	Disciplina	CGRAF
	Curso	SI
ALEXANDRE BELETTI FERREIRA	AULA - 14	

Nesta aula trabalharemos a parte de animação, para tanto, devemos, ver inicialmente alguns conceitos:

Background

Background é usado para especificar uma gradação de cores que simulam o chão e o céu no ambiente virtual, bem como um panorama que é colocado atrás de todos os objetos da cena e na frente do chão e do céu.

O "pano de fundo" especificado por Background não é modificado por operações de escala e translação sobre os componentes da cena, mas é alterado quando rotações são aplicadas aos objetos do ambiente. Isto significa que o usuário não consegue aproximar-se ou afastar-se do fundo, mas pode observá-lo em todas as direções e perceber que ele muda quando se observa o mundo a partir de pontos de vista diferentes.

Os campos:

frontUrl, backUrl, rightUrl, leftUrl, topUrl, bottomUrl – definem seis imagens, cada uma das quais será mapeada sobre uma face de um cubo. Este cubo forma um panorama que cerca os objetos do mundo virtual por todos os lados, e que, geralmente, é constituído de paisagens, nuvens, montanhas e outras imagens que lembra o mundo real.

No nó **appearance**, utilizaremos o campo **texture** que contém entre outros o nó **ImageTexture**

ImageTexture especifica um mapa de textura e parâmetros para aplicar esse mapa aos objetos. O mapa de textura é uma imagem bidimensional que se estende de 0 a 1 na horizontal (*s*) e na vertical (*t*).

Imagens de textura podem ser formadas de uma (escala de cinza), duas (escala de cinza mais transparência), três (cores RGB) ou quatro (cores RGB mais transparência) componentes. A imagem modifica a cor difusa (*diffuseColor*) e a transparência (*transparency*) especificados no nó Material de um objeto, afetando os cálculos de iluminação que são feitos para produzir a imagem final deste objeto.

Na utilização deste nó iremos explorar o campo **url**:

Url do arquivo de textura. Se múltiplos urls são especificados, eles são listados em ordem descendente de precedência (o browser carrega a primeira imagem da lista que conseguir). A principio, as texturas podem estar nos formatos JPEG ou PNG, mas a maioria dos broesers suportam também o formato GIF.

Nó TimeSensor

Um nó TimeSensor gera eventos com o passar do tempo. Pode ser usado para iniciar **animações** (geralmente em conjunto com um interpolador), para causar a ocorrência de uma única ação (como ligar ou desligar um alarme) em um certo horário, ou para gerar eventos a intervalos regulares de tempo.

Geralmente, o campo *startTime* é setado por um evento de tempo roteado a partir de outro sensor ou de um script, em resposta a ações do usuário. O sensor de tempo não tem utilidade até *startTime* seja alcançado. Nesse momento, o sensor gera um evento *isActive* com valor TRUE e começa a gerar os eventos *time*, *fraction_changed* e *cycleTime*.

Os eventos *time* sempre têm valor igual ao horário corrente. Os outros eventos relacionados com tempo são gerados seguindo padrões chamados *ciclos*. Se *loop* é FALSE, o sensor roda por um único ciclo de duração *cycleInterval* (ou até que *stopTime* seja atingido, se isso ocorrer antes do fim do primeiro ciclo). Se *loop* é TRUE, o sensor continua a rodar indefinidamente, a não ser que *stopTime* seja alcançado ou que *enabled* seja setado para FALSE. No início de cada ciclo, o sensor envia um evento *fraction_changed* com valor 0, e um evento *cycleTime* com valor igual ao horário corrente. Durante o período do ciclo, toda vez que o browser permite que o sensor gere um evento, o valore *fraction_changed* aumenta, indicando a fração do ciclo atual que já se passou. Quando *fraction_changed* atinge 1, o ciclo atual termina e o próximo é iniciado. O evento *cycleTime* é gerado apenas no início de cada ciclo.

Nada garante uma freqüência fixa para a geração de eventos de um sensor de tempo, mas a maioria dos browsers gera esses eventos uma vez a cada quadro de animação. Se o browser é capaz de mostrar 15 quadros por segundo, por exemplo, o sensor poderia gerar 15 eventos de tempo a cada segundo.

Destacaremos os seguintes campos:

cycleInterval define a duração de cada ciclo, em segundos. Deve ser maior que zero;

loop se for TRUE indica se o sensor deve repetir indefinidamente ou se for FALSE para depois de um ciclo;

Nó OrientationInterpolator

OrientationInterpolator é outro nó de interpolação usado para animações. Neste caso, os valores interpolados correspondem a diferentes orientações do objeto.

Campos:

key – é uma lista de tempos, cada qual representado como uma fração de tempo total da animação.

keyValues – é uma lista de valores de orientação (um para cada quadro da animação) que serão interpolados.

Rotas

A conexão entre o nó que gera um evento e o nó que o recebe é chamada de *rota*. Um evento de um nó pode ser roteado para um evento do mesmo tipo de outro nó de acordo com a seguinte sintaxe:

ROUTE NodeName1.eventOutName_changed TO NodeName2.set_eventInName

onde NodeName1 é o nome do nó que envia o envento, eventOutName é o nome do evento de saída, NodeName2 é o nome do nó que recebe o evento e eventInName é o nome do evento de entrada.

Quando um evento é roteado para outro, eles não precisam ter o mesmo nome, mas têm que ser, necessariamente, do mesmo tipo.

É importante destacar que rotas não são nós. A palavra-chave **ROUTE** é meramente uma construção sintática usada para estabelecer um caminho de mensagens entre nós através da estrutura da cena.

Utilizando os conceitos discutimos, temos:

ROUTE mercurioClock.fraction_changed TO mercurioPath.set_fraction

ROUTE mercurioPath.value_changed TO mercurio.set_rotation

Vejamos agora a implementação num exemplo que simula parcialmente o sistema solar, exibindo o Sol e os três planetas iniciais, isto é, Mercúrio, Vênus e Terra.

```
#VRML V2.0 utf8

Group {
   children [

        Background {
        backUrl       "D:\Beletti\Fals\CGRAF\Sistma_solar\universo2.jpg"
        frontUrl      "D:\Beletti\Fals\CGRAF\Sistma_solar\universo2.jpg"
        leftUrl       "D:\Beletti\Fals\CGRAF\Sistma_solar\universo1.jpg"
        rightUrl      "D:\Beletti\Fals\CGRAF\Sistma_solar\universo1.jpg"
        topUrl        "D:\Beletti\Fals\CGRAF\Sistma_solar\universo.jpg"
        bottomUrl     "D:\Beletti\Fals\CGRAF\Sistma_solar\universo.jpg"
   }

   #Sol
        DEF   sol Transform {
              translation 0.0 0.0 0.0
              children Shape {
                   appearance Appearance {
                     texture   ImageTexture
                        {url "D:\Beletti\Fals\CGRAF\Sistma_solar\sol.jpg"}
                        }
                        #sol
                        geometry Sphere    {
                        radius 8
                        }
               }
        }
        DEF brilho_sol2 Transform {
          translation   0.0 0.0 0.0
         children Shape {
            appearance Appearance {
               material Material {
                                          diffuseColor 0 0 0
                                          specularColor .62 .55 .26
                                          ambientIntensity 0
                                          shininess .05
                                          emissiveColor 1 .49 .11
                                          transparency .549
                                    }
            }
            geometry Sphere { radius 10.5}
         }
      },
```

```
# Definição das orbitas
    # mercurio
            DEF   mercurio Transform {
                  translation 31.0 0.0 0.0
                  center -31.0 0.0 0.0
          children Shape {
                    appearance Appearance {
                          texture         ImageTexture
                          {url "D:\Beletti\Fals\CGRAF\Sistma_solar\mercurio.jpg"}

                  }
                  geometry Sphere    {radius 1}
             }
       },
    # venus
            DEF   venus Transform {
                  translation 40.0 0.0 0.0
                  center -40.0 0.0 0.0
            children Shape {
                    appearance Appearance {
                          texture         ImageTexture
                          {url "D:\Beletti\Fals\CGRAF\Sistma_solar\venus.jpg"}

                  }
                  geometry Sphere    {radius 3}
            }
       },
    # terra
            DEF   terra Transform {
                  translation 50.5 0.0 0.0
                  center -50.5 0.0 0.0
            children Shape {
                  appearance Appearance {
                          texture         ImageTexture
                          {url "D:\Beletti\Fals\CGRAF\Sistma_solar\terra.jpg"}

                  }
                  geometry Sphere    {radius 2.5}
        }
       },

  # Animação definindo o relogio um para cada planeta

   DEF mercurioClock TimeSensor {
      cycleInterval 3
      loop TRUE
   },

      DEF venusClock TimeSensor {
         cycleInterval 2.5
         loop TRUE
      },

      DEF terraClock TimeSensor {
         cycleInterval 3.5
         loop TRUE
      },

      # Animação caminho, um para cada planeta

      DEF mercurioPath OrientationInterpolator {
         key [ 0.0, 0.50, 1.0 ]
         keyValue [
            0.0 55.0 0.0  0.0,
            0.0 55.0 0.0  3.14,
            0.0 55.0 0.0  6.28
         ]
      },

      DEF venusPath OrientationInterpolator {
         key [ 0.0, 0.50, 1.0 ]
         keyValue [
            0.0 50.0 0.0  0.0,
            0.0 50.0 0.0  3.14,
            0.0 50.0 0.0  6.28
         ]
      },

      DEF terraPath OrientationInterpolator {
         key [ 0.0, 0.50, 1.0 ]
         keyValue [
            0.0 40.0 0.0  0.0,
            0.0 40.0 0.0  3.14,
            0.0 40.0 0.0  6.28
         ]
      }
   ]
}
ROUTE mercurioClock.fraction_changed   TO mercurioPath.set_fraction
ROUTE venusClock.fraction_changed   TO venusPath.set_fraction
ROUTE terraClock.fraction_changed   TO terraPath.set_fraction

ROUTE mercurioPath.value_changed TO mercurio.set_rotation
ROUTE venusPath.value_changed TO venus.set_rotation
ROUTE terraPath.value_changed TO terra.set_rotation
```