	Disciplina	CGRAF
	Curso	SI
ALEXANDRE BELETTI FERRERA	AULA - 12	

Nesta aula iremos explorar a translação e rotação de sólidos e suas respectivas construções.

Nó Transform

Transform é um nó de agrupamento que define um sistema de coordenadas para seus filhos que é relativo aos sistemas de coordenadas de seus antecessores no grafo de cena.

Os campos *bboxCenter* e *bboxSize* podem ser opcionalmente usados para especificar uma caixa limitante máxima para os objetos neste nó. O browser usa esta caixa em processos de otimização para determinar se o grupo necessita realmente ser desenhado. A caixa limitante deve ser grande o suficiente para conter completamente todos os filhos do brupo, incluindo os efeitos de todos os nós de som, luz e neblina que pertencem ao agrupamento. Se o tamanho do nó de agrupamento muda com o tempo porque os nós-filhos estão participando de uma animação, a caixa deve ser grande o suficiente para conter os objetos em todos os possíveis estágios da animação.

Os campos *translation*, *rotation*, *scaleOrientation* e *center* definem uma transformação geométrica 3D que é composta na seguinte ordem:

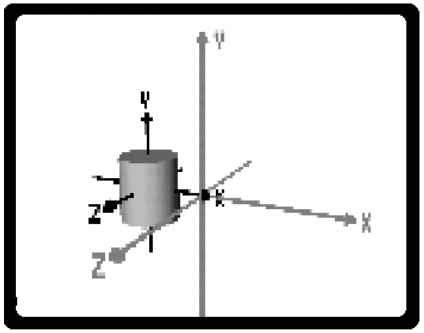
1. uma escala de valor arbitrário;
2. uma rotação de um ângulo qualquer em torno de um eixo arbitrário;
3. uma translação.

Campos:

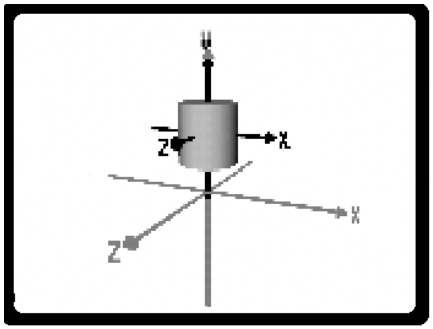
bboxCenter	centro da caixa limitante que circunda os nós-filhos deste nó Transform;
bboxSize	dimensões em *x*, *y* e *z* da caixa limitante que circunda os nós-filhos deste nó. O valor padrão significa que não há caixa limitante;
translation	especifica um vetor de translação;
rotation	especifica uma rotação expressa através de um eixo e de um ângulo em radianos;
scale	especifica uma escala, que pode ser não-uniforme (que distorce o objeto, modificando suas proporções originais);
scaleOrientation	especifica a orientação rotacional para a operação de escala;
center	especifica a origem para as operações de escala e rotação;
children	grupo de nós-filhos que são afetados pelas transformações especificadas neste nó Transform.

Vejamos agora alguns exemplos da aplicação deste nó, para tanto, digite as seguintes linhas de códigos:

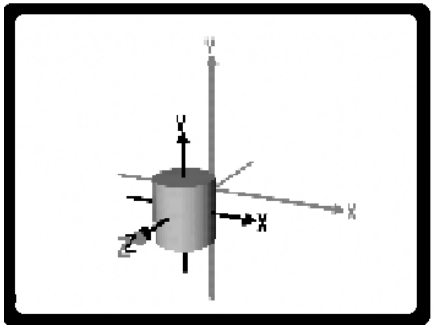
```
#VRML V2.0 utf8
Transform {
   translation -2.0 0.0 0.0
   children [
      Shape {
         appearance Appearance {
            material Material { }
         }
         geometry Cylinder { }
      }
   ]
}
```




```
#VRML V2.0 utf8
Transform {
   translation 0.0 2.0 0.0
   children [
      Shape {
         appearance Appearance {
            material Material { }
         }
         geometry Cylinder { }
      }
   ]
}
```

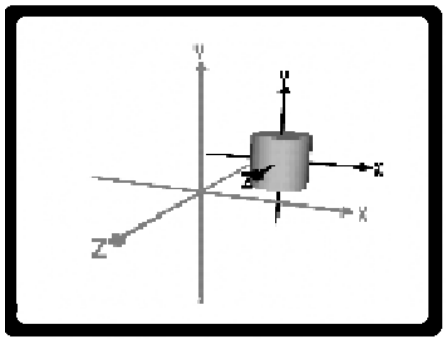



```
#VRML V2.0 utf8
Transform {
   translation 0.0 0.0 2.0
   children [
      Shape {
         appearance Appearance {
            material Material { }
         }
         geometry Cylinder { }
      }
   ]
}
```



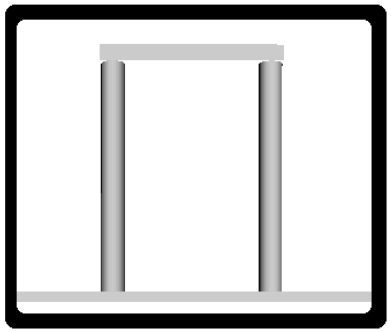
Vejamos agora a combinação da translação nos três eixos (x,y,z), assim sendo, digite as seguintes linhas de códigos:

```
#VRML V2.0 utf8
Transform {
   translation 2.0 1.0 -2.0
   children [
      Shape {
         appearance Appearance {
            material Material { }
         }
         geometry Cylinder { }
      }
   ]
}
```



Analisemos agora como gerar uma construção explorando estes recursos, então, digite as seguintes linhas de códigos:

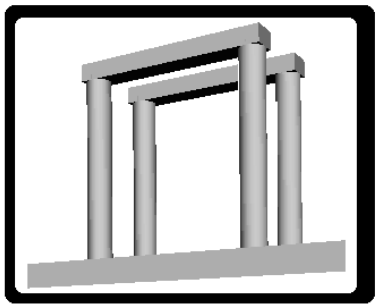
```
#VRML V2.0 utf8
Group {
   children [
      #Base
      Shape {
         appearance DEF Padrao Appearance {
            material Material { }
         }
         geometry Box {
            size 25.0 0.1 25.0}
      },
      # Coluna da esquerda
      Transform {
         translation -2.0 3.0 0.0
         children Shape {
            appearance USE Padrao
            geometry Cylinder {
               radius 0.3
               height 6.0}
         }
      },
      # Coluna da direita
      Transform {
         translation 2.0 3.0 0.0
         children Shape {
            appearance USE Padrao
            geometry Cylinder {
               radius 0.3
               height 6.0}
         }
      },
```



```
      # Coluna do teto
      Transform {
         translation 0.0 6.05 0.0
         children Shape {
            appearance USE Padrao
            geometry Box {
               size 4.6 0.4 0.6}
         }
      }
   ]
}
```

Criaremos agora uma composição dupla permitindo a geração de um cenário mais elaborado, para tanto, digite as seguintes linhas de códigos:

```
#VRML V2.0 utf8
Group {
   children [
   #Base
      Shape {
         appearance DEF Padrao Appearance {
            material Material { }
         }
         geometry Box {
            size 25.0 0.1 25.0}
      },
   #Primeira construção
   # Coluna da esquerda
      DEF Esquerda Transform {
         translation -2.0 3.0 0.0
         children DEF Coluna Shape {
            appearance USE Padrao
            geometry Cylinder {
               radius 0.3
               height 6.0}
         }
      },
   #Coluna da direita
      DEF Direita Transform {
         translation 2.0 3.0 0.0
         children USE Coluna
      },
   # Coluna do Teto
      DEF Teto Transform {
         translation 0.0 6.05 0.0
         children Shape {
            appearance USE Padrao
            geometry Box {
```

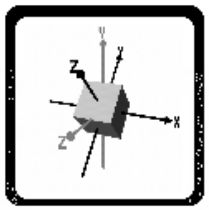


```
                size 4.6 0.4 0.6}
            }
        },
    # Segunda construção
        Transform {
            translation 0.0 0.0 -2.0
            children [
                USE Esquerda,
                USE Direita,
                USE Teto
            ]
        }
    ]
}
```

Iremos estudar agora o campo rotação, para tanto, digite as seguintes linhas de códigos:

Neste caso temos uma rotação em relação ao eixo X com um ângulo de 45º no sentido horário.

```
#VRML V2.0 utf8
Transform {
    rotation 1.0 0.0 0.0  -0.785
    children [
        Shape {
            appearance Appearance {
                material Material { }
            }
            geometry Box { }
        }
    ]
}
```

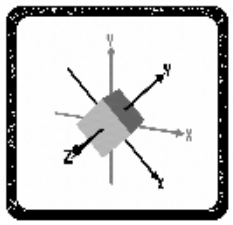


Neste exemplo, temos a rotação em relação ao eixo Y com um ângulo de 45º

```
#VRML V2.0 utf8
Transform {
    rotation 0.0 1.0 0.0  0.785
    children [
        Shape {
            appearance Appearance {
                material Material { }
            }
            geometry Box { }
        }
    ]
}
```

Neste próximo exemplo, temos a rotação em relação ao eixo Z com um ângulo de 45º no sentido horário:

```
#VRML V2.0 utf8
    rotation 0.0 0.0 1.0  -0.785
    children [
        Shape {
            appearance Appearance {
                material Material { }
            }
            geometry Box { }
        }
    ]
}
```



A palavra-chave **DEF** define um nome para um nó, ao mesmo tempo em que cria um objeto daquele tipo. A palavra-chave **USE** indica que uma referência a um nó previamente nomeado deve ser inserida no grafo de cena. O resultado de uma cláusula USE é que um único nó é compartilhado em mais de uma localização da cena. Se o nó original é modificado, todas as referências àquele nó também são modificadas. Se vários nós recebem o mesmo nome, a última ocorrência de DEF é utilizada pelas cláusulas USE.

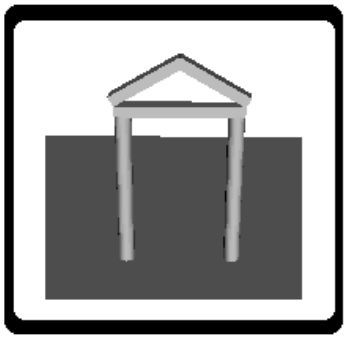
Façamos agora uma construção explorando os conceitos de rotação:

```
#VRML V2.0 utf8
Group {
    children [
    # Cilindro Central
        DEF Principal Shape {
            appearance Appearance {
                material Material { }
            }
            geometry Cylinder {
                height 1.0
                radius 0.1
            }
        },
    # Cilindro inclinado a 60º
        Transform {
            rotation 1.0 0.0 0.0  1.047
            children USE Principal
        },
    # Cilindro inclinado a 120º
        Transform {
            rotation 1.0 0.0 0.0  2.094
            children USE Principal
        }
    ]
}
```

No próximo exemplo, teremos a criação de uma construção explorando a rotação em relação aos eixos Ox e Oy, com variações angulares dando a condição de construir um sólido simétrico no espaço.

```
#VRML V2.0 utf8
Group {
   children [
   # Cilindro Principal
      DEF Principal Shape {
         appearance Appearance {
            material Material { }
         }
         geometry Cylinder {
            height 1.0
            radius 0.1 }
      },
   # Cilindro inclinado a 60º
      DEF Principal60 Transform {
         rotation 1.0 0.0 0.0  1.047
         children USE Principal
      },
   # Cilindro inclinado a 120º
      DEF Principal120 Transform {
         rotation 1.0 0.0 0.0  2.094
         children USE Principal
      },
   # Cilindros rotacionados em Oy num angulo de 102
      Transform {
         rotation 0.0 1.0 0.0  1.785
         children [
            USE Principal60,
            USE Principal120
         ]
      }
   ]
}
```

Com estes conceitos explorados temos condição de construir um portal grego, dando inicio a construções mais elaboradas.



```
#VRML V2.0 utf8
Group {
   children [
   # Ground
      Shape {
         appearance DEF Padrao Appearance {
            material Material { }
         }
         geometry Box {
            size 25.0 0.1 25.0
         }
      },
   # Pilastras
   # Coluna da esquerda
      DEF ColunaEsq Transform {
         translation -2.0 3.0 0.0
         children DEF Coluna Shape {
            appearance USE Padrao
            geometry Cylinder {
               radius 0.3
               height 6.0
            }
         }
      },
   # Coluna da direita
      DEF ColunaDir Transform {
         translation 2.0 3.0 0.0
         children USE Coluna
      },
   # Viga teto
      DEF Viga Transform {
         translation 0.0 6.05 0.0
         children Shape {
            appearance USE Padrao
            geometry Box{
               size 4.6 0.4 0.6
            }
         }
      },
   # Viga esquerda
      DEF VigaEsq Transform {
         translation -1.15 7.12 0.0
         rotation 0.0 0.0 1.0  0.524
         children DEF Teto Shape {
            appearance USE Padrao
            geometry Box {
               size 2.86 0.4 0.6
            }
         }
      },
   # Viga direita
      DEF VigaDir Transform {
         translation 1.15 7.12 0.0
         rotation 0.0 0.0 1.0  -0.524
         children USE Teto
```

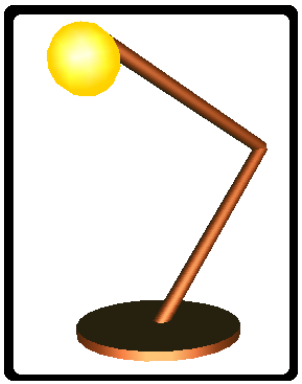
```
        }
    ]
}
```

Vamos fazer agora a construção de uma luminária de mesa, para tanto, digite as linhas de códigos abaixo:

```
#VRML V2.0 utf8

Group{
    children [
    # Base da luminaria
        Shape {
            appearance DEF Padrao Appearance {
                material Material {
                                        diffuseColor .44 .1 0
                                        specularColor 1 .68 .51
                                        ambientIntensity .0833
                                        shininess .07
                                        emissiveColor .15 .13 .06
                                    }
            }
            geometry Cylinder {
                radius 0.1
                height 0.01
            }
        },
    # Posição da composição da luminaria
        Transform {
            translation 0.0 0.15 0.0
            rotation    1.0 0.0 0.0  -0.7
            center      0.0 -0.15 0.0

    children [
          # Primeira duto da luminaria
            DEF Duto Shape {
                appearance USE Padrao
                geometry Cylinder {
                    radius 0.01
                    height 0.3
                }
            },
          # Segundo duto da luminaria
            Transform {
                translation 0.0 0.3 0.0
                rotation    1.0 0.0 0.0  1.9
                center      0.0 -0.15 0.0
```



```
                children [
                # ativa o duto criado acima
                    USE Duto
                ]
            },
            Transform {
                translation 0.0 0.3 0.2
                rotation    1.0 0.0 0.0  2.6
                center 0.0 -0.15 0.0
            # Lampada
                children        [
                    # gera lampada
                    DEF Lampada Shape     {
                            appearance Appearance    {
                                material Material {
                                diffuseColor 0 0 0
                                specularColor 1 1 1
                                emissiveColor 1 .82 0
                                ambientIntensity 0
                                shininess .05
                                }
                            }
                        geometry Sphere {
                            radius 0.04
                            }
                    }
                ]
            }
        ]
    }
  ]
}
```