Basic 3D Math Practice

Wentao Lu

2020-09-15

1 Pinhole Camera Model

For a pinhole camera, the focal length is the distance from the pinhole to the film plane. The dimensions of a frame of 35-mm film are about 24mm×36mm (width×height). The field of view of the pinhole camera is the angle made by the largest object that the camera can image on its film plane, but a real field of view is hard to deal with, so people use surface angles of view such as the one in the xz-plane, the yz-plane, or in the planes passing through one of the diagonals of the film plane. Assume that the human visual system has an angle of view of 90°, what focal length should we use with 35-mm film to achieve a natural view in

1. the xz-plane?
2. the yz-plane?
3. any of the planes passing through one of the diagonals of the film plane?

For any 3D point (x, y, z) lying in the field of view of the pinhole camera with focal length d, what is the projected point (u, v) onto the film plane?

Solution: ....

A sectional view of the pinhole camera model consists of two pairs of similar right triangles. To achieve a natural view when the human visual system has 90° field of view, the surface angle of view in any plane must also be 90°, so the interior angles of these triangles are $90/2 = 45°$. Since $tan 45$ is 1, the height and base of the triangle must be equal, so in our case, the focal length $d = h/2$, where h is the height of the triangle.

For the xz-plane, h is the width of the 35-mm film frame along x-direction, which is 24mm, so $d = 12$mm. For the yz-plane, h is the height along y-direction, which is 36mm, so $d = 18$mm. For the plane that passes through the diagonals, the diagonal length will substitute for h, so d is half of $\sqrt{24^2 + 36^2}$ which is 21.63mm.

For perspective projection, the result coordinates onto the image plane are

$$u = -\frac{xd}{z},\ v = -\frac{yd}{z}$$

2 Lookup Table

If we use direct coding of RGB values with 10 bits per primary color, how many possible colors do we have for each pixel? If we use 12-bit pixel values in a lookup table representation, how many entries does the lookup table have?

SOLUTION: ....

10 bits in each RGB channel allows 2^{10} = 1024 colors, so we have 2^{30} possible colors for each pixel. 12-bit pixel values can store 2^{12} = 4096 indices of the lookup table, so the table has 4096 entries, each encodes a true RGB color.

3 Vectors and Planes

1. Find the equation of the plane determined by the three points:

   ⋄ $P_0(1, 5, -7)$ ⋄ $P_1(2, 6, 1)$ ⋄ $P_2(0, 1, 2)$

2. Find the equation of the line passing through $P_0(1, -5, 2)$ and $P_1(6, 7, -3)$.

3. Let a plane be determined by the normal $n(1, -1, 1)$ and the point $P_0(2, 3, -1)$. Find the distance from point $P(5, 2, 7)$ to the plane.

4. Given the plane $5x - 3y + 6z - 7 = 0$, find a normal vector to the plane, determine whether $P_1(1, 5, 2)$ and $P_2(-3, -1, 2)$ are on the same side of the plane.

SOLUTION: ....

1. We can find a normal of the plane using the cross product of two vectors:

   $$n = \overrightarrow{P_0P_1} \times \overrightarrow{P_1P_2} = (1, 1, 8) \times (-2, -5, 1) = (41, -17, -3)$$

   so the plane equation can be written as:

   $$41x - 17y - 3z + D = 0$$

   plug $P_2(0, 1, 2)$ into the equation and solve for D, we have $D = 23$, so the equation is:

   $$41x - 17y - 3z + 23 = 0$$

2. $\overrightarrow{P_0P_1}$ = $(5, 12, -5)$, so the equation of the line can be written as:

   $$\frac{x - x_1}{5} = \frac{y - y_1}{12} = \frac{z - z_1}{-5}$$

   substitute in $P_0(1, -5, 2)$, so we have

   $$\frac{x - 1}{5} = \frac{y + 5}{12} = \frac{z - 2}{-5}$$

3. The distance from $P(5,2,7)$ to the plane is the projected length of $\overrightarrow{P_0P}$ onto the normal

$$\frac{|\overrightarrow{P_0P} \cdot n|}{|n|} = \frac{|(3,-1,8) \cdot (1,-1,1)|}{\sqrt{1^2 + (-1)^2 + 1^2}} = \frac{12}{\sqrt{3}} = 4\sqrt{3}$$

4. One of the normal vector is simply $n = (5,-3,6)$. Randomly choose a point on the plane, such as $P_0(-1,0,2)$, we have two vectors from P_0

$$\overrightarrow{P_0P_1} = (2,5,0), \ \ \overrightarrow{P_0P_2} = (-2,-1,0)$$

whose dot products with the normal vector are

$$\begin{aligned} \overrightarrow{P_0P_1} \cdot n &= (2,5,0) \cdot (5,-3,6) = -5 < 0 \\ \overrightarrow{P_0P_2} \cdot n &= (-2,-1,0) \cdot (5,-3,6) = -7 < 0 \end{aligned}$$

Since both of them yield a negative dot product, it implies that they are in the opposite direction from the normal, and their angles from the normal are between 90° and 180°. Hence, P_1 and P_2 must be on the same side of the plane. (if the dot product is 0, the point lies on the plane)

4 Collinearity and Convexity

1. Three vertices (in 3D) determine a triangle if they do not lie in the same line. Devise a test for collinearity of three vertices.

2. Given a set of vertices, find a test to determine whether the polygon that they determine is planar or not.

3. Devise a test for the convexity of a two-dimensional polygon.

SOLUTION: ....

1. Consider three arbitrary vertices $P_1(x_1,y_1,z_1)$, $P_2(x_2,y_2,z_2)$, $P_3(x_3,y_3,z_3)$. If they are collinear, one vertex is the linear combination of the other two, so

$$\det \begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ z_1 & z_2 & z_3 \end{bmatrix} = 0$$

2. Pick three vertices at random, if they are not collinear, they uniquely determine a plane. Next, evaluate the plane equation for all other vertices, if it evaluates to 0, the vertex must be on the same plane, otherwise the polygon is not planar.

3. This is similar to what we did to find the convex hull. Given a two-dimensional polygon, we start from a random vertex and traverse along the sides of the polygon in a clockwise manner, for each side we visit, define the side as a vector in the forward direction. If the polygon is convex, we must be making only right turns in each step.

Mathematically, we can check this by computing the dot product of adjacent vectors as we traverse. Consider two adjacent vectors v_1 and v_2, and v_2' is the vector of v_2 rotated by 90° clockwise, then v_2 is on the right hand side of v_1 if and only if

(1) $v_1 \cdot v_2 \geq 0$
(2) $v_1 \cdot v_2' \leq 0$

5 Transformations

1. Prove that a y-reflection (reflection about the x-axis) followed by a reflection through the line $y = -x$ is a pure rotation.

SOLUTION: . . . .

The reflection matrix about the x-axis is $r_1 = \left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right)$
The reflection matrix about line $y = -x$ is $r_2 = \left(\begin{smallmatrix} 0 & -1 \\ -1 & 0 \end{smallmatrix}\right)$

$r_1 \cdot r_2 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$ is a rotation around the z-axis by 90°.

2. Find the vertices of the rotated triangle obtained by performing a 45° rotation of triangle $A(0,0), B(1,1), C(5,2)$.

(a) about the origin
(b) about $P(-1,-1)$

SOLUTION: . . . .

Let $s = sin45$, $c = cos45$, $\Rightarrow$ a 45° counter-clockwise rotation is given by

$$\begin{pmatrix} c & s \\ -s & c \end{pmatrix} \text{ where } s = c = \frac{\sqrt{2}}{2}$$

therefore, after the rotation, the new coordinates will be:

$$\begin{aligned} x' &= x \cdot c - y \cdot s \\ y' &= x \cdot s + y \cdot c \end{aligned}$$

To rotate around the origin, we can directly apply the formulas above

$$\begin{aligned} A'(x', y') &= (0 \cdot c - 0 \cdot s, 0 \cdot s + 0 \cdot c) = (0, 0) \\ B'(x', y') &= (1 \cdot c - 1 \cdot s, 1 \cdot s + 1 \cdot c) = (0, \sqrt{2}) \\ C'(x', y') &= (5 \cdot c - 2 \cdot s, 5 \cdot s + 2 \cdot c) = (\frac{3\sqrt{2}}{2}, \frac{7\sqrt{2}}{2}) \end{aligned}$$

To rotate around $P(-1,-1)$, we must first translate P to the origin by adding 1 to each component x and y, then do the rotation, and finally translate back (substract 1)

$$\begin{aligned}
A'(x',y') &= (1\cdot c - 1\cdot s - 1, 1\cdot s + 1\cdot c - 1) = (-1, \sqrt{2}-1)\\
B'(x',y') &= (2\cdot c - 2\cdot s - 1, 2\cdot s + 2\cdot c - 1) = (-1, 2\sqrt{2}-1)\\
C'(x',y') &= (6\cdot c - 3\cdot s - 1, 6\cdot s + 3\cdot c - 1) = (\frac{3\sqrt{2}}{2}-1, \frac{9\sqrt{2}}{2}-1)
\end{aligned}$$

3. Write the transformation matrix that magnifies the triangle with vertices $A(0,0)$, $B(1,1)$, $C(5,2)$ to twice its size while keeping C fixed.

SOLUTION: ....

To keep $C(5,2)$ fixed, we translate it to the origin by substracting 5 from the x component and 2 from the y component, then scale by a factor of 2, and finally translate back

$$T = \begin{bmatrix} 1 & 0 & -5 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{bmatrix}, S = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow M = T^{-1}\cdot S \cdot T = \begin{bmatrix} 2 & 0 & -5 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$|A'B'C'| = M \cdot |ABC| = M \cdot \begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} -5 & -3 & 5 \\ -2 & 0 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\Rightarrow A' = (-5,-2),\;\; B' = (-3,0),\;\; C' = (5,2)$$

4. Let L be the line that passes through the origin in the direction of $u(1,1,1)$. Find the matrix for the rotation about L with angle 45°.

SOLUTION: ....

★ **Coordinate Axis Approach**

The unit vector in the direction of $(1,1,1)$ is

$$v = \frac{u}{||u||} = \frac{(1,1,1)}{\sqrt{1^2+1^2+1^2}} = (\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}) = (x,y,z)$$

To align this unit vector with the z-axis, we carry out two rotations

$$R_x(\theta_x) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & z/d & -y/d & 0 \\ 0 & y/d & z/d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, R_y(\theta_y) = \begin{bmatrix} d & 0 & -x & 0 \\ 0 & 1 & 0 & 0 \\ x & 0 & d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where $d = \sqrt{y^2+z^2}$ on the y-z plane.

Since both of them are rotation matrices, which are orthogonal, we have

$$R_x(-\theta_x) = R_x^{-1}(\theta_x) = R_x^T(\theta_x)$$
$$R_y(-\theta_y) = R_y^{-1}(\theta_y) = R_y^T(\theta_y)$$

Hence, the rotation matrix about L with angle 45° is

$$\begin{aligned} R(\theta) &= R_x(-\theta_x)R_y(-\theta_y)R_z(45)R_y(\theta_y)R_x(\theta_x) \\ &= R_x^T(\theta_x)R_y^T(\theta_y)R_z(45)R_y(\theta_y)R_x(\theta_x) \end{aligned}$$

To compute this matrix, we can use the numpy package from PYTHON.

```python
import numpy as np

def coordinate_axis_method(x, y, z, theta):
    d = np.sqrt(y**2 + z**2)

    r_x = np.array([
            [1, 0, 0, 0],
            [0, z/d, -y/d, 0],
            [0, y/d, z/d, 0],
            [0, 0, 0, 1]])

    r_y = np.array([
            [d, 0, -x, 0],
            [0, 1, 0, 0],
            [x, 0, d, 0],
            [0, 0, 0, 1]])

    s = np.sin(np.radians(theta))
    c = np.cos(np.radians(theta))

    r_z = np.array([
            [c, -s, 0, 0],
            [s, c, 0, 0],
            [0, 0, 1, 0],
            [0, 0, 0, 1]])

    return r_x.T @ r_y.T @ r_z @ r_y @ r_x

if __name__ == "__main__":
    x = y = z = 1 / np.sqrt(3)  # x, y, z of the unit vector
    M = coordinate_axis_method(x, y, z, 45)
    print(M)
```

The output matrix is $R(\theta) = \begin{bmatrix} 0.8047 & -0.3106 & 0.5059 & 0. \\ 0.5059 & 0.8047 & -0.3106 & 0. \\ -0.3106 & 0.5059 & 0.8047 & 0. \\ 0. & 0. & 0. & 1. \end{bmatrix}$

★ General Approach

This time the 3x3 rotation matrix is given by

$$M = uu^T + cos\theta(I - uu^T) + sin\theta \cdot U, \text{ where } U = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}$$

The code below yields the same matrix as the one we computed above.

```python
import numpy as np

def general_method(u, theta):
    s = np.sin(np.radians(theta))
    c = np.cos(np.radians(theta))

    U = np.array([
            [0, -z, y],
            [z, 0, -x],
            [-y, x, 0]])

    r = u @ u.T + c * (np.eye(3) - (u @ u.T)) + s * U

    # convert the matrix 3x3 to 4x4 in homogeneous coordinates
    m3_2_m4 = np.zeros((4, 4))
    m3_2_m4[-1, -1] = 1
    m3_2_m4[:3, :3] = r

    return m3_2_m4

if __name__ == "__main__":
    x = y = z = 1 / np.sqrt(3)
    u = np.array([[x, y, z]]).T  # unit vector (column vector)
    M = general_method(u, 45)
    print(M)
```

5. Let L be the line that passes through $(1, 1, 2)$ in the direction of $u(1, 1, 1)$. Find the matrix for the rotation about L with angle 60°.

Solution: ....

Now that the line does not pass through the origin, we have to translate the point $(1, 1, 2)$ to the origin first, apply the same routines as in the previous problem to obtain the rotation matrix R, then translate back.

The translation matrix from $(1, 1, 2)$ to the origin is

$$T = \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

its inverse is the matrix with the opposite signs on each of the translation components.

$$T^{-1} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Hence, the overall transformation matrix is

$$M = T^{-1} \cdot R \cdot T$$

In code, this can be computed as

```
p = (1, 1, 2)  # the fixed point

t_op = np.array([  # translation matrix from the origin to p
        [1, 0, 0, p[0]],
        [0, 1, 0, p[1]],
        [0, 0, 1, p[2]],
        [0, 0, 0, 1]])

t_po = np.array([  # translation matrix from p to the origin
        [1, 0, 0, -p[0]],
        [0, 1, 0, -p[1]],
        [0, 0, 1, -p[2]],
        [0, 0, 0, 1]])

if __name__ == "__main__":
    x = y = z = 1 / np.sqrt(3)
    u = np.array([[x, y, z]]).T  # unit vector (column vector)
    print(t_op @ coordinate_axis_method(x, y, z, 60) @ t_po)
    print(t_op @ general_method(u, 60) @ t_po)
```

Both approaches yield $M = \begin{bmatrix} 0.6667 & -0.33333 & 0.6667 & -0.6667 \\ 0.6667 & 0.6667 & -0.3333 & 0.3333 \\ -0.3333 & 0.6667 & 0.6667 & 0.3333 \\ 0. & 0. & 0. & 1. \end{bmatrix}$

6. Find the matrix of transformation that aligns the vector $u(1,1,1)$ with vector $v(2,1,1)$.

SOLUTION: . . . .

In order to align u with v, we take 2 steps to decompose the orientation.

(1) align u with the z-axis

(2) align the z-axis with v

In step (1), the unit vector of $(1,1,1)$ is

$$\frac{u}{||u||} = \frac{(1,1,1)}{\sqrt{1^2+1^2+1^2}} = (\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}) = (x,y,z)$$

The matrix $R_{uy}(\theta_{uy}) \cdot R_{ux}(\theta_{ux})$ will align u with the z-axis.

$$R_{ux}(\theta_{ux}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & z/d & -y/d & 0 \\ 0 & y/d & z/d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, R_{uy}(\theta_{uy}) = \begin{bmatrix} d & 0 & -x & 0 \\ 0 & 1 & 0 & 0 \\ x & 0 & d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where $d = \sqrt{y^2 + z^2} = \sqrt{2/3}$.

In step (2), the unit vector of $(2, 1, 1)$ is

$$\frac{v}{||v||} = \frac{(2,1,1)}{\sqrt{2^2 + 1^2 + 1^2}} = (\frac{2}{\sqrt{6}}, \frac{1}{\sqrt{6}}, \frac{1}{\sqrt{6}}) = (x, y, z)$$

The matrix $R_{vy}(\theta_{vy}) \cdot R_{vx}(\theta_{vx})$ will align v with the z-axis. Or to put it another way, using the inverse of this matrix, we can align the z-axis with v.

$$R_{vx}(\theta_{vx}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & z/d & -y/d & 0 \\ 0 & y/d & z/d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, R_{vy}(\theta_{vy}) = \begin{bmatrix} d & 0 & -x & 0 \\ 0 & 1 & 0 & 0 \\ x & 0 & d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where $d = \sqrt{y^2 + z^2} = \sqrt{1/3}$

Now let's combine $R_{uy}(\theta_{uy}) \cdot R_{ux}(\theta_{ux})$ in step 1 and the inverse of $R_{vy}(\theta_{vy}) \cdot R_{vx}(\theta_{vx})$ in step 2 to obtain the final transformation matrix. Note however that, since we are solving for the matrix instead of applying it, the order of multiplication must be reversed:

$$M = \text{step2} \times \text{step1} = (R_{vy}(\theta_{vy}) \cdot R_{vx}(\theta_{vx}))^{-1} \cdot (R_{uy}(\theta_{uy}) \cdot R_{ux}(\theta_{ux}))$$

With the help of code, we can easily compute the matrix like so

```
import numpy as np

# step 1
x = y = z = 1 / np.sqrt(3)
d = np.sqrt(y**2 + z**2)

r_x_u = np.array([
    [1, 0, 0, 0],[0, z/d, -y/d, 0],[0, y/d, z/d, 0],[0, 0, 0, 1]])

r_y_u = np.array([
    [d, 0, -x, 0],[0, 1, 0, 0],[x, 0, d, 0],[0, 0, 0, 1]])

A = r_y_u @ r_x_u  # aligns u with the z-axis

# step 2
x = 2 / np.sqrt(6)
y = z = 1 / np.sqrt(6)
d = np.sqrt(y**2 + z**2)

r_x_v = np.array([
    [1, 0, 0, 0],[0, z/d, -y/d, 0],[0, y/d, z/d, 0],[0, 0, 0, 1]])

r_y_v = np.array([
    [d, 0, -x, 0],[0, 1, 0, 0],[x, 0, d, 0],[0, 0, 0, 1]])

B = r_y_v @ r_x_v             # aligns v with the z-axis
B_inverse = np.linalg.inv(B)  # aligns the z-axis with v

if __name__ == "__main__":
    with np.printoptions(suppress=True, precision=4):
        print(B_inverse @ A)
```

The output matrix is $M = \begin{bmatrix} 0.9428 & 0.2357 & 0.2357 & 0. \\ -0.2357 & 0.9714 & -0.0286 & 0. \\ -0.2357 & -0.0286 & 0.9714 & 0. \\ 0. & 0. & 0. & 1. \end{bmatrix}$

6 3D Projections

1. Derive the matrix of the parallel projection onto the plane $x + y + z = 1$ in the direction of projector $v(-1, -1, -1)$.

SOLUTION: ....

$v(-1, -1, -1)$ is in the opposite direction of $(1, 1, 1)$, which is a normal of the projection plane, but the projection plane is not parallel with the principal planes, so this is an axonometric projection. Consider a reference point of the plane $R(x_0, y_0, z_0) = (1, 0, 0)$ that satisfies the plane equation $x + y + z = 1$, and a unit normal vector of the plane

$$n = \frac{(1,1,1)}{\sqrt{1^2 + 1^2 + 1^2}} = (\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}) = (a, b, c)$$

To project a point $P(x, y, z)$ to its image $P'(x', y', z')$ on the plane, apply the transform

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 - a^2 & -ab & -ac & ad_0 \\ -ab & 1 - b^2 & -bc & bd_0 \\ -ac & -bc & 1 - c^2 & cd_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

where $d_0 = ax_0 + by_0 + cz_0 = a = \frac{1}{\sqrt{3}}$.
so the projection matrix is

$$\begin{bmatrix} 1 - a^2 & -ab & -ac & ad_0 \\ -ab & 1 - b^2 & -bc & bd_0 \\ -ac & -bc & 1 - c^2 & cd_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 2/3 & -1/3 & -1/3 & 1/3 \\ -1/3 & 2/3 & -1/3 & 1/3 \\ -1/3 & -1/3 & 2/3 & 1/3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2. Derive the matrix of general perspective projection onto the plane with normal vector $\boldsymbol{n}(a,b,c)$ and reference point $R(x_0,y_0,z_0)$ and using $C(c_x,c_y,c_z)$ as the center of projection.

SOLUTION: ....

Let $P(x,y,z)$ be a point in 3D and $P'(x',y',z')$ be its projection on the plane. By definition, vector $P'P$ must pass through the center of projection C, so $CP' = \lambda CP$, i.e.

$$\begin{cases} x' - c_x = \lambda(x - c_x) \\ y' - c_y = \lambda(y - c_y) \\ z' - c_z = \lambda(z - c_z) \end{cases} \quad \Leftrightarrow \quad \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \lambda \begin{bmatrix} x \\ y \\ z \end{bmatrix} + (1-\lambda) \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix} \tag{1}$$

The perpendicular distance from center of projection C to the plane can be computed as

$$\begin{aligned} d_{cr} &= CR \cdot \boldsymbol{n} \\ &= a(x_0 - c_x) + b(y_0 - c_y) + c(z_0 - c_z) \\ &= (ax_0 + by_0 + cz_0) - (ac_x + bc_y + cc_z) \\ &= d_r - d_c \end{aligned}$$

Here, we decompose d_{cr} into two constants that are geometrically meaningful: d_r is the perpendicular distance from the origin to the plane, and d_c measures the distance from the origin to the center of projection C along the projector direction.

Since $CR = CP' + P'R$ and $CP' = \lambda CP$, it follows that

$$\begin{aligned} d_{cr} &= CR \cdot \boldsymbol{n} \\ &= CP' \cdot \boldsymbol{n} + P'R \cdot \boldsymbol{n} \\ &= \lambda CP \cdot \boldsymbol{n} + 0 \\ &= \lambda[a(x - c_x) + b(y - c_y) + c(z - c_z)] \\ &= \lambda[(ax + by + cz) - (ac_x + bc_y + cc_z)] \\ &= \lambda[(ax + by + cz) - d_c] \end{aligned}$$

therefore, we can derive the factor λ

$$\Rightarrow \lambda = \frac{d_{cr}}{(ax + by + cz) - d_c}$$

substitute λ into the above equation (1), we obtain

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \frac{d_{cr}}{(ax + by + cz) - d_c} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + (1 - \frac{d_{cr}}{(ax + by + cz) - d_c}) \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix}$$

In order to convert this into homogeneous matrix form, first we need to translate COP to the origin so it reduces to a simple case, then apply the homogeneous transform, and finally undo the translation. Therefore, the matrix in this general case can be computed as

$$M_C = T_C \cdot M_0 \cdot T_{-C} = \begin{bmatrix} 1 & 0 & 0 & c_x \\ 0 & 1 & 0 & c_y \\ 0 & 0 & 1 & c_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_{cr} & 0 & 0 & 0 \\ 0 & d_{cr} & 0 & 0 \\ 0 & 0 & d_{cr} & 0 \\ a & b & c & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 1 & -c_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

after factoring and combining like terms, finally we have

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = M_C \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} ac_x + d_{cr} & bc_x & cc_x & -d_r c_x \\ ac_y & bc_y + d_{cr} & cc_y & -d_r c_y \\ ac_z & bc_z & cc_z + d_{cr} & -d_r c_z \\ a & b & c & -d_c \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

if we expand M_C, the projection matrix will be

$$M = \begin{bmatrix} ax_0+b(y_0-c_y)+c(z_0-c_z) & bc_x & cc_x & -(ax_0+by_0+cz_0)c_x \\ ac_y & a(x_0-c_x)+by_0+c(z_0-c_z) & cc_y & -(ax_0+by_0+cz_0)c_y \\ ac_z & bc_z & a(x_0-c_x)+b(y_0-c_y)+cz_0 & -(ax_0+by_0+cz_0)c_z \\ a & b & c & -(ac_x+bc_y+cc_z) \end{bmatrix}$$

3. Using the origin as the center of projection, derive the perspective transformation onto the plane passing through $(1, 0, 0)$ and having the normal $n(1, 1, 1)$.
What are the principal vanishing points for this perspective transformation?

SOLUTION: ....

Using $(1, 0, 0)$ as the reference point $R(x_0, y_0, z_0)$ and $(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}) = (a, b, c)$ as the unit normal vector, we have $d_0 = ax_0 + by_0 + cz_0 = \frac{1}{\sqrt{3}}$, so the transformation is

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} d_0 & 0 & 0 & 0 \\ 0 & d_0 & 0 & 0 \\ 0 & 0 & d_0 & 0 \\ a & b & c & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{3}} & 0 \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Consider the family of parallel lines in the direction of (u_x, u_y, u_z)

$$x = u_x t + \alpha, \quad y = u_y t + \beta, \quad z = u_z t + \gamma$$

where α, β, γ are constants and t is any real number.

Apply the projection matrix to the points on the line, we have

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{3}} & 0 \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x/\sqrt{3} \\ y/\sqrt{3} \\ z/\sqrt{3} \\ (x+y+z)/\sqrt{3} \end{bmatrix} \sim \begin{bmatrix} x/(x+y+z) \\ y/(x+y+z) \\ z/(x+y+z) \\ 1 \end{bmatrix}$$

as $t \to \infty$, this becomes

$$\begin{bmatrix} x/(x+y+z) \\ y/(x+y+z) \\ z/(x+y+z) \\ 1 \end{bmatrix}_{t\to\infty} = \begin{bmatrix} \frac{u_xt+\alpha}{(u_x+u_y+u_z)t+\alpha+\beta+\gamma} \\ \frac{u_yt+\beta}{(u_x+u_y+u_z)t+\alpha+\beta+\gamma} \\ \frac{u_zt+\gamma}{(u_x+u_y+u_z)t+\alpha+\beta+\gamma} \\ 1 \end{bmatrix}_{t\to\infty} = \begin{bmatrix} u_x/(u_x+u_y+u_z) \\ u_y/(u_x+u_y+u_z) \\ u_z/(u_x+u_y+u_z) \\ 1 \end{bmatrix}$$

when $u_x = 1, u_y = u_z = 0$, the x-axis vanishing point is $(1,0,0,1)$.
when $u_y = 1, u_x = u_z = 0$, the y-axis vanishing point is $(0,1,0,1)$.
when $u_z = 1, u_x = u_y = 0$, the z-axis vanishing point is $(0,0,1,1)$.

4. Assume given the canonical perspective projection where the view plane is parallel to the xy-plane and located at distance $d > 0$ on the positive z-axis, and the center of projection located at the origin.

   (a) What is the projected image of a point $P(x,y,0)$ located in the xy-plane.

   (b) What is the projected image of the line segment joining the points $P_1(-1,1,-d)$ and $P_2(2,-2,d)$.

SOLUTION: . . . .

Given the canonical camera setup with a focal length of d, the perspective projection on any point (x,y,z) where z is not zero is given by

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{d} & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ z/d \end{bmatrix} \sim \begin{bmatrix} dx/z \\ dy/z \\ d \\ 1 \end{bmatrix}$$

1. Since $P(x,y,0)$ is located in the xy-plane, $z = 0$, we cannot use the formula above. In fact, the line from P to the center of projection (the origin) is on the xy-plane, which does not intersect with the view plane. Hence, P is invisible to the viewer.

2. OP_1 and OP_2 are not in the same direction, so they will be projected onto different end points on the view plane.

$$P_1^* = (-d/-d, d/-d, d, 1) = (1,-1,d,1)$$
$$P_2^* = (2d/d, -2d/d, d, 1) = (2,-2,d,1)$$

   therefore, their 2D coordinates on the view plane are P_1^*=$(1,-1)$,P_2^*=$(2,-2)$, the projected image will be the line segment joining P_1^* and P_2^*.

7 Change of Frame

Find the matrix that converts the coordinates $a^T = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ in the frame $[u_1, u_2, u_3, P_0]$ into coordinates $b^T = (\beta_1, \beta_2, \beta_3, \beta_4)$ in the new frame $[v_1, v_2, v_3, Q_0]$, given by

$$v_1 = u_1,\ v_2 = u_2,\ v_3 = u_2 + u_3,\ Q_0 = P_0 + u_1$$

If $a^T = (1, 1, 1, 1)$, what are the coordinates of b?

Solution: ....

$$\begin{cases} v_1 = 1 \cdot u_1 + 0 \cdot u_2 + 0 \cdot u_3 \\ v_2 = 0 \cdot u_1 + 1 \cdot u_2 + 0 \cdot u_3 \\ v_3 = 0 \cdot u_1 + 1 \cdot u_2 + 1 \cdot u_3 \\ Q_0 = 1 \cdot u_1 + 0 \cdot u_2 + 0 \cdot u_3 + P_0 \end{cases} \Rightarrow M = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

this matrix converts homogeneous coordinates between the two frames, like so:

$$b = (M^T)^{-1} a = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$