**sin함수를 이용한 파티클**

float\* Points = new float[(pointCount + 1) \* 4];

for (int i = 0; i <= pointCount; i++)

{

Points[i \* 4 + 0] = (i / (float)pointCount) \* 2 - 1;

Points[i \* 4 + 1] = (float)rand()/(float)RAND\_MAX;

Points[i \* 4 + 2] = (float)rand() / (float)RAND\_MAX;

Points[i \* 4 + 3] = 1;

if ((float)rand() / (float)RAND\_MAX > 0.5f)

{

Points[i \* 4 + 0] \*= -1;

}

}

z값은 랜덤 값을 저장하기 위한 용도.

x값은 -1~1사이 범위로 나오게 함.

y값도 랜덤 값을 저장. 0~1사이

float xValue = a\_position.x;

float yValue = sin(u\_time\*3.1415)\*0.3;

사인 곡선은 진폭이 0.3만큼 줄어들고 시간에 따라 위 아래로 이동

모든 y값이 동일

u\_time은 0~2사이를 왔다갔다하는 값

float yValue = sin(u\_time\*3.1415\*a\_position.y)\*0.3;

y값이 랜덤임으로 점들이 사인곡선상에서 랜덤 한 위치에 존재

하지만 아직 일직선상에 똑 같은 점들이 너무 많다.

전체적인 모양이 사인곡선이 아니다.

float yValue = sin(u\_time\*3.1415\*a\_position.y)\*a\_position.z\*0.3;

진폭에도 랜덤값을 곱해서 좀더 일직선 상에 점이 적게 한다.

float newTime = u\_time +1 - a\_position.x;

float xValue = a\_position.x;

float yValue = sin(newTime\*3.1415\*a\_position.y)\*a\_position.z\*0.3;

곱해지는 시간 값을 x좌표에 따라 차등을 둔다.

**픽셀셰이더 sin곡선**

` float points[] =

{

-1,1,0,

-1,-1,0,

1,-1,0,

1,-1,0,

1,1,0,

-1,1,0

};

화면을 덮는 좌표를 설정합니다.

in vec3 a\_position;

out vec2 p\_position;

void main()

{

p\_position = a\_position.xy;

gl\_Position = vec4(a\_position,1);

}

정점쉐이더에선 포지션을 현재 포지션의 x,y좌표를 넘겨줍니다.

그럼 보간되서 넘어가는데

uniform float u\_time;

void main()

{

float y = sin(p\_position.x\*2+u\_time);

if(p\_position.y>y)

{

FragColor = vec4(1);

}

else

{

FragColor = vec4(0);

}

}

이 값을 받아 현재 x좌표에 따른 sin값보다 클 때만 흰색으로 표시합니다. 동일하게 u\_time을 더해주면 이동하게 됩니다.

void main()

{

float xValue = a\_position.x;

float yValue = a\_position.y + sin(a\_position.x\*3.141592+u\_time)\*0.3;

gl\_Position = vec4(xValue,yValue,0,1);

}

사인 곡선으로 진동