Introducción a VRML

Algoritmos de Computación Gráfica

Mag. Ing. Walter Daniel Remicio Minami

Universidad Nacional Tecnológica de Lima Sur

Escuela Profesional de Ingeniería de Sistemas

VRML (Virtual Reality Model Language)

Es un lenguaje para modelar objetos en tercera dimensión.

El código se trabaja en archivos/ficheros de texto plano.

Utilizamos el VRMLpad como editor especializado.

Necesita de un visor llamado Cortona para poder visualizarlo de manera local. Pero también existen visualizadores en línea.

Estructura

Similar a las clases utilizadas en POO, en VRML se usan objetos y etiquetas para dar características y funcionalidades al mundo virtual.

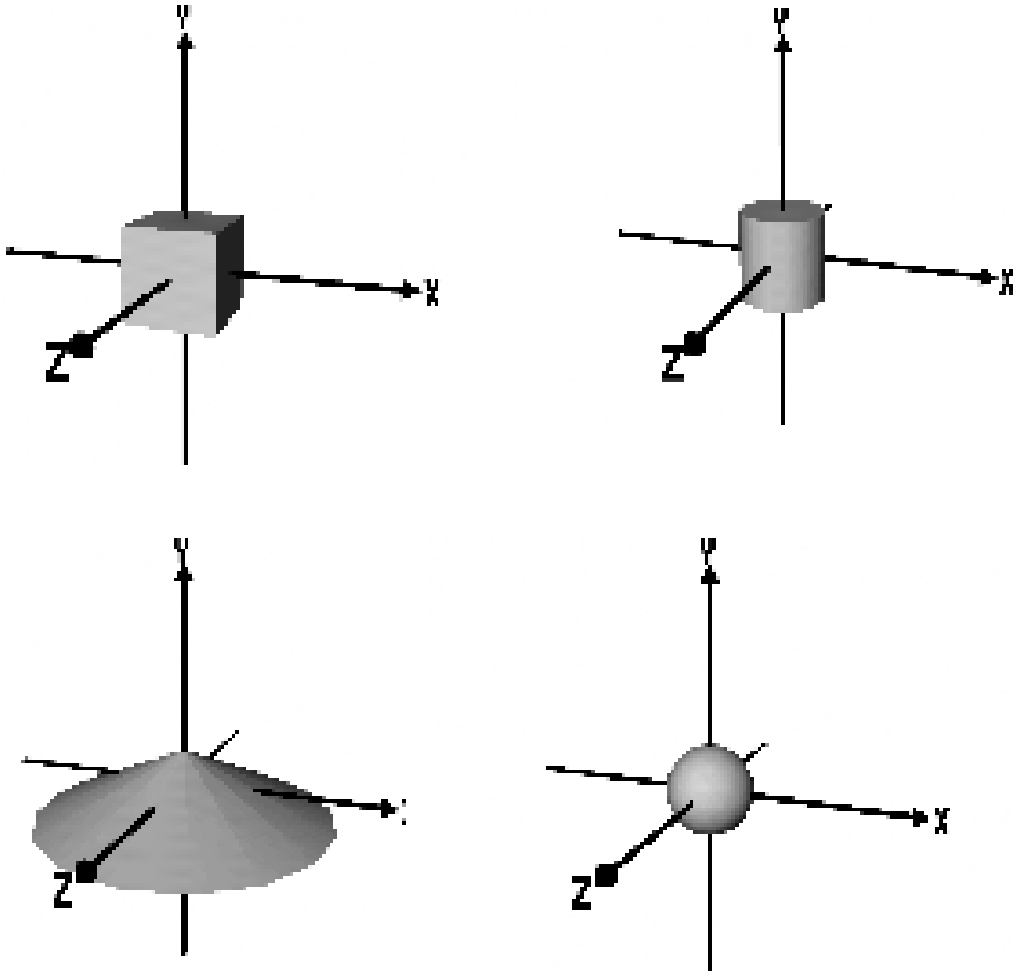
Los objetos inicialmente se crean en la coordenada 0,0,0 (x,y,z) por lo que debemos ir desplazándolos para poder apreciarlos adecuadamente en un mismo escenario.

Se pueden crear objetos en archivos separados y luego importarlos a un proyecto principal.

Shape

Permite definir un objeto tridimensional que se basa en 4 figuras geométricas base: cubo, esfera, cono, cilindro.

Las formas se establecen en el atributo GEOMETRY.



Box

Se usa para definir un cubo.

Con su atributo SIZE se establece el ancho (eje x), el alto (eje y) y la profundidad (eje z)

```
Shape {
     geometry Box {
          size 1 2 4
     }
}
```

Sphere

Se usa para definir una esfera.

Con su atributo RADIUS se establece el tamaño de su radio.

```
Shape {
     geometry Sphere {
          radius 1}
     }
```

Cylinder

Se usa para definir un cilindro.

Con su atributo RADIUS se establece el tamaño de su radio y con HEIGHT la altura.

```
Shape {
    geometry Cylinder {
        radius 0.5
        height 2.5
    }
```

Cone

Se usa para definir un cono.

Con su atributo HEIGHT se establece la altura con BOTTOMRADIUS el radio de la base.

```
Shape {
    geometry Cone{
        height 4
        bottomRadius 1.5
    }
```

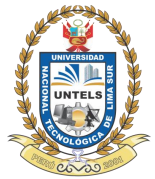
Apariencia

Sirve para establecer algunas características visuales sobre los objetos.

Por ejemplo para establecer un color, este se define en el atributo MATERIAL del atributo APPEARENCE.

```
Shape {
    appearance Appearance{
        material Material{
            diffuseColor  0 0 0
        }
    }

    geometry Box {
        size 1 1 1
    }
}
```



Apariencia

Los colores establecidos son bajo una codificación RGB binario

1 0 0 - Color Rojo
0 0 0 - Color Negro
0 0 1 - Color Azul rey
0 1 0 - Color Verde
1 1 1 - Color Gris
1 0 1 - Color Rosa
1 1 0 - Color Amarillo
0 1 1 - Color Azul cielo

Apariencia

También podemos establecerle una imagen como textura.

La imagen puede encontrarse localmente en un directorio o desde el internet.

```
Shape {
    appearance Appearance{
        texture ImageTexture {
            url ["http....jpg"]
        }
    }

    geometry Box {
        size 5 5 5
    }
}
```

Transform

Mediante la función TRANSFORM podremos mover y rotar la posición de nuestro objeto.

Transform - Translation

Con su atributo TRANSLATION se establece la nueva coordenada donde se ubicará nuestro objeto.

La nueva ubicación se expresa en eje X, Y, Z

```
Transform{
     translation 0 0 0
     children[
          Shape{
          }
     ]
}
```

Transform - Rotation

Con su atributo ROTATION indicamos cuantos radianes girará el objeto.

Además debe indicarse el eje en el que rotará.

Ejemplo: 90 grados en radianes es 1,57

Rad = Grados * PI / 180

```
Transform{
    rotation 0 1 0 1.57
    children[
        Shape {
        }
    ]
}
```

Transform - Scale

Con su atributo SCALE podemos establecer la escala del objeto. Es decir podemos redimensionarlo.

Se establece la escala en los 3 ejes también.

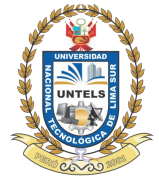
```
Transform{
     scale 2 1 1
     children[
          Shape {
          }
     ]
}
```

DEF

Permite definir un nombre propio a los distintos objetos dentro del archivo VRML.

Puede ser a un SHAPE o hasta propiedades como definir un APPEARANCE o MATERIAL.

```
DEF CuboAzul Shape {
    appearance Appearance {
        material Material {
            diffuseColor 0 0 1
        }
    }
    geometry Box {
        size 1 1 1
        }
    }
```



DEF appearance

```
Shape {
    appearance DEF aparienciaAzul Appearance {
        material Material {
            diffuseColor 0 0 1
        }
    }
}

Shape {
    appearance DEF aparienciaAmarillo Appearance {
        material Material {
            diffuseColor 1 1 0
        }
    }
}

Shape {
    appearance USE aparienciaAmarillo
    geometry Cone {
        height 2
        bottomRadius 1
    }
}
```

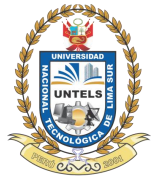
DEF material/texture

```
Shape {
    appearance Appearance {
        material DEF azul Material {
            diffuseColor 0 0 1
        }
    }
}

Shape {
    appearance Appearance {
        material USE azul
    }
    geometry Cone {
        height 2
        bottomRadius 1
    }
}
```

```
Shape {
    appearance Appearance {
        texture DEF Ladrillo ImageTexture {
            url ["ladrillo.jpg"]
        }
    }
}

Shape {
    appearance Appearance {
        texture USE Ladrillo
    }
    geometry Cone {
        height 2
        bottomRadius 1
    }
}
```



GROUP

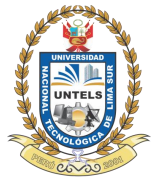
Permite agrupar varios Shape (figuras).

Una vez agrupadas puede aplicarse TRANSFORM a todo el conjunto.

Podemos definir un nombre a todo el GROUP usando DEF.

```
Group {
    children [
        Shape {
            appearance Appearance {
                texture ImageTexture {
                    url ["madera.jpg"]
                }
            }
            geometry Box{
                size 2,2,2
            }
        }

        Transform {
            translation 0,1.5,0
            children [
                Shape {
                    appearance Appearance {
                        texture ImageTexture {
                            url ["tejado.jpg"]
                        }
                    }
                    geometry Cone{
                        height 1
                        bottomRadius 1.5
                    }
                }
            ]
        }
    ]
}
```



Inline

Sirve para poder utilizar un objeto definido en otro archivo VRML.

Al conjunto importado podemos definirle un nombre también utilizando DEF.

```
DEF cubo Inline {
    url "cuboAzul.wrl"
}

Transform {
    translation 4,0,0
    children [USE cubo]
}
```