2022 WebGL高级课程

WebGL Advanced Course

WebGL视图变换

讲解人：冰老师
讲解时间：20221107

冰鉴韵
西藏 阿里

扫一扫上面的二维码图案，加我微信

WebGL交流群

目录

Lorem Ipsum is simply dummy text of the printing and typesetting industry

01

前言

前言

一个camera一般有如下四个属性，

- 向前向量（direction），相当于Z轴
- 向上向量（up vector），相当于Y轴
- 向右向量（right vector），相当于X轴
- 位置（position）

前段时间买了个相机，（佳能的M50），其实在webgl世界里面相机也是类似的。

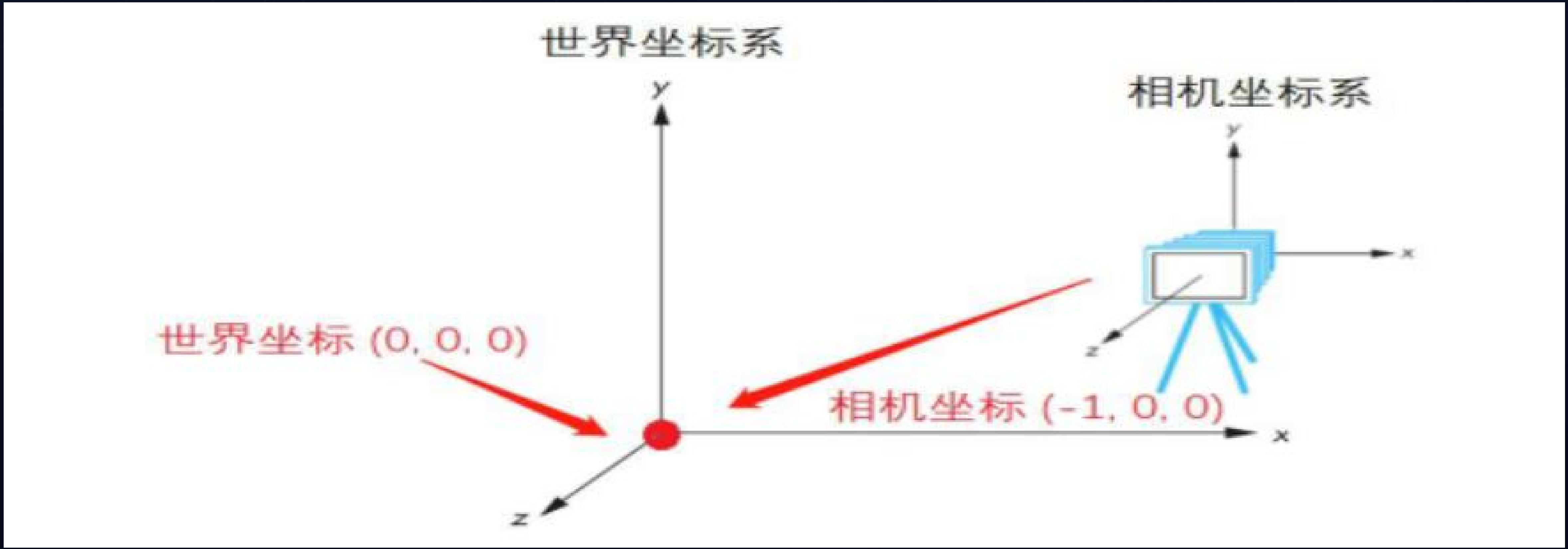
Lorem Ipsum is simply dummy text of the printing and typesetting industry

02

视图变换

定义

视图变换：进入世界坐标系空间之后，物体与WebGL相机虽然建立了联系，但是并没有进一步确定观察物体的状态。当我们把相机的位置进行移动的时候，相机坐标系和世界坐标系不再重合。这意味着我们直接将世界坐标作为最终的坐标绘制，并不能正确的描述观察者和物体之间的位置关系。如图，我们将相机沿着 x 轴正方向移动 1 个单位。此时世界坐标系的原点（0，0，0）在相机坐标系中的坐标就变成（-1，0，0），这说明我们需要在两个坐标系之间进行转换！这个时候就需要调整相机位置姿态，也就是视图变换。



Lorem Ipsum is simply dummy text of the printing and typesetting industry

03

公式推导

公式推导

https://zhuanlan.zhihu.com/p/362713511

其中前三个向量要求是相互垂直的。假设我们分别用d（向前向量（direction）），u（向上向量（up vector）），r（向右向量（right vector））和p（位置（position））来表示这四个变量。并假设待求的视图矩阵为V，根据前面的介绍我们知道，V的作用就是将摄像机移动到原点，并将摄像机的三个向量分别与坐标轴对齐，d与z轴正方向对齐，u与y轴正方向对齐，r与x轴正方向对齐。假设将摄像机与坐标轴对齐的矩阵为V，那么V的推导过程如下。

1、先将相机位置移动到原点位置（0，0，0），位移矩阵是Mt^{-1}

$$M_T{}^{-1} = \begin{bmatrix} 1 & 0 & 0 & -eye_x \\ 0 & 1 & 0 & -eye_y \\ 0 & 0 & 1 & -eye_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2、世界坐标实际上是E单位矩阵，通过矩阵R变换，变成[(l*u),u,l)

RE=[(l*u),u,l)=> R = [(l*u),u,l)

$$\begin{bmatrix} (\vec{l} \times \vec{u})_x & u_x & l_x & 0 \\ (\vec{l} \times \vec{u})_y & u_y & l_y & 0 \\ (\vec{l} \times \vec{u})_z & u_z & l_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3、实际上我们是求从相机位置转换到原定位置，也就是当前矩阵的逆矩阵R^{-1}

$$\begin{bmatrix} (\vec{l} \times \vec{u})_x & (\vec{l} \times \vec{u})_y & (\vec{l} \times \vec{u})_z & 0 \\ u_x & u_y & u_z & 0 \\ l_x & l_y & l_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

正交矩阵的逆矩阵等于转置矩阵

公式推导

$$V = RM = \begin{bmatrix} (\vec{l} \times \vec{u})_x & (\vec{l} \times \vec{u})_y & (\vec{l} \times \vec{u})_z & 0 \\ u_x & u_y & u_z & 0 \\ l_x & l_y & l_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad M_T{}^{-1} = \begin{bmatrix} 1 & 0 & 0 & -eye_x \\ 0 & 1 & 0 & -eye_y \\ 0 & 0 & 1 & -eye_z \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} (l \times u)_x & (l \times u)_y & (l \times u)_z & -(l \times u)_x eye_x - (l \times u)_y eye_y - (l \times u)_z eye_z \\ U_x & U_y & U_z & -U_x eye_x - U_y eye_y - U_z eye_z \\ L_x & L_y & L_z & -L_x eye_x - L_y eye_y - L_z eye_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

谢谢观看

冰鉴韵

西藏 阿里

扫一扫上面的二维码图案，加我微信

WebGL交流群