

Aula prática 3

Objetivos

- Construir um pequeno exemplo de programa interativo
- Explorar a utilização do *fragment shader* para criar efeitos 2D avaliando uma função no plano.

Exercício 3.1

Pretende-se construir uma aplicação interativa que use o canvas como uma tela interativa. O programa deverá desenhar um polígono (o modelo) cobrindo toda a superfície do canvas. O trabalho será todo realizado ao nível do *fragment shader*.

Passo 1

Para começar, adapte o exemplo disponibilizado no final por forma a desenhar um quadrilátero que cubra todo o canvas. Ainda se lembra dos limites do plano que são mapeados nas margens do canvas?

Passo 2

Adapte os *shaders* por forma a que no *fragment shader* se consiga saber qual a posição, no quadrilátero que foi desenhado (coordenadas do modelo), a que corresponde cada fragmento. *Hint*: *varying*...

Passo 3

Altere o *fragment shader* por forma a conseguir visualizar a posição 2D no polígono de origem. Não se esqueça que cada componente da cor varia no intervalo [0.0, 1.0].

Hint: a melhor forma de fazer debug no *fragment shader* é colocar na sua saída (`gl_FragColor`) os valores que pretendemos observar ou uma transformação simples desses valores para a gama de valores permitidos para a cor.

Passo 4

Acrescente no seu programa funções para tratar dos seguintes eventos, desde que ocorram no canvas:



- botão do rato para baixo
- botão do rato para cima
- Deslocamento do rato

Passo 5

Use as funções acima, de forma concertada, para calcular um deslocamento associado a uma ação de arrastamento (*drag*) efetuada com o rato. Este deslocamento deverá ser calculado em unidades do modelo (e não em pixels), valor este que deverá ser passado ao *fragment shader*. Torne esse valor visível para ter a certeza que o comportamento da aplicação está correto. Calcular um deslocamento em unidades do ecrã é trivial, mas para converter em unidades do modelo será necessário fazer alguns cálculos.

Passo 6

Modifique o seu *fragment shader* por forma a que o programa atribua uma cor a um pixel de acordo com a sua distância à origem. O valor do deslocamento passado ao fragment shader no passo anterior deverá fazer deslocar a imagem que se vê no canvas de acordo.

TITLE	LAST MODIFIED
 square.html	9/30/18 Fernando Birra
 square.js	9/30/18 Fernando Birra