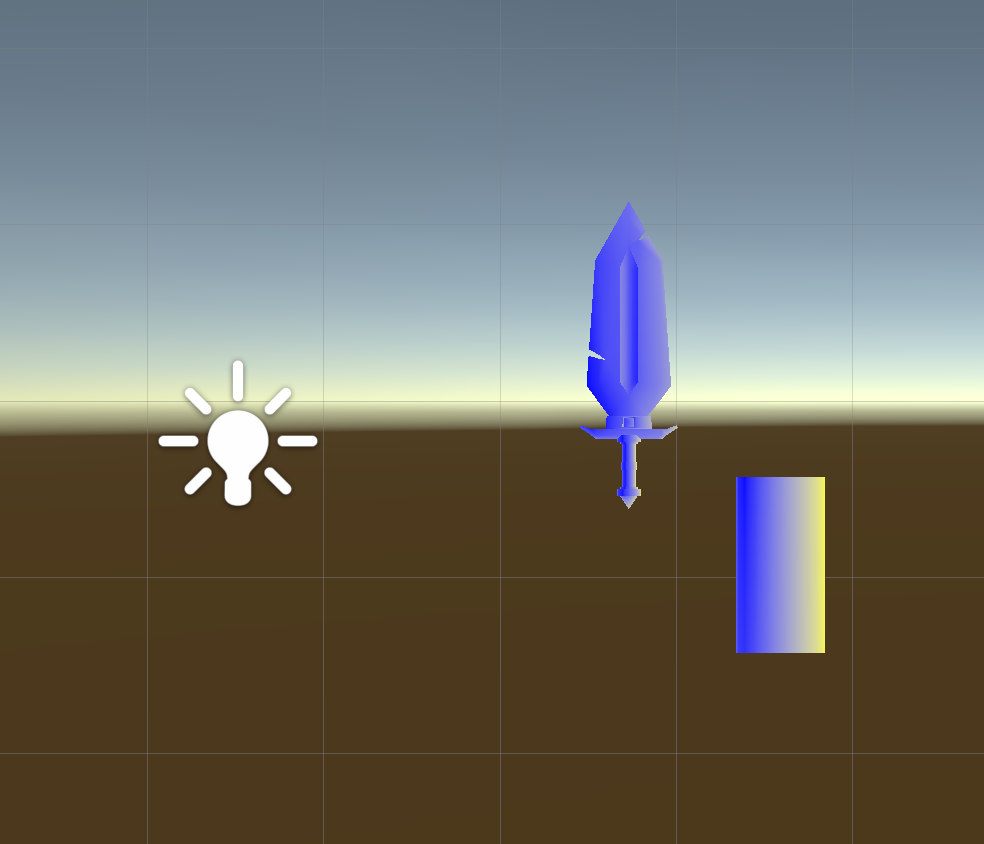
# HW2\_problem1

* Vertex sader
* output.diffAlmost = \_LightColor0.rgb \*max(0, dot(normalWS, lightDir)) \* atten;
* output.scale = ((1.0 + dot(normalWS, lightDir)) / 2.0);
* output.uv = TRANSFORM\_TEX(input.uv, \_BaseTex);
* return output;
* }
* float4 FragVertexLit(v2f input) : COLOR {
* //float3 blue = (0.0, 0.0, 1.0);
* //float3 yellow = (1.0, 1.0, 0.0);
* //float3 base =
* //float4 base = tex2D(\_BaseTex, input.uv);
* float3 output = lerp(float3(1.0 , 1.0 ,0.0), float3(0.0 , 0.0 ,1.0),input.scale );
* return(float4(output,1));
* }

Output:



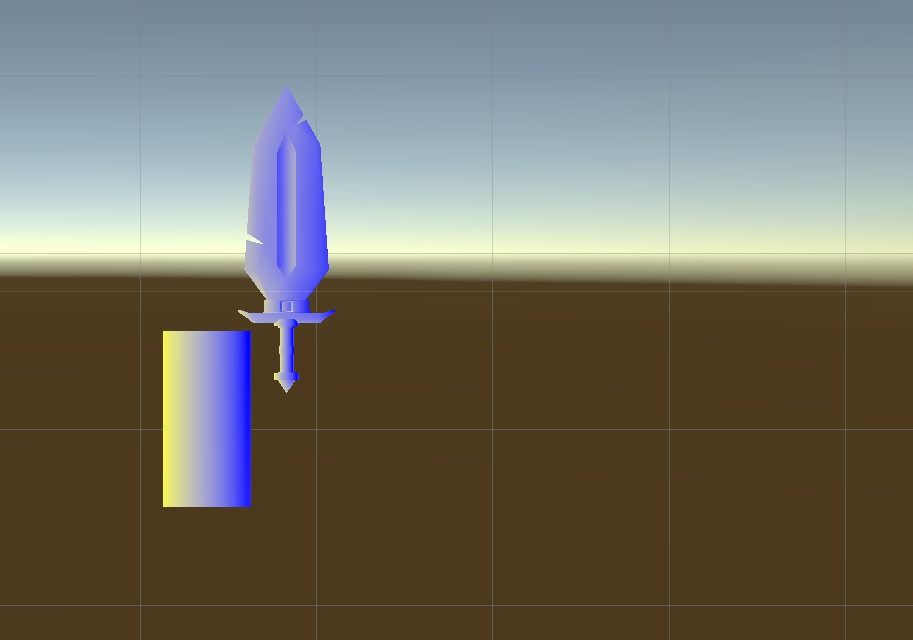
* Pixel shader

float3 diffAlmost = \_LightColor0.rgb \* max(0, dot(n, lightDir)) \* atten;

float3 output = lerp(float3(1.0, 1.0, 0.0), float3(0.0, 0.0, 1.0), ((1.0 + dot(n, lightDir)) / 2.0));

return(float4(output,1));

Output :



# HW2\_Problem2

* I used the the pixelit shader and modified the code for it to look like wavy .

Code:

float3 diffAlmost = \_LightColor0.rgb \* max(0, dot(n, lightDir)) \* atten;

float A = 6;

float B = 5;

float C = 16;

float D = 10;

float E = 4;

float F = 17;

input.uv.x += A \* sin(B\*input.uv.y) \* sin(C \* \_Time.x);

input.uv.y += D \* sin(E \* input.uv.x) \* sin(F \* \_Time.x);

float4 base = tex2D(\_BaseTex, input.uv);

float3 output = diffAlmost \* base.rgb;

return(float4(output,1));

}

Output:



# HW2\_Problem3

Code:

Properties{

\_BaseTex("Base (RGB) Gloss (A)", 2D) = "white" {}

\_NormalMap("Normalmap", 2D) = "bump" {}

\_AttenFactor("Atten Factor", float) = 1

\_Amplitude("Amplitude", float) = 1.0

\_Rate("Rate", float) = 1.0

}

Then

float \_AttenFactor;

float \_Amplitude;

float \_Rate;

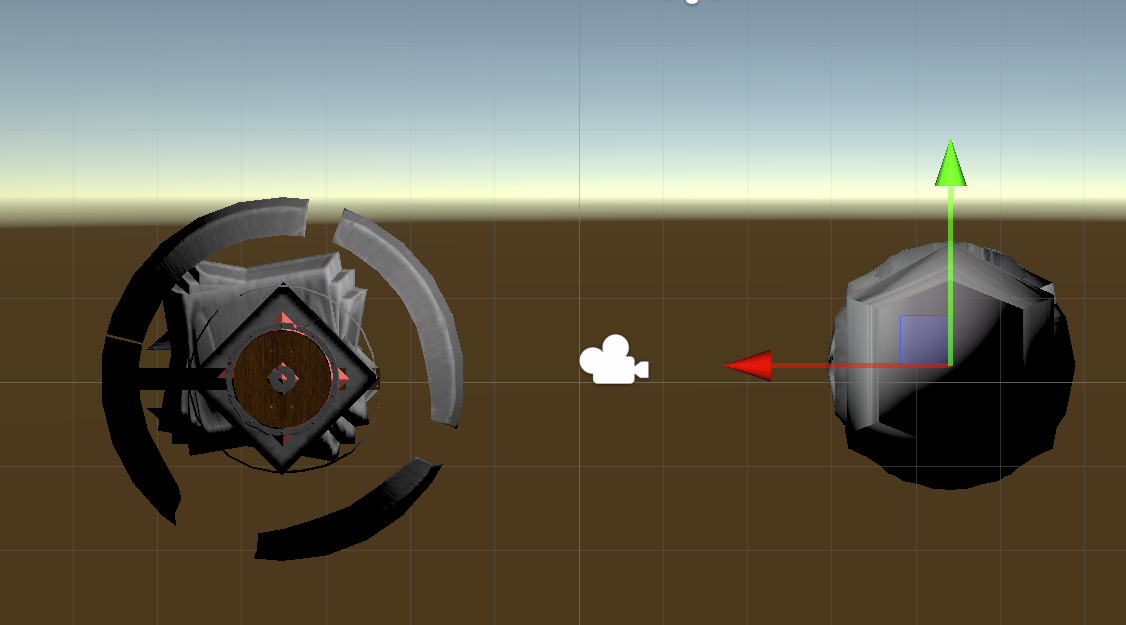
then

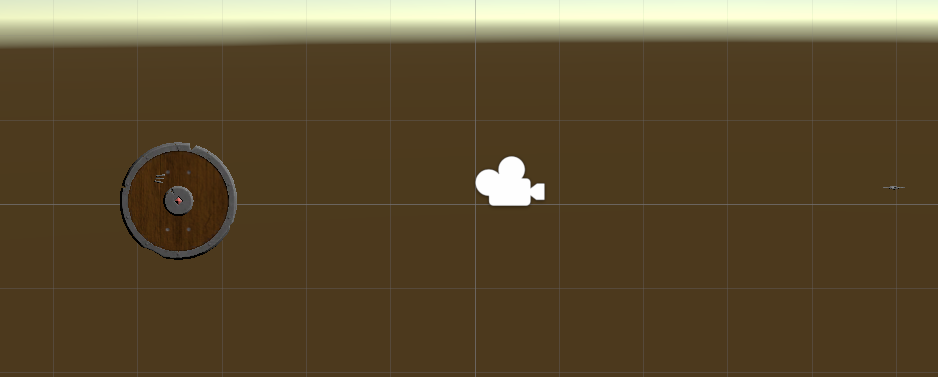
v2f output;

input.positionOS.xyz += \_Amplitude \* input.normalOS \* (1 + sin(\_Rate\*\_Time.x));

float4 positionWS = mul(unity\_ObjectToWorld, float4(input.positionOS.xyz, 1.0));

Output:





# HW2\_problem4

Code:

Properties {

\_BaseTex ("Base (RGB) Gloss (A)", 2D) = "white" {}

\_NormalMap ("Normalmap", 2D) = "bump" {}

\_NormalMap2 ("Normalmap2", 2D) = "bump" {}

\_AttenFactor ("Atten Factor", float) = 1

\_Rate("Rate", float) = 1.0

}

Then

sampler2D \_BaseTex;

float4 \_BaseTex\_ST;

sampler2D \_NormalMap;

float4 \_NormalMap\_ST;

sampler2D \_NormalMap2;

float4 \_NormalMap2\_ST;

float \_AttenFactor;

float \_Rate;

then

float normX;

float normY;

float normZ;

float2 nMapXY = 2 \* tex2D(\_NormalMap, input.nmap\_uv).ag - 1;

float2 nMap2XY = 2 \* tex2D(\_NormalMap2, input.nmap\_uv).ag - 1;

float nMapRecreatedZ = sqrt(1 - saturate(dot(nMapXY,nMapXY)));

float nMap2RecreatedZ = sqrt(1 - saturate(dot(nMap2XY, nMap2XY)));

// we are renormalizing because the GPU's interpolator doesn't know these are unit vectors

float3 nWS = normalize(input.normalWS);

float3 tWS = normalize(input.tangentWS);

float3 btWS = normalize(input.bitangentWS);

float3 newNormal = tWS \* nMapXY.x + btWS \* nMapXY.y + nWS \* nMapRecreatedZ;

newNormal = normalize(newNormal);

float3 newNormal2 = tWS \* nMap2XY.x + btWS \* nMap2XY.y + nWS \* nMap2RecreatedZ;

newNormal2 = normalize(newNormal2);

normX = lerp(newNormal.x, newNormal2.x, 0.5\*(1.0 + sin(\_Rate\*\_Time.x)));

normY = lerp(newNormal.y, newNormal2.y, 0.5\*(1.0 + sin(\_Rate\*\_Time.x)));

normZ = lerp(newNormal.z, newNormal2.z, 0.5\*(1.0 + sin(\_Rate\*\_Time.x)));

float3 norm = (normX,normY,normZ);

Output:

