Graphics Programming Final
Ben Hubner and Stephen Hill
Repository: https://github.com/Ryse245/GraphicsCoursework.git
Branch: final

We decided to iterate on our multiple fractal patterns in glsl, and texture them to the given objects. We have added another fractal pattern (Koch) to our existing patterns (Mandelbrot and Julia), and have added the ability to increase the number of iterations and location of the patterns in the scene. For Mandelbrot, we were able to include controls to zoom in/out and move the center to allow for detailed viewing of the fractal. We also tried to add a gradient color for the fractal patterns, but unfortunately it did not end up working in time, we were able to apply a gradient to objects by changing their textures directly to one, which unfortunately means that those objects do not have their original textures outside of the fractal demonstration. Ben worked on the fractal patterns and iteration changes, while Stephen worked on the real-time change implementations(zoom, changing iteration number, and moving the center of the mandelbrot fractal) and demo curves implementation. Unfortunately, we still were not able to get our glsl files working with our own render passes, but we are able to display our fractal work within the given drawPhong_multi_forward_mrt_fs4x file. This project is a real-time effect that is applied to the objects in the scene, and we consider this to be an example of an intermediate real-time effect, such as was described in the project requirements. As for the other requirements, we did not complete this fractal effect in an earlier class assignment, and we would use both vertex (fractalPattern_vs) and fragment (drawFractalPattern_fs) shaders if our own render passes had worked. Our project still uses vertex and fragment shaders (drawPhong_multi_forward_mrt_fs4x and passTangentBasis_transform_instanced_vs4x), but we were forced to add to existing shaders instead of using our own. Our UML diagram is at the bottom of this document, and the video recording for our code, as well as the code itself, is in our repository.

(**Bold** will indicate new additions from midterm to final
GLSL:
Mandelbrot fractal: drawPhong_multi_forward_mrt_fs4x, lines 275-305 **(zoomNum, center, scale, zoomFactor and anything relating to it were added from last time, as well as iterations being a uniform now. Mandelbrot was changed the most)**

Mandelbrot fractal projection: drawPhong_multi_forward_mrt_fs4x, lines 311-334

Julia fractal: drawPhong_multi_forward_mrt_fs4x, lines 337-354

Fractal noise: drawPhong_multi_forward_mrt_fs4x, lines 166-240

Koch fractal: drawPhong_multi_forward_mrt_fs4x, lines 357-380

These are also in our drawFractalPattern_fs shader, but we were unable to generate our own render passes

CSS:
Demo_Curves.h: added target names and pass name (lines 114, 146-150)

DemoState.h: added fractal demo state shader program (lines 361-363), added framebuffer for fractal(line 403), **Added variables to be used for keeping track of data relating to the real-time editing** (**lines 121,123,219-222)**

DemoState_initialize.c: Initialize added variables for the real-time editing(lines 281-285)

Demo_Pipelines.h: added additional render passes and targets for fractals (lines 116-119, 158-160)

Demo_Pipelines_idle_render.c: added render pass in pipeline section (lines 774-799, 839-841)

Demo_Curves_initialize.c: initialized target index and count for fractals (lines 61, 76)

Demo_Curves_idle_render.c: added pass and target names for fractal mode(lines 93, 119-122, 138), added text for targets displayed (lines 119-122), added fbo for fractals (lines 293, 311), added render pass in curves section (lines 648-658, 702), added gradient texture(lines 246, 434). **Send uniforms for use in the fractal shader(lines 430-432).**

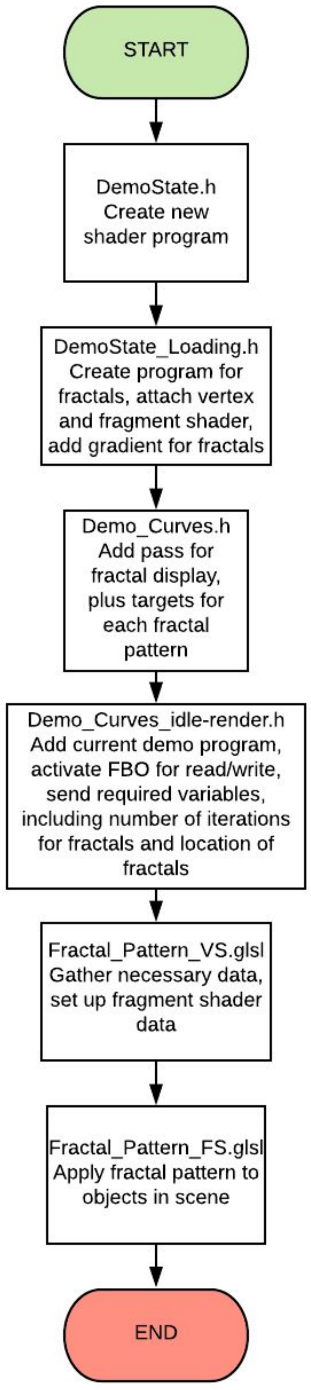
DemoState_loading.c: added fractal shaders to shader list(lines 436,483, 511, 550), made program and attached shaders to it(lines 733-736), added fbo for fractal(lines 1030-1033), added gradient file location (lines 805, 903). **Prepare the uniforms for the real-time editing to be sent to the shader(lines 834-837)**

DemoShaderProgram.h: Added uniform handles for the uniforms that are used for the real-time editing

DemoState_idle-input.c: Added inputs for changing the uniforms that are used in the shader(lines 159-168,229-233)

DemoState_idle-render: Added text that shows the new variables as they are being edited (lines 176-182)

UML:


START
DemoState.h
Create new
shader program
DemoState_Loading.h
Create program for
fractals, attach vertex
and fragment shader,
add gradient for fractals
Demo_Curves.h
Add pass for
fractal display,
plus targets for
each fractal
pattern
Demo_Curves_idle-render.h
Add current demo program,
activate FBO for read/write,
send required variables,
including number of iterations
for fractals and location of
fractals
Fractal_Pattern_VS.glsl
Gather necessary data,
set up fragment shader
data
Fractal_Pattern_FS.glsl
Apply fractal pattern to
objects in scene
END