Project Description

Video link: https://youtu.be/5OAvQKFTV8w

Github: https://github.com/Jinaisrz/Coding-one-Jin-Yu.git

I want to use gradient graphics as creative elements, and try to use shader and sounds to create excellent visual art works.

First, I extended OES_standard_derivatives to add GLSL derivatives.

Set uniform, then set the precision at the beginning of the shader, then set the canvas size in resolution, mouse and time, the mouse pixel position setting and the number of seconds after loading.

Set the rotation point and Angle, then do floating-point arithmetic a by sin and cos, and eventually return control to "vec2(p.x*c+p.y*s,p.x*s-p.y*c)".

In Void main, coordinate transformation is set by "vec2 position = (gl_FragCoord.xy / resolution.xy) + mouse". Then set the position of the mouse, use frp (Functional Reactive Programming) to set the fractal position, and then use the flp (floating Point) to take down the integer. Next, use double-precision floating-point values, equal to "(length(frp-0.5))". Finally, add rot to it by adjusting flp or frp, so as to set whether to rotate and divide the graphics.

In the float color, cos, sin and tan are used to formulate the graphic color and its change. I prefer to use the sine function, which can make the graphic visual look more transitional. Adding tan function makes the graphic have a more obvious line segment rotation . Finally, use gl_FragColor to set the fragment color value of vec4, and then use the tan and sin functions to make the graphics alternate between gradient and vector.

Then I made the BGM sound "magic" by setting up the oscillators, and importing 4 audios, setting both tempo and ticks to 10, and then exporting the sounds.

I set the pixel to 1 to make the graphics clearer. Set the camera position, create the scene, and the y and z positions of the PlaneBufferGeometry vertex (because x has been set by uniforms, no need to set it again). Then use GLSL shaders to do the actual rendering in Javascript and Three.js. Secondly, create a mesh, and add true judgments to the window size and mouse movement, then form an animation, set the rotation speed and render the final scene and camera, and finally form an automatic rotation animation that can control the visual effect with the mouse.