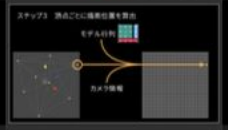
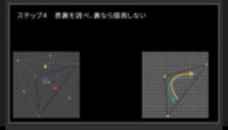
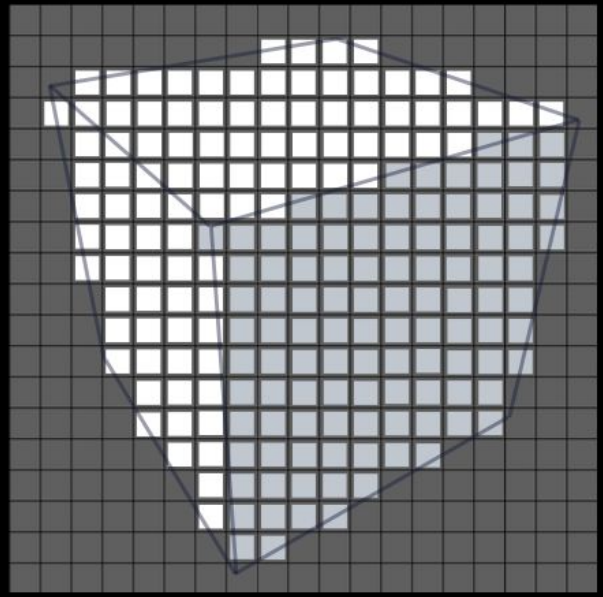
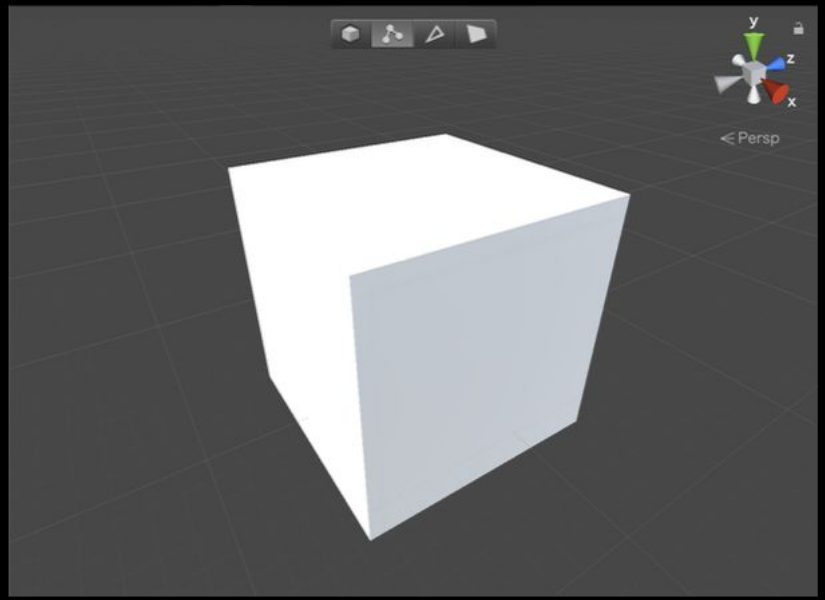
ステップ1		3Dモデルを準備
ステップ2		Transformの値を4x4行列に変換
ステップ3		頂点ごとに描画位置を算出
ステップ4		表裏を調べ、裏なら描画しない
ステップ5		描画点を確定
ステップ6		描画点をデプスバッファと比較
ステップ7		描画点に打つべき色を確定
ステップ8		点を打つ

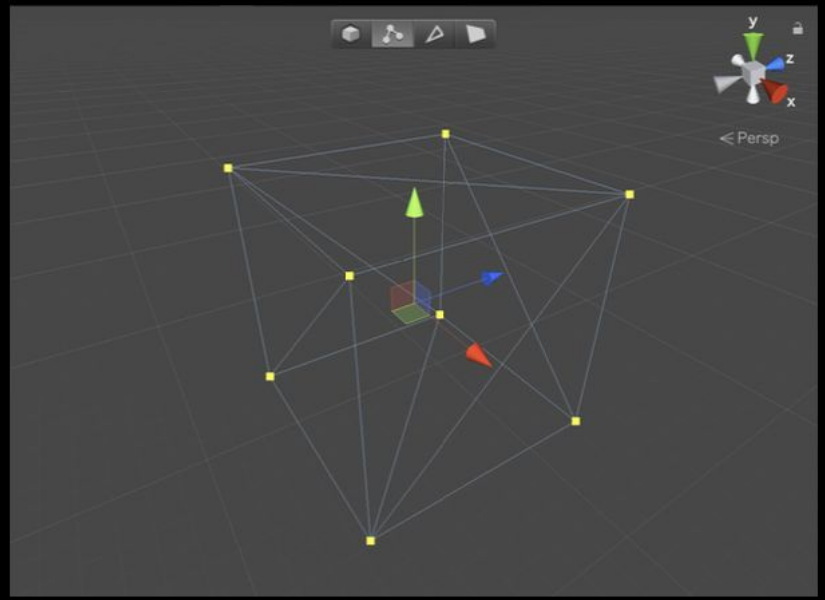
描画とは



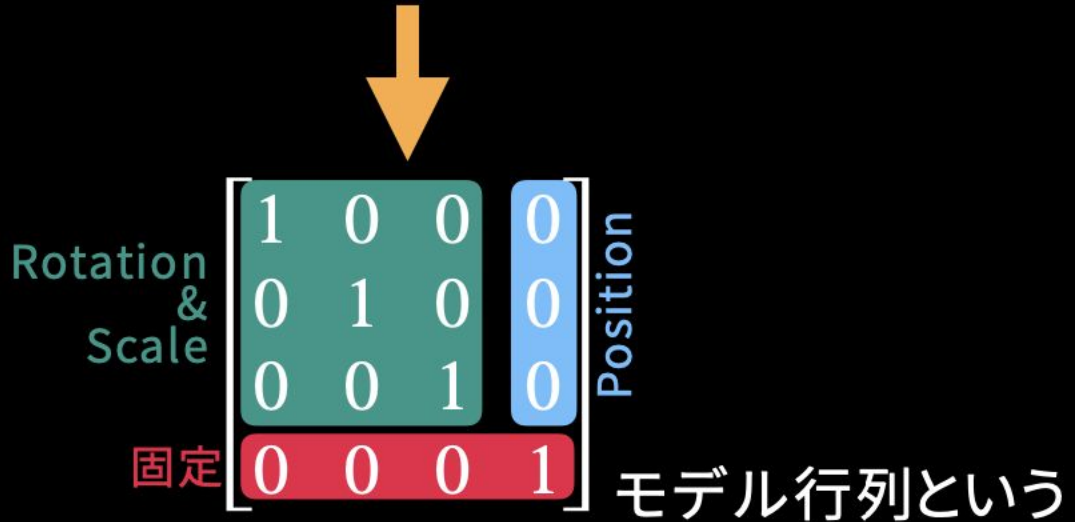
モニタ上の画素に色付きの点を打つこと

ステップ1　3Dモデルを準備





ステップ2　Transformの値を4x4行列に変換



モデル行列という

ステップ3　頂点ごとに描画位置を算出

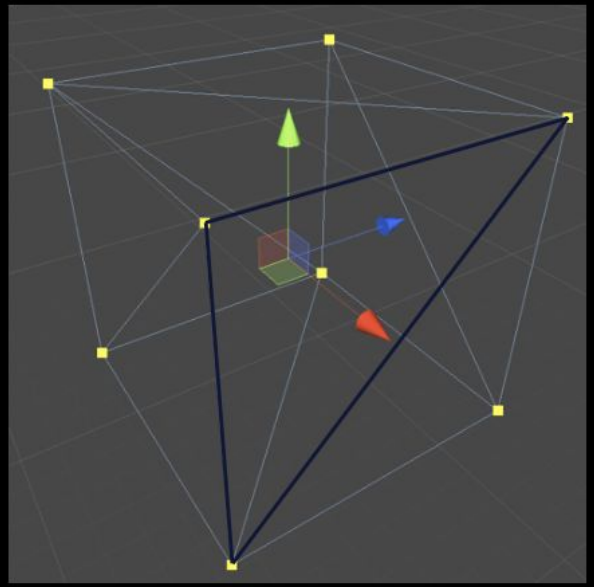
モデル行列

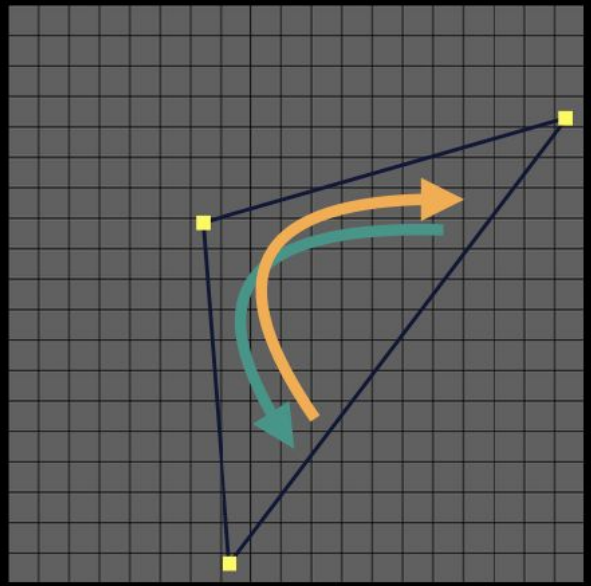
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

カメラ情報

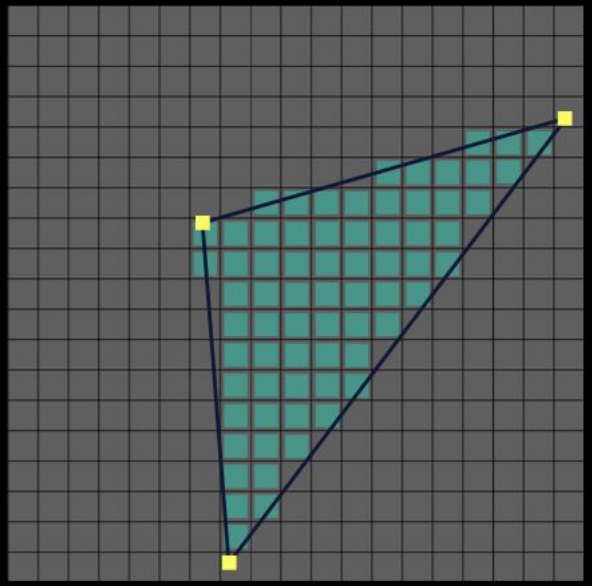
ステップ4　表裏を調べ、裏なら描画しない

3点そろったら



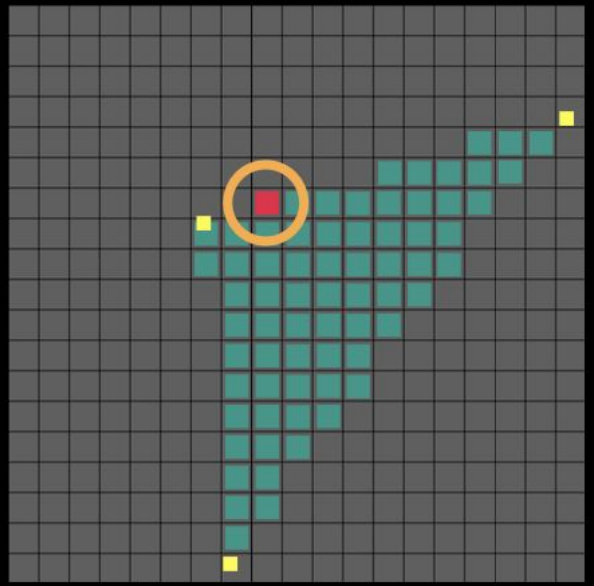


ステップ5　描画点を確定

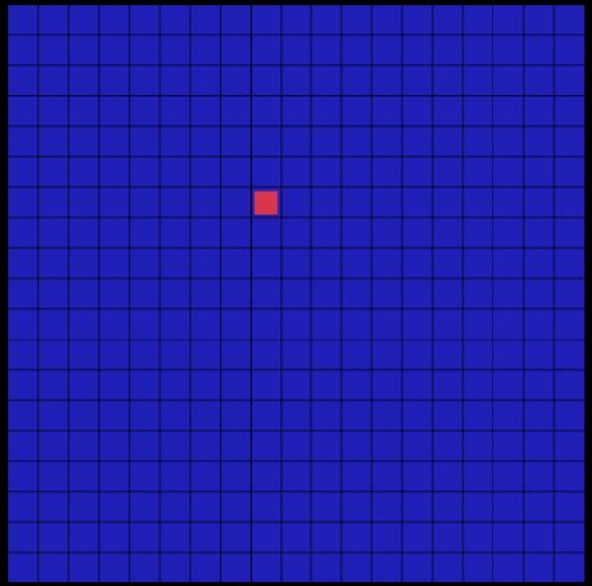


ピクセルの中心が三角形の内側にあるかどうか

ステップ6　描画点を深度バッファと比較
描画済みの点が手前にあるなら描画しない

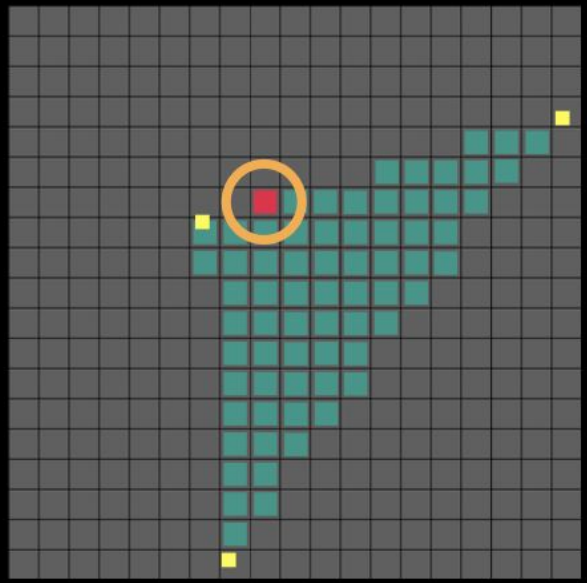


深度バッファ



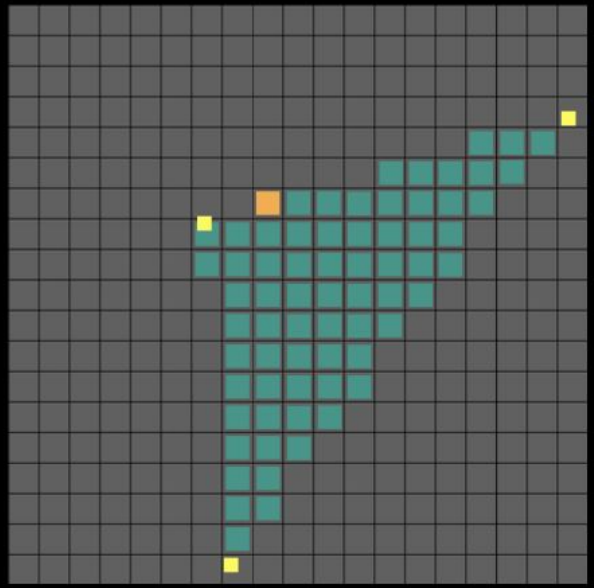
※GPUごとに実装が異なります

ステップ7　描画点に打つべき色を確定



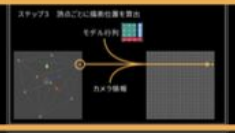
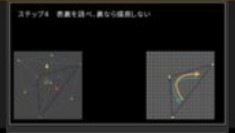
テクスチャ、ライティング、シャドウ、フォグなどなど考慮

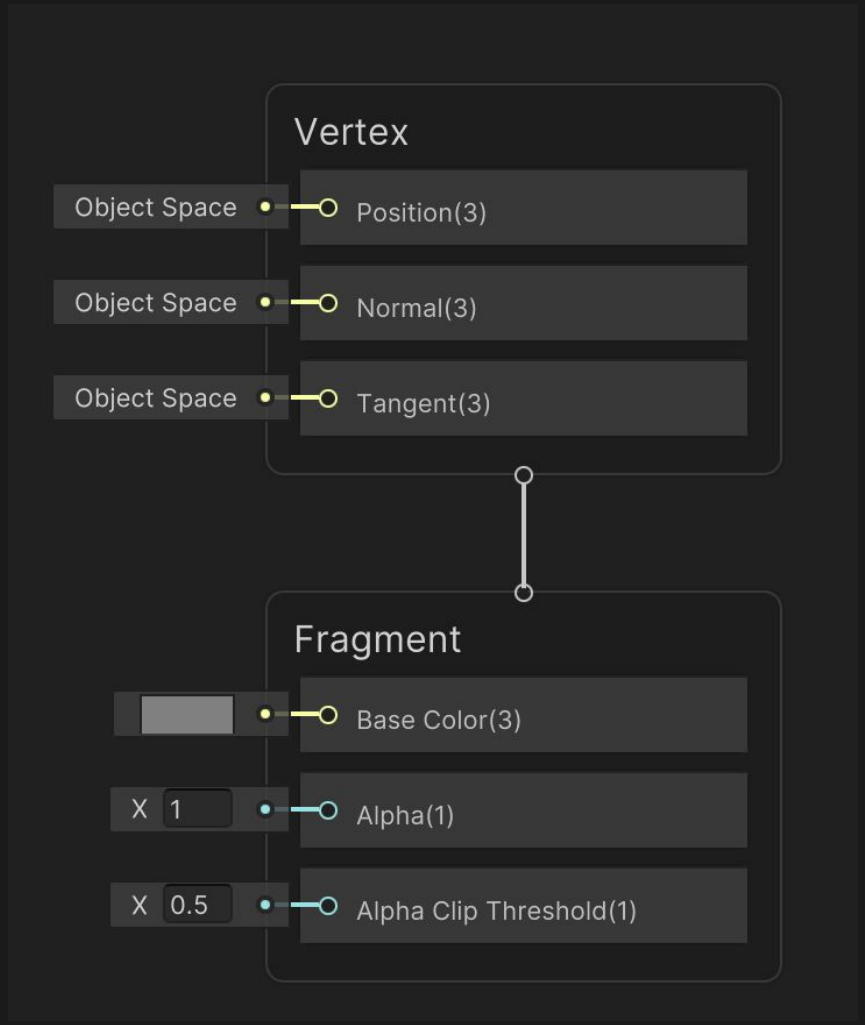
ステップ8　点を打つ



ブレンド関数を指定可

深度バッファも更新

CPU	ステップ1		3Dモデルを準備
CPU	ステップ2		Transformの値を4x4行列に変換
GPU	ステップ3		頂点ごとに描画位置を算出 頂点シェーダ
GPU	ステップ4		表裏を調べ、裏なら描画しない
GPU	ステップ5		描画点を確定
GPU	ステップ6		描画点をデプスバッファと比較
GPU	ステップ7		描画点に打つべき色を確定 フラグメントシェーダ
GPU	ステップ8		点を打つ


Vertex
Object Space
Position(3)
Object Space
Normal(3)
Object Space
Tangent(3)
Fragment
Base Color(3)
X 1
Alpha(1)
X 0.5
Alpha Clip Threshold(1)