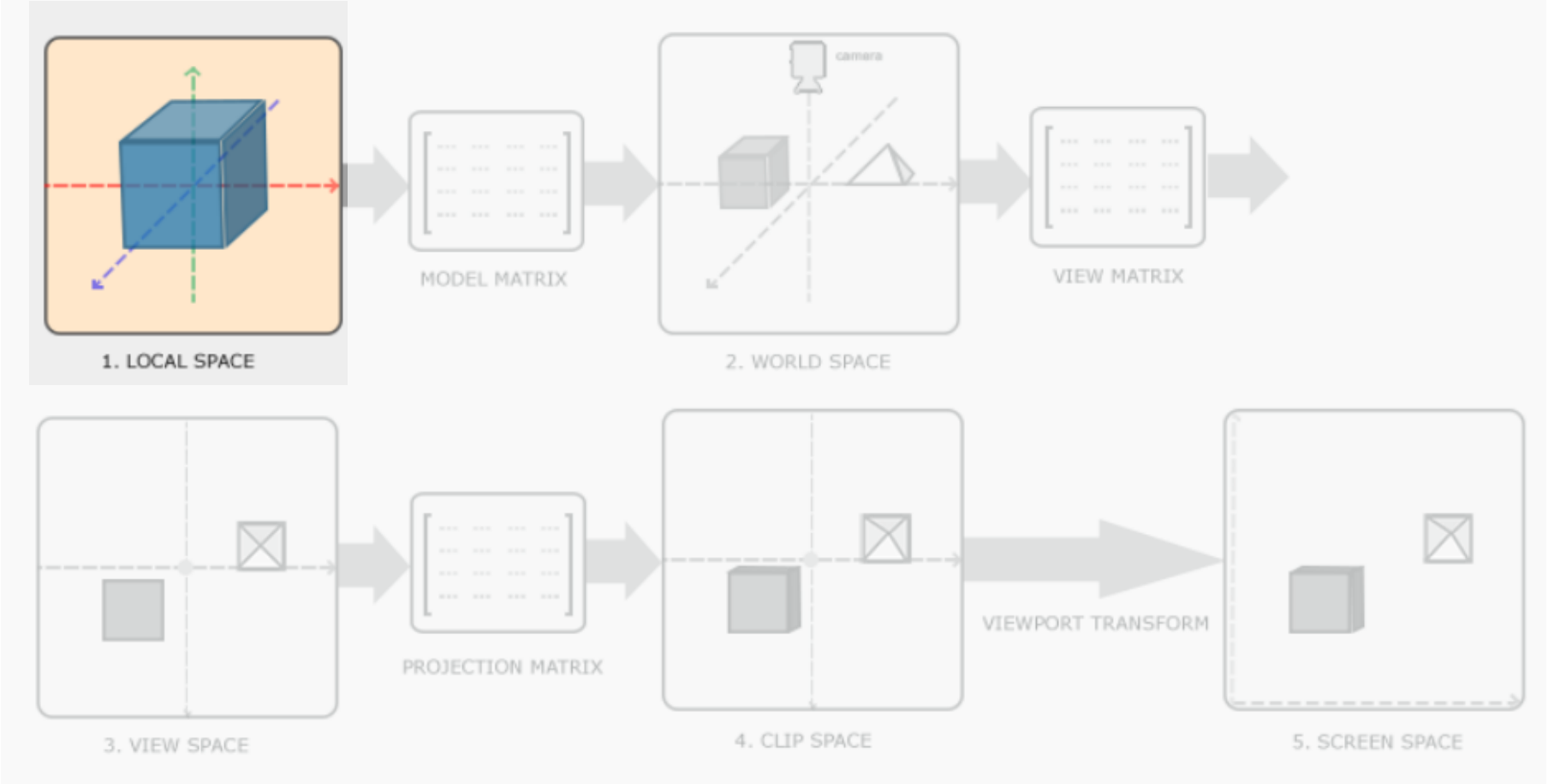
- **Espaço Visão:**
  - Transformação nos vértices (mundo) para que os objetos sejam visualizados a partir de um ponto de referência (observador ou câmera)
  - Sistema de coordenadas da câmera.

- **Espaço Clip:**
  - Transformação nos vértices (visão) para determinar o que de fato será exibido na tela.

- **Espaço de tela:** Coordenadas de pixel.

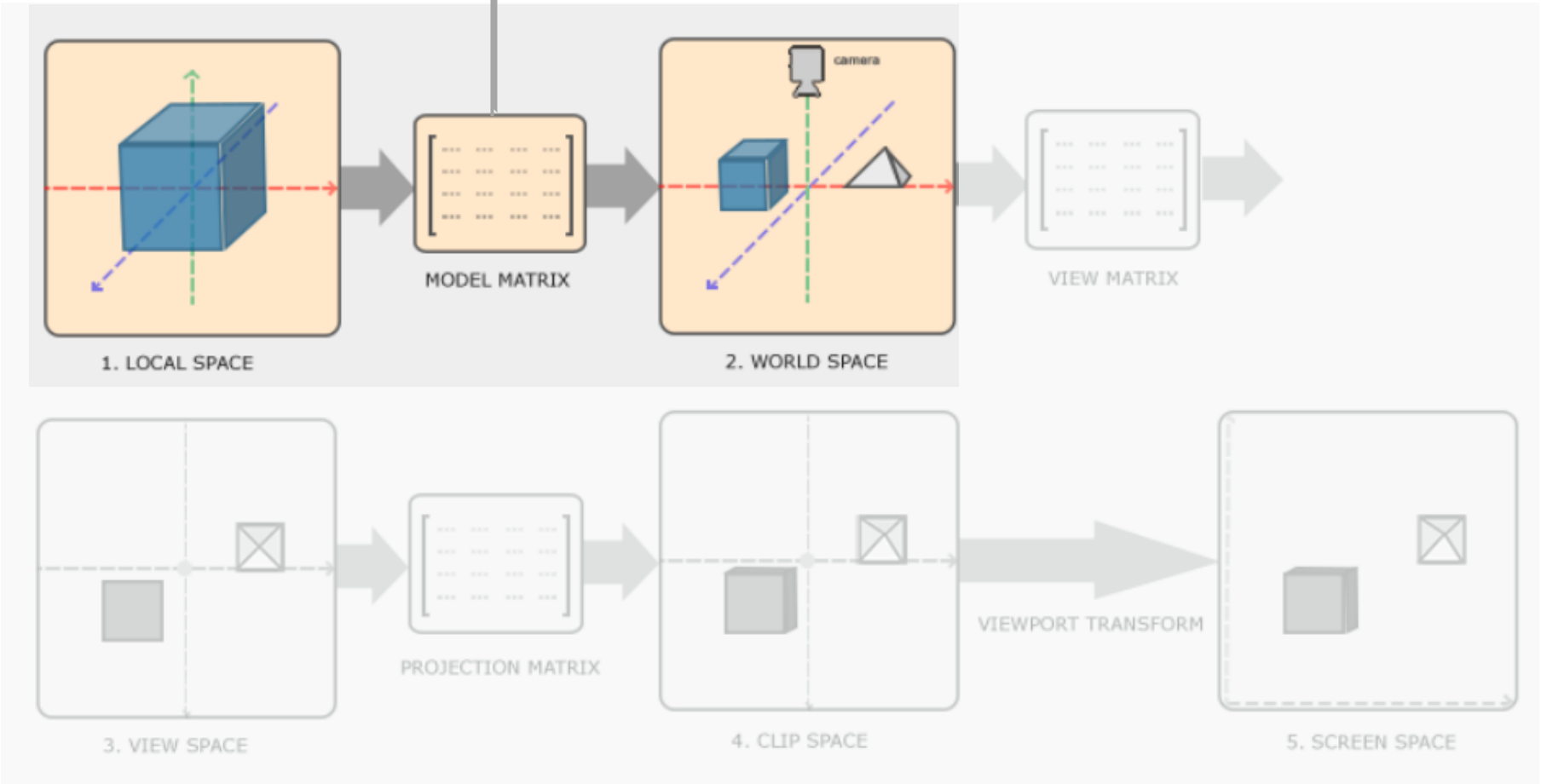
Pipeline de transformações

Coordenadas iniciais dos vértices (objetos)



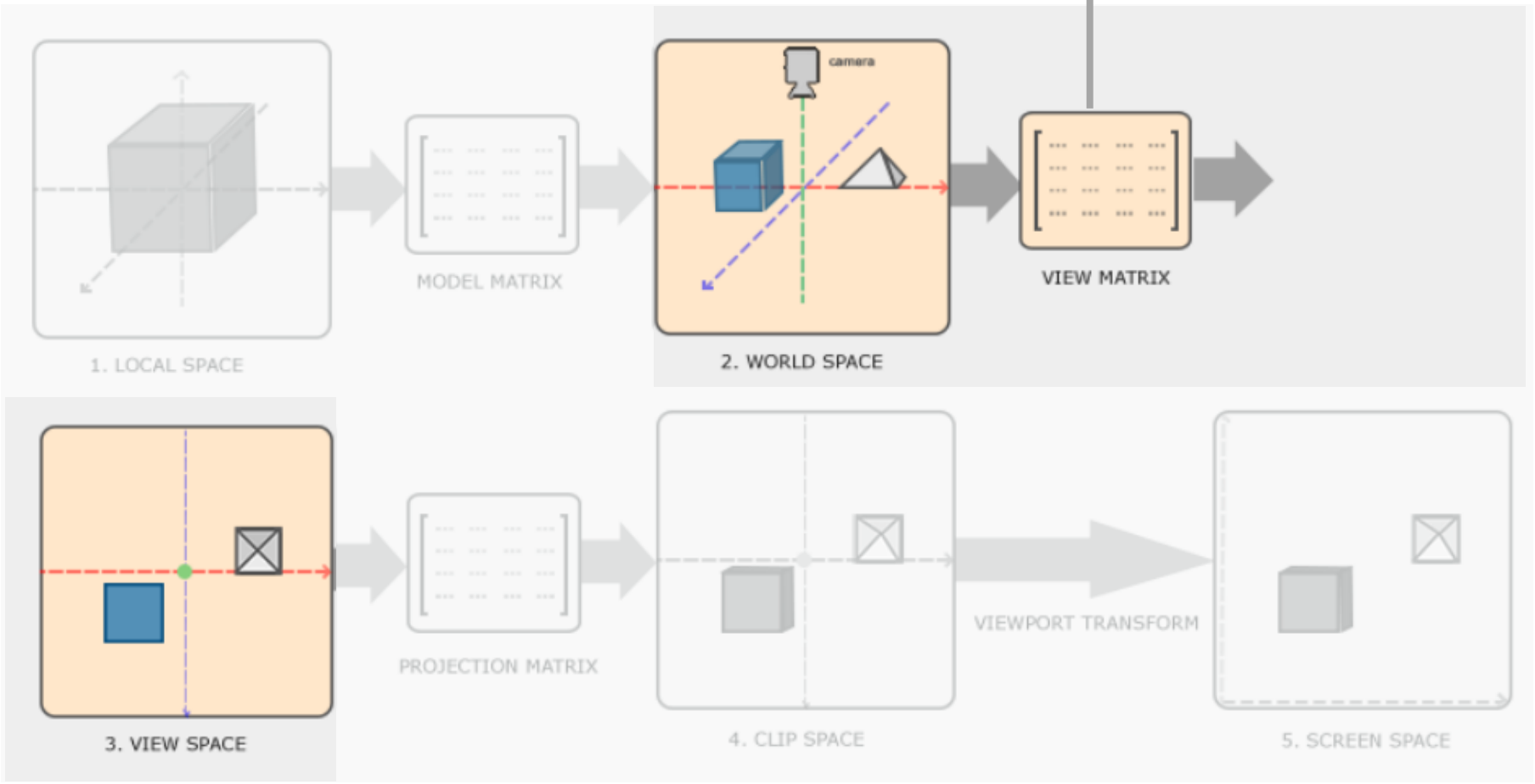
Pipeline de transformações

Transformações geométricas



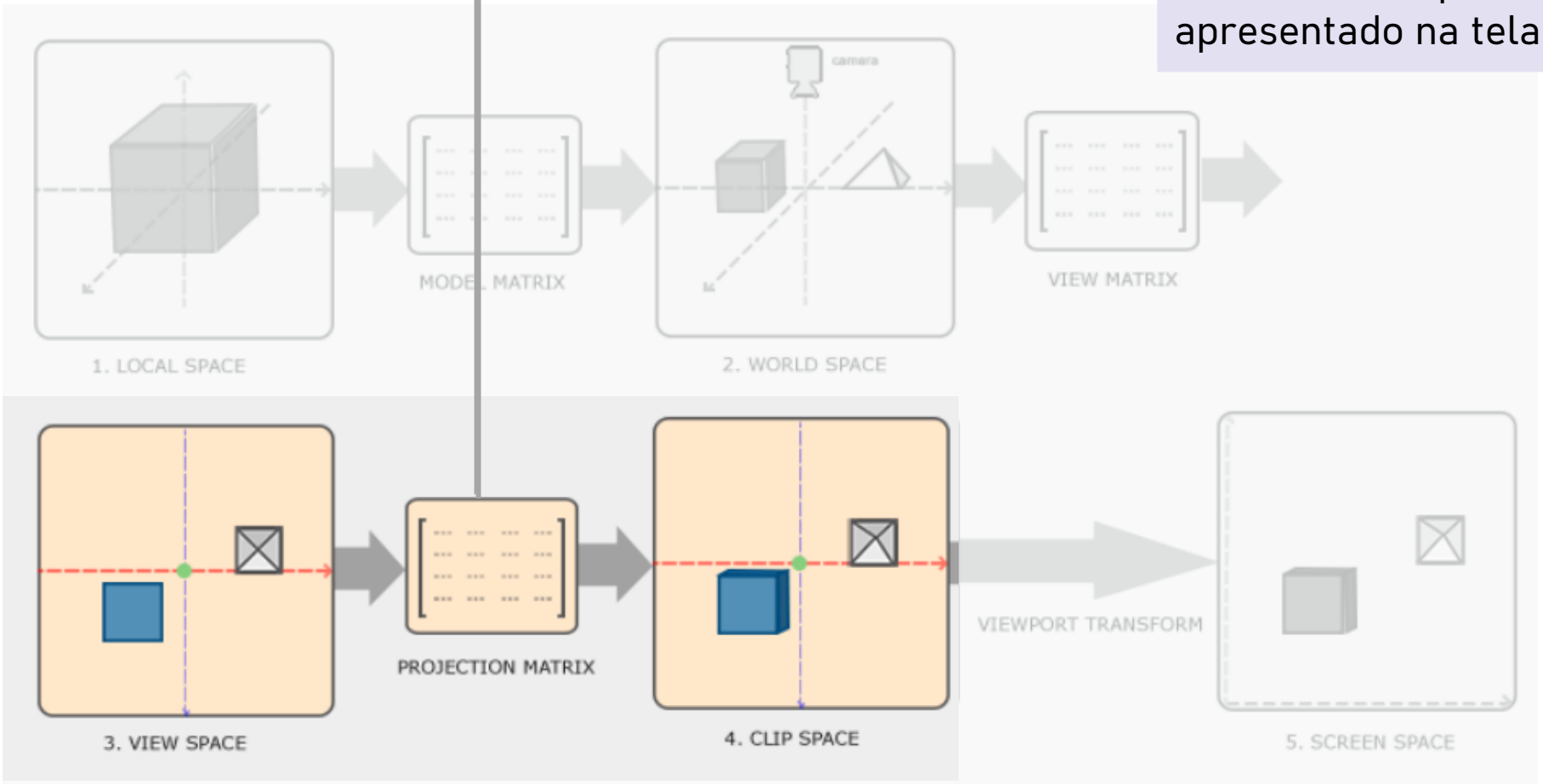
Pipeline de transformações

Transformação para visualizar os objetos a partir da câmera



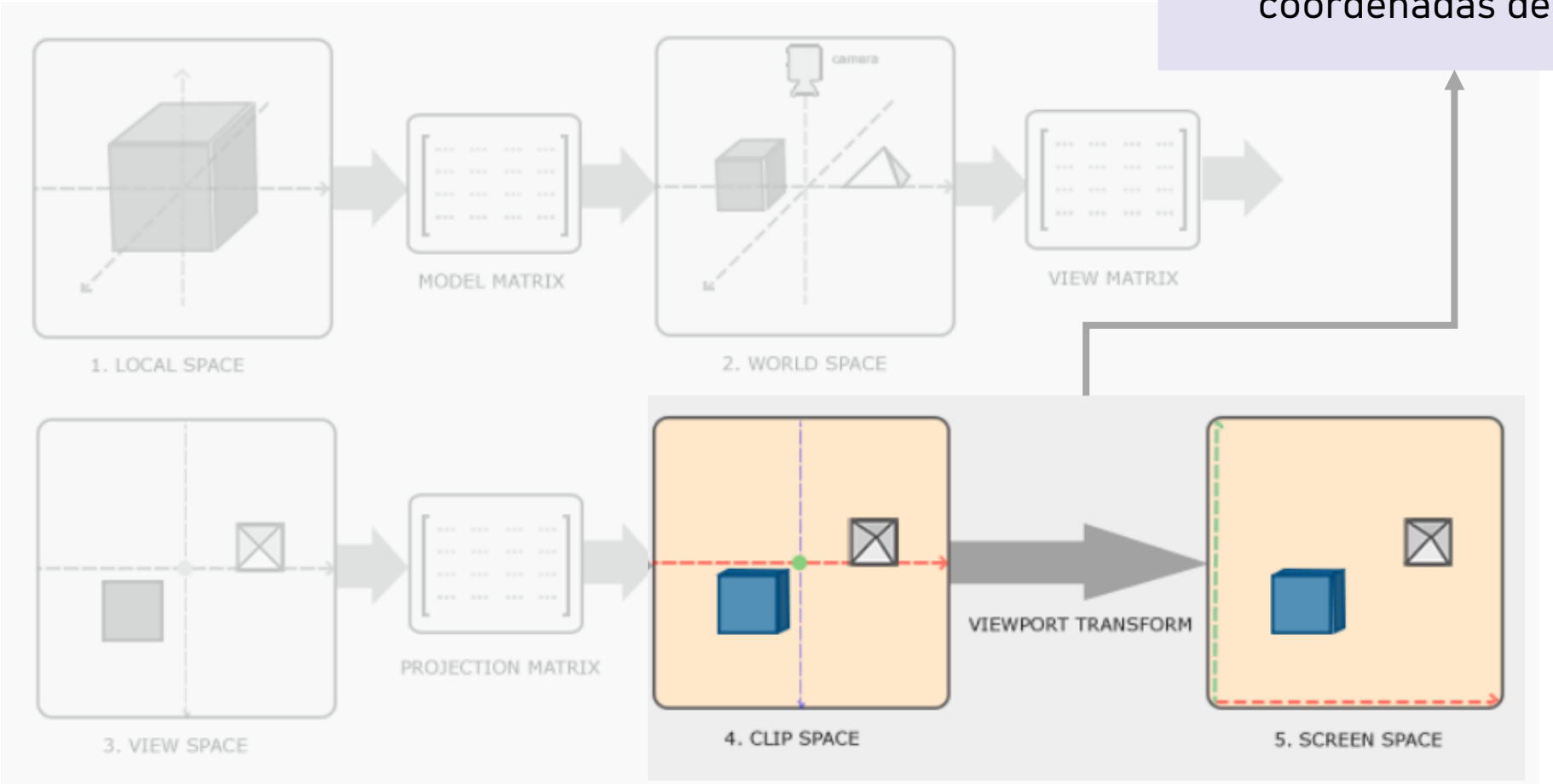
Pipeline de transformações

Transformação para determinar o que será apresentado na tela



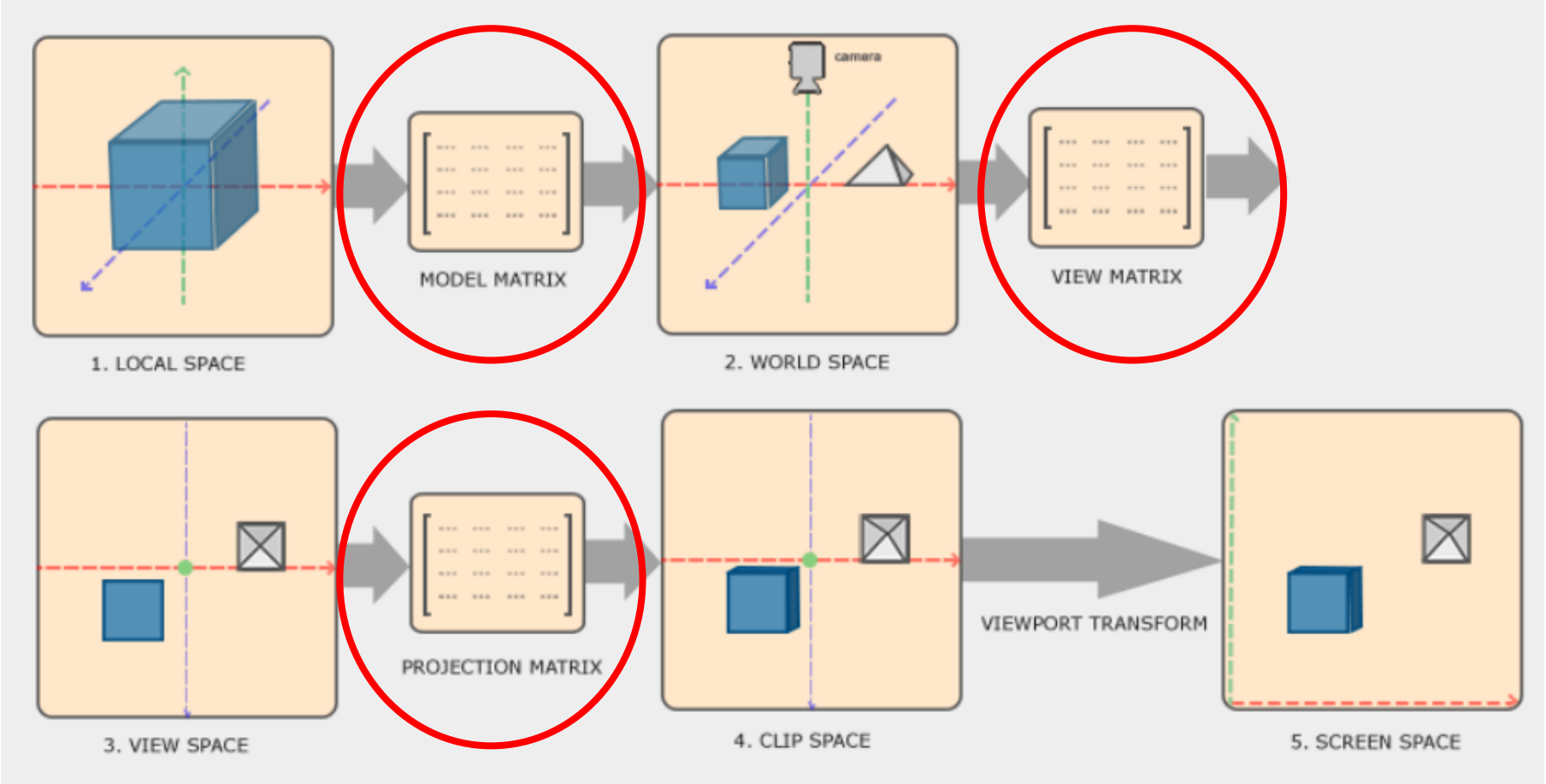
Pipeline de transformações

Mapeamento para coordenadas de pixel

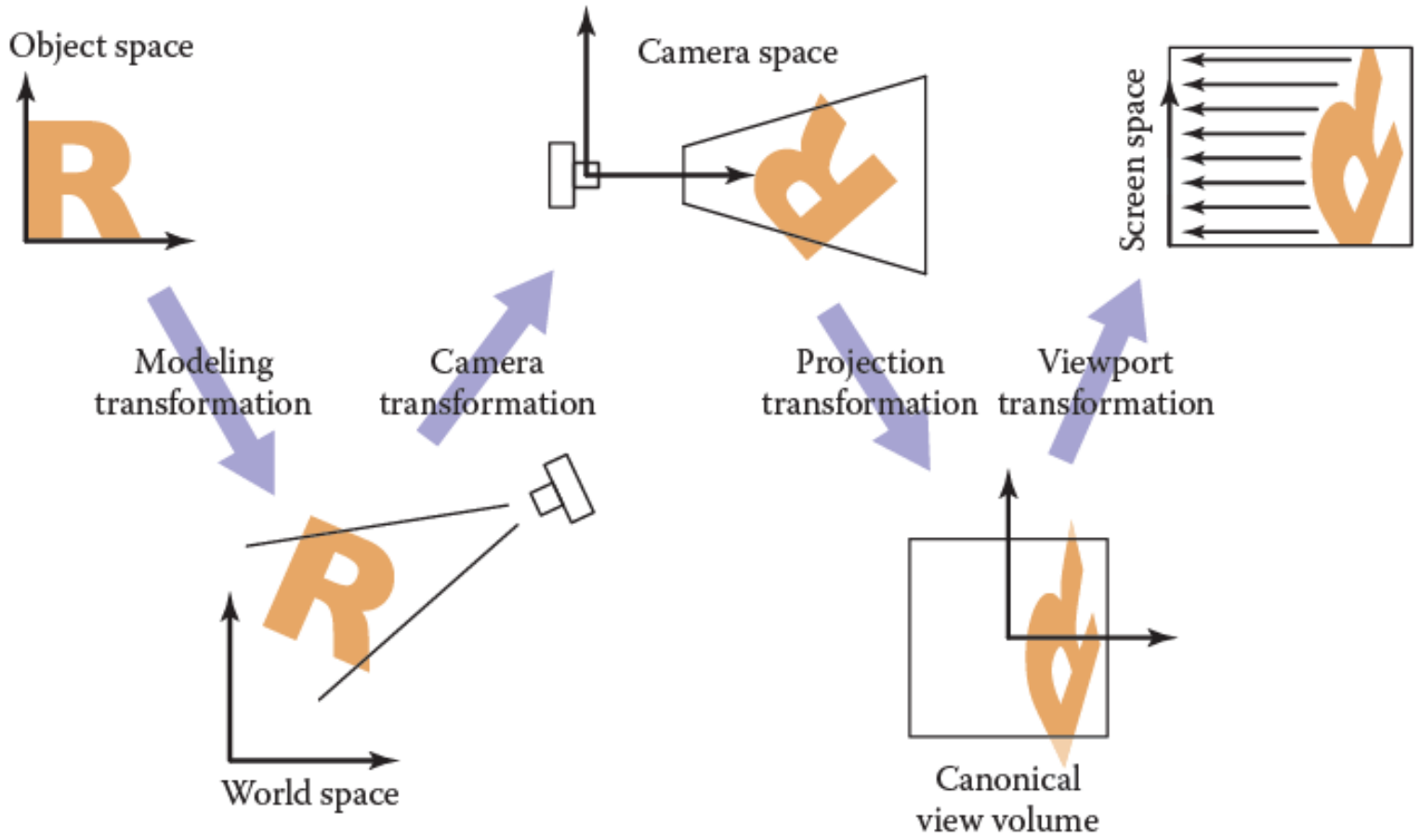


Pipeline de transformações

P' = Projection x View x Model x P



Pipeline de transformações



Fonte da imagem: Fundamentals of Computer Graphics (4ª edição), página 141.

Pipeline de transformações

- No nosso shader de vértices...

ANTES:

```
attribute vec3 position;
uniform mat4 mat_transformation;
void main(){
    gl_Position = mat_transformation * vec4(position,1.0);
}
```

DEPOIS:

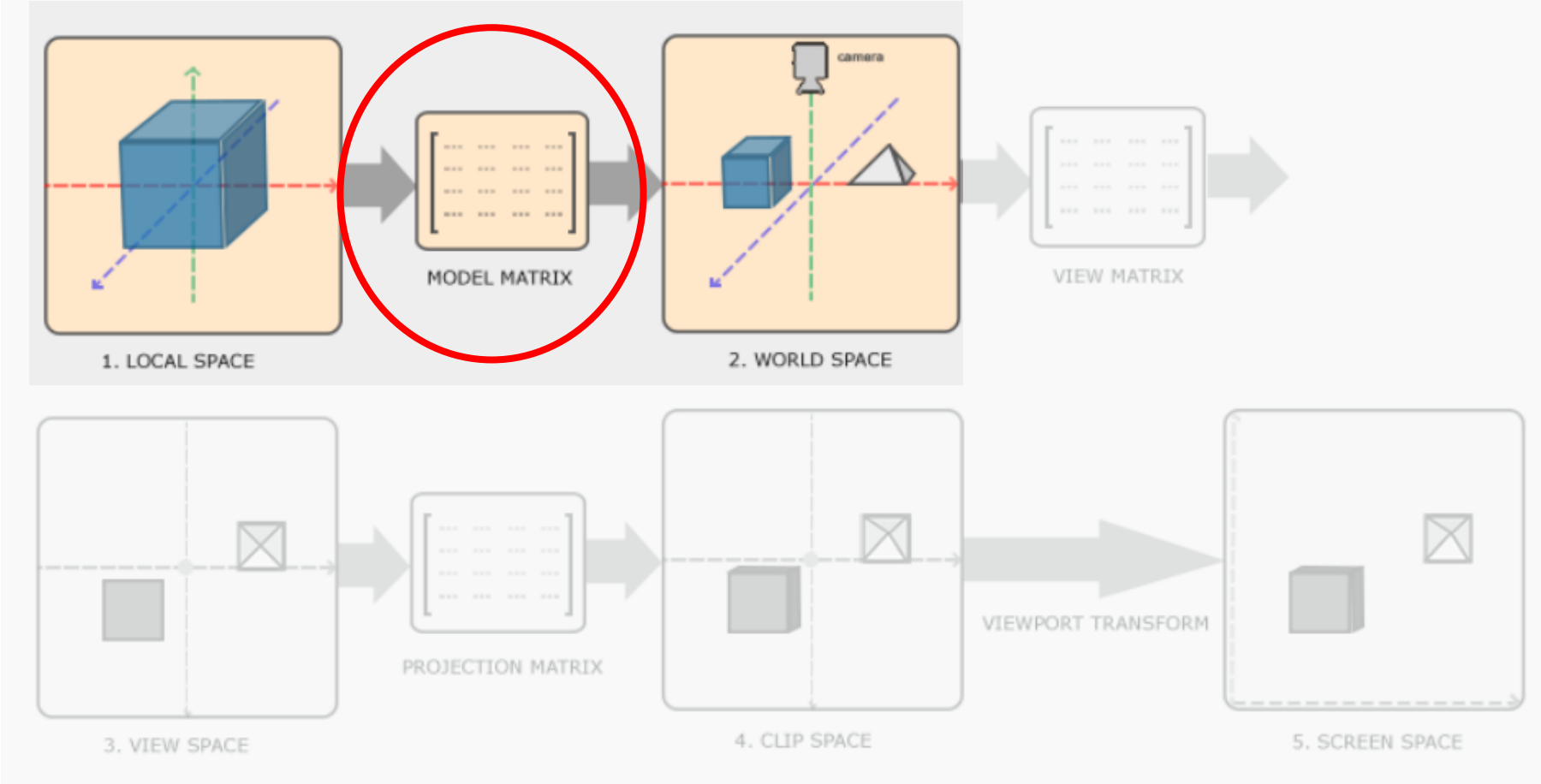
```
attribute vec3 position;

uniform mat4 model;
uniform mat4 view;
uniform mat4 projection;

void main(){
    gl_Position = projection * view * model * vec4(position,1.0);
}
```

Pipeline de transformações

Jupyter Notebook (Aula 08.Ex01)

Model Matrix

P’ = Projection x View x **Model** x P

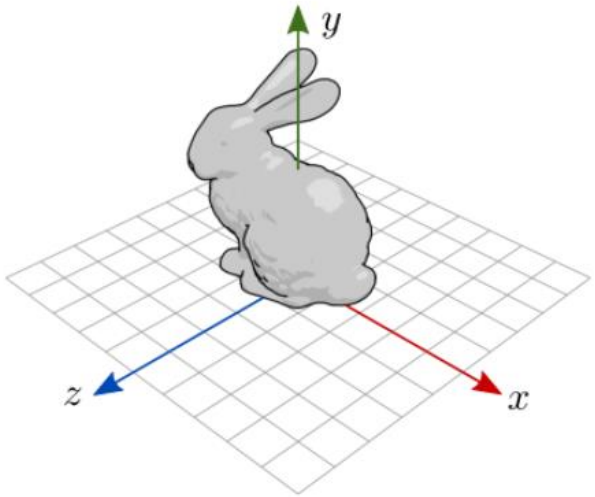


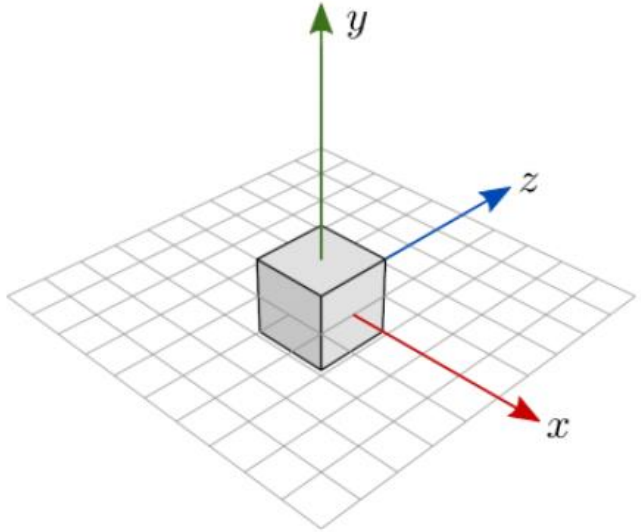
P’ = Projection x View x **Model** x P

Model Matrix

- Espaço Local ⟶ Espaço Mundo

- É comum que cada objeto seja inicializado com seu centro na origem do sistema de coordenadas local (i.e., do objeto).



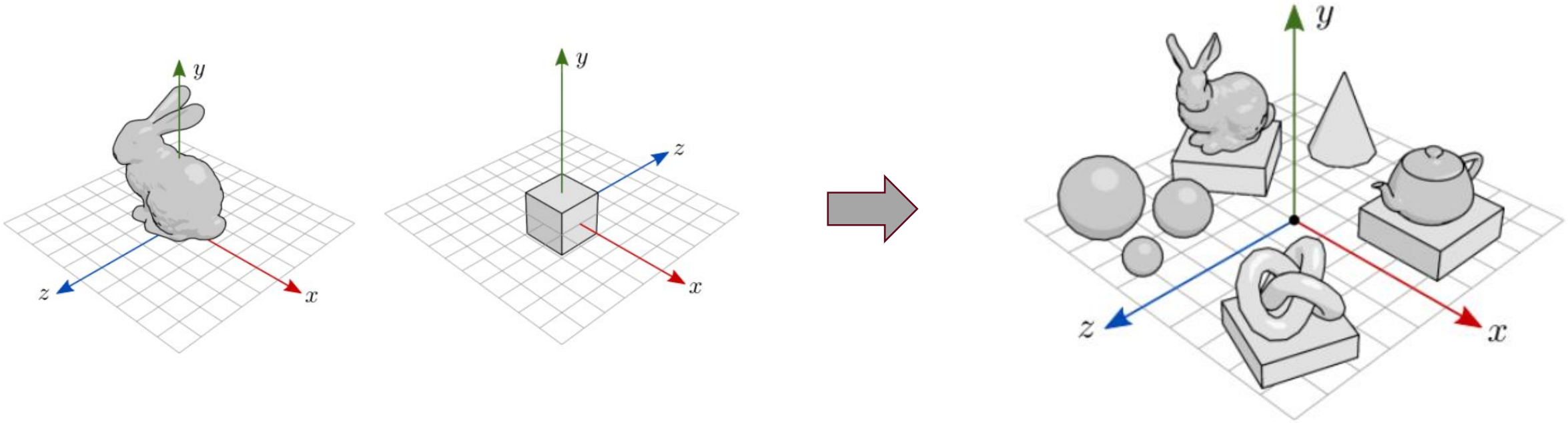


P’ = Projection x View x **Model** x P

Model Matrix

- Espaço Local → Espaço Mundo

- Precisamos colocá-los em uma única cena, i.e., em um único mundo



P’ = Projection x View x **Model** x P

Model Matrix

- Espaço Local → Espaço Mundo
- Precisamos colocá-los em uma única cena, i.e., em um único mundo
- Quem posiciona e orienta um objeto na cena é a **model matrix**, uma matriz que faz **transformações de modelagem** (usando rotação, escala e translação).

P’ = Projection x View x **Model** x P

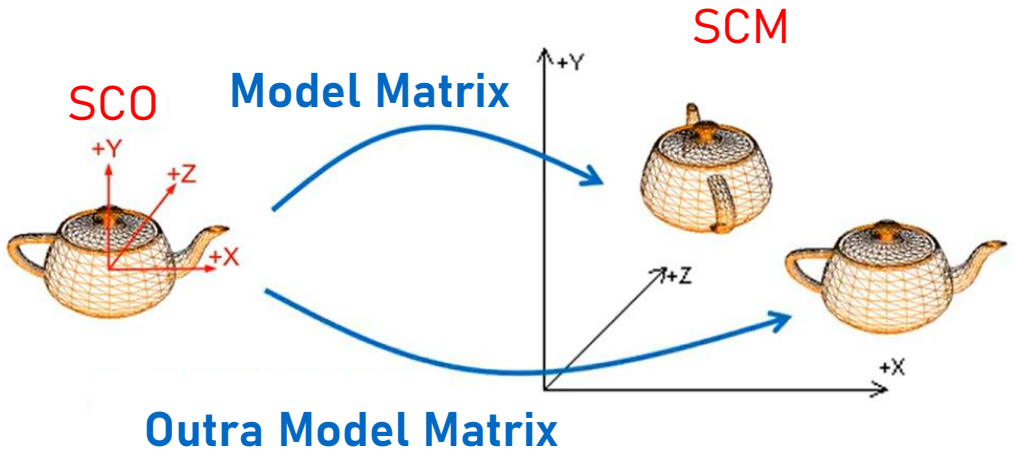
Model Matrix

- Espaço Local ⟶ Espaço Mundo
- Precisamos colocá-los em uma única cena, i.e., em um único mundo
- Quem posiciona e orienta um objeto na cena é a **model matrix**, uma matriz que faz **transformações de modelagem** (usando rotação, escala e translação).
- **Cada objeto de uma cena deve ter sua própria model matrix.**

P’ = Projection x View x **Model** x P

Model Matrix

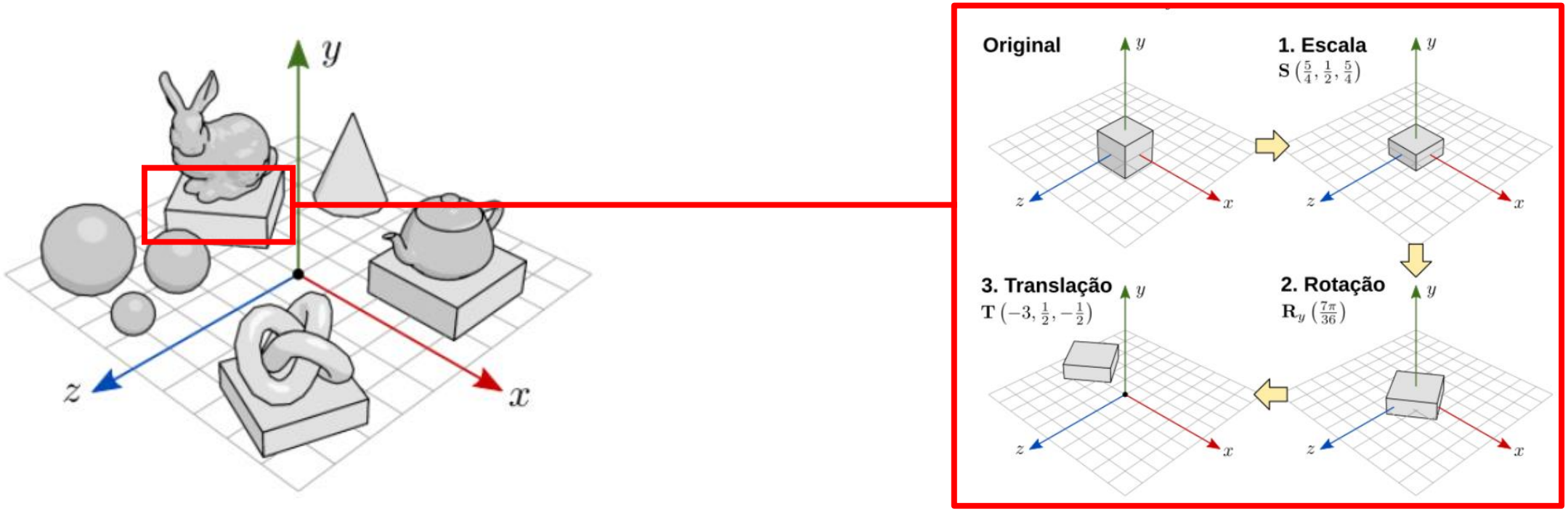
- Cada objeto de uma cena deve ter sua própria model matrix.



P’ = Projection x View x **Model** x P

Model Matrix

- Cada objeto de uma cena deve ter sua própria model matrix.



Model Matrix

P’ = Projection x View x **Model** x P

Jupyter Notebook
(Aula 08.Ex02)

Bibliografia

- Essa aula foi baseada no seguinte material:

- https://www.brunodorta.com.br/cg/glspaces *(Acessado em 23/08/2024)*

- Computação Gráfica, Slides de Ricardo Marcacini, Maria Cristina, Alaor Cervati. ICMC/USP.

- Hughes, J. F., Van Dam, A., Foley, J. D., McGuire, M., Feiner, S. K., & Sklar, D. F. (2014). Computer graphics: principles and practice. Terceira Edição. Pearson Education.