

**LIGA DE ENSINO DO RIO GRANDE DO NORTE
FACULDADE NATALENSE PARA O DESENVOLVIMENTO DO RN
BACHARELADO EM SISTEMAS DE INFORMAÇÃO**

CONSTRUÇÕES GEOMÉTRICAS E VRML

CONSTRUÇÕES GEOMÉTRICAS E VRML

Esta disciplina é apresentada ao curso de Bacharelado em Sistemas de Informação da Faculdade Natalense para o Desenvolvimento do Rio Grande do Norte como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador:

Prof. Cláudio Damasceno

Andiere Rosendo Dantas Júnior

NATAL/RN

**NATAL/RN
2004**

ANDIERE ROSENDO DANTAS JUNIOR

LIGA DE ENSINO DO RIO GRANDE DO NORTE
FACULDADE NATALENSE PARA O DESENVOLVIMENTO DO RN
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

CONSTRUÇÕES GEOMÉTRICAS E VRML

CONSTRUÇÕES GEOMÉTRICAS E VRML

Andiere Rosendo Dantas Júnior

Monografia apresentada e aprovada em 15 de 23 de março de 2004, pela
banca examinadora composta pelos seguintes membros:

**Monografia apresentada ao curso de
Bacharelado em Sistemas de Informação
da Faculdade Natalense para o
Desenvolvimento do Rio Grande do
Norte como requisito parcial para a
obtenção do título de Bacharel em
Sistemas de Informação.**

Orientador:

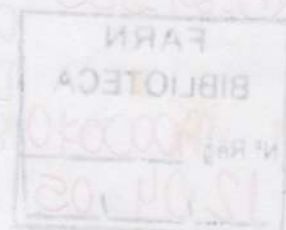
Luiz Gonzaga Damasceno

José Carlos Alves
Membro Examinador

Robson Santana Pacheco
Membro Examinador

NATAL/RN

2004



ANEXO DE ROSARIO VILLAS TORRES

ENCUENTRO DE CRONISTAS VILLAS

El presente documento no tiene
carácter de informe de investigación
ni de informe de gestión, sino que
tiene como finalidad servir de
base para la elaboración de
informes de investigación.

Observaciones

El presente documento

RB-21335

M

004.43:514.12

FARN

BIBLIOTECA

D163c

Nº Reg

0000070

2004

12.04.05

**LIGA DE ENSINO DO RIO GRANDE DO NORTE
FACULDADE NATALENSE PARA O DESENVOLVIMENTO DO RN
BACHARELADO EM SISTEMAS DE INFORMAÇÃO**

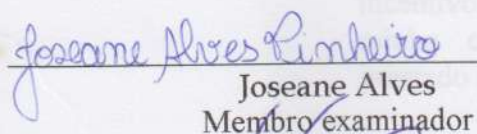
CONSTRUÇÕES GEOMÉTRICAS E VRML

Andiere Rosendo Dantas Júnior

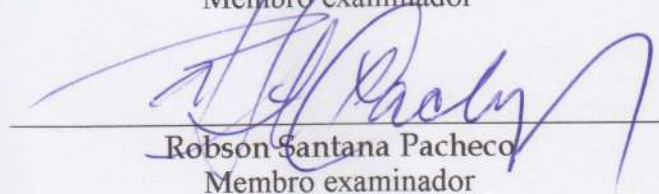
Monografia apresentada e aprovada em 15 de DEZEMBRO de 2004, pela banca examinadora composta pelos seguintes membros:



Luiz Gonzaga Damasceno
Orientador



Joseane Alves
Membro examinador



Robson Santana Pacheco
Membro examinador

Natal, 15 de DEZEMBRO de 2004

DEDICATÓRIA

AGRADECIMENTOS

A Deus, autor da vida, mistério que se faz acontecer através da nossa existência.

Aos meus pais pelo carinho, atenção e o esforço por me dar esta oportunidade, sei que hoje é muito difícil, mas vocês não mediram esforços para me dar as três principais coisas da vida de um ser humano: a vida, o amor, e a educação. Agradeço pela vida, pelo amor e atenção que nunca devem faltar e principalmente por dar-me uma educação exemplar, sei que vocês não tiveram a mesma oportunidade que estão me dando, mas mesmo assim, eu hoje sou o que sou graças a vocês.

A meu querido orientador, Professor Luis Gonzaga Damasceno e demais professores do Curso de Graduação em Bacharelado de Sistemas de Informação, pelos ensinamentos e apoio recebido durante a realização do mesmo.

Aos meus pais dedico este trabalho, como resultado do meu esforço e também do apoio que deles recebi em todos os momentos de minha vida. Ao meu irmão Ranyere pelo incentivo, embora que mesmo de longe e a minha companheira Izabelle pelo apoio prestado nos momentos mais difíceis.

AGRADECIMENTOS

SUMÁRIO

ÍNDICE DE FIGURAS

viii

ÍNDICE A Deus, autor da vida, mistério que se faz acontecer através da nossa existência. ix

RESUMO Aos meus pais pelo carinho, atenção e o esforço por me darem esta oportunidade, em que hoje é muito difícil, mas vocês não mediram esforços para me dar as três principais coisas da vida de um ser humano: a vida, o amor e a educação. Agradeço pela vida, pelo amor e atenção que nunca deixam faltar e principalmente por darem uma educação exemplar, sei que vocês não tiveram a mesma oportunidade que estão me dando, mas mesmo assim, eu hoje sou o que sou graças a vocês.

1.2.2 A meu querido orientador, Professor Luiz Gonzaga Damasceno e demais professores do Curso de Graduação em Bacharelado de Sistemas de Informação, pelos ensinamentos e apoio recebido durante a realização do mesmo.

CAPÍTULO 2

6

2.1 - Referencial Teórico

6

2.2 - VRML

7

2.2.1 - VRML 1.0

8

2.2.2 - VRML 2.0

9

2.2.3 - Características do VRML 2.0

11

2.2.4 - VRML 97

14

2.2.5 - Importância do VRML

14

2.2.6 - Visualizadores

15

2.2.7 - O Futuro do VRML

15

2.2.8 - Conceitos Básicos

16

2.2.9 - Gráficos 3D

18

2.2.10 - Conceitos de Figuras Geométricas

22

2.2.11 - Estudo das Construções Geométricas

23

2.2.11.1 - Circunferência

23

2.2.11.2 - Elipse

24

2.2.12 - Exemplos de Figuras Geométricas em VRML

27

2.2.12.1 - Box

28

2.2.12.2 - Cilindro

29

SUMÁRIO

ÍNDICE DE FIGURAS	viii
ÍNDICE DE ALGORITMOS	ix
RESUMO	x
ABSTRACT	xi
CAPÍTULO 1	1
1.1 - Introdução	1
1.2 - Objetivos	3
1.2.1 - Objetivos Gerais	3
1.2.2 - Objetivos Específicos	3
1.3 - Justificativa	4
1.4 - Estrutura do Texto	5
CAPÍTULO 2	6
2.1 - Referencial Teórico	6
2.2 - VRML	7
2.2.1 - VRML 1.0	8
2.2.2 - VRML 2.0	9
2.2.3 - Característica do VRML 2.0	11
2.2.4 - VRML 97	14
2.2.5 - Importância do VRML	14
2.2.6 - Visualizadores	15
2.2.7 - O Futuro da VRML	15
2.2.8 - Conceitos Básicos	16
2.2.9 - Gráficos 3D	18
2.2.10 - Conceitos de Figuras Geométricas	22
2.2.11 - Estudo das Construções Geométricas	23
2.2.11.1 - Circunferência	23
2.2.11.2 - Elipse	24
2.2.12 - Exemplos de Figuras Geométricas em VRML	27
2.2.12.1 - Box	28
2.2.12.2 - Cilindro	29

2.2.12.3 – Cone	31
2.2.12.4 – Esfera	33
Figura 2.2.12.1: Exemplo de Arquivo VRML	33
CAPÍTULO 3	34
3.1 – Introdução	34
Figura 3.1.1: Exemplo de Rotação Sobre o eixo X	34
CAPÍTULO 4	47
4.1 – Contribuições Finais	47
CAPÍTULO 5	48
5.1- Conclusões	48
CAPÍTULO 6	50
Bibliografia	50
Figura 2.2.12.1: Elaboração da fórmula da elipse	26
Figura 2.2.12.1.1: Exemplo de um cubo Branco	28
Figura 2.2.12.2.1: Exemplo de um cilindro cinza	30
Figura 2.2.12.3.1: Exemplo de um cone cinza	32
Figura 2.2.12.4.1: Exemplo de um esfera verde	33
Figura 3.1.1: Exemplo 1	36
Figura 3.1.2: Exemplo 2	38
Figura 3.1.3: Exemplo 3	43
Figura 3.1.4: Exemplo 4	46

ÍNDICE DE FIGURAS

ÍNDICE DE ALGORITMOS

<i>Figura 2.2.3.1 : Exemplo de Arquivo VRML</i>	13
<i>Figura 2.2.9.1 : Exemplo de Eixos XYZ</i>	19
<i>Figura 2.2.9.2 : Exemplo de Rotação Sobre o eixo Y</i>	20
<i>Figura 2.2.9.3 : Exemplo de Rotação Sobre o eixo X</i>	20
<i>Figura 2.2.9.4 : Exemplo de Rotação Sobre o eixo Z</i>	21
<i>Figura 2.2.11.1.1: Exemplo da Circunferência</i>	23
<i>Figura 2.2.11.1.2: Exemplo da construção da circunferência</i>	23
<i>Figura 2.2.11.2.1: Elementos da elipse</i>	24
<i>Figura 2.2.11.2.2: Construção da elipse</i>	24
<i>Figura 2.2.11.2.3: Os Pontos principais da elipse</i>	25
<i>Figura 2.2.11.2.4:Elaboração da fórmula da elipse</i>	26
<i>Figura 2.2.12.1.1: Exemplo de um cubo Branco</i>	28
<i>Figura 2.2.12.2.1: Exemplo de um cilindro cinza</i>	30
<i>Figura 2.2.12.3.1: Exemplo de um cone cinza</i>	32
<i>Figura 2.2.12.4.1: Exemplo de um esfera verde</i>	33
<i>Figura 3.1.1: Exemplo 1</i>	36
<i>Figura 3.1.2: Exemplo 2</i>	38
<i>Figura 3.1.3: Exemplo 3</i>	43
<i>Figura 3.1.4: Exemplo 4</i>	46

ÍNDICE DE ALGORITMOS

<i>Algoritmo 1 : Construção do cone rotacionado</i>	<i>13</i>
<i>Algoritmo 2 : Construção do cubo</i>	<i>28</i>
<i>Algoritmo 3 : Construção do Cilindro</i>	<i>29</i>
<i>Algoritmo 4 : Construção do cone</i>	<i>31</i>
<i>Algoritmo 5 : Construção da esfera</i>	<i>33</i>
<i>Algoritmo 6 : Construção e Animação do cubo</i>	<i>35</i>
<i>Algoritmo 7 : Construção e Animação do cubo</i>	<i>37</i>
<i>Algoritmo 8 : Construção da circunferência</i>	<i>39</i>
<i>Algoritmo 9 : Construção da Elipse</i>	<i>44</i>

Palavras-Chave: Circunferência, cubo, Cilindro, Elipse, VRML.

RESUMO

O presente trabalho busca estabelecer um diálogo entre o processo de construção geométrica puramente racional e o processo paralelo de descoberta intuitiva da forma. Para tal fim se combinam: um princípio geométrico que funciona como lei de geração formal e duas rotinas interativas que permitem criar estruturas geométricas tridimensionais abstratas. O princípio geométrico utilizado é a afinidade entre o círculo, a elipse, a hipérbole e a parábola. As rotinas foram desenvolvidas em linguagem VRML (Linguagem para Modelagem em Realidade Virtual). Para apoiar o aluno na sua tarefa de análise se estabelece uma metodologia de classificação das formas que associa a cada uma delas uma sequência alfanumérica. A corrente alfanumérica reflete a transformação operada durante a criação de cada estrutura. Com auxílio da interatividade, busca-se forçar uma superposição de estados mentais em um exercício de descoberta espacial que tesa o intuitivo e o racional.

Palavras –Chave: Circunferência, cubo, Cilindro, Elipse, VRML

Abstract

This paper will try to establish a dialogue between the process of rational geometric construction and the intuitive process of discovering shapes. This will involve: a geometric principle that works as a geometric generation law and two interactive macros. The geometric principle is the affinity between circles and ellipses. The macros, developed in language reality VRML, allow the students to construct 3D abstract structures. In order to provide support for the student in the task of analysing the 3D structures a methodology has been established to classify shapes giving each one an alphanumeric string. The string reflects the transformation order during the creation of each spacial structure. With the aid of interactivity it is intended that this exercise in discovering space creates a tension allowing the rational and intuitive states of mind overlap.

KEYWORDS: box, circles, cylinder, ellipses, VRML

"Vejo as salas de aula como um ambiente de aprendizado artificial e insulso que a sociedade se fingia e inventa porque quer ambientes artificiais de aprendizado mesmo-se buscando para o aprendizado de diversas importantes do conhecimento, como a escrita, gramática ou matemática escolar. Acredito que a presença do computador nos permitirá mudar o ambiente de aprendizagem fora das salas de aula de tal forma que todo o processo que as escolas usam atualmente ensinar com grandes dificuldades, despesas e limitado sucesso, será possível como a criança aprende a falar, mesmo desconhecendo, com falta e sem instrução organizada, livre, ingênua, dinâmica, que nasce como se que conhecemos hoje não teria lugar no futuro" (Papert, 1993, p. 43)

Para a consolidação da hipótese colocada, sistematicamente utilizei em paralelo parte do aporte bibliográfico, aproveitei-se para fazer uma integração entre os campos da álgebra – geometria analítica – e da expressão gráfica – geometria descritiva. Deixei, na mesma forma, valores dos estudos recursos computacionais gráficos e de linguagem de

CAPÍTULO 1

Este capítulo faz uma breve introdução do assunto tratado nesta monografia, abordando os aspectos que motivaram a realização do trabalho, bem como de seus objetivos, finalizando com a disposição da organização dos capítulos ao longo do texto apresentado.

1.1 - Introdução

O presente trabalho teve como finalidade de destacar e proporcionar o estudo de Construções Geométricas existente na área didática, dentro do campo da geometria analítica, mais especificamente relacionado ao estudo das curvas cônicas e, dentre estas, da hipérbole, da circunferência, da elipse. Utilizando um pequeno histórico, consolidado por uma referência bibliográfica, aonde procura-se demonstrar o VRML na área da educação principalmente na área da Matemática, bem como seus conceitos, características e funcionalidade. Refere-se à apresentação da origem e geração das curvas cônicas, principalmente em termos de sua visualização plana e espacial, vinculadamente ao posicionamento entre as superfícies cônicas de duas folhas e o plano secante que define a curva.

“Vejo as salas de aula como um ambiente de aprendizado artificial e ineficiente que a sociedade foi forçada a inventar porque seus ambientes informais de aprendizado mostram-se inadequados para a aprendizagem de domínios importantes do conhecimento, como a escrita, gramática ou matemática escolar. Acredito que a presença do computador nos permitirá mudar o ambiente de aprendizagem fora das salas de aula de tal forma que todo o programa que as escolas tentam atualmente ensinar com grandes dificuldades, despesas e limitado sucesso, será aprendido como a criança aprende a falar, menos dolorosamente, com êxito e sem instrução organizada. Isso implica, obviamente, que escolas como as que conhecemos hoje não terão lugar no futuro” (Papert, 1985, p. 45)

Para a consolidação da hipótese colocada, sistematicamente omissa em razoável parcela do aporte bibliográfico, aproveitou-se para fazer uma integração entre os campos da álgebra – geometria analítica – e da expressão gráfica – geometria descritiva. Da mesma forma, valeu-se dos atuais recursos computacionais gráficos e de linguagem de

programação VRML que sedimentou e facilitou a visualização de geração da curva, para nos ajudar a visualizar o sistema como ele é ou como queremos. Constatou-se a necessidade dessa revisão e consolidação em desenho técnico com ênfase em computação gráfica, nos estudos de uma disciplina de geometria analítica, dentro do tópico relacionado às curvas cônicas. Atende-se os objetivos da proposta do curso, que seriam o aprofundamento dos estudos na área e o entrelaçamento entre a expressão gráfica tradicional e sua correspondente vinculação computacional. Esse foi um fator emulador para a realização do presente estudo, proporcionando uma pequena revisão histórica da geometria, no que se refere ao assunto em pauta, com o levantamento dos destaques para a utilização da representação gráfica de uma maneira mais clara, completa, eficaz e auxiliando progressivamente na construção ou modelagem das figuras geométricas.

- Adequar a qualquer superfície desejada;
- Identificar as aplicações de potenciais que podem melhorar se aproveitar desde modelo de renderização;
- Propor métodos que visam adaptar-se à geometria do objeto, permitindo extrair o máximo de informação possível no processo de amostragem;
- Facilitar a etapa de reconstrução da imagem, melhorando sua qualidade final;
- Mostrar os resultados obtidos com o uso desses ferramentas no estudo da geometria plana cada vez mais eficaz, clara e completa;
- Apresentar o computador na Educação Matemática.

1.2 - Objetivos

1.2.1- Objetivos Gerais

O presente trabalho tem como objetivo fazer uma avaliação geral do estudo de Construções Geométricas originadas no estudo da geometria analítica na área da matemática auxiliando na construção do software, melhorando cada vez mais a qualidade de aprendizagem dos alunos.

1.2.2 - Os objetivos Específicos:

- Possibilitar a realização de uma amostragem adaptativa de objetos, que melhor se adeque à qualquer superfície desejada;
- Identificar as aplicações de potenciais que podem melhor se aproveitar deste modelo de renderização;
- Propor métodos que visam adaptar-se à geometria do objeto, permitindo extrair o máximo de informação possível no processo de amostragem;
- Facilitar a etapa de reconstrução da imagem, melhorando sua qualidade final;
- Mostrar os resultados obtidos com o uso dessas ferramentas no estudo da geometria plana cada vez mais eficaz, clara e completa;
- Apresentar o computador na Educação Matemática;

1.3 - JUSTIFICATIVA

O Aluno ao explorar o mundo ao seu redor adquire informações e experiências que influenciam no seu desenvolvimento físico, mental, social e emocional, sendo tais aspectos de grande influência nos traços de sua personalidade.

Com a introdução do computador no ambiente escolar está ocorrendo uma verdadeira revolução no processo de ensino-aprendizagem, pois as diferentes formas que ele pode ser utilizado na educação, as inúmeras informações que podem ser acessadas rapidamente e a conexão de alunos e professores com o mundo, exigem do educador um posicionamento diante dessa realidade, como também um questionamento dos métodos e processos de ensino utilizado.

Com o estudo feito se constatou a afirmação que o computador pode ser um instrumento poderoso e versátil na área da educação, se for utilizado com inteligência e competência, tornando-se um excelente recurso pedagógico que o professor tem a sua disposição para trabalhar em sala de aula. A forma que o computador pode ser utilizado na educação dependerá do conhecimento que o professor tem, do potencial dos computadores e da filosofia de educação. No trabalho educativo diário, uso do computador em sala de aula, se define uma Informática educativa viável a realidade sociocultural.

Conforme coloca Silva:

“A tecnologia é importante, e temos na nossa época um processo irreversível onde a Informática está em todas as atividades da sociedade. Em termos da Educação, esta performance tem que ser analisada mais friamente, pois não é a simples transposição dos conteúdos didáticos dos livros para a tela de um computador que dará um ganho real de aprendizagem”. (Silva, 1998).

O aluno tem a oportunidade de rever conceitos que foram adquiridos no dia-a-dia através da geometria informal, e questionar-se sobre seus próprios conhecimentos, encarando os erros não como derrota e sim como etapas a serem vencidas, pois o esforço ocorre naturalmente resultando em novas relações e percepções.

1.4 Estrutura do Texto

O texto deste trabalho está dividido em 4 capítulos, distribuídos da seguinte forma:

Capítulo 2: Este capítulo introduz o estudo de Construções Geométricas, contextualizando o assunto através dos trabalhos realizados até o momento, e apresentando os passos gerais necessários para a renderização de imagens;

Capítulo 3: Apresenta a implementação das técnicas descritas nos capítulos anteriores bem como os resultados (e a análise destes) obtidos a partir dos experimentos realizados.

Capítulo 4: As principais contribuições realizadas neste trabalho são apresentadas neste capítulo, demonstrando técnicas de amostragem alternativas e procedimentos para a melhoria da qualidade das imagens geradas;

A monografia encerra no *capítulo 5*, ao abordar trabalhos que podem ser ainda desenvolvidos no que foi apresentado, e ainda resumindo as considerações realizadas, de forma a consolidar os conceitos de Construções Geométricas.

CAPITULO 2

2.1 – REFERENCIAL BIBLIOGRÁFICO

O estudo de Construções Geométricas é um importante aspecto da modelagem geométrica que é usada para criar e comunicar informações sobre sua forma. Ela envolve a criação e manutenção do modelo sólido para futuro acesso e análise.

Os primeiros sistemas de projeto assistido por computador (CAD) representavam desenhos planos (entidades bidimensionais) ao invés de sólidos tridimensionais.

Os métodos de representação das figuras denominam também as técnicas de modelagem de sólidos na VRML. (AZEVEDO; CONCI, 2003)

2.2 - VRML

A Virtual Reality Modeling Language, ou Linguagem para Modelagem em Realidade Virtual, é uma linguagem de programação de mundos virtuais para RV que permite a criação de ambientes virtuais, permitindo interagir com esses ambientes.

Uma grande vantagem no uso desta linguagem é que suas aplicações "rodam" sobre a maioria dos browsers, permitindo a utilização da aplicação numa rede, além disso, é totalmente livre, sem nenhum custo. Porém, sua desvantagem é que essa linguagem não é a mais poderosa.

O código VRML é um subconjunto do formato de arquivo ASCII do Open Inventor, da Silicon Graphics, com características adicionais para navegação na Web. Esta característica é equivalente às âncoras do HTML, ou seja, pode-se criar âncoras em um ambiente virtual que levem a outros ambientes virtuais.

A linguagem, trabalha com geometria 3D, permitindo a elaboração de objetos baseados em polígonos, possui alguns objetos pré-definidos como cubo, cilindro e esfera, suporta transformações como rotação, translação e escala, permite a aplicação de texturas, luz, sombreamento, etc. E para escrever o código VRML, é necessário apenas um editor de textos, pois os arquivos são gravados em formato de ASCII com a extensão.wrl. Na verdade, a linguagem apenas descreve como os ambientes tridimensionais devem ser representados, sendo que o arquivo não precisa ser compilado.

A diferença entre a versão de VRML 1.0 para a versão VRML 2.0, é que a primeira tem como principais características a criação de mundos 3D estáticos e a criação de âncoras para outros ambientes, sendo que na versão 2.0, os objetos do mundo virtual podem mover-se e responder a eventos baseados no tempo ou em iniciativas do usuário. Além disso, esta nova versão permite a utilização de objetos multimídia, como sons e filmes em uma cena 3D.

2.2.1 VRML 1.0

Final da década de 80, Tim Berners-Lee criou a World Wide Web, adicionando inovações à Internet em termos de conectividade e de interface.

1992, surgiu a biblioteca gráfica Inventor da Silicon Graphics, para permitir que programadores desenvolvessem, com rapidez, programas gráficos 3D interativos, baseados nos conceitos de cena e na descrição de objetos.

Maio de 1994, em Genebra, na 1ª Conferência da World Wide Web, o grupo de discussão de realidade virtual decidiu desenvolver uma linguagem de descrição de cena que pudesse ser usada na Web.

Outubro de 1994, na 2ª Conferência da World Wide Web, foi decidida a adoção de um formato baseado no metafile do Open Inventor, com modificações para sua utilização em rede.

Maio de 1995, a especificação da VRML 1.0 foi completada, permitindo a criação de cenas estáticas. Em janeiro de 1996 foi lançada a versão 1.0c, corrigindo alguns problemas.

1995, colaboradores da comunidade de VRML formaram o VRML Architecture Group (VAG). Iniciou-se uma discussão sobre extensões à linguagem que suportassem animação e interação. Foi lançado um call for proposals para uma redefinição da linguagem. A Silicon Graphics, Netscape e outras companhias criaram a Moving Worlds (Moving Worlds).

Quando um arquivo VRML (.wrl) é chamado, o *Web browser* executa o visualizador VRML, que possibilita a navegação e visualização desse mundo 3D. Estes mundos possibilitam a criação de objetos com hiperlinks associados para outros mundos, uma página HTML, ou qualquer outro tipo MIME.

Um mundo virtual VRML é um conjunto de objetos que podem conter geometrias, sons MIDI ou WAV, imagens JPEG, luzes, etc. Estes objetos são chamados nodos. Eles são organizados em estruturas hierárquicas chamadas *scene graphs*, que definem uma ordem nos nodos. Um mecanismo definido para delimitar os efeitos, chamados nodos separadores, permite que uma parte da *scene graph* seja isolada de outra.

2.2.2 VRML 2.0

Em 1996, o VRML Architecture Group (VAG) fez um pedido de propostas para a especificação do VRML 2.0. Seis propostas foram recebidas e debatidas por dois meses.

Foi realizada a votação pela comunidade VRML, e a proposta escolhida foi a *Moving Worlds*, da SGI, sendo oficializada em março de 1996.

O VRML 1.0 fornece meios para criar e visualizar mundos 3D estáticos, apresentando uma interface "fria" com o usuário. Na versão 2.0, isso foi melhorado através de mundos estáticos mais aperfeiçoados, interação, animação, possibilidade de *scripts* e prototipação.

A melhoria dos mundos estáticos é observada através da possibilidade de maior realismo nas cenas. Isso é alcançado através da criação de cenas com fundos apresentando o chão e o céu, com a adição de paisagens com montanhas e nuvens, embaçando objetos que estão mais distantes, criação de terrenos irregulares, etc. Além disso, oferece também recursos de som 3D para aumentar o realismo do ambiente gerado.

A aparência de se mover em uma cena "morta" é desfeita nessa versão com a adição de *efeitos interativos*. Novos nodos do tipo sensor proporcionam essa interação: quando você entra em uma área do mundo ou clica em um objeto, o sensor pode setar um evento como acender uma luz. Outro tipo de sensor pode controlar a passagem do tempo, podendo ser usado em animações repetitivas. Outro recurso é a detecção de colisão, assim não será mais possível atravessar por objetos sólidos.

Houve, também, a inclusão de objetos de animação chamados interpoladores, que permitem criar animações de objetos do mundo 3D e então executá-las em outro momento. Com eles, pode-se criar objetos móveis como um pássaro voando, uma porta se abrindo, um robô se movendo. Pode-se mudar a cor ou a forma de um objeto, entre outros recursos de animação. É possível, também, a criação de um passeio guiado por um caminho determinado no mundo virtual.

Um recurso incluído é o nodo tipo *script*, que é fundamental nessa nova versão. Ele possibilita a criação de objetos animados em um mundo, e dá a eles um aspecto mais "inteligente" atribuindo ações determinadas. Esses efeitos são obtidos por meio de eventos, um *script* lê entradas de um sensor e gera eventos baseados nessa entrada que mudará outros nodos nesse mundo.

Outro recurso disponível no VRML 2.0 é a prototipação. Ela permite o encapsulamento de grupo de nodos, originando um novo tipo de nodo, um protótipo. Pode-se, então, criar instâncias desse novo tipo com diferentes valores de campos característicos.

Dessa forma, um protótipo pássaro pode ser instanciado com diversas cores: um pássaro amarelo, outro vermelho; ou com outras características particulares.

Um objeto VRML é um arquivo texto com extensão .vrl, .vrx ou .vri (os dois últimos compatíveis com gzip). Um objeto em VRML inicia sempre com a linha "#VRML V2.0 utf8". O identificador "utf8" permite utilizar caracteres internacionais, como acentos, nos modelos VRML. Toda a especificação de objetos, *links*, atributos ou transformações, é colocada dentro de blocos que são definidos pela expressão *Transform*.

Os comentários em VRML iniciam sempre pelo símbolo # e valem na linha em que aparecem. VRML é case-sensitive, ou seja, diferencia letras maiúsculas e minúsculas.

Quando se cria um objeto em VRML, ele é colocado no centro da tela (0,0,0), isto se não houver aplicação de uma movimentação no objeto.

O sistema de coordenadas é o cartesiano, seguindo a regra da mão direita. Normalmente, os objetos são projetados na direção positiva do eixo Z, com a direção positiva do eixo X para a direita e a direção positiva do eixo Y para cima. Para mudar esse ponto de projeção muda-se a câmara virtual ou usam-se transformações. A unidade de comprimento é o metro e a de ângulo o radiano. O modelo de cores na VRML é o RGB, com valores normalizados entre zero e um.

As primitivas básicas da VRML são os nodos *Box*, *Cone*, *Cylinder* e *Sphere*. Já as formas avançadas são definidas por *Tex*, *Extrusion*, *Grid*, *Extrusion*, *IndexedFaceSet*, *IndexedLineSet* e *PointSet*. Os objetos em VRML podem sofrer transformações (escala, rotação e translação), terem definidos seus materiais, texturas (inclusive animadas) e transformações nas texturas. Os nodos podem ser definidos como um grupo em VRML, com a capacidade de definir um conjunto em que se possa ser reutilizado em outro local.

Também são características da VRML: *hyperlink* para outros modelos VRML, para uma página HTML ou outro tipo que o *browser* pode ler, iluminação do tipo pontual, *directional* ou *spot*, animações e *browser* que recebe uma coleção, com

2.2.3 Características da VRML 2.0

Como os objetos VRML podem ser visualizados através de um *browser*, naturalmente estes são independentes de plataforma, podendo ser visualizados por qualquer tipo de computador, sendo disponível para diferentes ambientes e lugares através da Internet.

Um objeto VRML é um arquivo texto com extensão *.wrl*, *.wrz* ou *.wrl.gz* (os dois últimos compactados com *gzip*). Um objeto em VRML inicia sempre com a linha “#VRML V2.0 utf8”. O identificador “utf8” permite utilizar caracteres internacionais, como acentos, nos modelos VRML. Toda a especificação de objetos, *links*, atributos ou transformações, é colocada dentro de blocos que são definidos pela expressão *Transform*.

Os comentários em VRML iniciam sempre pelo símbolo # e valem na linha em que este aparece. VRML é *case-sensitive*, ou seja, diferencia letras maiúsculas e minúsculas.

Quando se cria um objeto em VRML ele é colocado no centro da tela (0,0,0), isto se o usuário não aplicar uma movimentação no objeto.

O sistema de coordenadas é o cartesiano, seguindo a regra da mão direita. Inicialmente, os objetos são projetados na direção positiva do eixo Z, com a direção positiva do eixo X para a direita e a direção positiva do eixo Y para cima. Para mudar esse padrão de projeção muda-se a câmera virtual ou usam-se transformações. A unidade de comprimento é o metro e a de ângulo o radiano. O modelo de cores na VRML é o RGB, com valores normalizados entre zero e um.

As primitivas básicas da VRML são os nodos *Box*, *Cone*, *Cylinder* e *Sphere*. Já as formas avançadas são definidas por *Text*, *Evaluation Grid*, *Extrusion*, *IndexedFaceSet*, *IndexedLineSet* e *PointSet*. Os objetos em VRML podem sofrer transformações (escala, rotação e translação), terem definidos seus materiais, texturas (inclusive animadas) e transformações nas texturas. Os nodos podem ser definidos como um grupo em VRML com a capacidade de definir um conjunto em que se possa ser reutilizado em outro local.

Também são características da VRML: hyperlink para outros mundos VRML, para uma página HTML ou outro tipo que o *browser* pode ler, iluminações do tipo pontuais, direcionais ou *spot*, animações, o *browser* ser avisado quando ocorrer uma colisão, som

espacial (som 3D) usando arquivos tipo MIDI ou WAV, entre outros. Maiores informações sobre as características da VRML podem ser vistas em [WEB 99].

No exemplo a seguir (figura 2.2.3.1) veremos um exemplo de um arquivo VRML e sua imagem correspondente, no qual um cone irá ser rotacionado em 180° no eixo Z, enquanto o outro cone manterá sua posição normal, transladado de 4 unidades no eixo X.

```

1 #VRML V2.0 utf8
2 Transform {
3     translation    5 0 0
4     children [
5         Shape {
6             appearance Appearance {
7                 material Material {
8                     diffuseColor 0 0 1
9                 }
10            }
11            geometry Cone {
12                bottomRadius 2
13                height 4
14            }
15        }
16    ]
17 }
18
19 Transform {
20     rotation 1 0 0 3.14
21     children [
22         Shape {
23             appearance Appearance {
24                 material Material {
25                     diffuseColor 0 0 1

```

```

26 }
27 }
28 geometry Cone {
29     bottomRadius 2
30     height 4
31 }
32 }
33 ]
34 }

```

Algoritmo 1: Construção do Cone Rotacionado

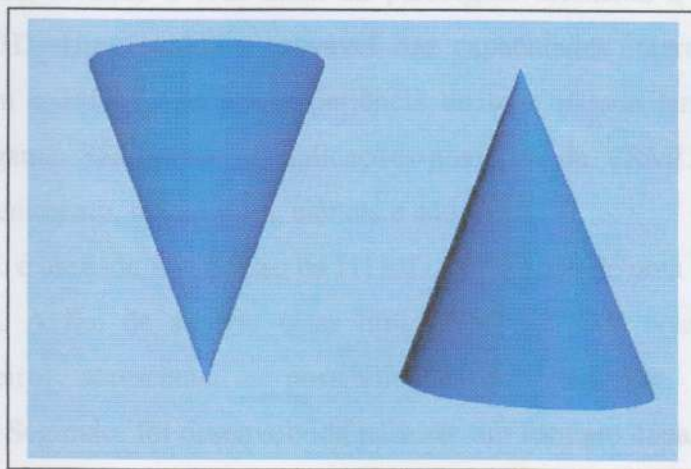


Figura 2.2.3.1: Exemplo de arquivo VRML.
código fonte VRML e imagem correspondente.

Existem atualmente várias páginas onde se pode encontrar maiores detalhes sobre a VRML e aprender a fazer mundos virtuais.

2.2.4 - VRML97

VRML97 é o nome informal do padrão internacional (ISO/IEC 14772-1:1997). É quase idêntico ao VRML 2.0, com muitas melhorias na documentação, mas com poucas diferenças funcionais.

O padrão internacional VRML97 foi desenvolvido pelo *Joint Technical Committee 1* (JTC 1) do ISO em parceria com o VRML Consortium no final de 1997. Para isso, teve como base o texto "VRML 2.0 Specification" (distribuído a partir de agosto de 1996).

2.2.5 Importância da VRML

A Internet, e especialmente a WWW, está mudando a maneira com que a informação é espalhada pelo mundo. A HTML é a base da WWW, fornece a interface para incorporar imagens gráficas bidimensionais (como GIF e JPEG). O paradigma é adequado para alguma informação quanto na capacidade de interação. A VRML permite o próximo nível de interação, levando a Web além do paradigma orientado a documentos para um mundo virtual 3D. Uma analogia provável das capacidades interativas da VRML em comparação com a HTML, seria a experiência de uma pessoa lendo um livro e outra jogando video game. São amplas as aplicações possíveis da VRML, como por exemplo, negócios, entretenimento, manufatura, ciência e educação.

A VRML é baseada no sucesso da HTML, que dispõe de uma interação limitada em duas dimensões. A fim de superar estas limitações, a VRML vem realizando grandes avanços. Primeiro, acrescenta a possibilidade de descrever objetos geométricos tridimensionais. Segundo, foi desenvolvida para ser um formato capaz de não só descrever estes objetos 3D, mas também descrever o procedimento da interação a ser aplicada quando um usuário encontra uma informação e aventura-se em interagir com esse objeto. Esta é a combinação da definição de uma descrição tridimensional com as novas capacidades introduzidas à VRML.

2.2.6 Visualizadores

A VRML se divide, basicamente, em duas áreas: Modelagem e Visualização . A primeira está relacionada com a construção dos objetos da cena a ser visualizada, utilizando bases matemáticas para isso. A segunda busca uma forma de representação visual dos objetos construídos. Esta visualização é uma seqüência de operações para transformar as informações dos objetos modelados no universo para o espaço de tela .

Sendo o objetivo deste trabalho fazer a integração das técnicas de modelagem com VRML, a modelagem geométrica será abordada mais detalhadamente no próximo capítulo. Como o objetivo da VRML é a Internet, a visualização é realizada através de *browsers*. Para isto, é necessário que se instale um *plug-in* adequado. Atualmente, existem alguns visualizadores disponíveis para diferentes plataformas. O mais usado para testar os modelos gerados foram: Microsoft VRML 2.0 Viewer.

2.2.7 O Futuro da VRML

Depois do desenvolvimento do padrão VRML (ISSO/IEC), foi decidida que características mais ambiciosas (como interação multi-usuário ou criaturas autônomas que podem reagir com o ambiente) não fazem parte da versão original da VRML. Atualmente estão sendo estudada implementações e testes em muitas áreas, como: Um formato comprimido para aumentar a velocidade de transmissão dos arquivos e do *parsing*, um dispositivo de interface entre o mundo VRML e um ambiente externo, animação de humanóides para dar uma representação VRML padrão para humanóides, entre outros.

2.2.8 – CONCEITOS BÁSICOS

É uma linguagem interpretada composta por nó, campo e comentário.

Usa-se : `#VRML V2.0 ascii` para que o interpretador reconheça este documento como *VRML* (*.wrl).

- Nó: comando que define formas e propriedades em *VRML*.

Ex. de Primitivas: cubo, esfera, cilindro e cone,

=> *Transform* (move, dimensiona e rotaciona objeto);

=> *AsciiText* (define texto ascii a ser mostrados);

=> *Groupind Nodes* (agrupa informações juntas - para reuso)

- Campo: atributos que os nós tem.

Ex.: Um cubo é descrito pelas propriedades **LARGURA**, **ALTURA** e **PROFUNDIDADE**.

- Comentário: " # " precede todo comentário. Um comentário pode ser feito em qualquer posição na linha.
- _Nodo (*Node*): É o componente fundamental de uma cena em *VRML*. Pode ser definido como abstrações dos objetos e conceitos do mundo real, por exemplo, esfera, cubos, luzes e descrição de materiais. Os nodos contém campos e eventos.
- _Nodos de Geometria (*Geometry node*): é representado pelos tipos *Box*, *Cone*, *Cylinder*, *ElevationGrid*, *IndexedFaceSet*, *IndexedLineSet*, *PointSet*, *Sphere*, *Text*.. Este nodo contém uma descrição de pontos tridimensionais, linhas, superfícies, *string* de textos e objetos sólidos.
- _Nodos filhos (*Children node*): São nodos nos quais vários nodos filhos são agrupados e estes são afetados pelas transformações feitas no nodo pai. Serve para criar uma hierarquia de transformações que são herdadas de pai para filho.
- _Eventos (*Event*): São utilizados para a troca de mensagens de um nodo para outro, através de uma rota. Os eventos sinalizam mudanças nos valores dos campos, estímulos externos, interações entre nodos, etc.

- **_ Campo (Field):** Os parâmetros do campo distinguem um nodo de outro do mesmo tipo.
- Campos podem conter vários tipos de dados e um ou mais valores.
- **_ Nodos de Aparência (Appearance node):** O nodo de Aparência é representado pelos tipos *Appearance*, *FontStyle*, *ImageTexture*, *Material*, *MovieTexture*, *PixelTexture* e *TextureTransform*, e seu objetivo é controlar a aparência renderizada dos nodos geométricos (objetos) as quais estão relacionados.
- **_ Rota (Route):** É a conexão entre o nodo gerador do evento e o nodo receptor do evento.

* *rotation*: muda a orientação de um objeto em graus (radianos).

* *scaling*: muda a dimensão dos objetos.

* *lighting*: aplica luz a cena.

* *texture mapping*: aplica imagens a superfície de um objeto.

Gráficos 3D definem espaços tridimensionais, como o sistema de eixos:

* *X*, que corresponde a largura.

* *Y*, a altura.

* *Z*, a profundidade.

2.2.9 - GRÁFICOS 3D

Exemplificaremos instruções básicas para a criação de mundos *VRML*. Usamos um editor de texto *ascii* e o *browser* de *VRML Live3D*. Eventualmente acrescentamos textura aos arquivos de imagem criados.

Há três elementos neste contexto: o observador, o objeto e a luz.

Um objeto mais complexo pode ser definido nas três coordenadas: X, Y e Z.

E pode-se atribuir certas propriedades como:

- * *material*: como o objeto interage com a luz.
- * *translation*: tradicionalmente é identificado como centro do objeto, modificando esta propriedade, modifica-se o centro do objeto (metros).
- * *rotation*: muda a orientação de um objeto em cena (radianos).
- * *scaling*: muda a dimensão dos objetos.
- * *lighting*: aplica luz a cena.
- * *texture mapping*: aplica imagem a superfície de um objeto.

Gráficos 3D definem espaços tridimensionais, como o sistema de três eixos:

- * X, que corresponde a largura;
- * Y, a altura;
- * Z, a profundidade.

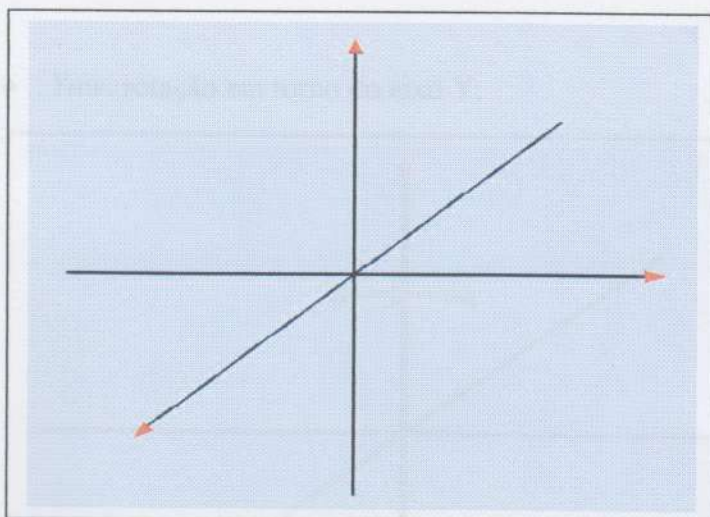


Figura 2.2.9.1 :Eixos XYZ

Assumindo um ponto no centro, a descrição de objetos em espaço tridimensional é expresso com as coordenadas representando com pontos absolutos num sistemas de coordenadas tridimensionais relativos ao ponto central.

Cada eixo tem direção positiva e negativa entendendo-se o ponto central como cenário. *VRML* padroniza que o eixo Y positivo aponta para cima, o eixo X positivo aponta para a direita e o eixo Z positivo aponta direto para o usuário, como mostra fig. 2.2.10.1.

Uma vez que a locação do objeto é estabelecida, relativa ao centro absoluto, pode ser orientada em uma ou mais de três direções:

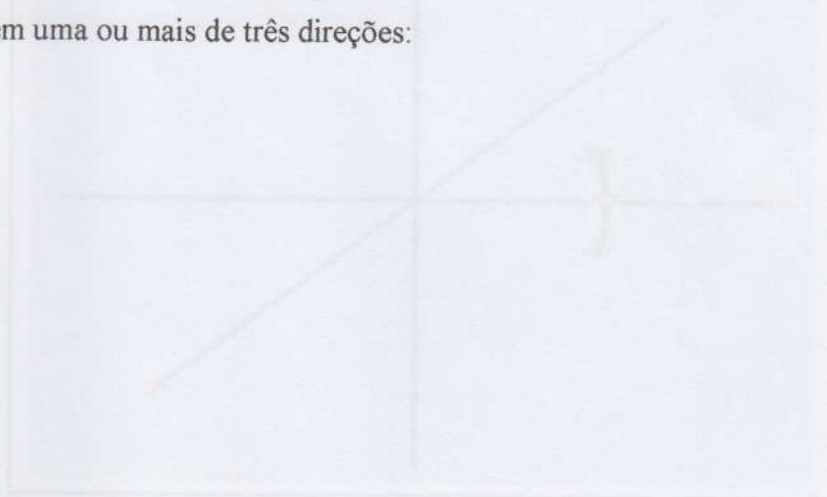


Figura 2.2.9.2 : Pitch, Rotação x;

- *Yaw*: rotação em torno do eixo Y;

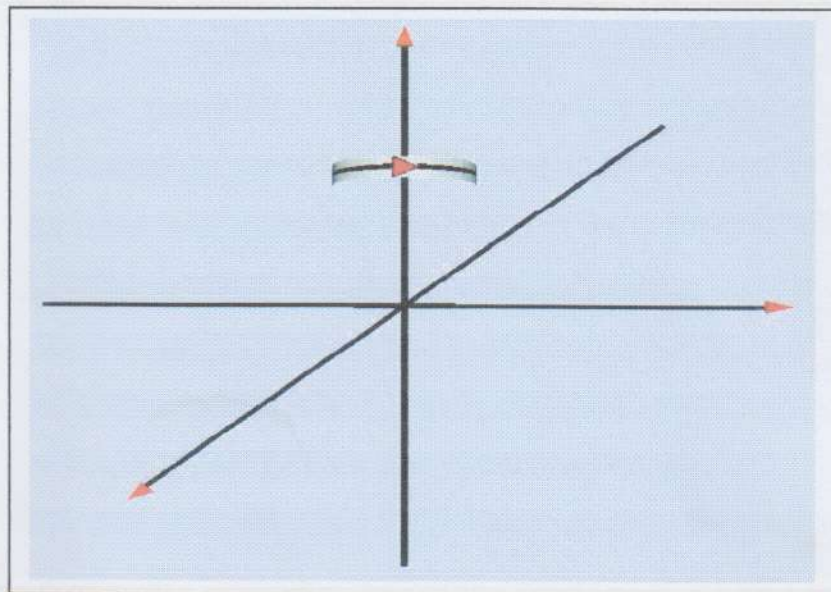


Figura 2.2.9.2: Yaw, Rotação Y

- *Pitch*: rotação em torno do eixo X;

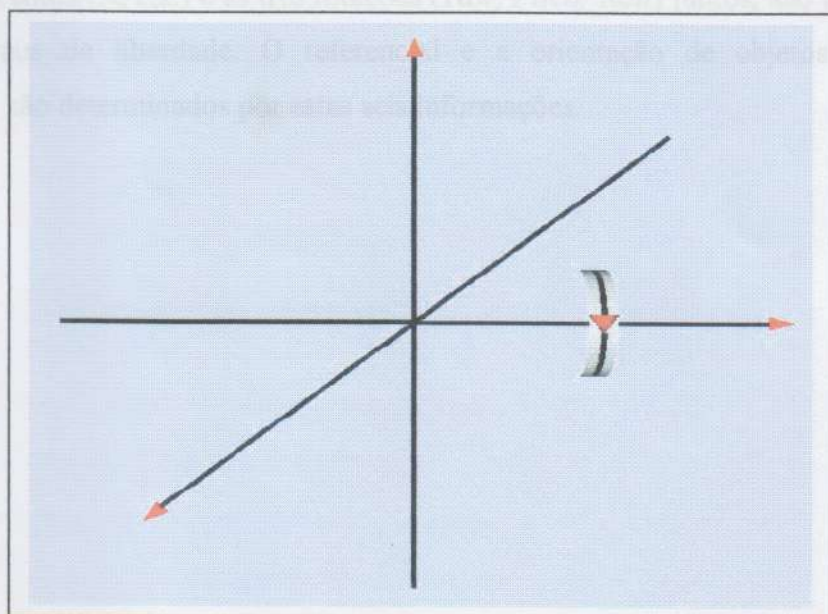


Figura 2.2.9.3 : Pitch, Rotação x;

- *Roll*: rotação em torno do eixo Z.

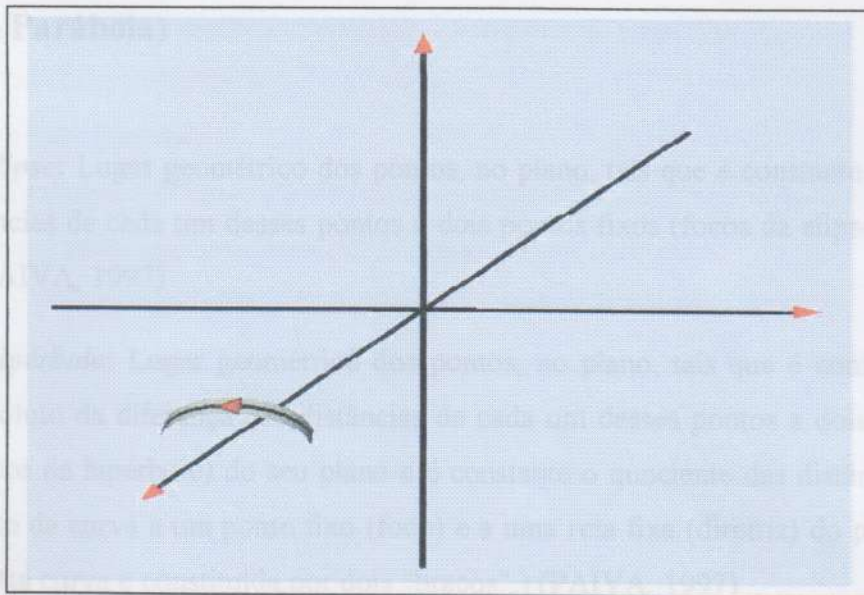


Figura 2.2.9.4 : Roll, Rotação z;

Os três eixos (X,Y,Z) e as três rotações (*Yaw*, *Pitch*, *Roll*) juntos, são referenciados como seis graus de liberdade. O referencial e a orientação de objetos em espaço tridimensional são determinados por estas seis informações.

2.2.10 – Conceitos de Figuras Geométricas (Circunferência, Elipse, Hipérbole, Parábola)

Elipse: Lugar geométrico dos pontos, no plano, tais que é constante a soma das distâncias de cada um desses pontos a dois pontos fixos (focos da elipse) desse plano. (PAIVA, 1997)

Hipérbole: Lugar geométrico dos pontos, no plano, tais que é constante o valor absoluto da diferença das distâncias de cada um desses pontos a dois pontos fixos (focos da hipérbole) do seu plano e é constante o quociente das distâncias de cada ponto da curva a um ponto fixo (foco) e a uma reta fixa (diretriz) do plano da curva. (Esta curva é constituída por dois "braços".) (PAIVA, 1997)

Parábola: Curva plana cujos pontos distam igualmente de um ponto fixo (foco) e de uma reta (diretriz), ambos situados no plano da curva. (PAIVA, 1997)

Circunferência: Curva plana fechada cujos pontos são eqüidistantes de um ponto fixo (centro) do seu plano. (PAIVA, 1997)



$$\begin{aligned} d_{CP} &= \sqrt{(X_p - X_C)^2 + (Y_p - Y_C)^2} \Rightarrow \sqrt{(x - a)^2 + (y - b)^2} = r \Rightarrow \\ &\Rightarrow (x - a)^2 + (y - b)^2 = r^2 \end{aligned}$$

Figura 2.2.11.1.2: exemplo da construção da circunferência

Portanto, $(x - a)^2 + (y - b)^2 = r^2$ é a equação padrão da circunferência e permite determinar os elementos essenciais para a construção da circunferência: as coordenadas do centro e o raio que pode ser vista na fig 2.2.11.1.2.

b) Equação reduzida

Quando o centro da circunferência estiver na origem $(C(0,0))$, teremos: $x^2 + y^2 = r^2$.

c) Equação geral Desenvolvendo a equação padrão, obtemos a equação geral da circunferência:

$$\begin{aligned} (x + a)^2 + (y - b)^2 &= r^2 \Rightarrow x^2 + 2ax + a^2 + y^2 - 2by + b^2 = r^2 \\ x^2 + y^2 + 2ax - 2by + a^2 + b^2 - r^2 &= 0 \end{aligned}$$

2.2.11– Estudo das Construções Geométricas

O estudo de Construções Geométricas baseia-se nas seguintes figuras :

2.2.11.1 - Circunferência

a) Equação padrão

Circunferência é o conjunto de todos os pontos de um plano equidistantes de um ponto fixo, desse mesmo plano, denominado centro da circunferência:

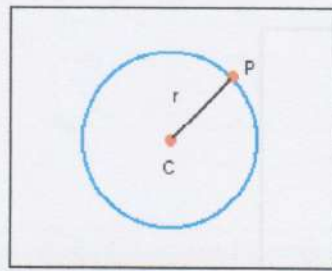
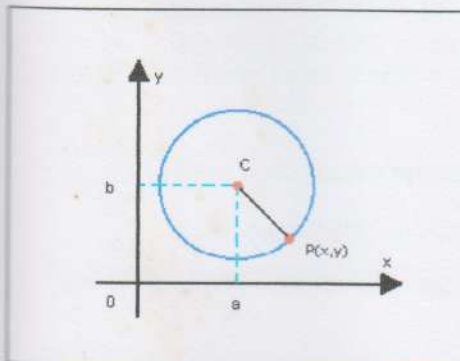


Figura 2.2.11.1.1: Exemplo da Circunferência

Assim, sendo $C(a, b)$ o centro e $P(x, y)$ um ponto qualquer da circunferência, a distância de C a $P(d_{CP})$ é o raio dessa circunferência. Então:



$$d_{CP} = \sqrt{(X_P - X_C)^2 + (Y_P - Y_C)^2} \Rightarrow \sqrt{(x - a)^2 + (y - b)^2} = r \Rightarrow \\ \Rightarrow (x - a)^2 + (y - b)^2 = r^2$$

Figura 2.2.11.1.2: exemplo da construção da circunferência

Portanto, $(x - a)^2 + (y - b)^2 = r^2$ é a equação padrão da circunferência e permite determinar os elementos essenciais para a construção da circunferência: as coordenadas do centro e o raio que pode ser vista na fig 2.2.11.1.2.

b) Equação reduzida

Quando o centro da circunferência estiver na origem ($C(0,0)$), teremos : $x^2 + y^2 = r^2$.

c) Equação geral Desenvolvendo a equação padrão, obtemos a equação geral da circunferência:

$$(x - a)^2 + (y - b)^2 = r^2 \Rightarrow x^2 - 2ax + a^2 + y^2 - 2by + b^2 = r^2 \\ \Rightarrow x^2 + y^2 - 2ax - 2by + a^2 + b^2 - r^2 = 0$$

2.2.11.2 - Elipse

Considerando, num plano, dois pontos distintos, F_1 e F_2 , e sendo $2a$ um número real maior que a distância entre F_1 e F_2 , chamamos de *elipse* o conjunto dos pontos do plano tais que a soma das distâncias desses pontos a F_1 e F_2 seja sempre igual a $2a$.

Por exemplo, sendo P, Q, R, S, F_1 e F_2 pontos de um mesmo plano e $F_1F_2 < 2a$, temos:

$$\left\{ \begin{array}{l} d_{F_1P} + d_{F_2P} = 2a \\ d_{F_1Q} + d_{F_2Q} = 2a \\ d_{F_1R} + d_{F_2R} = 2a \\ d_{F_1S} + d_{F_2S} = 2a \end{array} \right.$$

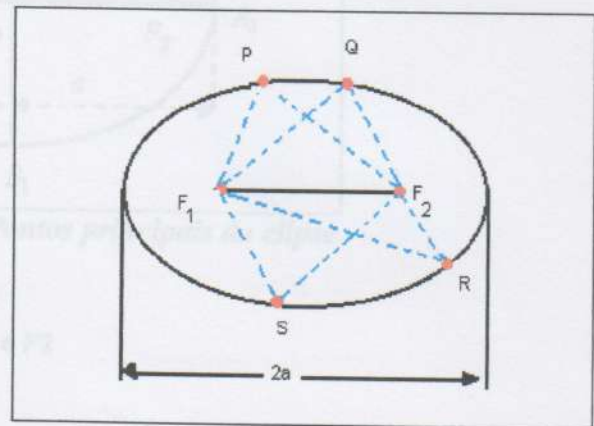


Figura 2.2.11.2.1: Elementos da elipse

Figura 2.2.11.2.2: Construção da elipse

A figura obtida é uma elipse.

Na figura acima, aplicando o Teorema de Pitágoras ao triângulo OF_1B_1 , retângulo em O , podemos escrever a seguinte relação fundamental:

c) Excentricidade

Chamamos de *excentricidade* o número real e tal que

Pela definição de elipse, $2c < 2a$, então $c < a$ e, consequentemente, $0 < e < 1$.
Observação: Quando os focos são muito próximos, ou seja, c é muito pequeno, a elipse se aproxima de uma circunferência.

a) Elementos

Observe a elipse a seguir. Nela, consideramos os seguintes elementos:

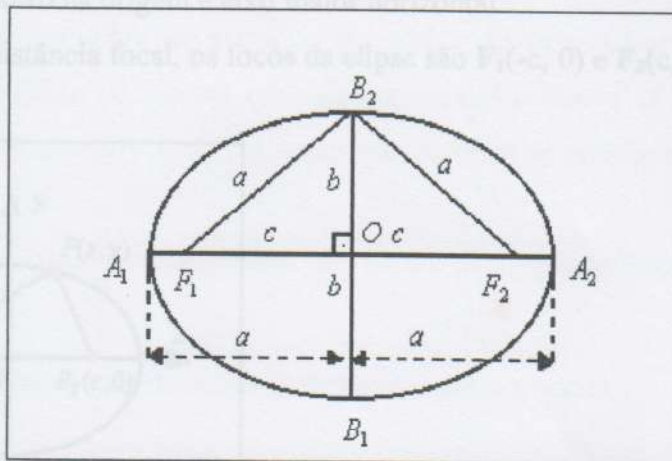


Figura 2.2.11.2.3: Os Pontos principais da elipse

- focos : os pontos F_1 e F_2
- centro: o ponto O , que é o ponto médio de F_1 e F_2
- semi-eixo maior: a
- semi-eixo menor: b
- semidistância focal: c
- vértices: os pontos A_1, A_2, B_1, B_2
- eixo maior: $|A_1A_2| = 2a$
- eixo menor: $|B_1B_2| = 2b$
- distância focal: $|F_1F_2| = 2c$

b) Relação fundamental

Na figura acima, aplicando o Teorema de Pitágoras ao triângulo OF_2B_2 , retângulo em O , podemos escrever a seguinte relação fundamental:

$$a^2 = b^2 + c^2$$

c) Excentricidade

Chamamos de *excentricidade* o número real e tal que:

$$e = \frac{c}{a}$$

Pela definição de elipse, $2c < 2a$, então $c < a$ e, conseqüentemente, $0 < e < 1$.

Observação: Quando os focos são muito próximos, ou seja, c é muito pequeno, a elipse se aproxima de uma circunferência.

d) Equações

Vamos considerar os seguintes casos:

a) elipse com centro na origem e eixo maior horizontal

Sendo c a semidistância focal, os focos da elipse são $F_1(-c, 0)$ e $F_2(c, 0)$:

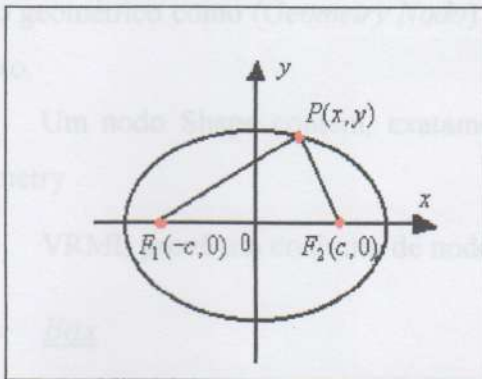


Figura 2.2.11.2.4: Elaboração da fórmula da elipse

Aplicando a definição de elipse

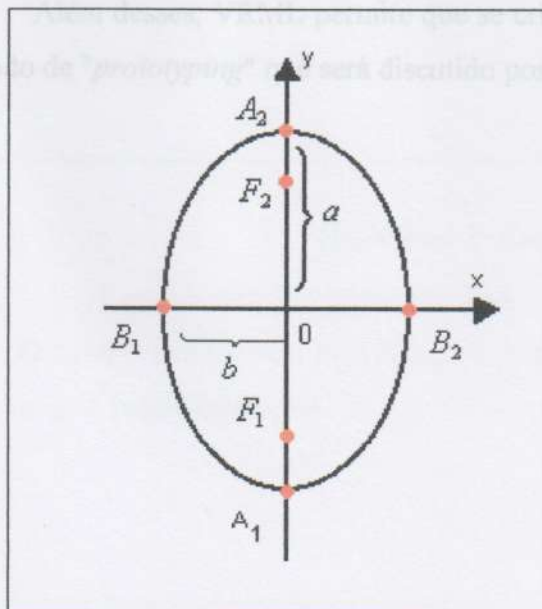
$$(d_{F_1P} + d_{F_2P} = 2a)$$

obtemos a equação da elipse:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

b) elipse com centro na origem e eixo maior vertical

Nessas condições, a equação da elipse é:



$$\frac{x^2}{b^2} + \frac{y^2}{a^2} = 1$$

2.2.12 — Alguns Exemplos de Figuras Geométricas em VRML

Como foi mostrado sobre os conceito de nodos do tipo Shape a cima vamos demonstrar como será exibido na tela no caso alguns exemplos aonde são associados um nodo geométrico como (*Geometry Nodo*) a nodos que definem as características visuais do objeto.

Um nodo Shape contem, exatamente, um único nodo geométrico em seu campo geometry .

VRML provê um conjunto de nodos geométricos como a seguir:

- Box
- Cone
- Cylinder
- ElevationGrid
- Extrusion
- IndexedFaceSet
- IndexedLineSet
- PointSet
- Sphere
- Text

Além desses, VRML permite que se criem outros nodos geométricos através do método de "prototyping" que será discutido posteriormente.

2.2.12.1 - Box

Um dos mais simples nodos geométricos é o box, que corresponde a um paralelepípedo centrado na origem (0,0,0) do sistema de coordenadas local e alinhado com os eixos cartesianos, tal qual indica a Fig.2.2.12.1.1.



Figura 2.2.12.1.1: Exemplo de um cubo Branco

Os valores default do nodo Box correspondem a um cubo de dimensões 2.5 x 2.5 x 2.5.

```
1  #VRML V2.0 utf8
2
3  # Desenho de um cubo Branco
4
5  Shape {
6      appearance Appearance {
7          material Material {
8              diffuseColor 3.0 3.0 3.0
9          }
10     }
11     geometry Box {
12         size 2.5 2.5 2.5
13     }
14 }
```

Algoritmo 2: Construção do Cubo

O campo *size* contém três números, indicando o comprimento do box ao longo dos eixos x, y, z respectivamente.

2.2.12.2 - Cilindro

Um cilindro é um sólido gerado pela rotação completa de um ângulo em torno de um de seus lados. O cilindro também é chamado de cilindro circular reto. Através do VRML daremos uma explicação melhor sobre a construção desse sólido de acordo com a fig. 2.2.12.2.

```
1 #VRML V2.0 utf8
2   # Desenho de um cilindro cinza
3   Shape {
4     appearance Appearance {
5       material Material {
6         diffuseColor 2.0 2.0 2.0
7       }
8     }
9     geometry Cylinder {
10      radius 2.0
11      height 5.0
12      side TRUE
13      top TRUE
14      bottom TRUE
15    }
16  }
```

Algoritmo 3: Construção do Cilindro

O cilindro é composto por três partes: lateral (*side*), a tampa (*top*) e o fundo (*bottom*). A cada uma dessas partes está associada uma variável do tipo *SPBool* (tipo pré-definido pela linguagem) que pode ser *TRUE*, indicando existência, ou *FALSE*, caso não exista.

Destaca-se, ainda, que o nodo Cilindro só apresenta faces externas, ou seja, quando visto de dentro para fora, os resultados são imprevisíveis.

2.2.12.3 Este nodo define um cilindro centrado na origem (0,0,0) do sistema de coordenadas cartesianas e com o eixo central ao longo do eixo y, como pode ser visto na Figura 2.2.12.2. Seus valores default correspondem a um cilindro de tamanho 5 nas três dimensões.



Figura 2.2.12.2: Exemplo de um cilindro cinza

O atributo *radius* refere-se ao raio do cilindro e *height* à sua altura ao longo do eixo central. Tanto *radius* quanto *height* devem ter valores maiores que 0.0.

O cilindro é composto por três partes: lateral (*side*), a tampa (*top*) e o fundo (*bottom*). A cada uma dessas partes está associada uma variável do tipo SFBool (tipo prédefinido pela linguagem) que pode ser TRUE, indicando existência, ou FALSE, caso não exista.

Destaca-se, ainda, que o nodo Cilindro só apresenta faces externas, ou seja, quando visto de dentro para fora, os resultados são imprevisíveis.

2.2.12.3 – Cone

Cone é um sólido gerado pela rotação completa de um triângulo retângulo em torno de seus catetos. Um cone também é chamado de cone circular reto. Através do VRML daremos uma explicação melhor sobre a construção desse sólido de acordo com a Fig. 2.2.12.3.1.

```

1      #VRML V2.0 utf8
2
3      # Desenho de um cone cinza
4
5      Shape {
6
7          appearance Appearance {
8
9              material Material {
10
11                  diffuseColor 2.0 2.0 2.0
12              }
13      }
14
15      geometry Cone {
16
17          bottomRadius 1.0
18
19          height      5.0
20
21          side        TRUE
22
23          bottom      TRUE
24      }
25  }
```

Algoritmo 4: Construção do Cone

Este nodo especifica um cone centrado na origem do sistema local de coordenadas e cujo eixo central está alinhado com o eixo y, tal qual mostra a Figura 2.2.12.3.1.

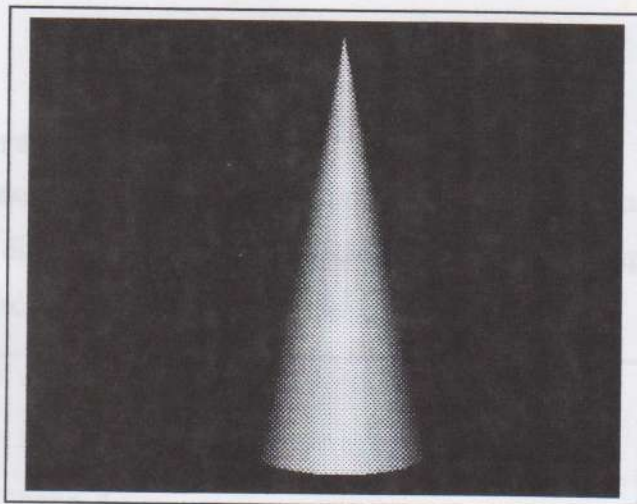


Figura 2.2.12.3.1: Exemplo de um cone Cinza

O campo **bottomRadius** especifica o raio da base do cone, **height** a altura do cone, desde o centro da base até o ápice e **bottom** se a base do cone é visível (TRUE) ou não (FALSE), enquanto **side** especifica se a lateral do cone é visível (TRUE) ou invisível (FALSE).

Algoritmo 5: Construção de Esfera

Este nó especifica uma esfera centrada na origem (0,0,0) do sistema local de coordenadas.

O campo **radius** define o raio da esfera e, por default, seu valor é 4.0, sendo que deve ser superior a 0.0.

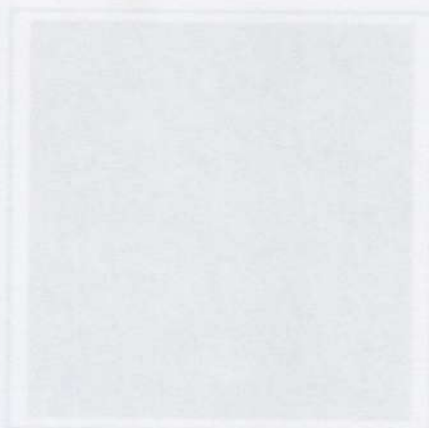


Figura 2.2.12.4.1: Exemplo de uma esfera Verde

2.2.12.4 - Esfera

É um sólido gerado pela rotação completa de um semicírculo em torno de um eixo que contém seu diâmetro. Através do VRML daremos uma explicação melhor sobre a construção desse sólido de acordo com a fig. 2.2.12.4.1.

```
1 #VRML V2.0 utf8
2
3 # Desenho de uma esfera verde
4
5 Shape {
6
7     appearance Appearance {
8         material Material {
9             diffuseColor 0.0 0.8 0.0
10
11         }
12     }
13
14     geometry Sphere {
15
16         radius 4.0
17
18     }
19 }
```

Algoritmo 5: Construção da Esfera

Este nodo especifica uma esfera centrada na origem (0,0,0) do sistema local de coordenadas.

O campo **radius** define o raio da esfera e, por default, seu valor é 4.0, sendo que deve ser superior a 0.0.

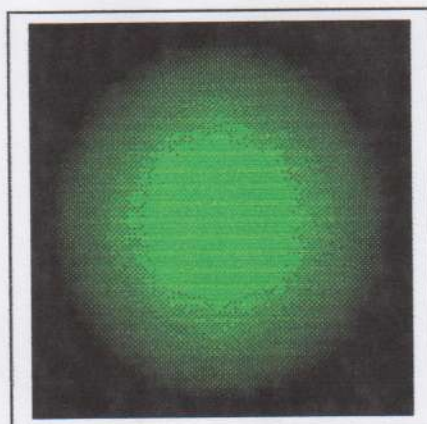


Figura 2.2.12.4.1: Exemplo de uma esfera Verde

CAPÍTULO 3

Implementação

3.1- Introdução

Na implementação, procurei utilizar vários conceitos e técnicas vistos durante as pesquisas feitas nesse período de elaboração dessa Monografia para construir a cena em Realidade Virtual como (Shape, Appearance, Material, Geometry, Transform, Group, Separator e Viewpoint) e animá-la (TimeSensor, PositionInterpolator, OrientationInterpolator, Script, PlaneSensor e TouchSensor).

As cenas e objetos em VRML são formados por cubos, esferas, cilindros, cones e superfícies poligonais. Na especificação de um objeto procurei criar em um arquivo texto, sem formatação, uma descrição de como este objeto deve ser exibido. Nesta descrição aparecem itens como suas dimensões, sua cor e sua orientação espacial.

Para ilustrar, o exemplo a seguir descreve um cubo que gira em torno do seu próprio eixo quando o usuário move o cursor sobre ele. Neste caso, o *eventOut isOver* do nó *TouchSensor* é enviado para o nó *TimeSensor*. O nó *TimeSensor* é enviado para o nó *OrientationInterpolator* que direciona a rota para o nó *Transform*. Quando o cursor move-se sobre o cubo. O nó *TouchSensor* envia *TRUE* usando o *eventOut isOver* que permite que o nó *TimeSensor* direcione o nó *OrientationInterpolator* e gire o cubo. Quando o cursor sai de cima do cubo, *FALSE* é enviado usando o *eventOut isOver* do nó *TouchSensor*, desabilitando o nó *TimeSensor* e parando a animação.

```
23         key [0.0, 0.50, 1.0]
24         keyValue [
25             0.01000000,
26             0.01000314,
27             0.01000628
28         ]
29     }
30 }
31 )
```

```

1 #VRML V2.0 utf8
2 Group{
3   children [
4     # Cubo que gira quando o usuário clica sobre ele
5     DEF Cube Transform {
6       children Shape {
7         appearance Appearance {
8           material Material { }
9         }
10        geometry Box { }
11      }
12    },
13    # Sensor, detecta quando o usuário clica em um objeto
14    DEF Touch TouchSensor { },
15    # relógio para controlar a animação
16    DEF Clock TimeSensor {
17      enabled FALSE
18      cycleInterval 4.0
19      loop TRUE
20    },
21    # Caminho da animação (movimento)
22    DEF CubePath OrientationInterpolator {
23      key [0.0, 0.50, 1.0]
24      keyValue [
25        0.0 1.0 0.0 0.0,
26        0.0 1.0 0.0 3.14,
27        0.0 1.0 0.0 6.28
28      ]
29    }
30  ]
31 }

```

32 ROUTE Touch.isOver	TO Clock.set_enabled
33 ROUTE Clock.fraction_changed	TO CubePath.set_fraction
34 ROUTE CubePath.value_changed	TO Cube.set_rotation

Algoritmo 6: Construção e Animação do Cubo

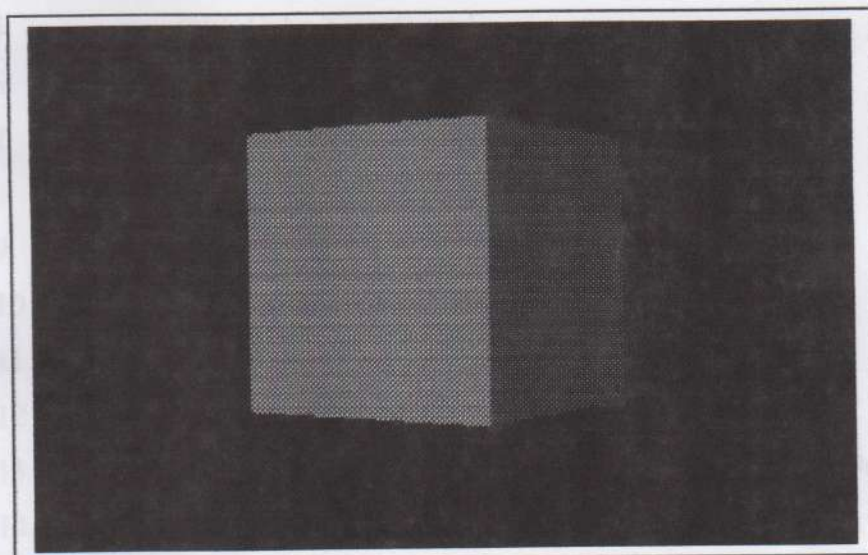


Figura 3.1.1: Exemplo 1

Neste outro caso, o *eventOut isOver* do nó *TouchSensor* é enviado para o nó *TimeSensor*. O nó *TimeSensor* é enviado para o nó *OrientationInterpolator* que direciona a rota para o nó *Transform*. O nó *TouchSensor* envia *TRUE* usando o *eventOut isOver* que permite que o nó *TimeSensor* direcione o nó *OrientationInterpolator* e gire o cubo automaticamente.

```

1 #VRML V2.0 utf8
2 Group{
3   children [
4     # Cubo que gira quando o usuário clica sobre ele
5     DEF Cube Transform {
6       children Shape {
7         appearance Appearance {
8           material Material { }
9         }
10        geometry Box { }
11      }
12    },
13    # Sensor, detecta quando o usuário clica em um objeto
14    DEF Touch TouchSensor { },
15    # relógio para controlar a animação
16    DEF Clock TimeSensor {
17      enabled TRUE
18      cycleInterval 4.0
19      loop TRUE
20    },
21    # Caminho da animação (movimento)
22    DEF CubePath OrientationInterpolator {
23      key [0.0, 0.50, 1.0]

```

```

24 keyValue [
25     0.0 1.0 0.0 0.0,
26     0.0 1.0 0.0 3.14,
27     0.0 1.0 0.0 6.28
28 ]
29 }
30 ]
31 }
32 # Rotas
33 ROUTE Touch.isOver TO Clock.set_enabled
34 ROUTE Clock.fraction_changed TO CubePath.set_fraction
35 ROUTE CubePath.value_changed TO Cube.set_rotation

```

Algoritmo 7: Construção e Animação do cubo

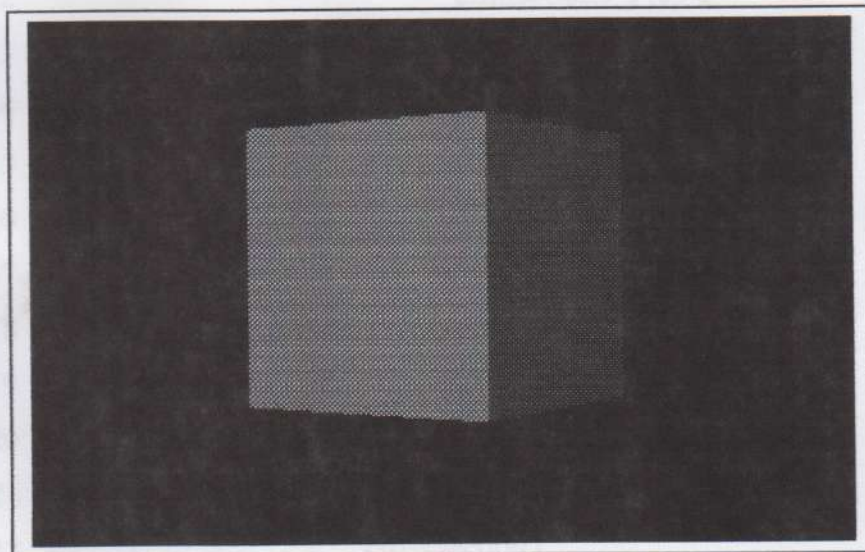


Figura 3.1.2: Exemplo 2

Baseados em pontos calculados e através do método Extrusion essa próxima construção dará uma melhor compreensão como uma circunferência é construída no seu plano e na tela:

```

1 #VRML V2.0 utf8
2 Background {
3     skyColor 0.7 0.7 0.7
4 }
5 DEF Tra1 Transform {
6     rotation 1 0 0 1.57
7     # scale 0.5 0.5 0
8     children [
9         Shape {
10             appearance Appearance {
11                 material Material {
12                     diffuseColor 1 .85 0
13                     specularColor .87 .25 .25
14                     ambientIntensity .157
15                     shininess 1
16                 }
17             }
18             geometry Extrusion {
19                 # rotation 1 0 0
20                 creaseAngle 1.57
21                 beginCap TRUE
22                 endCap TRUE
23                 crossSection [
24                     # Circle = Cálculo para Construção do Circulo
25                     -1.00 0.00
26                     -0.98 0.20
27                     -0.96 0.28
28                     -0.94 0.34
29                     -0.92 0.39 ....

```

```

29 ]
30 spine [
31 # Circle
32 -2.00 0.00 0.00
33 -1.96 0.00 0.40
34 -1.92 0.00 0.56
35 -1.88 0.00 0.68
36 -1.84 0.00 0.78
37 -1.80 0.00 0.87
38 -1.60 0.00 1.20
39 -1.40 0.00 1.43
40 -1.20 0.00 1.60 .....
41 ]
42 scale [
43 0.03 0.03 #, 1.95 1.95,
44 ]
45 }
46 children }
47 ]
48 }
49 Transform {
50 # rotation 1 0 0 1.57
51 children USE Tra1
52 }
53 DEF No1 Transform {
54 translation 0 1 0
55 # rotation 1 0 0 1.57
56 children [
57 Shape {
58 appearance Appearance {

```

```

59 material Material {
60     diffuseColor .541 .0902 .251
61     specularColor .81 .77 .75
62     ambientIntensity .487
63     shininess .83
64     transparency 0.5
65 }
66 }
67 geometry Cylinder {
68     radius 0.03
69     height 2
70 }
71 }
72 ]
73 }
74 DEF Tra2 Transform {
75     rotation 1 0 0 1.57
76     children [
77         Shape {
78             appearance Appearance {
79                 material Material {
80                     diffuseColor 1 0 0
81                     transparency 0.1
82                 }
83             }
84             geometry Sphere {
85                 radius 0.1
86             }
87         }
88     ]
89 }

```

```

90 DEF No2 Transform {
91   children [
92     DEF No4 Transform {
93       translation 0 2 0
94       # scale 3 3 3
95       # rotation 1 0 0 1.57
96       children USE Tra2
97     }
98     Transform {
99       children USE No1
100    }
101  ]
102 }

103 DEF TempoEsfera1 TimeSensor {
104   cycleInterval 4
105   loop TRUE
106 }
107 DEF OriEsfera1 OrientationInterpolator {
108   key [ 0 0.5 1 ]
109   keyValue [ 0 0 1 0,
110             0 0 1 3.14, # rotate right
111             0 0 1 6.28] # rotate down
112 }
113 ROUTE TempoEsfera1.fraction_changed TO OriEsfera1.set_fraction
114 ROUTE OriEsfera1.value_changed TO No2.set_rotation

```

Algoritmo 8: Construção da Circunferência

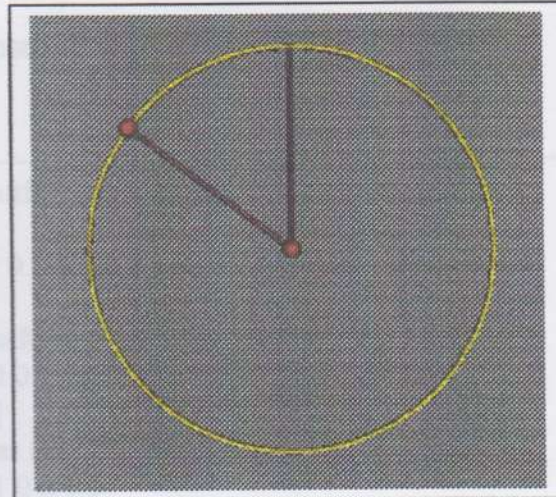


Figura 3.1.3: Exemplo 2

```

1 #VRML V2.0
2 Background [
3   skyColor
4 ]
5 DEF No3 Trans
6   rotating C
7   children [
8     Transform
9     rotation [
10      children [
11        Shape (
12          appearance Appearance (
13            material Material (
14              diffuseColor 1 .85 0
15              specularColor .87 2. 25
16              ambientIntensity .157
17              shininess [
18                ]
19            )
20          )
21          geometry Extrusion (
22            crossAngle 1.57
23            beginCap TRUE
24            endCap TRUE
25            crossSection [
26              # Circle
27              -2.00 0.00
28              -1.60 1.20
29              -1.20 1.60
30              -0.80 1.83
31              0.00 2.00
32              0.40 1.96
33              0.80 1.83
34              1.20 1.60
35              1.60 1.20
36              2.00 0.00
37              1.60 -1.20
38              1.20 -1.60
39              0.80 -1.83
40              0.40 -1.96
41              0.00 -2.00
42              -0.40 -1.96
43              -0.80 -1.83

```

Construção da Elipse:

```

1 #VRML V2.0 utf8
2 Background {
3     skyColor 0.7 0.7 0.7
4 }
5 DEF No3 Transform {
6     rotation 0 0 1 1.57
7     children [
8         Transform {
9             rotation 1 0 0 1.57
10            children [
11                Shape {
12                    appearance Appearance {
13                        material Material {
14                            diffuseColor 1 .85 0
15                            specularColor .87 .25 .25
16                            ambientIntensity .157
17                            shininess 1
18                        }
19                    }
20                    geometry Extrusion {
21                        creaseAngle 1.57
22                        beginCap TRUE
23                        endCap TRUE
24                        crossSection [
25                            # Circle
26                                -2.00    0.00
27                                -1.60    1.20
28                                -1.20    1.60
29                                -0.80    1.83
30                                -0.40    1.96
31                                0.00     2.00
32                                0.40    1.96
33                                0.80    1.83
34                                1.20    1.60
35                                1.60    1.20
36                                2.00    0.00
37                                1.60   -1.20
38                                1.20   -1.60
39                                0.80   -1.83
40                                0.40   -1.96
41                                0.00   -2.00
42                                -0.40  -1.96
43                                -0.80  -1.83.....

```

```

44 scale [
45     0.02 0.02 #. 1.95 1.95.
46 ]
47 children [
48     ]
49 ]
50 }
51
52 DEF No1 Transform {
53     children [
54         Shape {
55             appearance Appearance {
56                 material Material {
57                     diffuseColor 1 .85 0
58                     specularColor .87 .25 .25
59                     ambientIntensity .157
60                     shininess 1
61                 }
62             }
63             geometry Sphere {
64                 radius 0.002
65             }
66         ]
67     ]
68 }
69
70 Transform {
71     translation 0 1 0
72     scale 50 50 50
73     children USE No1
74 }
75 Transform {
76     translation 0 -1 0
77     scale 50 50 50
78     children USE No1
79 }
80 Transform {
81     translation 0.60 2.88 0
82     scale 50 50 50
83     children USE No1
84 }
85 Transform {
86     translation -1.63 -1.65 0.00
87     scale 50 50 50
88     children USE No1
89 }

```

```

90 Transform {
91 #   rotation 1 0 0 1.57
92 children [
93     Shape {
94         appearance Appearance {
95             material Material {
96                 diffuseColor 0 0 1.0
97                 specularColor .87 .25 .25
98                 ambientIntensity .157
99                 shininess 1
100             }
101         }
102         geometry Extrusion {
103             creaseAngle 1.57
104             beginCap TRUE
105             endCap TRUE
106             crossSection [
107                 # Circle
108                 -2.00    0.00
109                 -1.60    1.20
110                 -1.20    1.60
111                 -0.80    1.83
112                 -0.40    1.96
113                 0.00     2.00
114                 0.40     1.96....

```

Algoritmo 9: Construção da Elipse

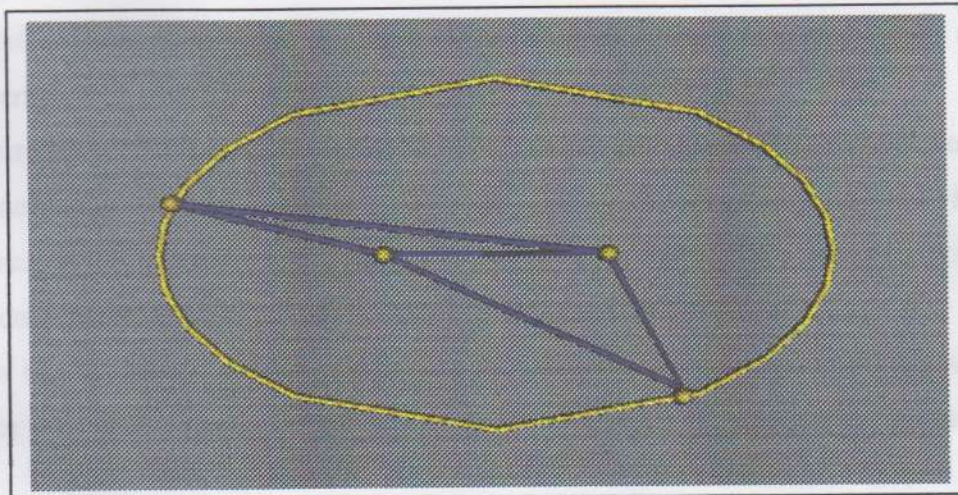


Figura 3.1.4: Exemplo 3

CAPÍTULO 4

4.1 - CONTRIBUIÇÕES FINAIS

Este estudo teve por objetivo demonstrar o VRML na área da educação principalmente na área da Matemática, bem como seus conceitos, características, funcionalidade entre outros citados acima.

O VRML juntamente com a Realidade Virtual encontram-se em um crescimento contínuo e exponencial, o qual faz refletir que a linguagem está apenas no início de sua longa caminhada na Educação.

Pelos estudos citados neste artigo nota-se que esta linguagem foi um potencial explorado, pois, pode ser aplicado nas mais diversas áreas da educação (Literatura, Medicina, Física, Química, Matemática, Arquitetura, etc.), devido a sua grande flexibilidade de se adaptar a diversos estilos de aprendizagem.

Foi notado que cada pessoa possui um estilo de aprendizagem que melhor lhe satisfaz, algumas de maneiras visuais, pois só aprendem se olharem para o objeto em estudo, outras verbais, tem que ouvir sobre o assunto em estudo até que este fique gravado, seja qual for a maneira utilizada, pode se adaptar ao VRML. Para o aprendizado das pessoas com problemas no entendimento de equações, teorias, e princípios, o VRML pode ser usado para materializar estas informações utilizando o VRML como uma ferramenta de grande auxílio na aprendizagem dos alunos, tornando uma aula diferente e ao mesmo tempo divertida e criativa, fazendo com que a matéria seja muito fácil.

O uso de VRML no estudo realizado foi e sempre será na educação basicamente um processo de exploração, de descoberta, de observação e de construção de uma visão do conhecimento a partir destas ações. Segundo Byrne "Este uso ainda é recente. Existe um consenso de que a Realidade Virtual pode ajudar no processo de ensino".

Por isso é que nós queremos utilizar a tecnologia como ferramenta para auxiliar os professores a atingir o objetivo que é atenção dos alunos e o interesse do próprio pela matéria.

CAPÍTULO 5

Conclusão

Este capítulo finaliza o trabalho apresentando conclusões gerais sobre a Estudo de Construções Geométricas e principalmente sobre a de imagens e o processo de amostragem de objetos. Também é apresentado novas propostas e extensões às técnicas discutidas e implementadas neste trabalho.

5.1 Conclusões

As técnicas apresentadas nesta monografia demonstram a evolução da VRML , e que este modelo alcançou um nível de maturidade que permitiu chegar a bons resultados. Técnicas apresentadas na construção das figuras geométricas descrito ao longo deste texto com o uso de Nodos, Nodo de Geometria, Eventos, Nodos filhos. Estes método caracteriza-se em representar modelos de objetos utilizando pontos como primitiva, ao invés de triângulos ou superfícies paramétricas como é feito tradicionalmente.

Duas etapas se destacam neste método: a amostragem de pontos em um objeto e a reconstrução da imagem. Vários aspectos peculiares a cada uma dessas etapas são discutidos e dois problemas relevantes são abordados de forma mais direta como contribuição deste trabalho.

Tendo como objetivo gerar imagens de melhor qualidade, uma melhor representação da superfície do objeto leva a reconstrução de uma imagem com menos falhas.

Diversos efeitos podem ser mais facilmente formulados e criados utilizando a tecnologia como VRML para representação de superfícies, e algumas entidades principalmente na área da Matemática voltado para o estudo da geometria aonde essas figuras já possuem característica de natureza no caso da hipérbole, circunferência, elipse, parábola.

Existem muitos problemas a serem resolvidos, e também a necessidade de adequar as representações de forma que possam ser utilizadas mais facilmente. Armazenar um objeto representado por estas técnicas citadas a cima pode exigir até dezenas de vezes mais memória do que representar este mesmo objeto com polígonos, o que tornaria inviável a manipulação de vários objetos simultaneamente. Porém, o desenvolvimento de técnicas que

permitam a utilização de *hardware* específico para processamento gráfico na etapa de demonstração dessas figuras pode aumentar bastante aprendizagem por parte dos alunos. Da mesma forma, uma grande vantagem do método é permitir que a construção das figuras seja trabalhada com um alto nível de detalhe, permitindo efeitos aplicados com qualidade. Com uma representação através do VRML, as mesmas operações podem ser realizadas sem a necessidade de tal equipamento sem um grande aumento do custo computacional.

Todos estes aspectos nos mostram o grande potencial da técnica tanto como alternativa quanto para complemento na geração dessas figuras. Com a evolução do modelo e dos procedimentos com essa tecnologia pode tornar-se mais uma ferramenta importante na área de educação em várias áreas não só em matemática mas em outras como física, química e várias outras.

4) FRENCH, Thomas. *Desenho Técnico*. Porto Alegre: Editora Globo, 1979;

5) IEZZI, Gelson; DOLCE, Osvaldo. *Geometria Analítica*. São Paulo: Editora Moderna Ltda, 1972;

6) OLIVEIRA, S. L. de. *Tratado de Metodologia Científica: Projetos de Pesquisas, TGI, TCC, monografias, dissertações e teses*. São Paulo: Pinus, 1997;

7) PAIVA, Manoel Rodrigues. *Matemática - Volume 2*, São Paulo, Moderna LTDA, 1997.

8) RANGEL, Aleyr Pinheiro, *Curvas*. Rio de Janeiro: UFRJ, 1974;

CAPÍTULO 6

Referências Bibliográficas

- 1) AMES, Andrea L.; NADEAU, David R.; MORELAND, John L. **VRML 2.0 Sourcebook**. 2nd ed. New York: John Wiley, 1997.
- 2) AZEVEDO, Eduardo; CONCI, Aura. **Computação Gráfica – Teoria e Prática**, São Paulo, Campus, 2003;
- 3) CARVALHO, Benjamin de. **Desenho Geométrico**, Rio de Janeiro, Ao livro Técnico, 1998 ;
- 4) FRENCH, Thomas. **Desenho Técnico**, Porto alegre: Editora Globo, 1979;
- 5) IEZZI, Gelson; DOLCE, Osvaldo. **Geometria Analítica**. São Paulo : Editora Moderna Ltda, 1972;
- 6) OLIVEIRA, S. L. de. **Tratado de Metodologia Científica: Projetos de Pesquisas, TGI, TCC, monografias, dissertações e teses**. São Paulo: Pioneira, 1997;
- 7) PAIVA, Manoel Rodrigues. **Matemática – Volume 2**, São Paulo, Moderna LTDA, 1997.
- 8) RANGEL, Alcyr Pinheiro, **Curvas**. Rio de Janeiro: UFRJ, 1974;