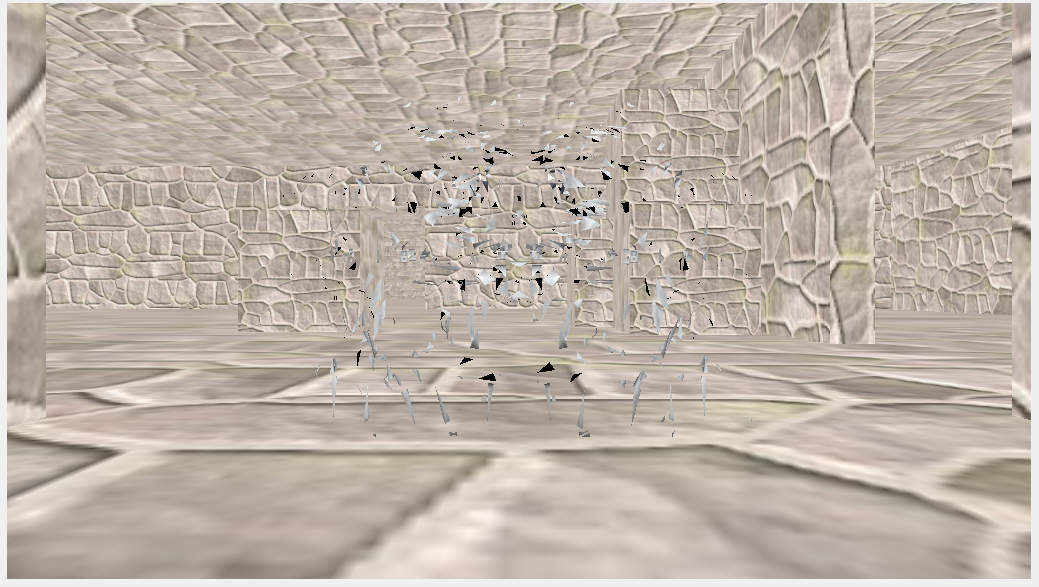
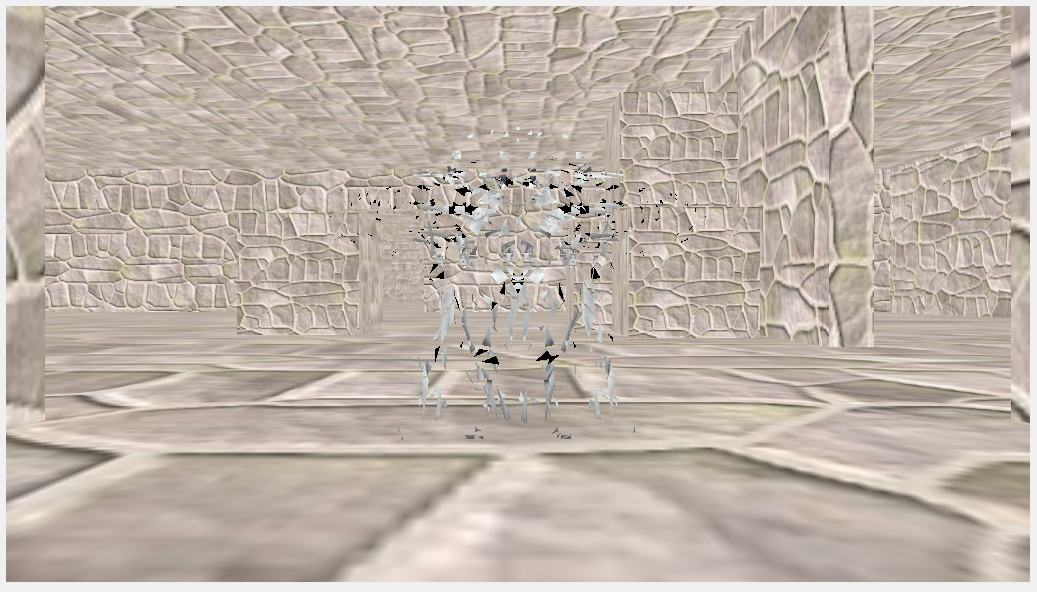
I made a shader takes a model and will move its triangles along its normal vector. This will work on all models but can look very strange sometimes. We don’t even need to take in the normals, we calculate them in the geometry shader. After calculating the normal we translate each vertex along its normal. The result is a 3D model that seems to continually explode its vertices over time after which it returns to normal again.







Vertex Shader

#version 430 core

layout (location = 0) in vec4 position;

layout (location = 1) in vec2 texCoords;

out VS\_OUT {

vec2 texCoords;

} vs\_out;

uniform mat4 projection;

uniform mat4 view;

uniform mat4 model;

void main()

{

gl\_Position = projection \* view \* model \* position;

vs\_out.texCoords = texCoords;

}

Geometry Shader

#version 430 core

layout (triangles) in;

layout (triangle\_strip, max\_vertices = 3) out;

in VS\_OUT {

vec2 texCoords;

} gs\_in[];

out vec2 TexCoords;

uniform float time;

vec4 explode(vec4 position, vec3 normal)

{

float magnitude = 2.0f;

vec3 direction = normal \* ((sin(time) + 1.0f) / 2.0f) \* magnitude;

return position + vec4(direction, 0.0f);

}

vec3 GetNormal()

{

vec3 a = vec3(gl\_in[0].gl\_Position) - vec3(gl\_in[1].gl\_Position);

vec3 b = vec3(gl\_in[2].gl\_Position) - vec3(gl\_in[1].gl\_Position);

return normalize(cross(a, b));

}

void main() {

vec3 normal = GetNormal();

gl\_Position = explode(gl\_in[0].gl\_Position, normal);

TexCoords = gs\_in[0].texCoords;

EmitVertex();

gl\_Position = explode(gl\_in[1].gl\_Position, normal);

TexCoords = gs\_in[1].texCoords;

EmitVertex();

gl\_Position = explode(gl\_in[2].gl\_Position, normal);

TexCoords = gs\_in[2].texCoords;

EmitVertex();

EndPrimitive();

}

Fragment Shader

#version 430 core

in vec2 TexCoords;

out vec4 color;

uniform sampler2D texture\_diffuse1;

void main()

{

color = texture(texture\_diffuse1, TexCoords);

}