几何变换详解

在三维图形学中，几何变换大致分为三种，平移变换（Translation），缩放变换（Scaling），旋转变换（Rotation）。以下讨论皆针对DirectX，所以使用左手坐标系。

平移变换

将三维空间中的一个点[x, y, z, 1]移动到另外一个点[x', y', z', 1]，三个坐标轴的移动分量分别为dx=Tx, dy=Ty, dz=Tz, 即

x' = x + Tx

y' = y + Ty

z' = z + Tz

平移变换的矩阵如下。

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ Tx & Ty & Tz & 1 \end{bmatrix}$$

缩放变换

将模型放大或者缩小，本质也是对模型上每个顶点进行放大和缩小（顶点坐标值变大或变小），假设变换前的点是[x, y, z, 1]，变换后的点是[x', y', z', 1]，那么

x' = x * Sx

y' = y * Sy

z' = z * Sz

缩放变换的矩阵如下。

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Shader(2)
Swift
Unity3D
Vim
XAudio2
编程语言(9)
代码片段(19)
数据结构与算法(26)
图形学(10)
移动开发(7)
杂(24)

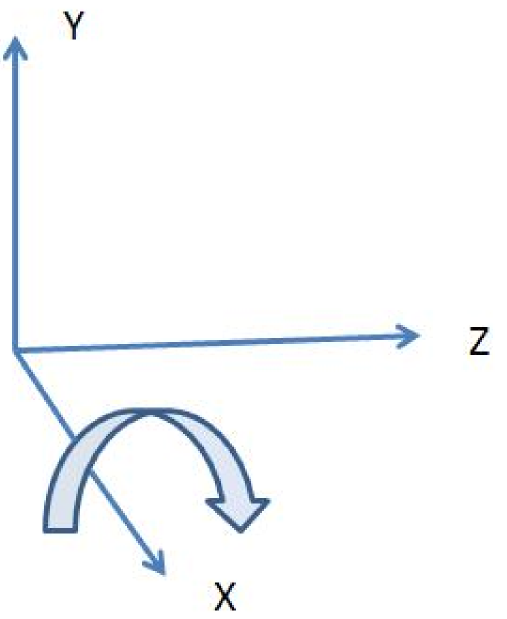
随笔档案

2015年12月 (1)
2015年11月 (1)
2015年10月 (2)
2015年8月 (5)
2015年7月 (2)
2014年11月 (2)
2014年6月 (2)
2014年5月 (1)
2013年9月 (1)
2013年7月 (1)
2013年6月 (1)
2013年4月 (2)
2013年3月 (3)
2013年2月 (3)
2012年11月 (2)
2012年10月 (2)
2012年9月 (5)
2012年8月 (6)
2012年7月 (5)
2012年6月 (2)
2012年5月 (2)
2012年4月 (1)
2011年9月 (2)
2011年7月 (3)
2011年6月 (2)
2011年5月 (4)
2011年4月 (3)
2011年3月 (3)
2011年2月 (4)
2010年12月 (3)
2010年11月 (3)
2010年9月 (1)
2010年8月 (11)
2010年7月 (17)
2010年6月 (9)
2010年5月 (12)
2010年4月 (2)
2010年3月 (16)
2010年2月 (3)
2010年1月 (2)
2009年12月 (2)
2009年11月 (9)
2009年10月 (11)
2009年9月 (3)
2009年8月 (1)
2009年7月 (4)
2009年6月 (10)
2009年5月 (2)

旋转变换

这是三种变换中最复杂的变换，这里只讨论最简单的情况，绕坐标轴旋转，关于绕任意轴旋转，在后续的随笔中介绍。

绕X轴旋转



绕X轴旋转时，顶点的x坐标不发生变化，y坐标和z坐标绕X轴旋转θ度，旋转的正方向为顺时针方向（沿着旋转轴负方向向原点看）。[x, y, z, 1]表示变换前的点，[x', y', z', 1]表示变换后的点。变换矩阵如下。

关于旋转的正方向，OpenGL与多数图形学书籍规定旋转正方向为逆时针方向（沿着坐标轴负方向向原点看），比如Computer Graphics C Version，p409。

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\theta & sin\theta & 0 \\ 0 & -sin\theta & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

绕Y轴旋转

Game Engine

DX11 Tutorials
Irrlicht
MathWords
Ogre

OpenGL

缤纷世界

3D Controls
3D Fractal
chaos files
ChaosinChinese
Cloth Simulation
DirectX Developer Center
DirectX document online
DX tutorials
EuclideanSpace
Fractal Video
geometrictools
Google C++ Style
HardCode
HyperGraph
In Framez
LatexEditor
Mame
Mandelbrot Set
OpenGL official page
OpenGL Tutorials
pouet
Ray tracing
Rthdribl
TechInterview
toymaker
W3SCHOOL
XNA/DirectX Forum
X-Zone
云风的Blog

其他

Aogo汇编小站
Channel 9
Code all in one
Emath
Math Circle
Math Game
Microsoft At Home
Microsoft At Work
Windows forum
中国DOS联盟

友情链接

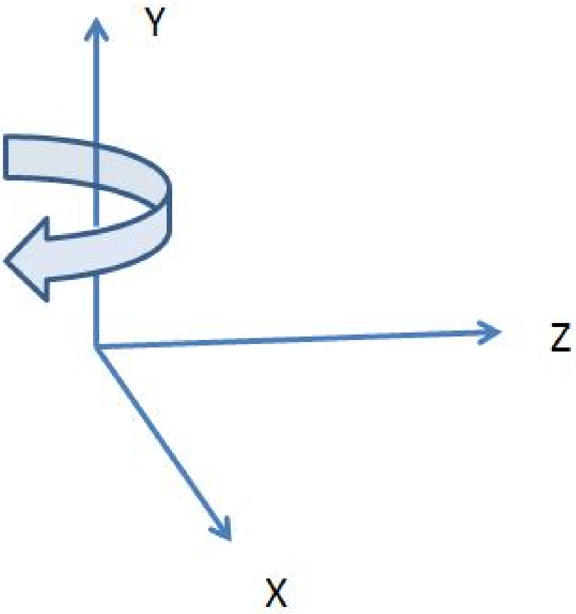
LittleStart
WW老弟

积分与排名

积分 - 189882
排名 - 786

最新评论

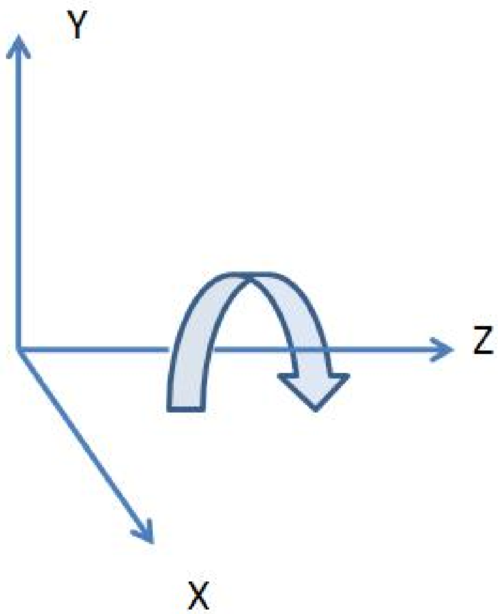
1. Re:判断点是否在三角形内



绕Y轴旋转时，顶点的y坐标不发生变化，x坐标和z坐标绕Y轴旋转θ度。[x, y, z, 1]表示变换前的点，[x', y', z', 1]表示变换后的点。变换矩阵如下。

$$[x' \quad y' \quad z' \quad 1] = [x \quad y \quad z \quad 1]\begin{bmatrix} cos\theta & 0 & -sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ sin\theta & 0 & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

绕Z轴旋转



绕Z轴旋转时，顶点的z坐标不发生变化，x坐标和y坐标绕Z轴旋转θ度。[x, y, z, 1]表示变换前的点，[x', y', z', 1]表示变换后的点。变换矩阵如下。

P = A + u * (C – A) + v * (B - A) P A B C 向量 如果只考虑二维(x,y)的话。这就是个二元一次方程组了。Px = Ax + u (Cx -Ax) + v *......
--doubi1900
2. Re:Direct2D教程（四）Path Geometry
struct D2D1_ARC_SEGMENT { D2D1_POINT_2F point; D2D1_SIZE_F size; FLOAT rotationAngle; D2......
--wyu_wmb
3. Re:用DirectX实现多视图渲染
@黑尾砂狐D3DXVECTOR3 eyePt(0.0f, 0.0f, -5.0f)；这样就可以水平了，模型实际上是平着的嘛，只是视角在变，平着看就是水平放置，绕y轴转就用D3DMatrixRotat......
--Drimoon
4. Re:GLFW初体验
@lsw1019那就按照我写的步骤试一下啊。...
--zdd
5. Re:GLFW初体验
glfw3.dll,与你的错误二相同。
--lsw1019

阅读排行榜

1. 点到平面的距离公式(36348)
2. 算法-求二进制数中1的个数(35286)
3. 字符串面试题（一）字符串逆序(34538)
4. 关于数组的几道面试题(34075)
5. C++ 初始化列表(33318)

评论排行榜

1. 程序员，请昂起你高贵的头！(364)
2. 关于数组的几道面试题(96)
3. 几何变换详解(76)
4. 判断点是否在三角形内(61)
5. 使用DirectX截屏(55)

推荐排行榜

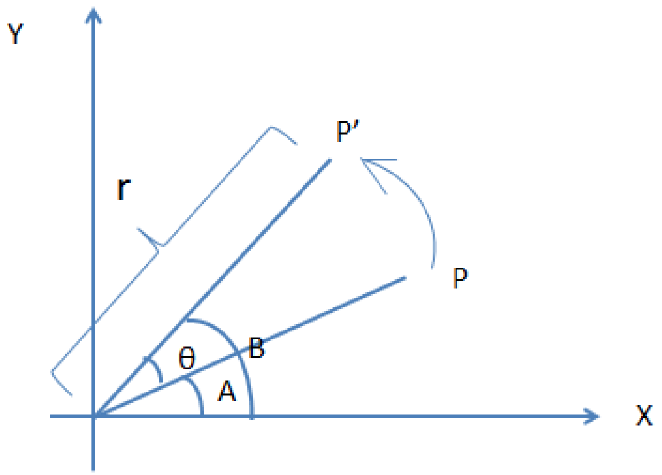
1. 程序员，请昂起你高贵的头！(172)
2. 算法-求二进制数中1的个数(26)
3. C++ 初始化列表(20)
4. 字符串面试题（一）字符串逆序(19)

$$[x' \quad y' \quad z' \quad 1] = [x \quad y \quad z \quad 1]\begin{bmatrix} cos\theta & sin\theta & 0 & 0 \\ -sin\theta & cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

绕坐标轴旋转的矩阵推导

上面三个旋转矩阵是如何得来的呢？我们推导一下，首先看一下二维的情况，再扩展到三维即可。实际上上面三种绕坐标轴旋转的情况属于特殊的二维旋转，比如绕Z轴旋转，相当于在与XOY平面上绕原点做二维旋转。

假设点P(x, y)是平面直角坐标系内一点，其到原点的距离为r，其与X轴的夹角为A，现将点P绕原点旋转θ度，得到点P'(x', y')，P'与X轴的夹角为B，则A = B - θ。（注意，在二维坐标中，逆时针旋转时角度为正，顺时针旋转时角度为负，下图中由P旋转到P'，角度为θ，若是由P'转到P，则角度为-θ）。



$$r = (x, y) = (rcosA + rsinA)$$
$$p' = (x', y') = (rcosB, rsinB) = (rcos(A+\theta), rsin(A+\theta))$$
$$rcos(A+\theta) = rcosAcos\theta - rsinAsin\theta = xcos\theta - ysin\theta$$
$$rsin(A+\theta) = rsinAcos\theta + rcosAsin\theta = ycos\theta + xsin\theta$$

于是可得下面的转换方程

$$\begin{cases} x' = xcos\theta - ysin\theta \\ y' = ycos\theta + xsin\theta \\ z' = z \end{cases}$$ （式一）

写成矩阵的形式就是

$$(x', y' z') = (x, y, z)\begin{bmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

5. 关于数组的几道面试题(16)

求得旋转矩阵为

$$\begin{bmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

由于这里使用齐次坐标，所以还需加上一维，最终变成如下形式

绕Z轴旋转矩阵

$$\begin{bmatrix} cos\theta & sin\theta & 0 & 0 \\ -sin\theta & cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

和前面给出的绕Z轴旋转矩阵完全吻合。

对于绕X轴旋转的情况，我们只需将式一中的x用y替换，y用z替换，z用x替换即可。替换后得到

$$\begin{cases} y' = ycos\theta - zsin\theta \\ z' = zcos\theta + ysin\theta \\ x' = x \end{cases} \quad \text{(式二)}$$

对应的旋转矩阵为

绕X轴旋转矩阵

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\theta & sin\theta & 0 \\ 0 & -sin\theta & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

对于绕Y轴旋转的情况，只需对式二做一次同样的替换即可，的到的变换方程为

$$\begin{cases} z' = zcos\theta - xsin\theta \\ x' = xcos\theta + zsin\theta \\ y' = y \end{cases}$$

对应的变换矩阵为

绕Y轴旋转矩阵

$$\begin{bmatrix} cos\theta & 0 & -sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ sin\theta & 0 & cos\theta & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

逆矩阵

平移变换矩阵的逆矩阵与原来的平移量相同，但是方向相反。

$$\begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 1 & 0\\T_x & T_y & T_z & 1\end{bmatrix}\begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 1 & 0\\-T_x & -T_y & -T_z & 1\end{bmatrix}=I$$

旋转变换矩阵的逆矩阵与原来的旋转轴相同但是角度相反。

$$\begin{bmatrix}1 & 0 & 0 & 0\\0 & cos\theta & sin\theta & 0\\0 & -sin\theta & cos\theta & 0\\0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}0 & 0 & 0 & 0\\0 & cos\theta & -sin\theta & 0\\0 & sin\theta & cos\theta & 0\\0 & 0 & 0 & 1\end{bmatrix}=I$$

$$\begin{bmatrix}cos\theta & 0 & -sin\theta & 0\\0 & 1 & 0 & 0\\sin\theta & 0 & cos\theta & 0\\0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}cos\theta & 0 & sin\theta & 0\\0 & 1 & 0 & 0\\-sin\theta & 0 & cos\theta & 0\\0 & 0 & 0 & 1\end{bmatrix}=I$$

$$\begin{bmatrix}cos\theta & sin\theta & 0 & 0\\-sin\theta & cos\theta & 0 & 0\\0 & 0 & 1 & 0\\0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}cos\theta & -sin\theta & 0 & 0\\sin\theta & cos\theta & 0 & 0\\0 & 0 & 1 & 0\\0 & 0 & 0 & 1\end{bmatrix}=I$$

缩放变换的逆矩阵正好和原来的效果相反，如果原来是放大，则逆矩阵是缩小，如果原来是缩小，则逆矩阵是放大。

$$\begin{bmatrix}S_x & 0 & 0 & 0\\0 & S_y & 0 & 0\\0 & 0 & S_z & 0\\0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\frac{1}{S_x} & 0 & 0 & 0\\0 & \frac{1}{S_y} & 0 & 0\\0 & 0 & \frac{1}{S_z} & 0\\0 & 0 & 0 & 1\end{bmatrix}=I$$

== Happy Coding ==