



1

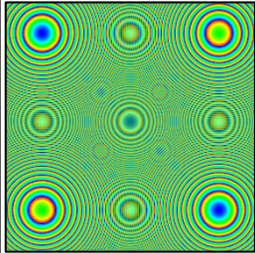
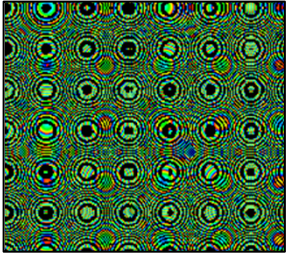
## Algorithmic Art



This work is licensed under a [Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License](https://creativecommons.org/licenses/by-nc-nd/4.0/)



**Oregon State  
University**  
Mike Bailey  
mjb@cs.oregonstate.edu

algart.pptx mjb - December 17, 2020

1

2

```

algart.glib
##OpenGL GLIB
Ortho -1. 1. -1. 1.
LookAt 0 0 1 0 0 0 0 1 0

Vertex algart.vert
Fragment algart.frag
Program AlgArt
    uScreen <true>
    uColor {0. 0. 0. 1.}
    uMod <1 2 8>
    uSide <.1 1. 3.>

QuadXY .2 2.
  
```

```

algart.vert
#version 330 compatibility

out float vX, vY;
out vec2 vST;

void
main( )
{
    vST = gl_MultiTexCoord0.st;
    vX = gl_Vertex.x;
    vY = gl_Vertex.y;
    gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;
}
  
```

mjb - December 17, 2020

2

3

```

algar.frag
#version 330 compatibility
uniform bool    uScreen;
uniform vec4    uColor;
uniform int     uMod;
uniform float   uSide;
in vec2         vX, vY;
in vec2         vST;

vec3
Rainbow( float t )
{
    ...
}

void
main( )
{
    vec2 xy;
    if( uScreen )
        xy = uSide * gl_FragCoord.xy;
    else
        xy = 200. * uSide * vST;

    float z = dot( xy, xy );    // z = x^2 + y^2
    int c = int( z );
    if( ( c % uMod ) != 0 )
    {
        //discard;
        gl_FragColor = vec4( uColor.rgb, 1. );
    }
    else
    {
        float t = float( c % 360 ) / 359.;
        vec3 rgb = Rainbow( t );
        gl_FragColor = vec4( rgb, 1. );
    }
}

```

This method is known as **Connett Circles**

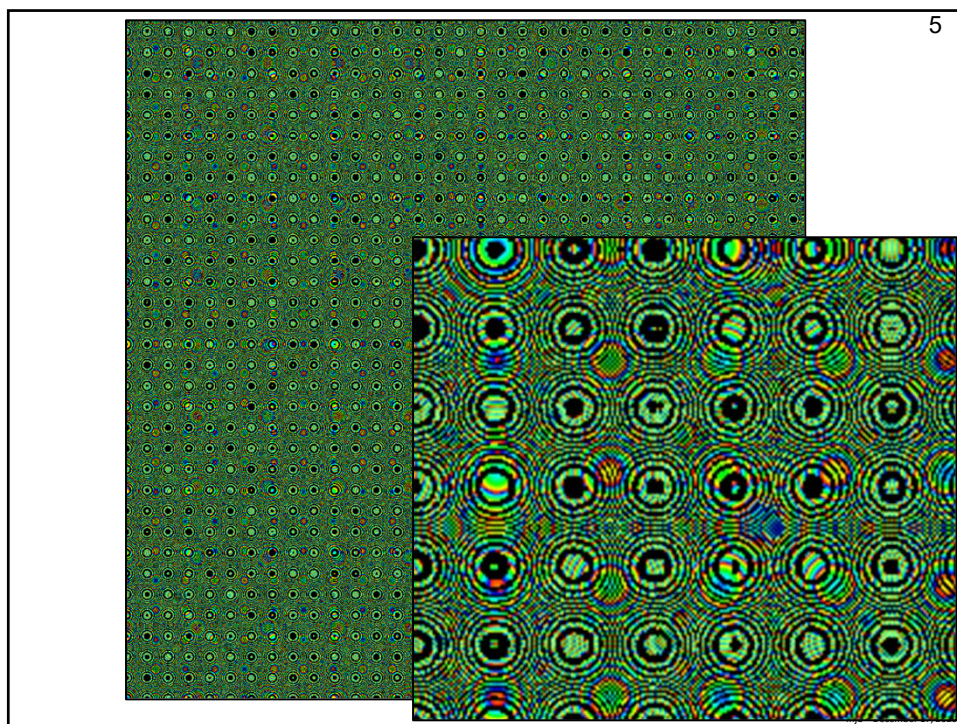
mjb - December 17, 2020

3

4

mjb - December 17, 2020

4



5