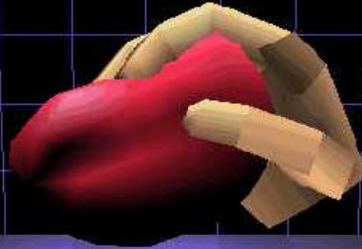


RV não imersiva

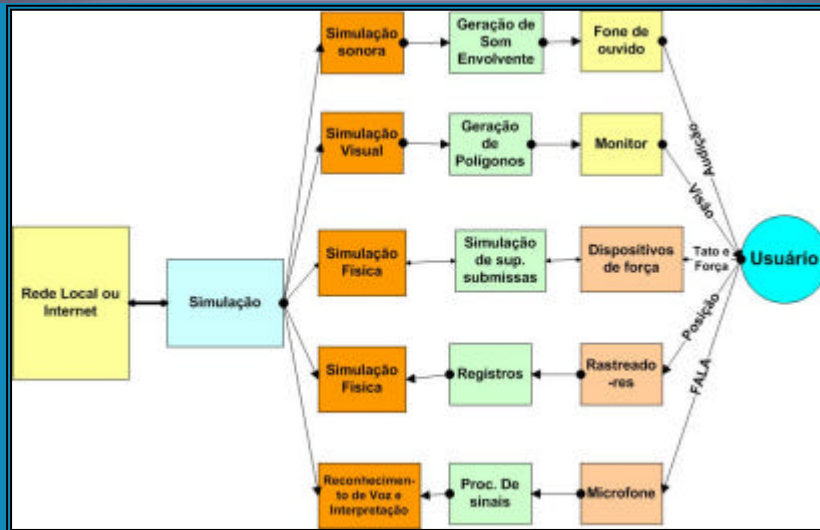


Prof. Dr. Alexandre Cardoso

Vantagens

- utilizar as vantagens da evolução da indústria de computadores;
- evitar as limitações técnicas e problemas decorrentes do uso de capacete;
- e facilidade de uso e custo.

Sistemas de RV não imersivo



Realidade Virtual - Prof. Dr. Alexandre Cardoso

Sistemas de desenv. de RV

- bibliotecas ampliáveis de funções orientadas a objeto;
- voltados para especificações de realidade virtual: um objeto simulado passa a ser uma classe e herda seus atributos inerentes (default);
- simplificam a tarefa de programar mundos complexos: bibliotecas ampliáveis

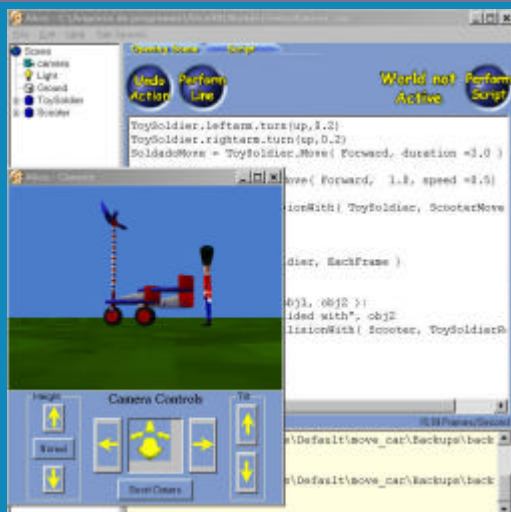
Realidade Virtual - Prof. Dr. Alexandre Cardoso

Sistemas de desenv. de RV



Realidade Virtual - Prof. Dr. Alexandre Cardoso

Alice



Realidade Virtual - Prof. Dr. Alexandre Cardoso

Características - Alice

- iniciou-se na "*University of Virginia*"
- continuou com uma equipe na "*Carnegie Mellon University*"
- desenvolvida em *Python*;
- provê suporte a requisitos de computação gráfica e realidade virtual, sem a exigência do aprendizado destes conceitos.

Realidade Virtual - Prof. Dr. Alexandre Cardoso

Alice

- Diversos objetos 3D já definidos
- Permite alterações e criações de cenários (dinâmicos ou não)
- Comandos de alto nível permitem:
 - Modificação de atributos dos objetos
 - Aplicações de transf. Geométricas
 - Desenvolvimento de cenários
 - Movimentação de câmera
 - Navegação

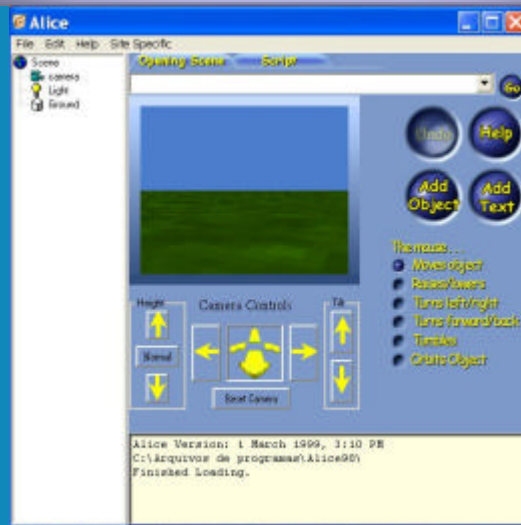
Realidade Virtual - Prof. Dr. Alexandre Cardoso

Alice

- A cena é composta por três elementos: *Camera*, *Light*, *Ground*:
- *Camera*: é usado para definição de ponto de vista do usuário, além de posição do mesmo
- *Light*: definir a iluminação da cena
- *Ground*: especificar os elementos que estarão na cena - objetos e seus atributos

Realidade Virtual - Prof. Dr. Alexandre Cardoso

Alice – GUI – versão 98



Realidade Virtual - Prof. Dr. Alexandre Cardoso

VRML - A Internet em 3D

- Introdução
- Geometria
- Animação
- Iluminação
- Elementos Complementares
- Conclusão
- Espaço Aberto

Realidade Virtual - Prof. Dr. Alexandre Cardoso

Introdução - o nascimento:

- 1994: Mark Pesce e Brian Behlendorf imaginam a possibilidade de desenvolvimento de interface aberta, gratuita, virtual, tridimensional, com multimídia animada e interativa
- VRML 1.0 - Geometria
- VRML 2.0 - Comportamento - Agosto/96

Realidade Virtual - Prof. Dr. Alexandre Cardoso

Introdução - Por que a emoção?

- VMRL tem o potencial de permitir um novo tipo de aplicações - baseadas na *WEB* com simulação distribuída, multiusuário, grupos de discussão em tempo real e até mesmo reuniões tridimensionais;
- Tais aplicações são o resultado de 3 importantes fatores:
 - conectividade em rede
 - interação multiusuário
 - interface com o usuário baseada em modelagem multimídia tridimensional - 3D
- Enfim, a meta final é construir ciberespaços.


Realidade Virtual - Prof. Dr. Alexandre Cardoso

Introdução

- Arquivo VRML = arquivo texto - .wrl
- cabeçalho: `#VRML V2.0 utf8`
- O arquivo texto será uma sequência de nós que conterá a descrição de objetos.
- Um nó pode:
 - conter outro nó - ter um filho - ex: Group
 - ter um conjunto de campos, que contém informações sobre o elemento e que podem estar escritos em qualquer ordem.

Realidade Virtual - Prof. Dr. Alexandre Cardoso

VRML - Como começar?

- Arquivo VRML é um arquivo texto - não tem o que Compilar!!!
- Para editar: qualquer editor de textos, indicado: [VRMLPad](#) 
- Instalar o *plug-in* para VRML, indicado: *CosmoPlayer* (ou Cortona)

O arranjo básico:

```
# VRML V2.0 utf8
Group{
  children [
    nó e campos ....
  ]
}
```


Dicas iniciais - estrutura básica

- Abra o editor de textos...
- Insira o cabeçalho : #VRML v2.0 utf8
- Edite o seu mundo virtual, inserindo as formas que deseja
- Salve no formato “.wrl”
- Abra o *IE Explorer* e carregue!

Descrição de Formas

- Descrição de uma forma:
 - Geometria
 - Aparência
- Geometria:
 - Box, Cone, Cylinder, Sphere
 - Text
 - Coordinate, IndexedLineSet, IndexedFaceSet
 - ElevationGrid, Extrusion

Descrição de Formas

- Estrutura da descrição:

```
Shape {  
  appearance Appearance {}  
  geometry Box {}  
}
```

```
- Box {  
  size 0.2 0.2 0.2}
```

Ⓢ Aqui é descrita a geometria do nó

Formas Básicas

- Sphere {
 radius 0.3}
- Cylinder{
 height 2.0
 bottom TRUE
 side TRUE
 radius 3.0}
- Cone {
 bottom TRUE
 bottomRadius 2.0
 height 3.0
 side TRUE }

Formas Básicas

- Text {
 string "texto"
 fontStyle FontStyle {
 justify "MIDDLE"
 size 2.0
 spacing 0.1
 style "BOLD"}
 length 1.0 }

Descrevendo a Aparência

- Aparência:
 - Definição de Cores como propriedade da Forma
 - Aplicação de Texturas
- Cores:

```
appearance Appearance {  
    material Material {  
        ambientIntensity 1.0  
        diffuseColor 1.0 1.0 1.0  
        specularColor 0.3 0.3 0.3  
        emissiveColor 0.3 0.3 0.4  
        transparency 1.0 }}
```

⑩ exemplo

Aparência - Texturas:

- Texturas:

```
texture ImageTexture {url ""}
```

Ⓢ Endereço da imagem
que será aplicada como
textura

```
textureTransform TextureTransform {  
    scale 3.0 3.0  
    center 0.0 0.0  
    rotation 0.0  
    translation 0.0 0.0}}
```

Ⓢ Fatores de
correção da
textura

- [*exemplo1*](#) - [*exemplo2*](#)

Transformações Geométricas

- Transformações:

- Translação
- Rotação
- Escala

- Nó que suporta as transformações:
transform

```
Transform {  
    translation    0.0 0.0 0.0  
    rotation 1.0 0.0 0.0 1.57  
    scale 2.0 2.0 2.0  
    children [ ] }
```

Ⓢ Lista dos nós que sofrerão as
transformações definidas

Alterando Posição

- Translação:
 - Pode ser nos eixos x , y e/ou z :
 - *translation x y z* (valores em real)
 - Cuidado com números exagerados: risco de perder a forma no cenário definido
- exemplo

Rotacionando a forma

- rotação:
 - Pode ser em torno de qualquer eixo: x , y ou z :
 - *rotation x y z ang* (valor em rad)
 - Cuidado com o eixo que escolhe
- exemplo

Escala

- **escala:**
 - deve ser usado um fator de escala: o mesmo para os três eixos
 - *scale fat_x fat_y fat_z* (valores em real)
 - Observe que a posição pode influir na escala
- exemplo

Reutilização de formas

- Eventualmente, é necessário reutilizar uma forma ou uma aparência, já definida
- Neste caso, a utilização dos termos DEF e USE permite que seja efetivada a reutilização
- DEF *nome*
- USE *nome*
- exemplo

Reutilizando arquivos

- Deseja-se reutilizar um dado arquivo, que define uma forma ou um conjunto de formas
- Neste caso, é preciso indicar o caminho “url” do arquivo, que pode ser local ou arquivo da internet (necessita o caminho completo: <http://...>)
- use o *inline*
- [exemplo](#)

Fundos - cenários

- Veja:
 - [ex. anterior com fundo adequado](#)
- Nó que define fundos: background
- permite usar imagens para composição ou a definição de cores, exclusivamente
- é possível trocar fundos dinamicamente:
 - [exemplo](#)

Iluminação

- Fontes de Luz em VRML
 - headlight
 - Pointlight
 - Directionallight
 - Spotlight
- A importância de uma boa iluminação no realismo
- ausência de sombras!!!!
- Barreiras físicas X fontes de luz

Pointlight

- Luz que emana de um ponto particular no espaço e se espalha igualmente em todas as direções
- pincel de luz: divergente, sem que ocorra uma forma de direcionamento específica
- variáveis importantes:
 - atenuação
 - cor
 - intensidade
 - localização
 - raio

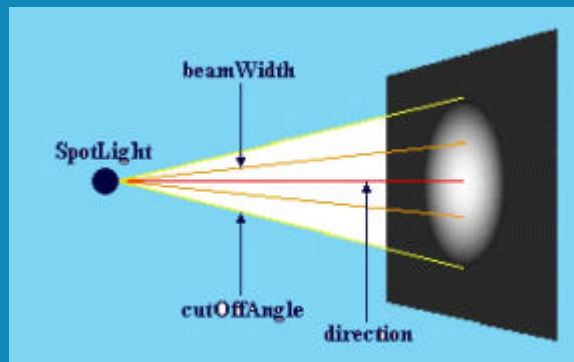
⑩ [Exemplo1](#)

⑩ [Exemplo2](#)

Spotlight

- Luz que emana de um ponto particular no espaço e se espalha, na forma de um cone, em uma direção específica
- variáveis importantes:
 - atenuação
 - cor
 - intensidade
 - localização
 - raio
 - direção
 - ângulo de abertura

Spotlight



⑩ exemplo

Directionallight

- Luz que parte de uma fonte de luz, segundo um pincel paralelo em uma única direção
- Simula um projetor
- variáveis importantes:
 - cor da luz
 - direção
 - intensidade
 - interação com a iluminação ambiente
- exemplo

Exercícios

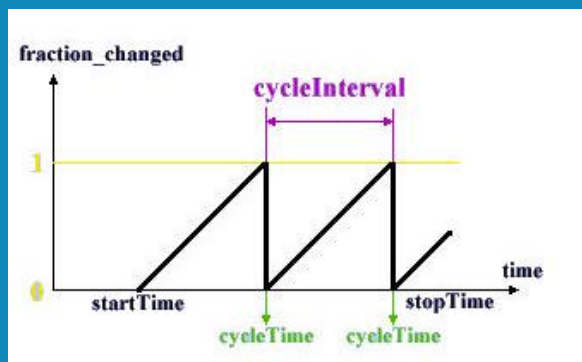
1. Simule um exemplo, onde, aplicando textos, seja apresentado um dipolo elétrico
2. Monte um poste de luz, que gere iluminação adequada
3. Aplicando texturas, simule um muro, onde foi feita uma pixação. Apresente uma abertura para este muro
4. Componha um sistema que, adequadamente, simule um sistema solar

Animando os mundos

- Uma animação presssupõe:
 - tempo
 - alteração de elementos no tempo
 - disparo eventual (?)
- nós relacionados
 - PositionInterpolator
 - OrientatorInterpolator
 - timesensor
 - sensores

Sensores

- TimeSensor: é basicamente, um timer
 - variáveis importantes:



Sensores

- **VisibilitySensor**: relativo a um volume no formato de uma caixa (invisível) que envia eventos quando o usuário entra ou sai dela
 - variáveis importantes:
 - centro da caixa
 - tamanho da caixa
 - se está ou não habilitado
- **ProximitySensor**: similar ao **VisibilitySensor**, exceto por parâmetros de saída
- [exemplo](#)

Sensores

- **Collision**
 - controla a colisão do navegante com as formas que estão definidas dentro de seu campo *children*
 - variáveis importantes:
 - elementos definidos dentro do nó
 - tamanho da região de colisão
 - collide - TRUE ou FALSE
- [exemplo](#)

Montando animações

- O contador de tempo:

```
DEF cont_tempo TimeSensor {  
    startTime 0.0  
    loop TRUE  
    cycleInterval 5.0  
},
```
- Este nó envia frações de tempo a outro nó, quando há uma definição de rota adequada
- Sem a definição de rota, não há qualquer informação de saída deste nó
- é preciso inserir o controlador da animação

Montando animações

- O controlador de animação (posições, neste caso):

```
DEF cont_pos PositionInterpolator {  
    key    [ 0.0 0.2 0.4 0.6 0.8 1.0]  
    keyValue [  
        0.0 0.0 0.0  
        1.0 0.0 0.0  
        1.0 1.0 0.0  
        0.0 1.0 0.0  
        0.0 2.0 0.0  
        1.0 2.0 0.0  
    ] }  
}
```

Montando animações

- Resta agora, definir a forma que será animada:

```
DEF esfera Transform {  
    children Shape{  
        appearance Appearance {  
            material Material { }  
        }  
        geometry Sphere {  
            radius 0.3}  
        }  
    }  
}
```

Montando animações

- Finalmente, são feitas as rotas que controlam a animação, de forma que eventos de um nó sejam enviados a outro nó - troca de mensagens entre os objetos do cenário virtual:
- ROUTE cont_tempo.fraction_changed TO cont_pos.set_fraction

⑩Nó que recebe

⑩Nó que envia

- ROUTE cont_pos.value_changed TO esfera.set_translation
- [exemplo1](#)
- [exemplo2](#)

Montando animações

- A animação não necessita ser de posição, pode ser, por exemplo, de escala do elemento
- Neste caso, os fatores que são utilizados no elemento de controle devem ser mapeados para valores de escala:
- ROUTE cont_esc.value_changed TO elemento.set_scale
- [exemplo](#)

Montando animações

- Para animar a rotação de uma forma, é necessário definir em torno de que eixo a mesma ocorrerá e será usado outro nó:
 - Orientator Interpolator
- Neste caso, as rotas devem alterar a rotação de um nó transform:
- [exemplo](#)

Sons e Filmes

- A inserção de sons e de filmes pode tornar o ambiente virtual mais realístico
- é possível inserir som ativado pelo toque, aproximação ou movimentação do usuário
- é possível inserir um som ambiente - música de fundo
- permite-se: midi, wav, mpeg
- não se permite o mp3

Adicionando Som

- **Nós utilizados para sons e filmes:**
 - AudioClip - descrição de uma fonte de som
 - MovieTexture - inserção de filmes
 - Sound - emissores de sons
- AudioClip {
 url ""
 startTime 0.0
 stopTime 0.0
 loop FALSE
 description ""
 pitch 1.0 }

Adicionando Som

```
• Sound {  
    source AudioClip {}  
    direction 0.0 0.0 0.0  
    location 0.0 0.0 0.0  
    spatialize TRUE  
    maxBack      10.0  
    maxFront 10.0  
    minBack      0.0  
    minFront 0.0  
    intensity 1.0  
    priority 0.0  
}
```

 [exemplo](#)

Links para outros AV ou sites

- Dentro de um ambiente, podem ser adicionados links para outros ambientes virtuais ou outras páginas web (a critério do desenvolvedor)
- **Nó relacionado: Anchor**
- **nó Anchor é um nó de agrupamento**
 - formas que servem para ativar os links devem estar declaradas no campo 'children' deste nó
 - pode ser descrito o link para qual será levado o navegador - aparece na barra de status do browser

Adicionando Links

Anchor {

url ""
description ""
parameter []
children []

}

⑩Permite, dinamicamente:

⑩addChildren

⑩removeChildren

exemplo