1. Czy program napisze linie Brasenhama

Tak lub Nie. Tu trzeba patrzeć w kodzie jaki znak stoi przy zmiennej y. Jeśli y+=1 to linia algorytm narysuje linię do góry. Jeśli y-=1 to w dół. Jeśli nie będzie tam dodawania ani odejmowania to linia będzie pozioma.

1. Rodzaje shadera w OpenGL:

- Vertex Shader

- Geometry shader

- Pixel Shader, Fragment Shader

1. Offset i stride we floatach Vertex (xyz,rgb, st) x3 na bajtach od 0 do 92.
2. Mnożenie macierzy, żeby powstało przekształcnie z 3d na ekran monitora (Mcam i jeszcze 2). M = M\_vp \* M\_per \* M\_cam dla rzutowania perspektywistycznego (powstaje przez przemnożenie macierzy P przez macierz rzutowania 3D)

M = M\_vp \* M\_ortho \* M\_cam dla rzutowania 3D

1. Pseudokod do czy punkt w trójkącie i drawtriangle (resteryzacja używając współrzędnych barycentrycznych).

Wektory bazowe wektor c-a, b-a, początek układu współrzędnych jest wierzchołek a. Mając tak zdefiniowany układ współrzędnych każdy punkt p można opisać w takim układzie jako przysunięcie od początku układu współrzędnych a o pewne złożenie wielokrotności wektorów bazowych, wektor (b-a)\*współczynnik beta, a wektor (c-a)\*dla potrzeb naszych rozważań przez współczynnik gamma. Współczynniki te stojące przy wektorach bazy to współrzędne, więc będziemy tutaj mówić o współrzędnych beta i gamma.

1. Czy światło podające na ścianę obiektu będzie miało te same parametry w cieniowaniu płaskim (Tak)
2. Uniform czy zmienna jest unikalna dla każdego shadera (Tak)
3. Jeśli punkt p ma jedną wspórzędną barycentryczną równą 0.1, a pozostałe mają wartość w przedziale (0, 1) to leży on wewnątrz trójkąta
4. Uzupłnij zawartość instrukcji warunkowej, tak aby sprawdzała ona czy współrzędne barycntryczne var są w przedziale (0, 1). Bar zawiera rzy pola: x, y, z

WZÓR: (bar.x>=0 && bar.x<=1) && (bar.y>=0 && bar.y<=1) && (bar.z>=0 && bar.z<=1)

1. Czy następujące zdanie jest prawdziwe?

W cieniowaniu płaskim każdy piksel danej ściany ma indywidualnie wyliczane parametry oświetlenia. (Fałszywe)

1. Uzupełnij:

Algorytm backface culling sprawdza, dla każdej ściany czy jest ona obrócona przodem do kamery. Jeśli nie jest to jej nie wyrysowuje na obrazie.

1. Zdefiniuj atrybuty danych znajdujących si w buforze wierzchołków (Vertex buffer Object), aby odzwierdiedlić ułożenie danych jak na poniższym diagramie:
2. Zdefiniuj atrybuty danych znajdujących się w buforze wierzchołków (Vertex Buffer Obejct), aby odzwierdcidlić ułożenie danych jak na poniższym diagramie:

Vertex 1 Vertex 2 Vertex 3 …

X Y Z S T R G B X Y Z S T R G B X Y Z S T R G B

Użyj funkcji glVertexAttribPointer. Jej deklaracja wygląda następująco:

Void glVertexAttribPointr(GLuint index, GLint size, GLenum type, GLboolean normalized, GLsize1 stride, const void \* pointere);

glVertexAttribPointer(0, 4, GL.flat, BL.false, 32, (void)\*0);

glVertexAttribPointer(1, 2, GL.flat, BL.false, 32, (void)\*12);

glVertexAttribPointer(2, 3, GL.flat, BL.false, 32, (void)\*20);

stride offset

1 32 0

2 32 12

3 32 20

1. Uzupełnij poniższy shader wierzchołków, tak aby korzystał z danych w buforze zdefiniowanym w zad. 1. Kolor i położenie powinny być przekazywane dalej do potoku graficznego (zmienne wyjściowe) natomiast położenie wierzchołka powinno być rozszerzone do wektora 4-elementowego (ostatni element ustaw na 1.0) i wykorzystane do ustawienia zmiennej gl\_position.
2. Napisz jak wyliczysz oświetlenie ściany w cieniowaniu płaskim:

- Definicja wektora światła

- Obliczenie wektora normalnego ściany.

- Obliczenie intensywnością oświetlenia na podstawie wektora normalego, wektora

Światła dla każdej ściany (św.x\*n.x+św.y\*n.y+św.z\*n.z)

- Wyrenderowanie ściany w jednolitym oświetleniu.

1. Zmodyfikuj poniższy algorytm rysowania linii tak, aby wykorzystał tylko zmienne typu int.

Public void drawLineBresenhan(int x0,int x1, int y0, int y1){

Int white = 255 | (255<<8)|(255<<16)|(255<<24);

Int err= 0;

Int d = math.abs(2\*(y1-y0));

Int y = y0;

For(int x=x0; x<x1; x++){

Render.setRGB(x,y,white);

Err+=d;

If(err>(x1-x0){

Y+=(y1>y0?1:-1);

Err-=2\*(x1-x0);

}

}

}

1. Jeśli punkt p ma jedną wspórzędną barycentryczną równą -0.05, a pozostałe mają wartość w przedziale (0, 1) to leży on po za trójkątem.
2. Opisz jak działa algorytm backface culling.
3. Opisz jak działa algorytm testu z-buffora.
4. Jeśli punkt p ma jedną współrzędną barycntryczną równą 0, a pozostałe mają wartość w przedziale (0, 1) to leży na przeciwnym boku trójkąta.
5. Opierając się na poniższym pseudokodzie:

// dia katdego sprawdzanego punktu obrazu render;

// oblica współrzędne bazyc.

// Jeśli punkt leży wewnątrz, zamaluj (patrz wykład)

i zakładając że mamy obraz z klasy BufferedImage (zmienna this.render) oraz dostarczoną metodę do wyznaczania współrzędnych barycentrycznych:

public Vec3f barycentric (Vec2f A, Vec2f B, Vec2f C, Vec2f P)

Napisz metodę:

void drawTriangle(Vec2f A, Vec2f B. Vec2f C)

która wypełnia trójkąty na obrazie this.render, na kolor biały (zakładamy że jest zdefiniowany pod zmienną white), ale używa tylko nieparzystych wierszy i kolumn obrazu. Piksele ustawiaj wywołując:

this.render.setRGB(x, y, white):

void drawTriangle(Vec2f A, Vec2f B, Vec2f C) {

for (int x=0,x<this.render width, x++) {

for(int y=0; y< this.render.height; y++) {

Point P = new Point (x, y)

Vec3f bar = barycentric(A, B, C, P);

If(x%2 == 1 && y%2 == 1) {

If((bar.x>=0 && bar.x <= 1) && (bar.y>=0 && bar.y <= 1) && (bar.z>=0 && (bar.z >= 0 && bar.z <= 1)) {

This.render.setRGB(x,y,white);

}

}

}

}

1. Uzupełnij poniższy shader wierzchołków, tak aby korzystał z danych w buforze zdefiniowanym w zad. 1. Kolor i położenie tekstury powinny być przekazywane dalej do potoku graficznego (zmienne wyjściowe) natomiast położenie wierzchołka powinno być przeliczone – wszystkie współrzędny podzielone przez ostatni komponent (współrzędna w) i wykorzystane do ustawienia zmiennej gl\_position.

#version 330 core

Layout(location=0) in ver4 aPos;

Layout(location=2) in ver3 aColor;

Layout(location=1) in ver2 aTexCoord;

In Vec4 Pos

Out vec2 gl\_position;

Void main() {

Gl\_position = vec4(oPos.x, oPos.y,oPos.w)

Nw = vec3(gl\_position.x/gl\_position.w, gl.position.y\y1.position.w, gl\_positiob.z/gl.position.w)

}

1. Jeżeli wartość w z-buforze o danych współrzędnych (x,y) jest ... niż przetwarzanego piksela, to framebuffer nie ejst aktualizowany.

MNIEJSZA

1. Napisz pseudokod do barycentric

public Vec3f barycentric(Vec3f A, Vec3f B, Vec3f C, Vec3f P) {

// wektor składający się ze współrzędnych x wektorów AB, AC, PA

Vec3f v1 = new Vec3f(B.x - A.x, C.x - A.x, A.x - P.x);

// wektor składający się ze współrzędnych y wektorów AB, AC, PA

Vec3f v2 = new Vec3f(B.y - A.y, C.y - A.y, A.y - P.y);

// iloczyn skalarny v1 i v2

Vec3f cross = this.scalarDot(v1, v2);

// wektor postaci: cross.x / cross.z, cross.y / cross.z

Vec2f uv = new Vec2f(cross.x / cross.z, cross.y / cross.z);

// współrzędne barycentryczne, uv.x, uv.y, 1- uv.x - uv.y

Vec3f barycentric = new Vec3f(uv.x, uv.y, 1 - uv.x - uv.y);

return barycentric;

}

1. Zmienna typu uniform jest unikalna dla każdego programu shadera i może być dostępna z dowolnego shadera na dowolny etapie potoku graficznego.

TAK

1. Napisz w pseudokodzie metodę dokonująca rasteryzacji trójkąta o współrzędnych ABC, zakładając, że mamy do dyspozycji metodę barycentric(Point A, Point B, Point C, Point P), wyliczającą współrzędne barycentryczne. Należy uwzględnić Z-Buffer.

public void drawTriangle(Vec3i A, Vec3i B, Vec3i C, Vec3i color) {

Integer[] xs = {A.x, B.x, C.x};

Integer[] ys = {A.y, B.y, C.y};

int minx = Collections.min(Arrays.asList(xs));

int maxx = Collections.max(Arrays.asList(xs));

int miny = Collections.min(Arrays.asList(ys));

int maxy = Collections.max(Arrays.asList(ys));

for (int i = minx; i <= maxx; i++) {

for (int j = miny; j <= maxy; j++) {

Vec2i P = new Vec2i(i,j);

Vec3f Bar = barycentric3i(A, B, C, P);

float z = A.z\*(1 - Bar.z) + B.z\*Bar.z ;

if ((Bar.x > 0 && Bar.x < 1 && Bar.y < 1 && Bar.y > 0 && Bar.z > 0 && Bar.z < 1)) {

if (ZBuffer[i][j] >= z){

drawPoint(i, j, color);

ZBuffer[i][j] = z;

}

}

}

}

}

1. Podaj sekwencję mnożenia macierzy, aby uzyskać macierz dokonująca transformacji współrzędnych wirtualnego świata 3D do współrzędnych ekranu:

M = ...............

Macierze oznaczaj dużą literą M, z odpowiednim indeksem dolnym

M = Mvp Mortho Mcam