1. 利用scolar在线编辑工具实现FBM（分形布朗运动模型（“也叫做分形噪声”）基于各种角度参数下的图片实现）

shader在线实现代码的网站：<https://thebookofshaders.com/edit.php>

1.fbm的二维实现：不断叠加循环所展现出的分形图案

// Author:

// Title:

#ifdef GL\_ES

precision mediump float;

#endif

uniform vec2 u\_resolution;

uniform vec2 u\_mouse;

uniform float u\_time;

float random (in vec2 st) {

return fract(sin(dot(st.xy,

vec2(12.9898,78.233)))\*

43758.5453123);

}

// Based on Morgan McGuire @morgan3d

// https://www.shadertoy.com/view/4dS3Wd

float noise (in vec2 st) {

vec2 i = floor(st);

vec2 f = fract(st);

// Four corners in 2D of a tile

float a = random(i);

float b = random(i + vec2(1.0, 0.0));

float c = random(i + vec2(0.0, 1.0));

float d = random(i + vec2(1.0, 1.0));

vec2 u = f \* f \* (3.0 - 2.0 \* f);

return mix(a, b, u.x) +

(c - a)\* u.y \* (1.0 - u.x) +

(d - b) \* u.x \* u.y;

}

#define OCTAVES 6

float fbm (in vec2 st) {

// Initial values

float value = 0.0;

float amplitude = .5;

float frequency = 0.;

//

// Loop of octaves

for (int i = 0; i < OCTAVES; i++) {

value += amplitude \* noise(st);

st \*= 2.;

amplitude \*= .5;

}

return value;

}

void main() {

vec2 st = gl\_FragCoord.xy/u\_resolution.xy;

st.x \*= u\_resolution.x/u\_resolution.y;

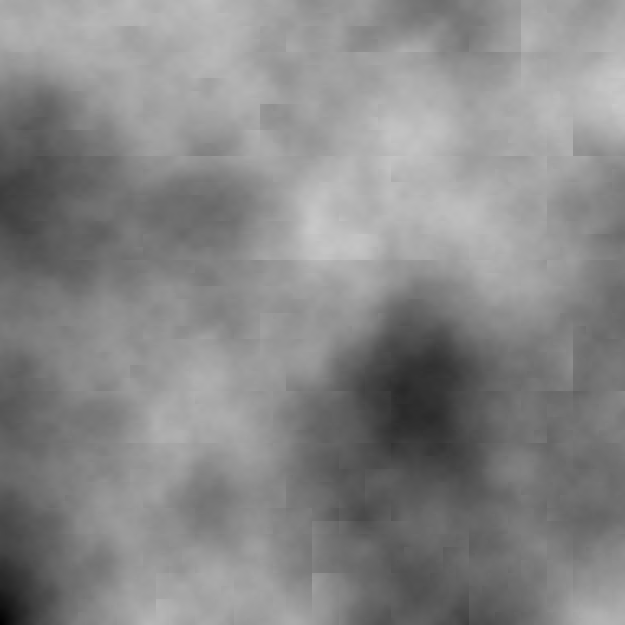
vec3 color = vec3(0.0);

color += fbm(st\*3.0);

gl\_FragColor = vec4(color,1.0);

}

效果图案：



1. 基于域翘曲（Domain Warping）的方法利用分形布朗运动模型生成云一般的纹理效果：

#ifdef GL\_ES

precision mediump float;

#endif

uniform vec2 u\_resolution;

uniform vec2 u\_mouse;

uniform float u\_time;

float random (in vec2 \_st) {

return fract(sin(dot(\_st.xy,

vec2(12.9898,78.233)))\*

43758.5453123);

}

// Based on Morgan McGuire @morgan3d

// https://www.shadertoy.com/view/4dS3Wd

float noise (in vec2 \_st) {

vec2 i = floor(\_st);

vec2 f = fract(\_st);

// Four corners in 2D of a tile

float a = random(i);

float b = random(i + vec2(1.0, 0.0));

float c = random(i + vec2(0.0, 1.0));

float d = random(i + vec2(1.0, 1.0));

vec2 u = f \* f \* (3.0 - 2.0 \* f);

return mix(a, b, u.x) +

(c - a)\* u.y \* (1.0 - u.x) +

(d - b) \* u.x \* u.y;

}

#define NUM\_OCTAVES 5

float fbm ( in vec2 \_st) {

float v = 0.0;

float a = 0.5;

vec2 shift = vec2(100.0);

// Rotate to reduce axial bias

mat2 rot = mat2(cos(0.5), sin(0.5),

-sin(0.5), cos(0.50));

for (int i = 0; i < NUM\_OCTAVES; ++i) {

v += a \* noise(\_st);

\_st = rot \* \_st \* 2.0 + shift;

a \*= 0.5;

}

return v;

}

void main() {

vec2 st = gl\_FragCoord.xy/u\_resolution.xy\*3.;

// st += st \* abs(sin(u\_time\*0.1)\*3.0);

vec3 color = vec3(0.0);

vec2 q = vec2(0.);

q.x = fbm( st + 0.00\*u\_time);

q.y = fbm( st + vec2(1.0));

vec2 r = vec2(0.);

r.x = fbm( st + 1.0\*q + vec2(1.7,9.2)+ 0.15\*u\_time );

r.y = fbm( st + 1.0\*q + vec2(8.3,2.8)+ 0.126\*u\_time);

float f = fbm(st+r);

color = mix(vec3(0.101961,0.619608,0.666667),

vec3(0.666667,0.666667,0.498039),

clamp((f\*f)\*4.0,0.0,1.0));

color = mix(color,

vec3(0,0,0.164706),

clamp(length(q),0.0,1.0));

color = mix(color,

vec3(0.666667,1,1),

clamp(length(r.x),0.0,1.0));

gl\_FragColor = vec4((f\*f\*f+.6\*f\*f+.5\*f)\*color,1.);

}

效果图案（动态）：

