Projeto 1: Visualização de funções complexas

Funções Complexas de Variável Complexa

Neste projeto vamos programar uma aplicação que nos permitirá visualizar uma função complexa f() de variável complexa **z**. Uma função complexa tem a característica do seu resultado, **w**, ser um número complexo:

w = f(z)

Sendo os números complexos representados normalmente num plano cartesiano, a duas dimensões, para visualizarmos as nossas funções complexas de variável complexa necessitaríamos dum espaço a 4 dimensões (2 para o argumento e outras 2 para o resultado). Felizmente existe uma alternativa a essa forma de visualização, denominada por "Coloração do Domínio" .

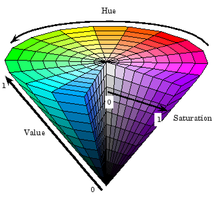
Coloração do Domínio

Como referimos anteriormente, um número complexo poderá corresponder a uma posição no plano cartesiano 2D. Vamos usar essa correspondência para o argumento **z** das nossas funções. Já para o resultado, **w**, vamos utilizar uma cor. Assim, ao aplicar a função a uma determinada região do plano obtemos um padrão colorido que representa o resultado da função numa parte do seu domínio.

Modelo de cor HSV

A especificação duma cor à custa das suas componentes primárias R, G e B é bastante útil para utilização em dispositivos que reproduzem a cor por adição de cores primárias (CRTs e LCDs). Contudo, existem outras formas de especificar cores, usando outros modelos de cor. Um modelo bastante utilizado, pois captura os atributos psico-fisiológicos da cor, é o modelo HSV (Hue, Saturation, Value). A cor é igualmente especificada usando 3 componentes independentes, mas agora com um significado mais perto da forma como nós a percepcionamos:

- Hue (Tom) - a tonalidade da cor (vermelho, laranja, amarelo, verde, ...)
- Saturation (Saturação) - quanto maior for a saturação, mais afastada a cor estará dos cinzentos. O valor da saturação varia entre 0 (para os cinzentos) e 1.
- Value (Valor) - está relacionado com o brilho que a cor aparenta. O preto é a cor com menor valor (V=0). Por oposição, o branco (mas também as cores principais do arco-íris) têm V=1.



Mapeamento dum número complexo numa cor

Para mapearmos o resultado da nossa função complexa numa cor vamos fazê-lo usando coordenadas polares. Qualquer número complexo pode ser representado por coordenadas polares, visto corresponder a uma posição no plano. Assim sendo, dado um complexo **w**, resultado da nossa função, transformamos em primeiro lugar para um par (*r*, *theta*), onde r representa a distância do ponto à origem (o raio) e *theta* o ângulo formado com a horizontal quando ligamos o ponto **w** à origem.

Finalmente o mapeamento para uma cor em HSV far-se-á usando os seguintes mapeamentos das componentes individuais da cor:

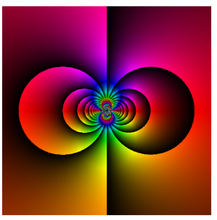
- H = *theta*
- S = 1
- V = frac(log2(*r*))

Exemplos de funções complexas

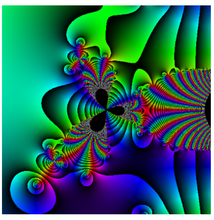
Seguem-se alguns exemplos de funções. Em alguns casos as imagens foram obtidas após várias operações de arrastamento e ampliação ou redução.

w = z

w = z^2



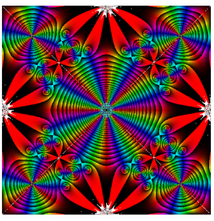
exp(100 / z)



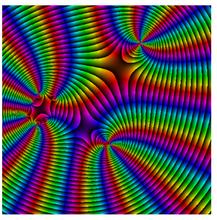
f3(z)=exp(z) + [(z - 2)^2 (z + 1 - 2i) (z + 2 + 2i)] / (z^3), para 2 iterações

f4: (z^2 + 1) / (z^2 - 1)

f4(z), para 3 iterações



f4(z), para 6 iterações



[(z+2)^2[(z-1-2i)(z+i), para 2 iterações

Descrição do trabalho

Pretende-se construir uma aplicação interativa que permita visualizar e explorar funções complexas de variável complexa. Cada pixel do canvas terá associada uma cor que resulta do mapeamento do valor da função complexa numa cor, conforme descrito anteriormente.

O programa deverá permitir ajustar os seguintes parâmetros:

- **centro** - permite definir o ponto do plano dos números complexos que será mostrado no centro do canvas. Inicialmente será 0+0i. A operação deverá ser por arrasto (drag) sobre o canvas. A imagem deverá deslocar-se solidária com o movimento do ponteiro do rato.
- **escala** - permite definir qual a dimensão do plano dos números complexos a visualizar no canvas.
- **func** - um identificador inteiro que servirá para selecionar a função a aplicar (de entre um conjunto de funções previamente programadas)
- **n** - permite ajustar o número de vezes que função será aplicada, como se duma potência se tratasse.

O trabalho deverá ser capaz de reproduzir qualquer das funções aqui referidas a título de ilustração, devendo cada uma delas estar disponível na interface de forma acessível. O utilizador poderá ainda escolher o número de vezes que a função será aplicada ao seu próprio resultado (parâmetro **n**).

Extras

O trabalho será avaliado para 18 valores, sendo os restantes 2 valores atribuídos à realização das seguintes componentes:

- Adição duma variável tempo que possa ser usada dentro das funções e assim criar animações
- Possibilidade de usar um canvas não quadrado que maximize a área disponível na janela do browser, mas sem deformar a imagem das funções.

Detalhes técnicos

Geometria a desenhar

A única primitiva a desenhar neste trabalho será um TRIANGLE_STRIP que cobrirá o espaço [-1,1]x[-1,1]. Todo o trabalho será executado ao nível do *fragment shader*. O papel do *vertex shader* nesta aplicação é mínimo.

Parâmetros/Variáveis obrigatórias do programa GLSL

Os parâmetros (variáveis uniform) necessários (e obrigatórios) neste programa são:

- um inteiro `func`
- um inteiro `n`
- um número complexo `center`
- a escala `scale` - permite converter de pixels para unidades no plano cartesiano dos complexos (ou no sentido inverso)

Regras e Informação Adicional

Composição dos grupos

Os trabalhos práticos deverão ser realizados por grupos de 2 alunos dum mesmo turno prático. A entrega do trabalho a título individual terá que ser devidamente justificada e autorizada pelo respectivo docente do turno prático.

Entrega

O trabalho será entregue via Google Classroom, usando a classe do respectivo turno prático.

Avaliação

Os trabalhos serão avaliados pelo respetivo docente das aulas práticas e discutidos com os respetivos alunos em data a definir oportunamente.