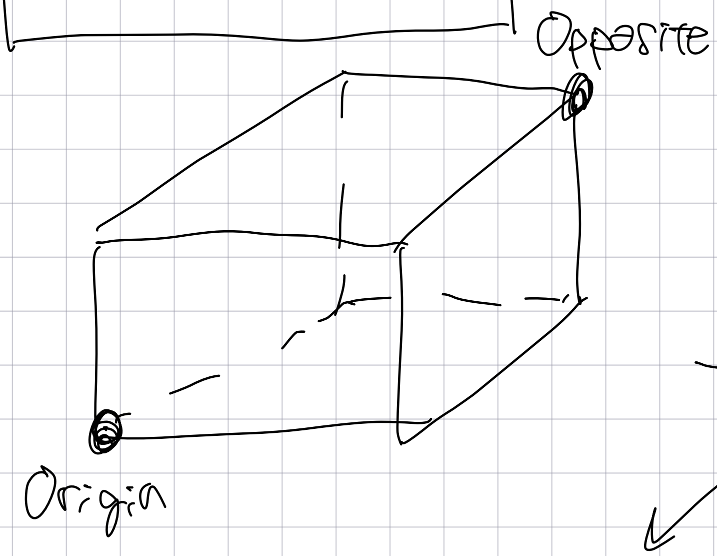
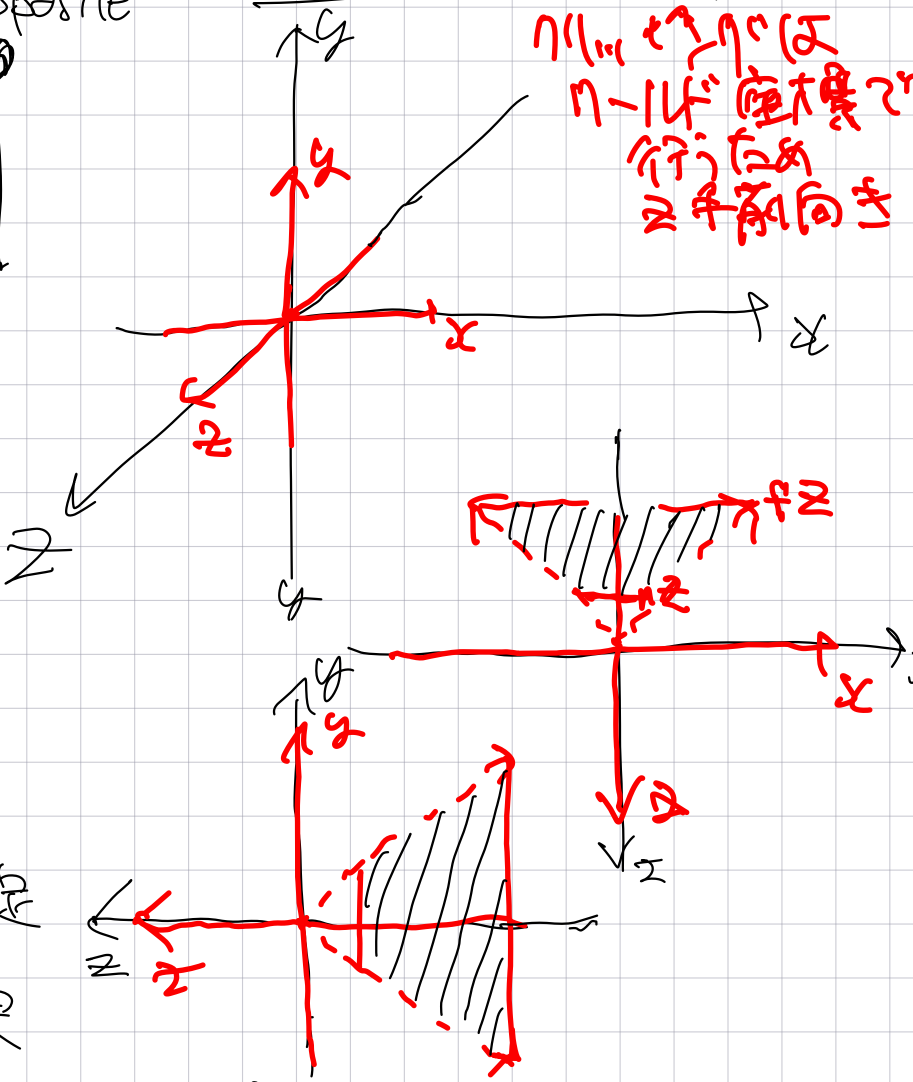
RANGE_CUBE



CLIPPING



〈処理順〉

1、xz軸での交差判定

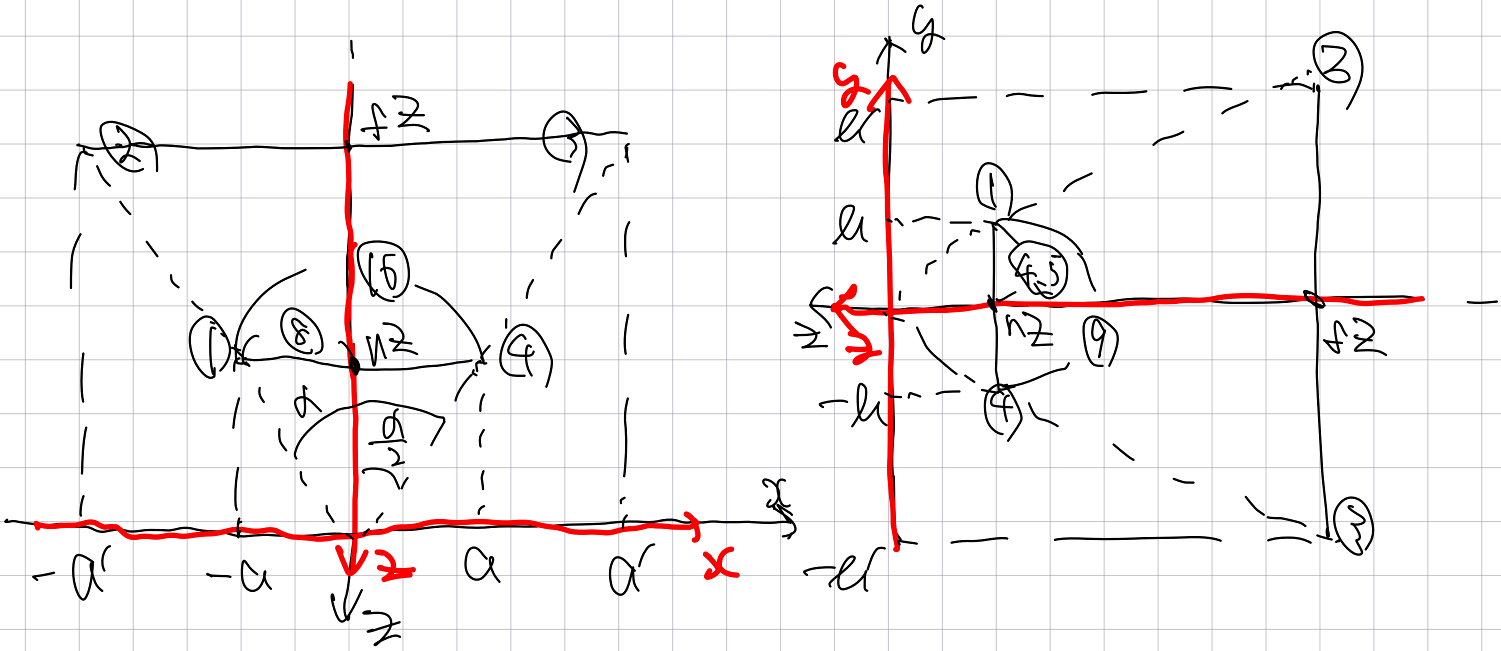
2、yz軸での交差判定

3、1及び2が真の場合、交差判定は真となり、
一方のみの場合、偽となる

<クリッピング領域の決定処理>

● 処理に関わる数値
- アスペクト比 ⟶ x成分X、y成分Y
- 視野角 ⟶ α
- nearZ ⟶ nZ
- farZ ⟶ fZ
- カメラ位置
- カメラ回転角度

このうち固定値は、アスペクト比、nZ、fZ



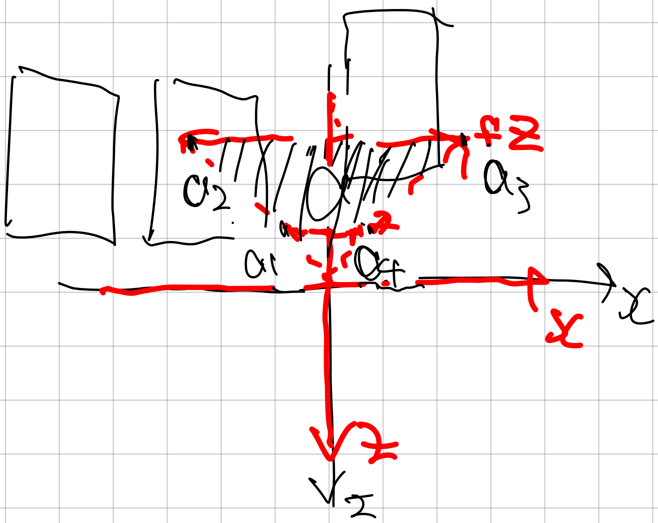
$$\tan\frac{\alpha}{2} = \frac{a}{nZ} \Leftrightarrow a = \tan\frac{\alpha}{2}\cdot nZ$$

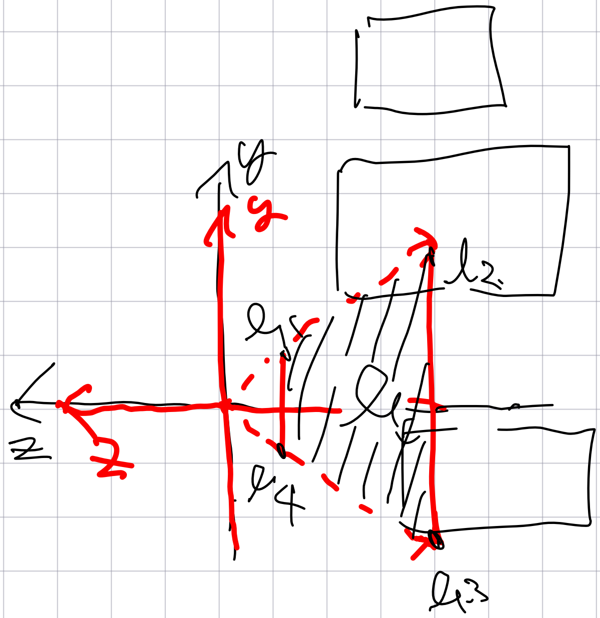
$$\frac{a}{8} = \frac{\tan\frac{\alpha}{2}\cdot nZ}{8}$$

$$a = \tan\frac{\alpha}{2}\cdot nZ \quad , \quad b = \tan\frac{\alpha}{2}\cdot nZ\cdot\frac{Y}{X}$$

$$a' = \tan\frac{\alpha}{2}\cdot fZ \quad , \quad b' = \tan\frac{\alpha}{2}\cdot fZ\cdot\frac{Y}{X}$$

<交差判定>





・xz軸でのクリッピング領域をα、yz軸でのものをlとする

$(x_1, y_1)(x_2, y_2)$

2点を通る直線の方程式は

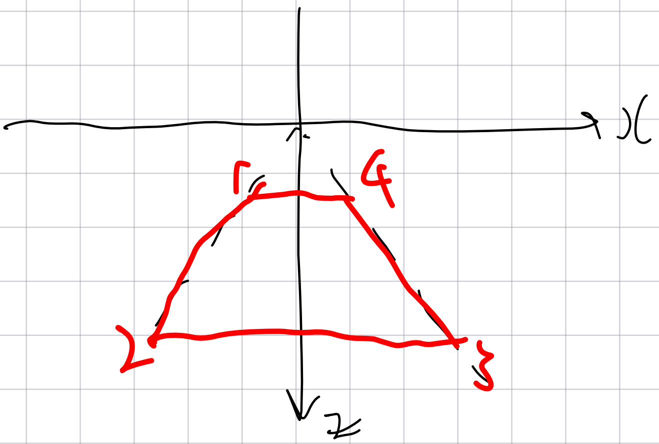
$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) \iff y = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) + y_1$$

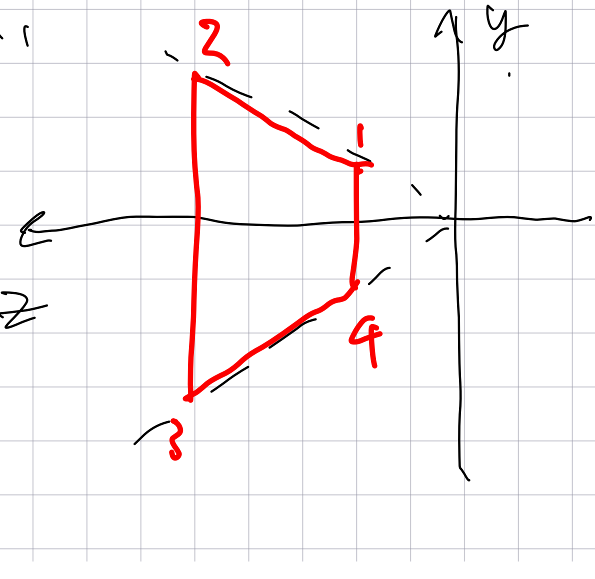
xz軸、yz軸で共有の軸はz軸であるよりx、yをα、lとして上の式を変えて

$$z = \frac{z_2 - z_1}{\alpha_2 - \alpha_1}(\alpha - \alpha_1) + z_1$$

これに不等号を使用して4つの式でα、lそれぞれの領域を定義して交差での判定を行う。

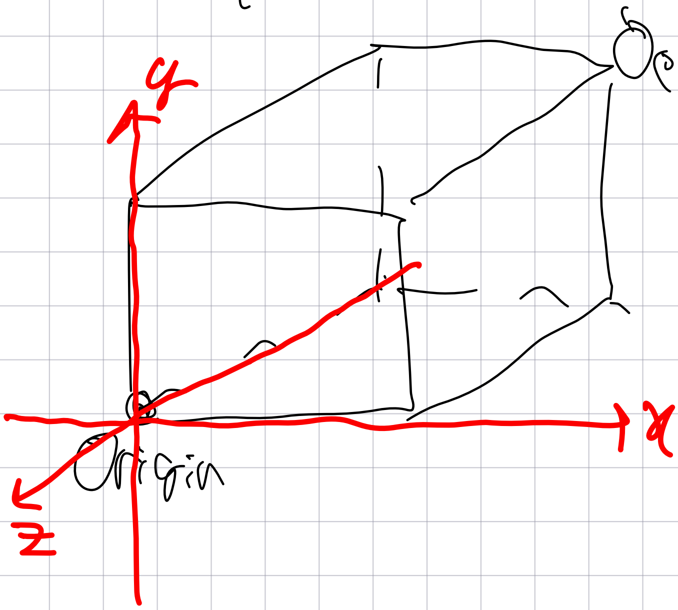
$$\alpha = \frac{\alpha_2 - \alpha_1}{z_2 - z_1}(z - z_1) + \alpha_1$$



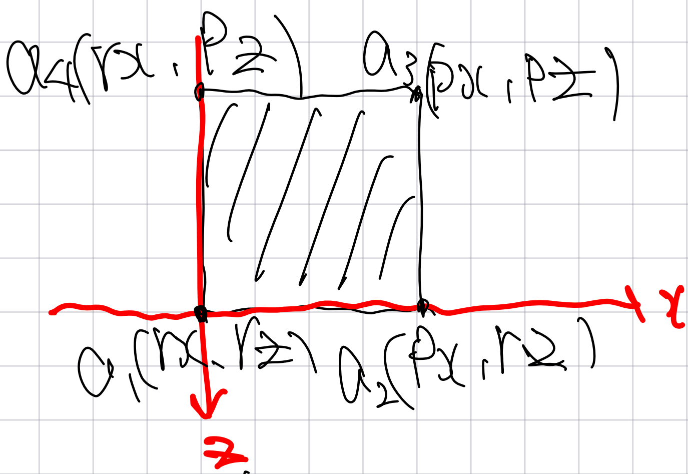


<RANGE_CUBEのxz, yz軸での領域>

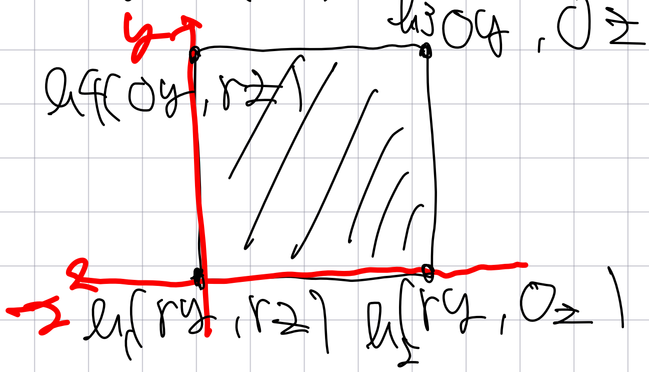
Origin→r, Opposit→p



<xz軸>



<yz軸>



$$z = \frac{z_2 - z_1}{x_2 - x_1}(x - x_1) + z_1$$

と不等号を用いて領域を絞る

<RangeCubeの回転による値の変化>

・Origin, Oppositだけでなく長方体の8頂点の座標を格納する必要がある。

・8頂点の移動、回転後、新たにOrigin, Oppositを決める

（オブジェクトのアニメーション処理はどこでする？）

・その後、領域の重なりによる判定を行う。

<ポリゴンの表裏判定とクリッピング>

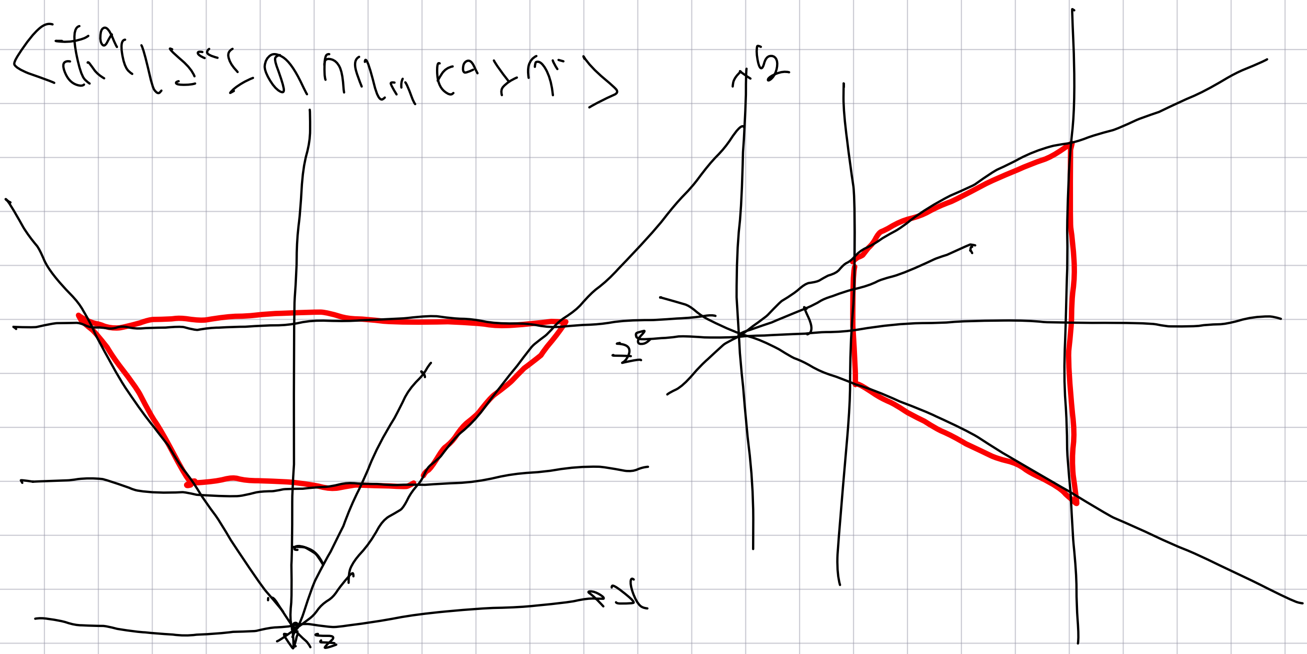
<表裏判定>

ポリゴンの法線ベクトル（3点の点のベクトルはすべて等しいことが条件）と、カメラPosから面へ向かうベクトルとの内積を行い、その値が正の場合にのみ面は表を向いているとする

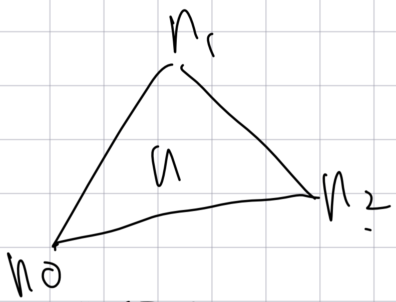
$$\vec{a}\begin{vmatrix} a_1 \\ a_2 \\ a_3 \end{vmatrix} \cdot \vec{b}\begin{vmatrix} b_1 \\ b_2 \\ b_3 \end{vmatrix} = a_1b_1 + a_2b_2 + a_3b_3$$ で内積が求められる

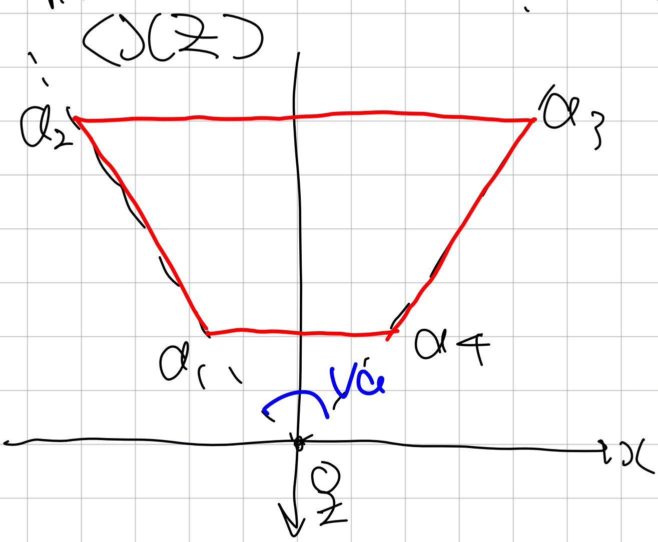
◎ 3×1行列と3×1行列の内積の値を求め正負の結果を配列に保存するGPU関数が必要

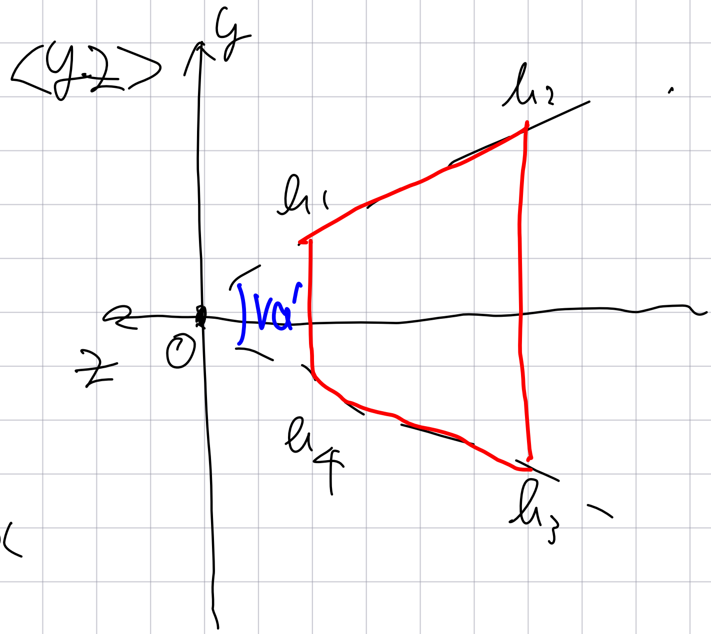
<ポリゴンのクリッピング>



<ポリゴンのクリッピング>







□処理順

← これ解そう

1. ポリゴンの頂点と原点Oとの角度をxz軸、yz軸それぞれで求める

2. ポリゴンの3辺でビューボリュームの8面での交点を求め、交点が1つでもある場合、ポリゴンはビューボリュームの内側に一部でも存在することになる(辺x,y,zで不等号で領域指定)
(もしくはVa, Va', nz, fzを用いて角度で判定) ← これダメそう

3、2で交点が無かった面に対してビューボリュームの12辺とポリゴンの内部を表すベクトルで交点を求めそのベクトルの大きさがポリゴン内部の点を指すものであればポリゴンはビューボリュームの内側に一部でも存在することになる

12辺でなくてもwidthとheightでそれぞれ4本ずつで良さそう

＜計算量比較＞

①角度の場合

・すべてのポリゴンの(num1番号)頂点に対して角度を求める計算を行う

・1点も角度内でかつz範囲にないポリゴンに対して、ポリゴンの辺で、ビューボリュームの平面との交点を求め、その値が平面の値のはんい内かで調べる

・ポリゴンの3辺とビューボリュームの8角のどれも範囲外のポリゴンに対してビューボリュームの4辺とポリゴンの交点を求めそれがビューボリュームの辺の辺上にあるかどうか判定を行う。

②最初から交点を求める場合(交点取得法とする)

①ポリゴンの3辺とビューボリュームの6面の平面との交点を求め、それがx, y, z軸それぞれで辺上にあるか判定する

②判定でポリゴンの3辺のどれもfalseの場合、
ポリゴンの3辺とビューボリュームの8角のどれも範囲外のポリゴンに対してビューボリュームの4辺とポリゴンの交点を求めそれがビューボリュームの辺の辺上にあるかどうか判定を行う。

<交点取得法①>

□直線の方程式

ポリゴンの頂点を $n1$, $n2$, $n3$ とするとき直線は

n_1n_2, n_2n_3, n_3n_1 の辺

ここでは1つの直線を $A\begin{pmatrix} x \\ y \\ z \end{pmatrix}$, $B\begin{pmatrix} l \\ m \\ n \end{pmatrix}$ で表すとする

□平面の方程式

・平面上の1点はビューボリュームの対角の2点の座標から求められる
2点を α, β とすると

$$\alpha\begin{pmatrix} x \geqq VP1 \\ y \geqq VP4 \\ x \geqq VP1 \end{pmatrix}, \quad \beta\begin{pmatrix} x \geqq VP3 \\ y \geqq VP2 \\ x \geqq VP3 \end{pmatrix}$$ となる

各面の法線ベクトルは $\vec{n}\begin{pmatrix} p \\ q \\ r \end{pmatrix}$ とし、各面の1点を $A\begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix}$

とすると

$$p(x-x_0)+q(y-y_0)+r(z-z_0)=0$$

$$px-px_0+qy-qy_0+rz-rz_0=0$$

$$px+qy+rz-px_0-qy_0-rz_0=0$$

各成分を a, b, c, d とすると

$$a=p,\quad b=q,\quad c=r,\quad d=-px_0-qy_0-rz_0$$

となる

〈直線と平面の交点の取得〉

・式の変形　$x = x + tl$, $y = y + tm$, $z = z + tn$

各面の法線ベクトルは $\vec{n}\begin{pmatrix} p \\ q \\ r \end{pmatrix}$ とし、各面の1点をA$\begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix}$

$$a = p,\ b = q,\ c = r,\ d = -px_0 - qy_0 - rz_0$$ より

$$p(x + tl) + q(y + tm) + r(z + tn) + d = 0$$

$$px + ptl + qy + qtm + rz + rtn + d = 0$$

$$t(pl + qm + rn) = -px - qy - rz - d$$

$$t = \frac{-px - qy - rz + px_0 + qy_0 + rz_0}{pl + qm + rn}$$

$$= \frac{p(-x + x_0) + q(-y + y_0) + r(-z + z_0)}{pl + qm + rn}$$

tを直線の方程式に代入することで交点の座標が求まる
交点をIとすると以下の式になる

$$I\begin{pmatrix} Ix \\ Iy \\ Iz \end{pmatrix} = \begin{pmatrix} x + tl \\ y + tm \\ z + tn \end{pmatrix}$$

<交点取得法②>

① 直線(ビューボリュームの4辺)の方程式

4辺はnzスクリーン面とfzスクリーン面の頂点をそれぞれつなぐ2辺とnz,fzスクリーンの横向きの辺です
viewPointの番号で直線を表す(辺はLです)

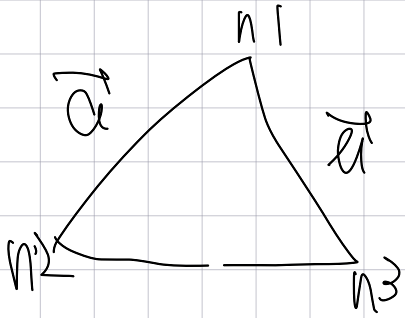
L1 → VP4, VP5
L2 → VP5, VP1
L3 → VP1, VP0
L4 → VP0, VP4

$$z = \frac{z_2 - z_1}{\alpha_2 - \alpha_1}(\alpha - \alpha_1) + z_1$$

$$\alpha = \frac{\alpha_2 - \alpha_1}{z_2 - z_1}(z - z_1) + \alpha_1$$

により求められる

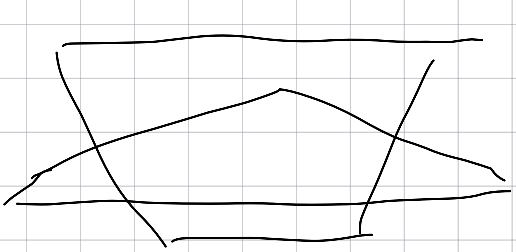
② 三角形上の頂点の表し方



$\overrightarrow{n_1 n_2}$, $\overrightarrow{n_1 n_3}$ の単位ベクトルを $\vec{a}$, $\vec{b}$ とし、それぞれの大きさを $|\vec{a}|$, $|\vec{b}|$ または s と t を

$s + t \leq 1$, $s \geq 0$, $t \geq 0$

とし三角形上の一点は($\vec{p}$)



$$\vec{p} = s\vec{a} + t\vec{b}$$

1. 面と直線の交点を求める。($\vec{p}$を求める) αにはx,yが入る
2. 交点が辺上かどうか判定
3. 辺上の場合、三角形内部か判定
4. TRUEで描画対象

$$\vec{p}(\alpha) = s\vec{a}(\alpha) + t\vec{b}(\alpha)$$

$$s = \frac{\vec{p}(x) - t\vec{b}(x)}{\vec{a}(x)} = \frac{\vec{p}(y) - t\vec{b}(y)}{\vec{a}(y)}$$

$$s=\frac{\vec{p}(x)-t\vec{l}(x)}{\vec{a}(x)}=\frac{\vec{p}(y)-t\vec{l}(y)}{\vec{a}(y)}$$

$$\overrightarrow{ay}\left(\overrightarrow{px}-t\overrightarrow{lx}\right)=\overrightarrow{ax}\left(\overrightarrow{py}-t\overrightarrow{ly}\right)$$

$$\overrightarrow{ay}\cdot\overrightarrow{px}-\overrightarrow{ax}\cdot\overrightarrow{py}=\overrightarrow{ay}\cdot t\overrightarrow{lx}-\overrightarrow{ax}\cdot t\overrightarrow{ly}$$

$$=t\left(\overrightarrow{ay}\cdot\overrightarrow{lx}-\overrightarrow{ax}\cdot\overrightarrow{ly}\right)$$ より

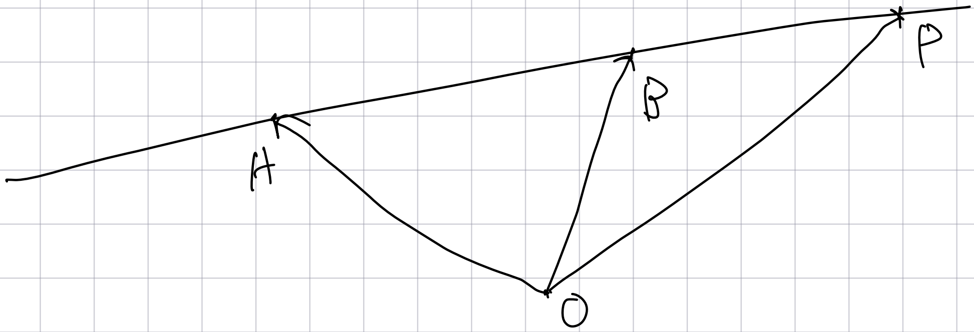
$$t=\frac{\overrightarrow{ay}\cdot\overrightarrow{px}-\overrightarrow{ax}\cdot\overrightarrow{py}}{\overrightarrow{ay}\cdot\overrightarrow{lx}-\overrightarrow{ax}\cdot\overrightarrow{ly}}$$

sはtを代入し求めることで、

$s\geq 0$　$t\geq 0$，$s+t\leq 1$ のとき$\vec{p}$は三角形の辺上及び内部にあるより

$\vec{p}$に対応するポリゴンは描画対称

〈直線の点が線分上の点かどうかの判定〉



$\vec{P} = \vec{A} + t(\vec{B}-\vec{A})$, $\vec{B}-\vec{A}$を正規化すると
AB上にPがある場合tは$|\vec{AB}|$より小さくなる

〈3点,3辺すべてがビューボリューム外にある場合〉

・ポリゴンのx_{min}, y_{min}, z_{min}, x_{max}, y_{max}, z_{max}をIFで取得

・そこからRANGE_CUBEを作成

・ObjectのClippingRangeと同様の処理を行う

・範囲内にあれば、描画対象としてpolyInViewVolumeにpush_backする。

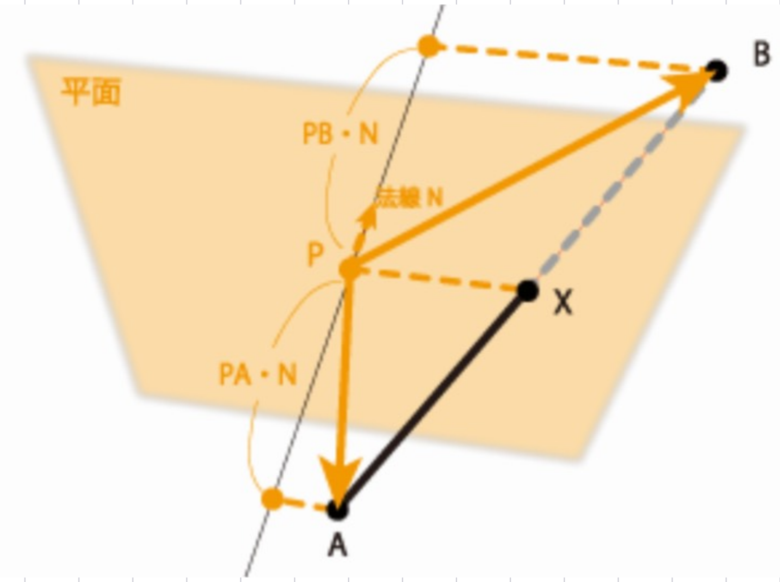
＜線分と多角形面の交点取得＞

1. 線分と平面の交点を求める：PHとN，PBとNの
それぞれの内積
交点の座標

2 PHとN，PBとNの内積から交点が存在するかの
判定を行い、交点が存在するポリゴン情報構造体の
インデックス番号及び、交点の座標を格納。

3、インデックス番号からポリゴン情報構造体のどのポリゴ
ンかを特定し、対応する交点がポリゴン内部か
判定し内部なら格納

〈線分と平面の交点の求め方〉

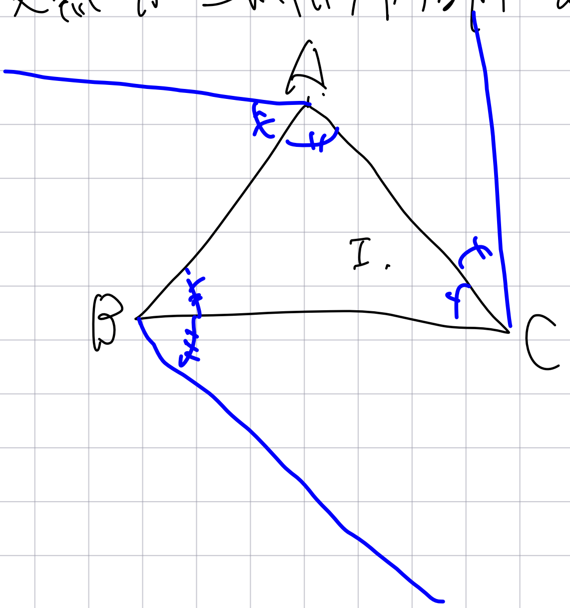


PAとNの内積と
PBとNの内積の符号が
異なれば交点は存在する
また $|AX|:|XB| = |PA \cdot N|:|PB \cdot N|$
※絶対値

※PAとN、PBとNの内積が0の場合は0.0001など近似値を使用する

よりXの3次元座標は $\vec{A} + \vec{AX} \cdot \dfrac{|XB| \cdot |PA \cdot N|}{|PB \cdot N|}$

〈交点が三角形内部にあるか判定〉



以下の3つの条件を同時に満たすとき内側に存在

① ABとAIの成す角が
ABとACの成す角より小さい

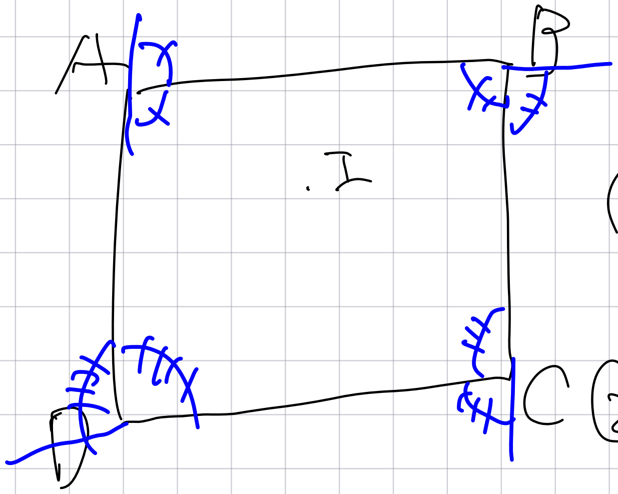
② BCとBIの成す角が
BCとBAの成す角より小さい

③ CAとCIの成す角が
CAとCBの成す角より小さい

LineVec → AB, BC, CA
[CA]

facePoint → A B C

<四角形の内側に点が存在するかどうか>



以下の3つの条件を同時に満たすとき内側に存在

① AB と AI の成す角が AB と AD の成す角より小さい

② BC と BI の成す角が BC と BA の成す角より小さい

③ CD と CI の成す角が CD と CB の成す角より小さい

④ DA と DI の成す角が DA と DC の成す角より小さい

LineVec → AB , BC , CD , DA
(DA)

facePoint → A B C D

LineVec[A+1] と (i − facePoint[A]) の成す角が
LineVec[A+1] と LineVec[A]×−1 の成す角より小さい

↑LineVec と facePoint を以上のように定義すれば成立っ

※(3)はベクトルを反転させるため−1をかける

1) (1)と(2)の成す角が(1)と(3)の成す角より小さい を言いかえると

$$\boxed{\begin{array}{ll} (1) = \vec{a}, & (2) = \vec{i} \\ (1) = \vec{a}, & (3) = \vec{e} \end{array}}$$

$\vec{a}$と$\vec{i}$の成す角が$\vec{a}$と$\vec{e}$の成す角より小さいときのことなり

$$\frac{\vec{a}\cdot\vec{i}}{|\vec{a}|\cdot|\vec{i}|} = \cos\theta$$

$$\mathrm{acos}\left(\frac{\vec{a}\cdot\vec{i}}{|\vec{a}|\cdot|\vec{i}|}\right)\cdot\frac{180}{\pi}$$

でθが求められる

$\vec{a}$と$\vec{e}$の場合は

$$\mathrm{acos}\left(\frac{\vec{a}\cdot\vec{e}}{|\vec{a}|\cdot|\vec{e}|}\right)\cdot\frac{180}{\pi}$$ となる

$$\mathrm{acos}\left(\frac{\vec{a}\cdot\vec{i}}{|\vec{a}||\vec{i}|}\right)\cdot\frac{180}{\pi} < \mathrm{acos}\left(\frac{\vec{a}\cdot\vec{e}}{|\vec{a}||\vec{e}|}\right)\cdot\frac{180}{\pi}$$

$$\mathrm{acos}\left(\frac{\vec{a}\cdot\vec{i}}{|\vec{a}||\vec{i}|}\right) < \mathrm{acos}\left(\frac{\vec{a}\cdot\vec{e}}{|\vec{a}|\cdot|\vec{e}|}\right)$$

(LineVec[A+1] と (i − facePoint[A]) の成す角が
LineVec[A+1] と LineVec[A]×−1 の成す角より小さい)

Aは3角形の場合 0〜2，四角形では0〜3

$$\mathrm{acos}\left(\frac{\vec{a}\cdot\vec{i}}{|\vec{a}||\vec{i}|}\right) < \mathrm{acos}\left(\frac{\vec{a}\cdot\vec{l}}{|\vec{a}|\cdot|\vec{l}|}\right)$$ から

$$\mathrm{acos}\left(\frac{LV[A+1]\cdot(i-fp[A])}{|LV[A+1]|\cdot|(i-fp[A])|}\right) \cdots ①$$

$$< \mathrm{acos}\left(\frac{LV[A+1]\cdot(LV[A]\cdot -1)}{|LV[A+1]|\cdot|(LV[A]\cdot -1)|}\right) \cdots ②$$

となる

GPUでは①、②をそれぞれ求める

<処理流れ>

・面の一点から線分の始点と終点への2つのベクトルを作り、それらと面の法線ベクトルの内積を計算する
・内積が0の場合は0.0001に変える
・交点の座標を求める
・三角形，四角形の場合でのそれぞれでのacosの値をすべて求める
・求めたacosから交点の内外判定を行う。

〈ポリゴンのビューボリュームによる多角形化について〉

①手順

1. ポリゴンの頂点のうち3つすべてビューボリューム内にある場合、そのポリゴンをレンダリング対象に加え、多角形化の対象から外す。また3点すべてビューボリューム外にあるポリゴンに対しては直方体範囲を作成し、ビューボリューム内に一部でも存在するか判定し、(メッシュのビューボリュームカリングと同様の処理)内部にないものは多角形化の対象から外す。

2. 手順1で対象ではないポリゴンは除外できているため、対象のポリゴンすべてに対して、ポリゴン線分とビューボリューム面との交点を求め格納する

3. 対象のポリゴンすべてに対してビューボリューム線分とポリゴン面との交点を求め、その交点が各ポリゴンの内部に存在する場合、格納する

4. 格納した頂点をすべてスクリーンピクセル座標に変換する

5、頂点データを格納しているデータの先頭の頂点と線分を成す頂点を探す(2次元での直線の方程式を使う)

(これで多角形化終わり)

12 GPUで行う処理

- 頂点ベクトルとビューボリューム軸ベクトルとの内積を求め、頂点がビューボリューム内にあるか判定し、ある場合、その頂点を格納する処理

- ベクトルの内積を求め、それから交差判定を行い、その後交点を求めることで平面と線分を求める行程をビューボリュームとポリゴンで相互に行い求めた頂点とビューボリューム内にある頂点すべてをスクリーンピクセル座標に変換する処理.