



计算机图形学小白入门

——从0开始实现OpenGL

多边形剪裁算法

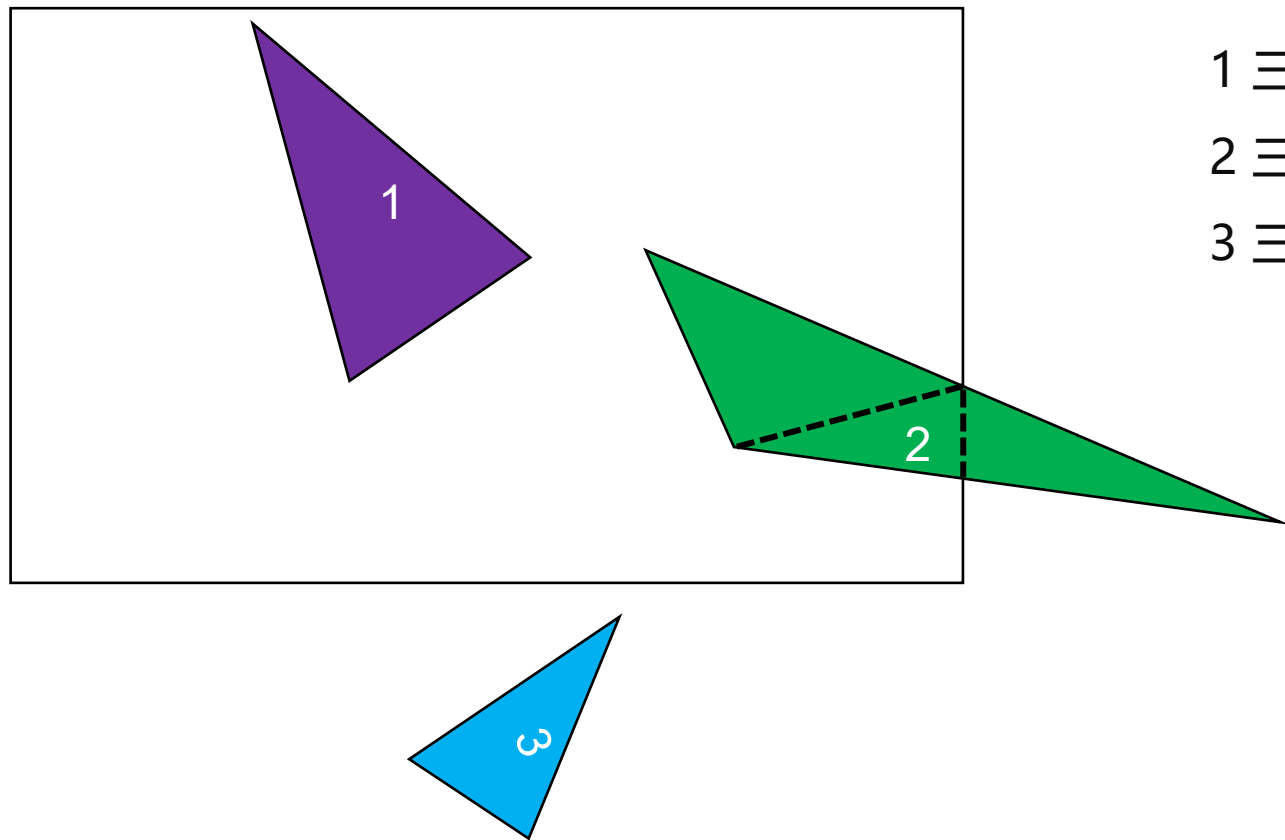


授课：赵新政
资深三维工程师

专注3D图形学技术
教育品牌

目标分析

- 分析对于一个Mesh的一次Draw调用，其中的三角形会产生三种情况，我们需要做到：



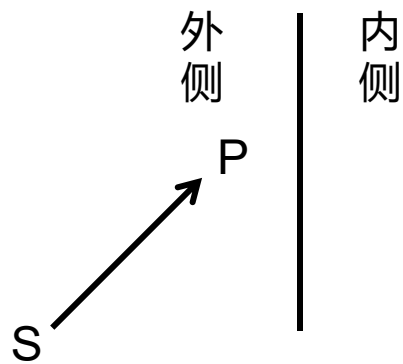
- 1 三角形在可视范围内，顶点都保留
- 2 三角形与可视范围相交，去掉外围点，产生新点
- 3 三角形不在可视范围内，丢弃所有顶点

Sutherland-Hodgman算法

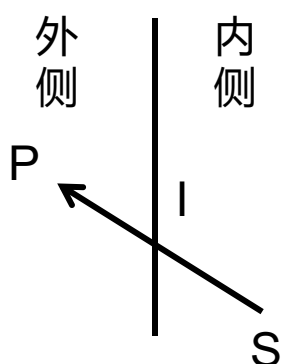
- Sutherland-Hodgman算法也叫逐边裁剪法，该算法采用了分割处理、逐边裁剪的方法。

算法思想（二维）

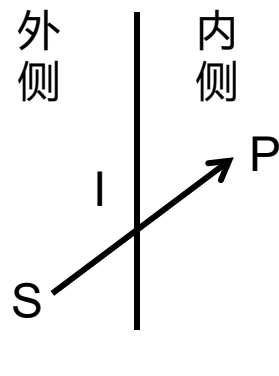
- 一次用窗口的一条边裁剪多边形，循环多次；
- 每次，构造一个空的点数组DST；原数组为SRC
- 每次，所有顶点从0号开始，作为S点，S后面的点为P点，依次进行测试，结果输出点到DST，剪裁原则如下图；
- SRC=DST;



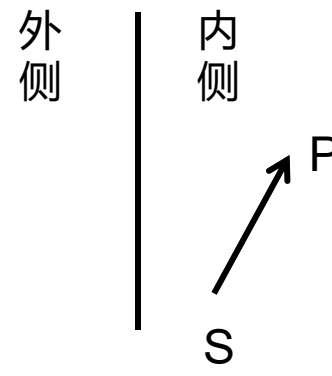
情况1：无输出



情况2：输出交点I



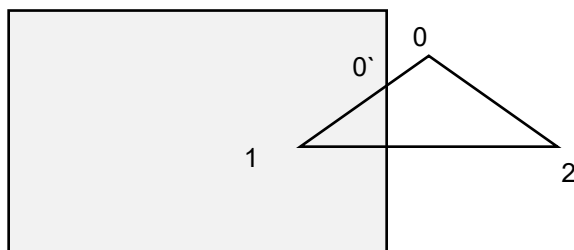
情况3：输出交点I与P



情况4：输出P

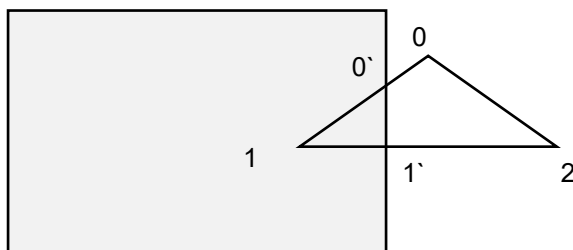
Sutherland-Hodgman算法举例（一）

输入0-1-2
考察**右侧边**



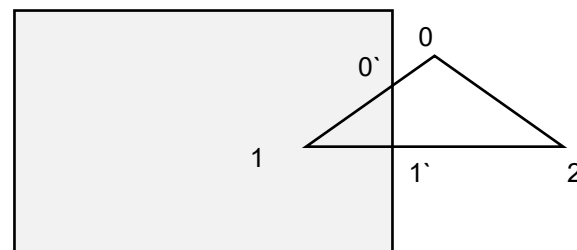
S=0, P=1 情况3

| | |
|----|---|
| 0' | 1 |
|----|---|



S=1, P=2 情况2

| | | |
|----|---|----|
| 0' | 1 | 1' |
|----|---|----|



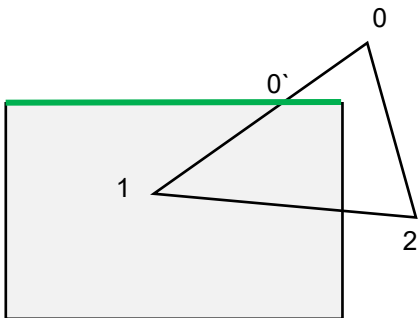
S=2, P=0 情况1

| | | |
|----|---|----|
| 0' | 1 | 1' |
|----|---|----|

如法炮制考察**剩余三边**后，输出三角形为0'-1-1'的点

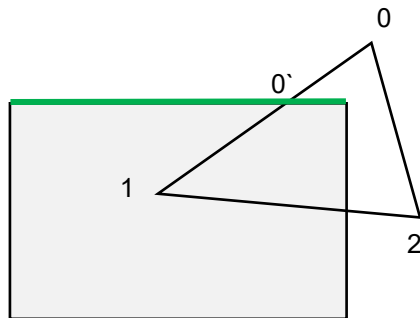
Sutherland-Hodgman算法举例（二）

输入0-1-2
考察上侧边



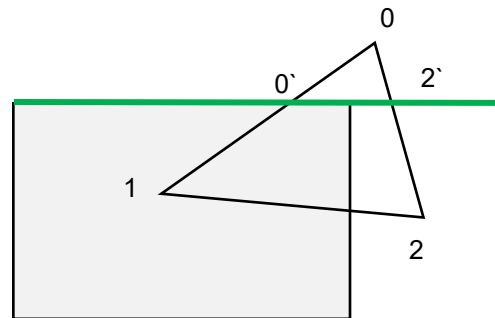
$S=0, P=1$ 情况3

| | |
|----|---|
| 0' | 1 |
|----|---|



$S=1, P=2$ 情况4

| | | |
|----|---|---|
| 0' | 1 | 2 |
|----|---|---|



$S=2, P=0$ 情况2

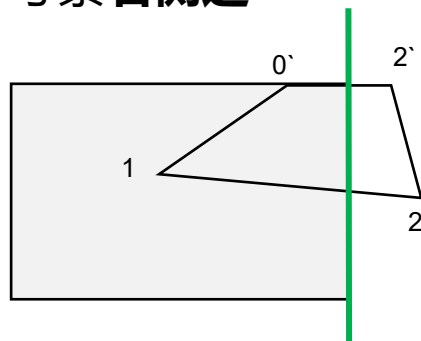
| | | | |
|----|---|---|----|
| 0' | 1 | 2 | 2' |
|----|---|---|----|

如法炮制考察剩余三边后，输出0'-1-2-2'

Sutherland-Hodgman算法举例（二）

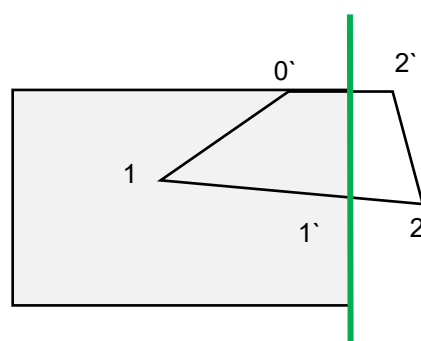
输入 $0'-1-2-2'$

考察右侧边



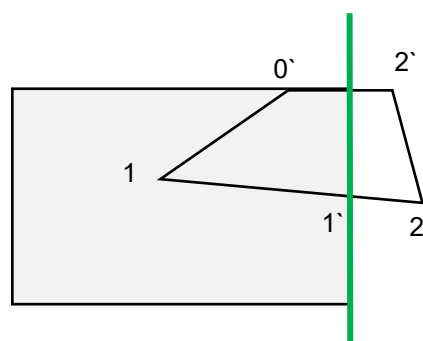
$S=0'$, $P=1$ 情况4

| |
|---|
| 1 |
|---|



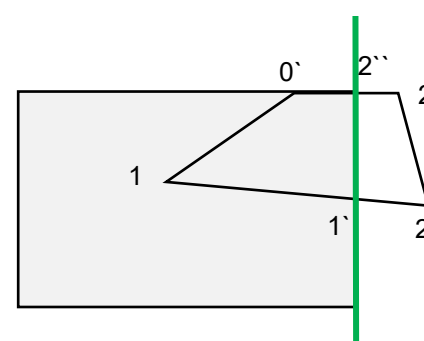
$S=1$, $P=2$ 情况2

| | |
|---|----|
| 1 | 1' |
|---|----|



$S=2$, $P=2'$ 情况1

| | |
|---|-----|
| 1 | 1'' |
|---|-----|



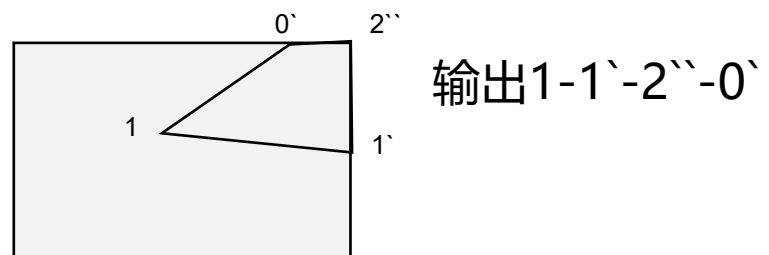
$S=2'$, $P=0'$ 情况3

| | | | |
|---|-----|-----|----|
| 1 | 1'' | 2'' | 0' |
|---|-----|-----|----|

如法炮制考察剩余三边后，输出 $1-1''-2''-0'$

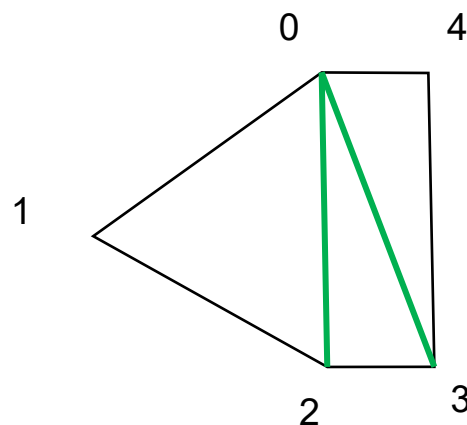
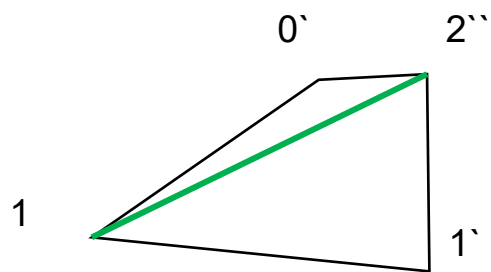
Sutherland-Hodgman算法举例（二）

对其他两条边如法炮制，最终可以获得下方所示数组：



接下来进行三角形重建：永远以数组第一个顶点为起始点，递进构建三角形：

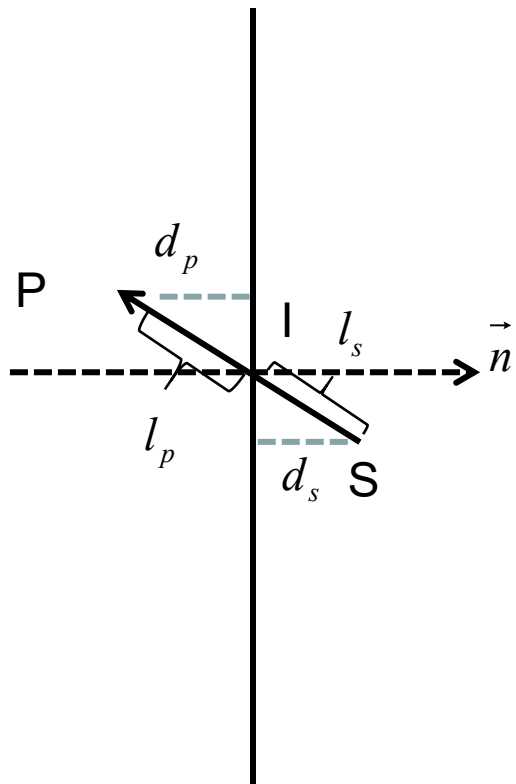
- 即：1-1'-2'' 1-2''-0'



新点属性插值

当两个点位于平面内外两侧，我们需要对其进行插值，从而得到相交点的位置/颜色/uv等属性

注意，下方的L/d都是有向距离



插值计算流程

- 得到边界线的方程表达式
- 讲p/s点带入求出与边界的距离
- 选择任何一条距离作为系数，执行线性插值

$$weight = \frac{l_s}{l_s - l_p} = \frac{d_s}{d_s - d_p}$$

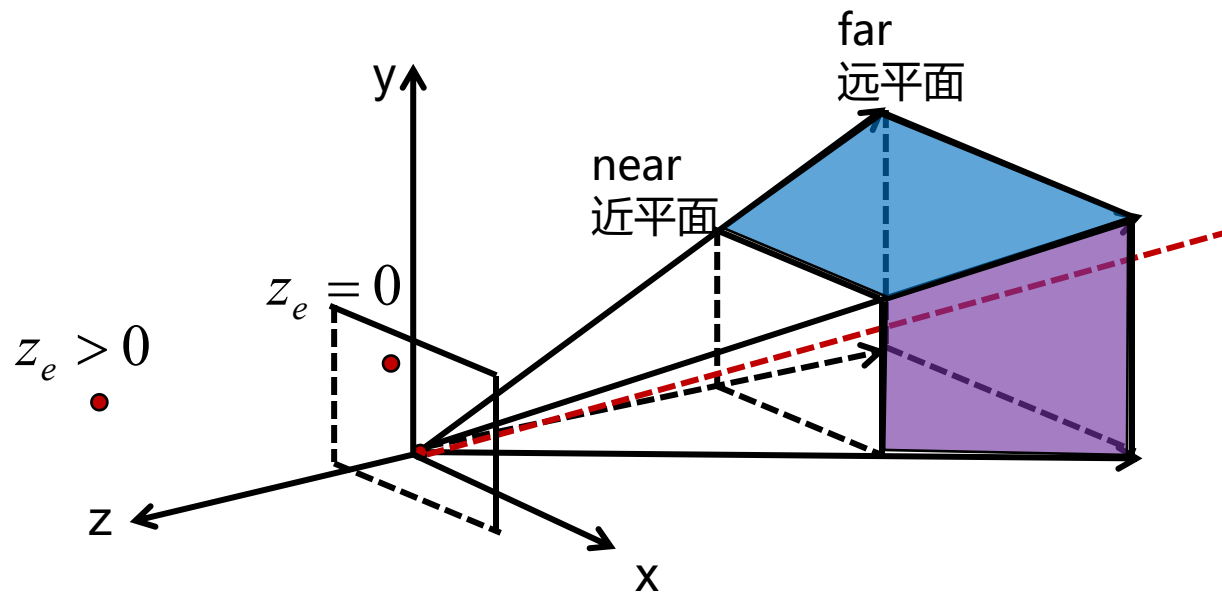
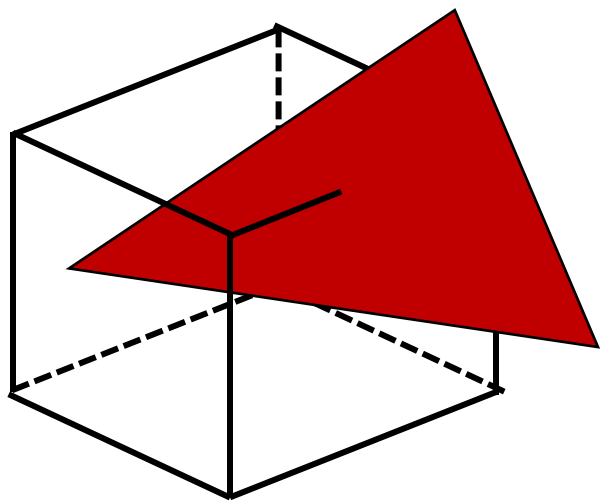
$$I_position = P_position.weight + S_position.(1 - weight)$$

$$I_color = P_color.weight + S_color.(1 - weight)$$

$$I_uv = P_uv.weight + S_uv.(1 - weight)$$

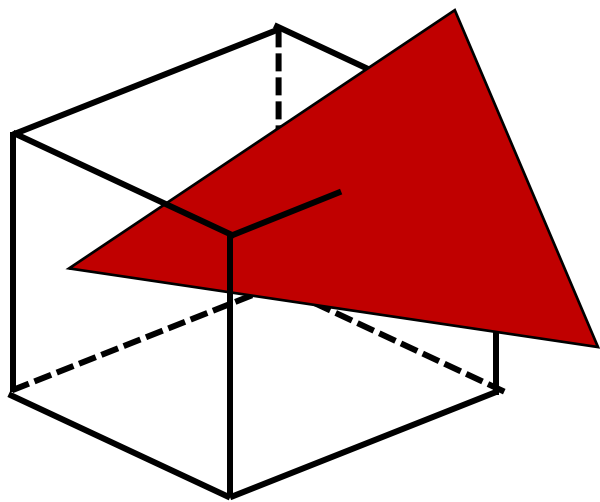
Sutherland-Hodgman算法三维

- 在维情况下，剪裁边变成了剪裁平面；我们选择在NDC构成的-1到1盒体内进行剪裁
- BUT 必须保证：
 - 顶点在摄像机前方
 - 顶点z值不为0（即不与摄像机重合）



判定条件

- 根据上述条件以及NDC下点坐标必须为-1到1内，可使用**剪裁空间坐标**对内外进行判定：



$$0 < w_c$$

$$-1 < \frac{x_c}{w_c} < 1$$

$$-1 < \frac{y_c}{w_c} < 1$$

$$-1 < \frac{z_c}{w_c} < 1$$

转化为平面方程



注意：优先判定了 w 大于0，所以可以直接把 w 乘到两边

$$w_c > 0$$

$$-x_c + w_c > 0$$

$$x_c + w_c > 0$$

$$-y_c + w_c > 0$$


$$y_c + w_c > 0$$

$$-z_c + w_c > 0$$

$$z_c + w_c > 0$$

判定条件研究

- 判定条件其实是一个四维空间平面，法线可以抽取，并且常数 $d=0$ ，比如：

| | | |
|--------------------------------------|--|--------------------|
| $0.x_c + 0.y_c + 0.z_c + 1.w_c > 0$ | | $(0 \ 0 \ 0 \ 1)$ |
| $-1.x_c + 0.y_c + 0.z_c + 1.w_c > 0$ | | $(-1 \ 0 \ 0 \ 1)$ |
| $1.x_c + 0.y_c + 0.z_c + 1.w_c > 0$ | | $(1 \ 0 \ 0 \ 1)$ |
| $0.x_c + -1.y_c + 0.z_c + 1.w_c > 0$ | 平面法线  | $(0 \ -1 \ 0 \ 1)$ |
| $0.x_c + 1.y_c + 0.z_c + 1.w_c > 0$ | | $(0 \ 1 \ 0 \ 1)$ |
| $0.x_c + 0.y_c + -1.z_c + 1.w_c > 0$ | | $(0 \ 0 \ -1 \ 1)$ |
| $0.x_c + 0.y_c + 1.z_c + 1.w_c > 0$ | | $(0 \ 0 \ 1 \ 1)$ |

- 任意点带入方程后，得到的结果，都是距离判定平面的“有向距离”
- 插值交叉点的方法，也可采用前述的距离插值方法