Note: Half-Space Triangle Rasterization

Fong Zi-Sing, 2022.3

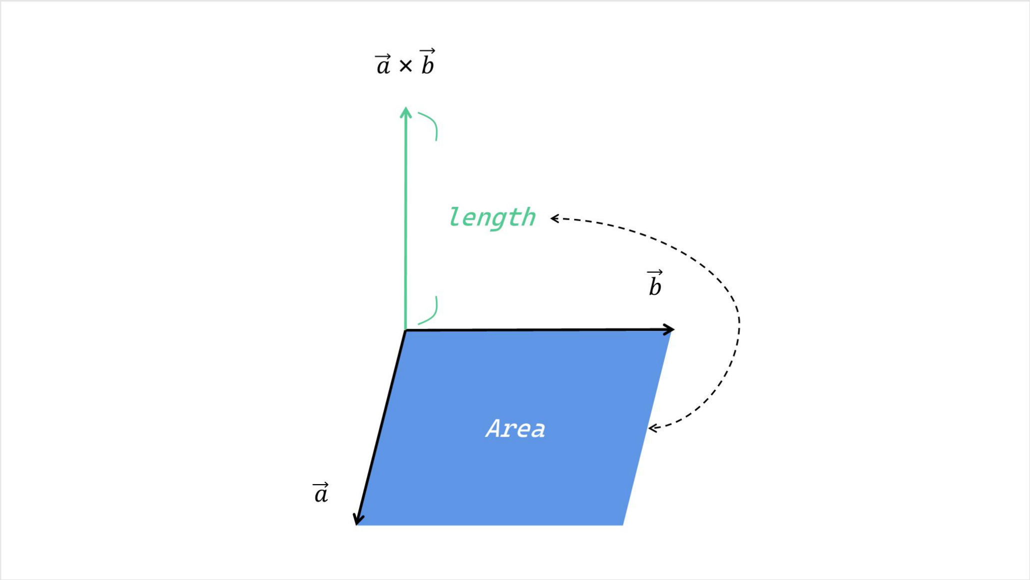
---

1. Cross Product (Perp Dot Product)

$$\left|\vec{a} \times \vec{b}\right| = |a||b| \sin\theta = a_x b_y - b_x a_y = \begin{vmatrix} a_x & a_y \\ b_x & b_y \end{vmatrix} \tag{1}$$

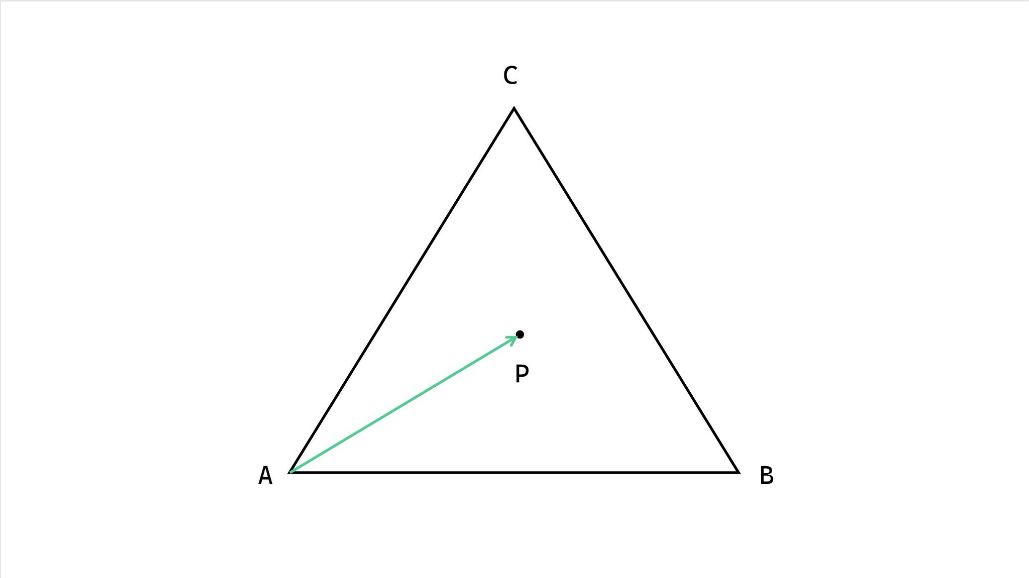
- $\left|\vec{a} \times \vec{b}\right|$ is also written as $a^{\perp} \cdot b$, means “perp dot product”.
  - $\left|\vec{a} \times \vec{b}\right| = 0$, $\vec{a}$, $\vec{b}$ are parallel.
  - $\left|\vec{a} \times \vec{b}\right| > 0$, $\vec{b}$ is counterclockwise from $\vec{a}$.
  - $\left|\vec{a} \times \vec{b}\right| < 0$, $\vec{a}$ is counterclockwise from $\vec{b}$.

- the direction of cross product obeys “right hand rule”.
- the magnitude of cross product equals the area of a parallelogram
- with vectors $\vec{a}$ and $\vec{b}$.



- Also can see [1].

2. Edge function[2]



- If the vertices A, B, C are given in counterclockwise order, then

  " P is inside the triangle "

  is equivalent to

$$\left|\overrightarrow{AB} \times \overrightarrow{AP}\right| > 0, \quad \left|\overrightarrow{BC} \times \overrightarrow{BP}\right| > 0, \quad \left|\overrightarrow{CA} \times \overrightarrow{CP}\right| > 0$$

- According to the formula [1], $\left|\overrightarrow{AB} \times \overrightarrow{AP}\right|$ can be described as:

$$\left|\overrightarrow{AB} \times \overrightarrow{AP}\right| = (B_x - A_x)(P_y - A_y) - (P_x - A_x)(B_y - A_y) \tag{2}$$

Perform the multiplications and by rearranging the factors, the above formula can be written as the "edge function":

$$F_{AB}(P_x\,,\ P_y) = (A_y - B_y) \cdot P_x + (B_x - A_x) \cdot P_y + (A_x B_y - A_y B_x) \tag{3}$$

- the other two edge functions are the following:

$$F_{BC}(P_x\,,\ P_y) = (B_y - C_y) \cdot P_x + (C_x - B_x) \cdot P_y + (B_x C_y - B_y C_x) \tag{4}$$

$$F_{CA}(P_x\,,\ P_y) = (C_y - A_y) \cdot P_x + (A_x - C_x) \cdot P_y + (C_x A_y - C_y A_x) \tag{5}$$

- For convenience of programming, this three edge functions are numbered as following:

$$I_{01} = (A_y - B_y),\ J_{01} = (B_x - A_x),\ K_{01} = (A_x B_y - A_y B_x)$$
$$I_{02} = (B_y - C_y),\ J_{02} = (C_x - B_x),\ K_{02} = (B_x C_y - B_y C_x)$$
$$I_{03} = (C_y - A_y),\ J_{03} = (A_x - C_x),\ K_{03} = (C_x A_y - C_y A_x)$$

$$F_{AB}(P_x\,,\ P_y) = F_{01}(P_x\,,\ P_y) = I_{01} \cdot P_x + J_{01} \cdot P_y + K_{01} \tag{6}$$

$$F_{BC}(P_x\,,\ P_y) = F_{02}(P_x\,,\ P_y) = I_{02} \cdot P_x + J_{02} \cdot P_y + K_{02} \tag{7}$$

$$F_{CA}(P_x\,,\ P_y) = F_{03}(P_x\,,\ P_y) = I_{03} \cdot P_x + J_{03} \cdot P_y + K_{03} \tag{8}$$

It is easy to find out that:

$$F_{01}(P_{x+1}, P_y) - F_{01}(P_x, P_y) = I_{01} \tag{9}$$

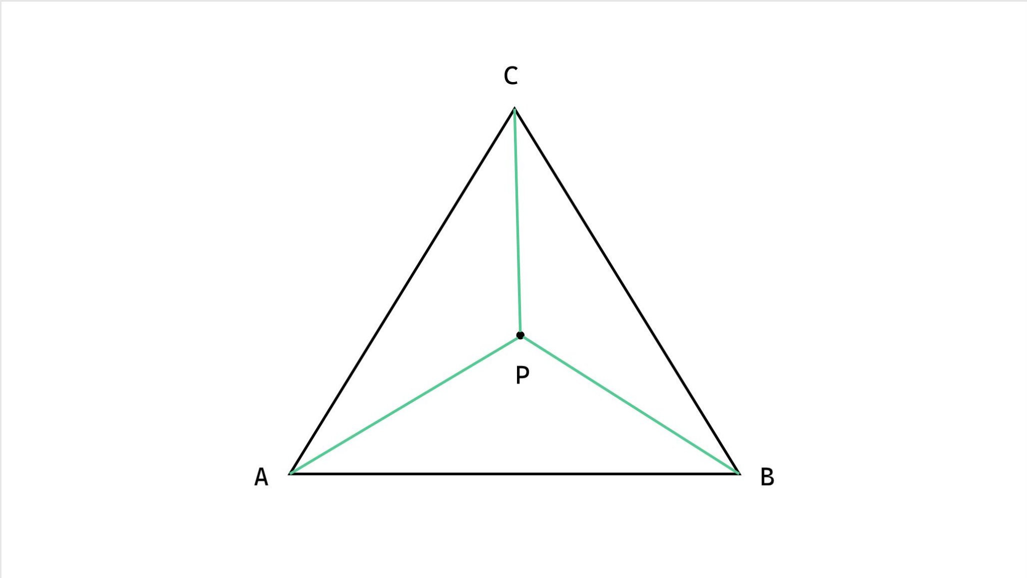
$$F_{01}(P_x, P_{y+1}) - F_{01}(P_x, P_y) = J_{01} \tag{10}$$

The same as the other two edge functions.

- Then, we can iterate triangle as the following pseudo code

```
[ minX, maxX, minY, maxY ] = CalcuteBoundingBox(triangle.vertices[3],screenSize)

[ I01, I02, I03 ] = CalcuteDeltaX(triangle.vertices[3])
[ J01, J02, J03 ] = CalcuteDeltaY(triangle.vertices[3])
[ K01, K02, K03 ] = CalcuteConst(triangle.vertices[3])

F01 = I01 * minX + J01 * minY + K01
F02 = I02 * minX + J02 * minY + K02
F03 = I03 * minX + J03 * minY + K03

Cy1 = F01
Cy2 = F02
Cy3 = F03

loop y = minY to maxY with step 1
{
    Cx1 = Cy1
    Cx2 = Cy2
    Cx3 = Cy3

    loop x = minX to maxX with step 1
    {
        // Determines whether the point (x, y) is inside the triangle.
        if (Cx1 >= 0 && Cx2 >= 0 && Cx3 >= 0)
        {
            Render(x, y)
        }

        Cx1 += I01
        Cx2 += I02
        Cx3 += I03
    }

    Cy1 += J01
    Cy2 += J02
    Cy3 += J03
}
```

3. Barycentric Interpolation



- The properties (attributes) P of the point P can be calculated by the barycentric coordiantes. includes depth, normal, color, texture coordinates, etc.

$$\mathrm{P}_P = u \cdot \mathrm{P}_A + v \cdot \mathrm{P}_B + w \cdot \mathrm{P}_C \,, \quad u + v + w = 1 \tag{11}$$

$$u = \frac{\mathrm{S}_{\Delta BCP}}{\mathrm{S}_{\Delta ABC}} \,, \; v = \frac{\mathrm{S}_{\Delta CAP}}{\mathrm{S}_{\Delta ABC}} \,, \; w = \frac{\mathrm{S}_{\Delta ABP}}{\mathrm{S}_{\Delta ABC}} \tag{12}$$

  And, if P is inside the triangle, $u \geq 0$, $v \geq 0$, $w \geq 0$

- According to formulas (1) and (6),

$$u = \frac{F_{02}}{F_{01}+F_{02}+F_{03}} = \frac{|\overrightarrow{BC}\times\overrightarrow{BP}|}{|\overrightarrow{AB}\times\overrightarrow{AP}|+|\overrightarrow{BC}\times\overrightarrow{BP}|+|\overrightarrow{CA}\times\overrightarrow{CP}|} = \frac{2\times\mathrm{S}_{\Delta BCP}}{2\times\mathrm{S}_{\Delta ABC}} \tag{13}$$

  the other two coefficients are the following:

$$v = \frac{F_{03}}{F_{01}+F_{02}+F_{03}} \tag{14}$$

$$w = \frac{F_{01}}{F_{01}+F_{02}+F_{03}} \tag{15}$$

Also can see [3].

4. Perspective-Correct Interpolation[4]

- Barycentric coordinates are not invariant under projection, it can not produce the perspective correct result. we should use the interpolation to get the correct z-depth as the following:

$$Z_p = \frac{1}{\frac{u}{Z_A}+\frac{v}{Z_B}+\frac{w}{Z_C}} \tag{16}$$

According to formulas (13,14,15) and rearranging the factors, the above formulas can be described as:

$$Z_p = \frac{1}{\frac{\frac{F_{02}}{F_{01}+F_{02}+F_{03}}}{Z_A}+\frac{\frac{F_{03}}{F_{01}+F_{02}+F_{03}}}{Z_B}+\frac{\frac{F_{01}}{F_{01}+F_{02}+F_{03}}}{Z_C}} = \frac{F_{01}+F_{02}+F_{03}}{\frac{F_{02}}{Z_A}+\frac{F_{03}}{Z_B}+\frac{F_{01}}{Z_C}} = \frac{2\times S_{\Delta ABC}}{\frac{F_{02}}{Z_A}+\frac{F_{03}}{Z_B}+\frac{F_{01}}{Z_C}} \tag{17}$$

Then,

$$\frac{2\times S_{\Delta ABC}}{Z_p} = \frac{F_{02}}{Z_A}+\frac{F_{03}}{Z_B}+\frac{F_{01}}{Z_C}$$

For convenience, written as:

$$\frac{2\times S_{\Delta ABC}}{Z_p} = D\left(P_x\,,\ P_y\right) = \frac{F_{02}}{Z_A}+\frac{F_{03}}{Z_B}+\frac{F_{01}}{Z_C} \tag{18}$$

And it easy to will find out that:

$$D\left(P_{x+1}\,,\ P_y\right) - D\left(P_x,\ P_y\right) = I_{00} = \frac{I_{02}}{Z_A}+\frac{I_{03}}{Z_B}+\frac{I_{01}}{Z_C} \tag{19}$$

$$D\left(P_x\,,\ P_{y+1}\right) - D\left(P_x,\ P_y\right) = J_{00} = \frac{J_{02}}{Z_A}+\frac{J_{03}}{Z_B}+\frac{J_{01}}{Z_C} \tag{20}$$

$$K_{00} = \frac{K_{02}}{Z_A}+\frac{K_{03}}{Z_B}+\frac{K_{01}}{Z_C} \tag{21}$$

$$F_{00} = I_{00}\cdot P_x + J_{00}\cdot P_y + K_{00} \tag{22}$$

So the perspective correct z-depth can also use for iteration-based algorithm.

- Now there is a vector $(F_{00}\,,\ F_{01}\,,\ F_{02}\,,\ F_{03})$, if divided by $F_{01}+F_{02}+F_{03}$ (the value is $2\times S_{\Delta ABC}$) , the $\left(\frac{1}{Z_p}\,,\ w\,,\ u\,,\ v\right)$ is given. It is friendly to CPU/GPU calculation, due to memory alignment and SIMD fitting.

- Additionally, use the interpolation to get other correct attributes as the following:

$$P_p = \left(u\cdot\frac{P_A}{Z_A}+v\cdot\frac{P_B}{Z_B}+w\cdot\frac{P_c}{Z_C}\right)/\,\frac{1}{Z_p} = \left(u\cdot\frac{P_A}{Z_A}+v\cdot\frac{P_B}{Z_B}+w\cdot\frac{P_c}{Z_C}\right)\cdot Z_p \tag{23}$$

- The pseudo code is the following:

```
[ minX, maxX, minY, maxY ] = CalcuteBoundingBox(triangle.vertices[3],screenSize)
[ I01, I02, I03 ] = CalcuteDeltaX(triangle.vertices[3])
[ J01, J02, J03 ] = CalcuteDeltaY(triangle.vertices[3])
[ K01, K02, K03 ] = CalcuteConst(triangle.vertices[3])
[ I00, J00, K00 ] = CalcuteConst(triangle.vertices[3],
                                 I01, I02, I03,
                                 J01, J02, J03,
                                 K01, K02, K03)

/** __mm128 is SIMD struct, details see intel.com */
I = Make __mm128(I00, I01, I02, I03)
J = Make __mm128(J00, J01, J02, J03)
K = Make __mm128(K00, K01, K02, K03)

F = I * minX + J * minY + K;

Area2 = F[1] + F[2] + F[3]
A = Make __mm128(Area2, Area2, Area2, Area2)

/** ( 1 / depth, gamma, alpha, beta ) at ( minX, minY ) */
F = F / A

/** dx, dy */
I = I / A
J = I / A


Cy = F
loop y = minY to maxY with step 1
{
    Cx = Cy
    loop x = minX to maxX with step 1
    {
        /** Determines whether the point (x, y) is inside the triangle. */
        if (Cx[1] >= 0 && Cx[2] >= 0 && Cx[3] >= 0)
        {
           Depth = 1 / Cx[0];

           /** Z-depth testing. */
           if (DepthBuffer(x, y) < Depth)
           {
             DepthBuffer(x, y) = Depth
             Render(x, y)

           }
        }
        Cx += I
    }
    Cy += J
}
```

References

[1] *vectors cross product*[DB/OL]
https://www.mathsisfun.com/algebra/vectors-cross-product.html

[2] Mileff P , K Nehéz , Dudra J . *Accelerated Half-Space Triangle Rasterization*[J]. Acta Polytechnica Hungarica, 2015, 12(7):2015-2217.

[3] *Barycentric coordinate system*[DB/OL]
https://en.wikipedia.org/wiki/Barycentric_coordinate_system

[4] Low K L . *Perspective-Correct Interpolation*[J].
Springer Proceedings in Mathematics & Statistics, 2002.