VALMIRÉ JOÃO RIBEIRO JUNIOR

TÉCNICAS DE OTIMIZAÇÃO DE AMBIENTES VIRTUAIS EXTENSOS E SUAS APLICAÇÕES

Joinville, Novembro/2004

UNIVERSIDADE DO ESTADO DE SANTA CATARINA

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

VALMIRÉ JOÃO RIBEIRO JUNIOR

TÉCNICAS DE OTIMIZAÇÃO DE AMBIENTES VIRTUAIS EXTENSOS E SUAS APLICAÇÕES

Trabalho de conclusão de curso submetido à Universidade Estadual de Santa Catarina como parte dos requisitos para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador:

Marcelo da Silva Hounsell, PhD

Joinville, Novembro/2004

TÉCNICAS DE OTIMIZAÇÃO DE AMBIENTES VIRTUAIS EXTENSOS E SUAS APLICAÇÕES.

VALMIRÉ JOÃO RIBEIRO JUNIOR

Este Trabalho de Conclusão de Curso foi julgado adequado para a obtenção do título de Bacharel em Ciência da Computação Área de Concentração em Sistemas de Computação e Sistemas de Conhecimento aprovada em sua forma final pelo Curso de Ciência da Computação Noturno do CCT/UDESC.

Cinara T. Menegazzo, MSc.

Banca Examinadora:

Marcelo da Silva Hounsell, PhD.

Rogério Eduardo da Silva, MSc.

Siovani Cintra Felipussi, DSc.

Primeiramente, a Deus, o Todo-Poderoso, pela vida e sabedoria. Aos meus pais, Valmiré e Marli, pelo amor, carinho e confiança.

Obrigado a todos que de alguma forma colaboraram na realização deste trabalho. Em especial, ao Rubens Redel por sua colaboração e ao Profº Marcelo pela orientação dispensada.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

SIGLA	Descrição
2D	Duas dimensões (X, Y)
3D	Três Dimensões (X, Y, Z)
ASCII	American Standard Code for Information Interchange
AV	Ambiente Virtual
BBOX	Bounding Box
CAD	Computer Aided Design (Projeto Auxiliado por Computador)
CAVE	Cave Automatic Virtual Environment
CCT	Centro de Ciências Tecnológicas
FEJ	Faculdade de Engenharia de Joinville
GIF	Graphics Interchange Format
HMD	Head Mounted Display (Vídeo-Capacete)
ISSO	International Standardization Organization
JPEG	Joint Photographic Experts Group
LARVA	Laboratório de Realidade Virtual Aplicada
LCD	Liquid Crystal Display (Visor de Cristal Líquido)
LOD	Level Of Detail (Nível de Detalhe)
LRV	Laboratório de Realidade Virtual
NCA	Núcleo de Computação Gráfica Aplicada
PNG	Portable Network Graphics
RV	Realidade Virtual
RVI	Realidade Virtual Imersiva
RVNI	Realidade Virtual Não-Imersiva
TCC	Trabalho de Conclusão de Curso
UDESC	Universidade do Estado de Santa Catarina
UFRGS	Universidade Federal do Rio Grande do Sul
UFSC	Universidade Federal de Santa Catarina
URL	Uniform Resourse Locater
USP	Universidade de São Paulo
VRML	Virtual Reality Modeling Language

WRL	Word Reality Language

RESUMO

O Campus Virtual da UDESC - Joinville foi implementado em trabalho anterior (SCHULZ, 2004) sem preocupações com performance somente com o realismo.

O realismo é um dos fatores determinantes em sistemas de Realidade Virtual, no entanto, quanto maior o nível de realismo, maiores serão as necessidades de processamento.

Uma dificuldade para a modelagem de ambientes de exteriores extensos a serem veiculados via *web*, diferentes de outros tipos de ambientes, é a limitação na configuração do usuário em termos de *hardware* e conexão via *Internet*. Deve-se supor uma baixa configuração de *hardware*, para que a visitação obtenha uma boa performance quando visualizada.

Com o objetivo de reduzir os custos de processamento permitindo que o Campus Virtual seja melhor visualizável em máquinas simples, estudar-se-ão e implementar-se-ão técnicas de otimização de ambientes virtuais extensos.

ABSTRACT

The Virtual Campus of the UDESC - Joinville was implamented in a previous work (SCHULZ, 2004) without concern about performance but with realism, as it can be seen on the figures below.

Realism is one of the basic issues in Virtual Reality Systems. However, the higher the realism level, the higher the demands for processing.

Some of the difficulties for the ambient modeling of extensive exteriors to be viewed on the web, considering this case different than the other ambients, types, are the limitations imposed by the hardware and type of internet connection of the common user.It should be assumed that the common user will have a hardware set-up low grade, in order to provide a good performance for the viewing when a visit is made.

With the goal of reducing costs of processing in order to allow the Virtual Campus to be better viewed in simpler computers, it will be studied and implamented optimizations techniques for extensive virtual ambients.

1 INTRODUÇÃO

Em termos computacionais, a realidade virtual é considerada a forma mais avançada de interface com o computador. Permite ao usuário obter a imersão, navegação e interação em um ambiente sintético tridimensional gerado por computador, utilizando canais multi-sensoriais (VINCE, 1998).

Os sistemas de realidade virtual são divididos em dois tipos principais: realidade virtual imersiva e realidade virtual não-imersiva. Nestes tipos de sistemas, se faz necessários equipamentos especiais de entrada e saída e potentes para produzir ambientes velozes e com gráficos com alta qualidade, a fim de para garantir, ao usuário, a sensação de imersão no ambiente. Por outro lado, a realidade virtual não-imersiva não necessita de *hardware* específico, pois os usuários podem utilizar simplesmente monitores, *CPUs* e *mouses* convencionais para acessar o mundo sintético. Deste modo, este mundo não demandará muita performance computacional (REDEL, 2004).

Pode-se citar como uma dificuldade para a veiculação na *WEB* de ambientes extensos a “configuração” do usuário em termos de *hardware* e conexão para a *internet*. O desenvolvedor não tem como saber ao certo a configuração que o usuário de seu sistema de realidade virtual disporá, por isso, deve-se supor uma baixa configuração de *hardware* e otimizar seu sistema de forma que tenha performance aceitável quando executado.

O trabalho de conclusão de curso “Campus Virtual” do 1º semestre de 2004 (SCHULZ, 2004) abordou e implementou aspectos visuais do Campus da Universidade do Estado de Santa Catarina – UDESC Joinville, não havendo uma preocupação excessiva em termos de performance. No entanto, o alto nível de realismo, gerou grande necessidade de processamento. Portanto, a fim de reduzir os custos de processamento na máquina do usuário, permitindo que o campus virtual seja visualizável também em máquinas mais simples, estudar-se-ão e implementar-se-ão técnicas de otimização de ambientes virtuais extensos.

Não se pode esquecer de citar como justificativa para este trabalho de conclusão de curso, o fato de não se conhecer ainda uma literatura, um

conjunto condensado de informações referentes às técnicas de desenvolvimento em realidade virtual para ambientes extensos. Estudar e entender as soluções de otimização para este tipo de ambiente ajudará a entender suas diferenças com outros tipos de ambientes, como os de interiores limitados.

1.1. OBJETIVOS

A seguir serão apresentados o objetivo geral e os específicos a serem alcançados neste trabalho.

1.1.1. Geral

Investigar características específicas dos ambientes virtuais extensos com ênfase nas estratégias de otimização da modelagem para veiculação na *web*.

1.1.2. Específicos

- Caracterizar comparativamente a outros tipos de ambientes, por exemplo, de interiores limitados.
- Levantar e caracterizar exemplos de diferentes sistemas, discutir a aplicabilidade e identificar as características funcionais no desenvolvimento deste tipo de sistema.
- Experimentar, implementar e testar algumas técnicas melhorando, no aspecto **performance,** o **Campus Virtual.**
- Possibilitar que o campus virtual seja acessado pela *internet* de maneira mais rápida e eficiente, sem haver necessidade dos usuários de ter máquinas mais robustas.
- Definir um guia de implementação que auxiliem na modelagem de ambientes virtuais extensos.

1.2. ESTRUTURA DO TRABALHO

Discursar como que o trabalho estará sendo apresentado indicando, capítulo a capítulo, qual o seu conteúdo, o objetivo, seqüência e porque está naquela ordem.

2 FUNDAMENTAÇÃO TEÓRICA

2.1. REALIDADE VIRTUAL

As palavras *realidade* e *virtual* possuem sentidos opostos no que diz respeito à existência concreta de alguma coisa. Porém juntas fazem sentido a uma das mais atuais e modernas técnicas de vivenciar um mundo sem estar efetivamente inserido nele (MIRANDA, 1999).

Muitas definições de realidade virtual foram concebidas por acadêmicos, desenvolvedores de software e pesquisadores com base em suas próprias experiências, o que gerou diversas definições na literatura.

Em termos conceituais, *realidade virtual* é uma realidade que é aceita como verdadeira, embora não necessariamente exista fisicamente (VINCE, 1998). Em termos computacionais, a realidade virtual é considerada a forma mais avançada de interface com o computador. Permite ao usuário realizar a imersão, navegação e interação em um ambiente sintético tridimensional gerado por computador, utilizando canais multi-sensoriais.

O principal conceito inserido no contexto de realidade virtual é o envolvimento do usuário com o sistema. Esta é mais importante que o detalhamento gráfico o que, nem sempre, é possível modelar de maneira realista (HOUNSELL e PIMENTEL, 2003).

2.1.1. Histórico da Realidade Virtual

O termo *realidade virtual* aparece pela primeira vez no início dos anos 80 por Jaron Lanier, fundador da VPL Research Inc., "*para diferenciar as simulações tradicionais feitas por computador de simulações envolvendo múltiplos usuários em um ambiente compartilhado*" (NETTO, 2001).

A realidade virtual começou na indústria de simulação, com os simuladores de vôo que a força aérea dos Estados Unidos passou a construir depois da Segunda Guerra Mundial (KIRNER, 2004).

Alguns anos depois, por volta de 1965, Ivan Sutherland apresentou para a comunidade científica a idéia de usar computadores para desenhar projetos diretamente na tela de um computador através do uso de uma caneta ótica - foi o início dos gráficos computadorizados (computação gráfica). Sutherland tornou-se o precursor da atual indústria de CAD e desenvolveu o primeiro videocapacete totalmente funcional para gráficos de computador no projeto *The Ultimate Display*. Com o uso deste videocapacete era possível ao usuário ver, através da movimentação de sua cabeça, os diferentes lados de uma estrutura de arame na forma de um cubo flutuando no espaço.

Em 1982, Thomas Furness demonstrava para a força aérea americana o VCASS (*Visually Coupled Airborne Systems Simulator*), conhecido como Super Cockpit – um simulador que imitava a cabine de um avião através do uso de computadores e videocapacetes interligados representando um espaço gráfico 3D. Os videocapacetes integravam a parte de áudio e vídeo. Assim, os pilotos podiam aprender a voar e lutar em trajetórias com 6 graus de liberdade (6DOF), sem decolar verdadeiramente, ficando praticamente isolados do mundo ao seu redor. O VCASS possuía uma alta qualidade de resolução nas imagens e era bastante rápido no *rendering* de imagens complexas. No entanto apresentava um problema: milhões de dólares eram necessários apenas para o capacete. Através do uso de uma nova tecnologia de visores de cristal líquido (LCD) Michael McGreevy começou a trabalhar no projeto VIVED (*Virtual Visual Environment Display*) em 1984 na NASA, no qual as imagens seriam estereoscópicas. A resolução das imagens era limitada em comparação ao VCASS, mas o custo era bastante atrativo. A parte de áudio e vídeo foi então montada sobre uma máscara de mergulho utilizando dois visores de cristal líquido com pequenos auto-falantes acoplados. Scott Fisher junta-se a esse projeto no ano de 1985 com o objetivo de incluir nele: luvas de dados, reconhecimento de voz, síntese de som 3D, e dispositivos de *feedback* tátil.

No final de 1986 a equipe da NASA já possuía um ambiente virtual que permitia aos usuários ordenar comandos pela voz, escutar fala sintetizada e som 3D, e manipular objetos virtuais diretamente através do movimento das

mãos. O mais importante é que através deste trabalho foi possível verificar a possibilidade de comercialização de um conjunto de novas tecnologias, sendo que o preço de aquisição e desenvolvimento tornava-se mais acessível.

Atualmente, com o desenvolvimento tecnológico dos últimos anos, a realidade virtual está sendo utilizada para os mais diversos fins nas diversas áreas da ciência, sendo que na última década aplicações médicas utilizando realidade virtual passaram a ser desenvolvidas, tornando essa área comercialmente e clinicamente importante em termos de tecnologia aplicada à medicina (SZÉKELY e SATAVA, 1999).

2.1.2. Realidade Virtual Imersiva e Não-Imersiva

Os sistemas de realidade virtual são divididos em dois tipos principais: realidade virtual imersiva e realidade virtual não-imersiva (TAXEN e NAEVE, 2002). A diferença entre as duas não é realmente relatada na habilidade de promover a sensação de imersão, mas o uso de dispositivos que “escondem” o mundo real do usuário ou não.

A realidade virtual imersiva (RVI) implica no uso de dispositivos de saída (tais como HMD – *Head Mounted Display*) e entrada (tal como um *Data Glove*) que transferem gestos do usuário para um mundo virtual. É necessário nestes tipos de sistemas, *hardware* muito potente para produzir ambientes com gráficos aceitáveis, e garantir a sensação de imersão no ambiente.

Um ótimo exemplo desse tipo de ambiente é a caverna digital. Esta é uma infra-estrutura do Núcleo de Realidade Virtual do LSI (Laboratório de Sistemas Integráveis), vinculado à Escola Politécnica da USP. Desenvolvido por pesquisadores do LSI-EPUSP, esse sistema é conhecido nos Estados Unidos como CAVE (*Cave Automatic Virtual Environment*) e na Europa como CUBE (Ver fig. 1).

FIGURA 1. CAVE da USP – Exemplo de RVI.

Por outro lado, a realidade virtual não imersiva (RVNI) não necessita de *hardware* específico, pois os usuários podem utilizar simplesmente monitores, CPU's e *mouses* convencionais para acessar o mundo sintético. Em muitos casos, a realidade virtual só precisa parecer precisa, não necessariamente ser precisa (FRANCIS e TAN, 1999). Deste modo, o mundo sintético não demandará muita performance computacional.

Um bom exemplo desse tipo de ambiente é a Cabine Virtual da Polícia Militar, projeto modelado pelos integrantes do LRV (Laboratório de Realidade Virtual) da Universidade Federal de Santa Catarina. É a modelagem de uma avenida no Kobrasol, bairro da cidade de São José, juntamente com uma cabine virtual de denúncias. Coube ao LRV da UFSC criar um exemplo em 3D do funcionamento da cabine para a Polícia Militar (Ver fig. 2).

FIGURA 2. Cabine Virtual da Polícia Militar (UFSC) – Exemplo de RVNI.

A realidade virtual não imersiva possui vantagens como: utiliza plenamente todas as vantagens da evolução da indústria de computadores, evita as limitações técnicas e problemas decorrentes do uso de dispositivos específicos e a facilidade de uso. Casos especiais de sistemas realidade virtual não imersiva são aqueles que podem ser explorados através da internet através de uma linguagem chamada VRML (*Virtual Reality Modeling Language*). A realidade virtual não imersiva, quando baseada na web, possui grandes vantagens sobre a realidade virtual imersiva, pois a não-imersiva é muito mais simples e barata de se implantar num site (KIRNER, 2004).

2.2. *VIRTUAL REALITY MODELING LANGUAGE* – VRML

VRML (*Virtual Reality Modeling Language*) é um formato de arquivo para descrever objetos e mundos tridimensionais interativos e foi projetado para ser

veiculada via *internet* e sistemas locais em uma variedade de aplicações (HARTMAN e WERNECKE, 1996).

O objetivo inicial era suportar o necessário para o desenvolvimento de mundos virtuais tridimensionais multiusuários na Internet, sem precisar de redes de alta velocidade. Baseando-se em parte do Open Inventor, da Silicon Graphics, com características para utilização na Internet permitindo fazer ligações entre mundos virtuais (CAREY e BELL, 1997).

O padrão VRML (ISO/IEC 14772) descreve quais e como os objetos são representados. Os objetos podem ser representados através de geometrias primitivas, transformações hierárquicas, fontes de luz, pontos de visão, animações, mapeamento de texturas, etc.

Um dos objetos constitutivos do VRML são os NÓS. Estes são os blocos básicos de construção de um ambiente em VRML, e o seu nome ou tipo indica o objeto que ele descreve, que são basicamente *shapes* e suas propriedades. Nós individuais descrevem *shapes*, cores, luzes, como posicionar e orientar *shapes*, etc (AMES, 1997).

A vantagem primária do VRML refere-se à independência da plataforma operacional, permitindo sua visualização em diversos sistemas operacionais desde que compatíveis com o padrão da ISO-IEC.

A fim de visualizar e interagir com ambientes e objetos virtuais um *plug-in* necessita ser instalado. O *plug-in*, como extensão ao navegador WWW, será ativado toda vez que acessar arquivos com a extensão “.WRL” na Internet.

2.3. DEFINIÇÕES

Para efeito desta monografia, os ambientes virtuais podem ser ainda classificados como Extensos, Massivos ou Limitados; cada um podendo ainda ser sub-classificado em Exterior, Interior ou Misto, conforme as definições e diferenciações a seguir.

2.3.1. Ambientes Exteriores

Como o próprio nome já diz, **ESTAR FORA**. Um ambiente virtual pode ser considerado do tipo Exterior quando os objetos agregados a ele possuem uma parte visível externa, podendo ter também a interna, mas a casca externa destes tem maior ênfase (importância) do que a interna. Usualmente, quando o ambiente não é todo fechado (circundante) tem-se ele como um Ambiente Exterior.

2.3.2. Ambientes Interiores

Como o próprio nome já diz, **ESTAR DENTRO**. Quando o ambiente virtual possui uma casca externa e interna, mas a percepção da casca pela parte interna (para o entendimento do ambiente e da aplicação pelo usuário) é mais importante, este é denominado Ambiente Interior. Se o ambiente possuir somente a casca interna, também ele dever ser denominado como um Ambiente Interior.

Às vezes, um ambiente virtual (prédio ou construção) pode transmitir ao seu usuário, tanto a sensação de estar fora (de um carro, por exemplo), mas dentro (de uma obra por exemplo) mas uma das duas é mais significativa do que a outra em relação aos objetivos gerais do ambiente, para definir se o ambiente virtual é Externo (Exteriores) ou Interno (Interiores).

No caso de uma aplicação na Engenharia Civil, a estrutura da construção geraria um ambiente onde o usuário estaria interno. Mas se o ambiente virtual fosse voltado para visualizar análises estáticas em vigas, então este seria um Ambiente Exterior.

Um outro quesito a ser avaliado para indicar se o ambiente virtual é exterior ou interior, é o tempo que se deseja ou se espera que usuário permaneça na parte de fora ou de dentro de um ambiente virtual.

2.3.3. Ambientes Extensos

Um ambiente pode ser denominado como Extenso quando uma das características principais do mesmo é a necessidade de muita navegação entre as partes do ambiente.

Uma das aplicações exemplo para este tipo de ambiente é o campus virtual da Universidade de Hong Kong (WONG et. al., 2002). O campus virtual de Hong Kong foi projetado para quiosque e teve como objetivo orientar os visitantes do campus. Possui um sistema de grafos que orienta o usuário sobre determinado caminho. Dado um ponto de partida e um de destino, o sistema automaticamente, através de algoritmos de caminho mínimo, indica ao usuário o menor caminho dentre os dois pontos do campus (Ver fig. 3).

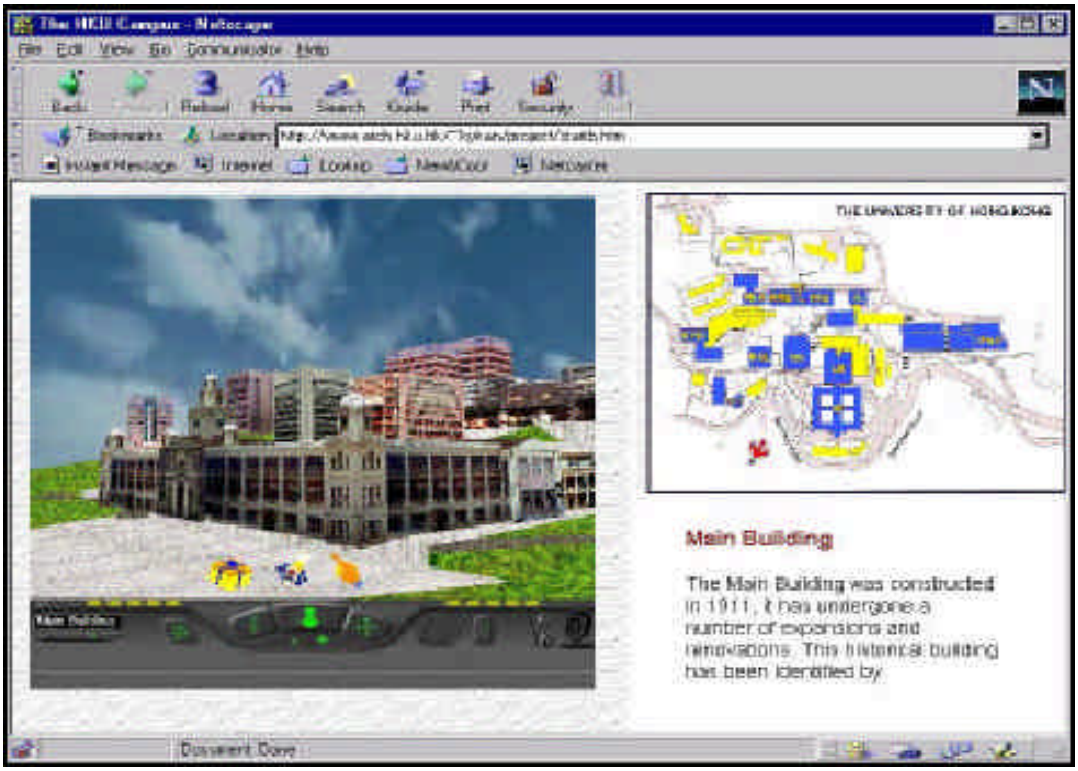

FIGURA 3. Campus Virtual da Universidade de Hong Kong.

Outro exemplo de ambientes extensos de interesse refere-se ao campus virtual da Universidade Federal do Rio Grande do Sul (UFRGS). Estando

disponível na internet (UFRGS, 2003), o ambiente virtual denominado como Campus Central em Realidade Virtual, foi desenvolvido pelo Núcleo de Computação Gráfica Aplicada (NCA) em conjunto com a Faculdade de Arquitetura. Este ambiente permite observar a riqueza do patrimônio histórico e arquitetônico da UFRGS. Fez-se a modelagem das edificações, que compõem o campus, através do programa *AutoCAD*. Em segunda fase, para a aplicação dos materiais (cores e texturas) foi utilizado o software *3D Studio*. Por fim, após a exportação dos mesmos para VRML, os prédios prontos foram reunidos em um único arquivo gerando o campus virtual da UFRGS (Ver fig. 4 e 5).

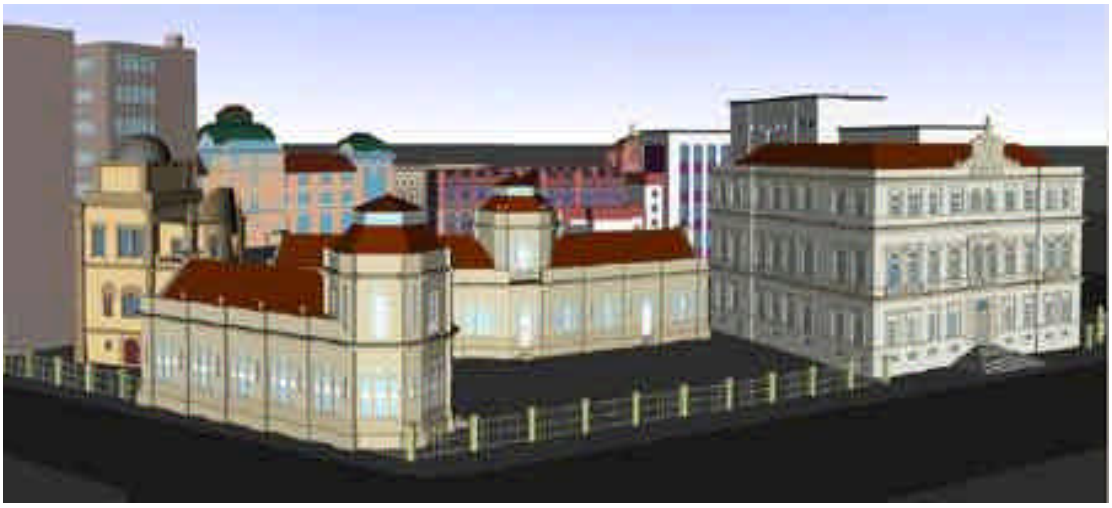

FIGURA 4. Vista Frontal do Campus Virtual da UFRGS.

FIGURA 5. Vista Lateral do Campus Virtual da UFRGS.

2.3.4. Outras Nomenclaturas

Ainda pode-se encontrar na literatura menção a outros tipos de ambiente que são considerados aqui, englobados como Ambientes Extensos; são eles: os ambientes massivos e os de grande escala.

Ambientes “Massivos” (CORSEUIL et. al., 2003) referem-se a ambientes que contam poucos objetos mas que estes são extremamente detalhados, de forma que geram um modelo geométrico massivo, grande.

Outros tipos de ambientes (RODRIGUES et. al., 2004) são constituídos de uma quantidade enorme de objetos que, individualmente podem até ser simples mas, a composição da cena gera um ambiente de “Grande Escala”.

Os dois exemplos acima, um objeto massivo e inúmeros objetos simples, seriam os limites extremos de uma graduação onde toda composição intermediária é chamado aqui de “Ambiente Extenso”.

2.3.5. Ambientes Limitados

Um ambiente virtual pode ser considerado do tipo Limitado quando o mesmo não requerer muita navegação entre as partes.

Um número grande e crescente de aplicações está utilizando ambientes virtuais limitados em áreas incluindo manufatura, negócios, entretenimento e medicina.

Um exemplo de aplicação de Ambiente Limitado é um simulador de robô, projeto do grupo LARVA - Laboratório de Realidade Virtual Aplicada do Departamento de Ciência da Computação – DCC. Este, trata-se de um ambiente limitado de programação para iniciantes em robótica. Com uma interface intuitiva e agradável, podem-se programar algumas tarefas e visualizar as operações do robô de forma tridimensional (Ver fig. 6).

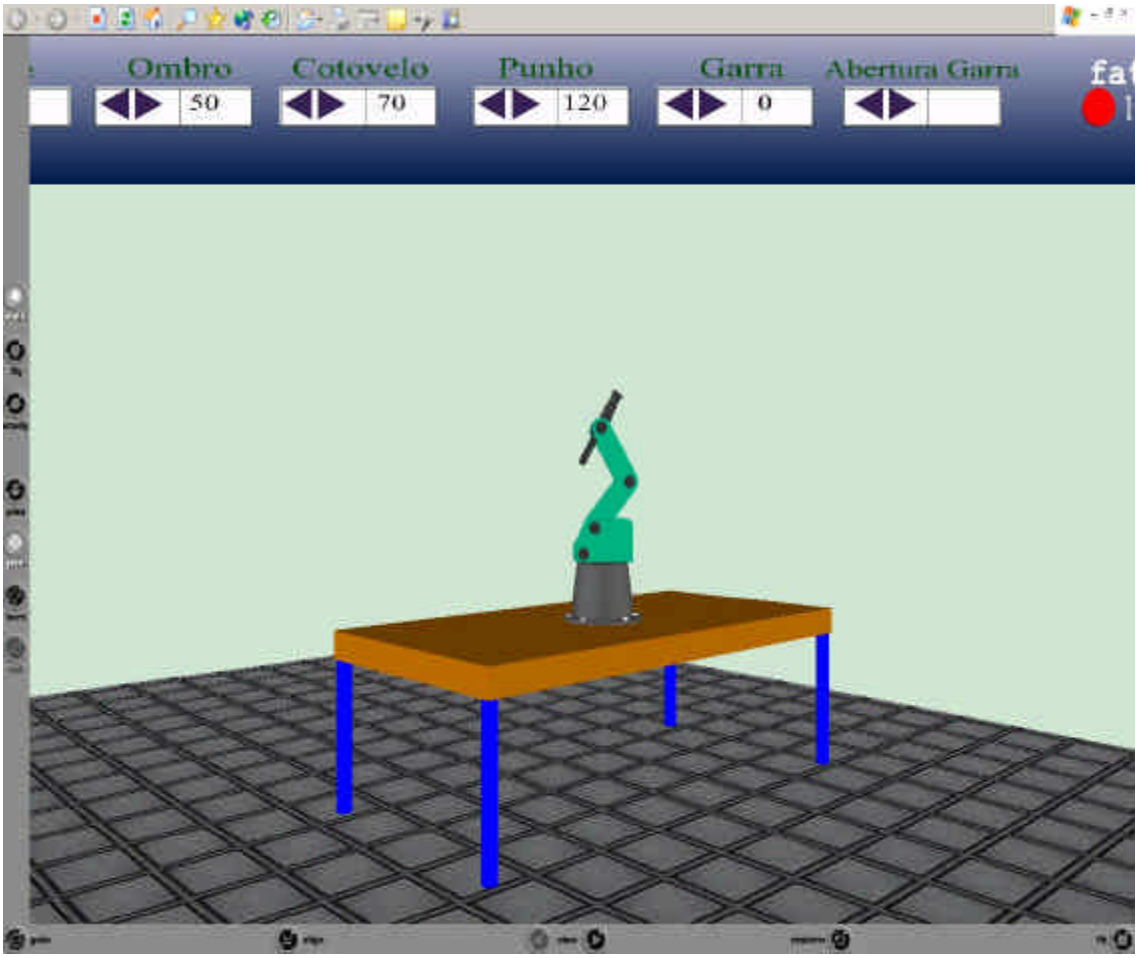

FIGURA 6. Simulador de Robô – Exemplo de Ambiente Limitado.

Um outro exemplo de ambiente limitado é a modelagem de um Caixa Eletrônico 24 Horas (Ver fig. 7). O ambiente pode ser usado para treinar os funcionários e clientes do banco, tendo a possibilidade do usuário realizar várias tarefas do cotidiano do caixa eletrônico virtualmente (EASTGATE, 2001).

Este ambiente é considerado de interior limitado, devido às interações feitas pelos usuários com os caixas eletrônicos (parte mais importante do ambiente) e estes estão circundantemente fechados através das paredes transparentes. Ainda apesar de ser visível por fora (externo), a maior parte do tempo espera-se que o usuário esteja dentro do ambiente manipulando o Caixa Virtual.

FIGURA 7. Caixa Eletrônico Virtual – Exemplo de Ambiente Limitado.

2.4. CONCLUSÕES

Com o uso de Ambientes Virtuais para, cada vez mais, representar ambientes extensos (como cidades, campus universitários, por exemplo) torna-se cada vez mais evidentes e necessárias pesquisas voltadas para a condução e orientação dos usuários nestes ambientes (PREECE, 1994) onde o uso de mapas auxiliares já foi enfaticamente defendido aos ambientes limitados e ainda; o uso cada vez mais criterioso de recursos de projetos voltados para a eficiência do projeto, tanto quanto ao seu carregamento via *internet* quanto a sua performance (FPS – *Frames per Second*) quando visualizado.

Pode-se dizer que estes dois fatores (orientação e performance) seriam bastante diferenciados para o caso de projetos de Ambientes Limitados. O Trabalho apresentado aqui se aprofunda no segundo aspecto, ou seja, nas formas de otimizar a performance dos Ambientes Extensos.

3 OTIMIZAÇÃO DE UM AMBIENTE VIRTUAL

Na construção de um ambiente virtual de exterior extenso em VRML, a otimização é uma parte essencial. Dadas as limitações, que serão vistas a seguir, impostas pela atual tecnologia, um ambiente virtual que não seja cuidadosamente otimizado é certamente muito lento para captar o interesse do usuário.

Será desenvolvido no decorrer deste capítulo um estudo de técnicas (baseadas em comandos já existentes e disponíveis na linguagem VRML). Não será alterado, e nem proposta, nenhuima alteração de nenhum programa ou código interno do *plug-in*. E sim, serão resgatados comandos dos códigos de descrição (*Markup Language*).

Acredita-se que o compêndio de recursos que serão neste capítulo enfatizados podem (e devem) ser considerados como uma etapa de otimização do ambiente virtual já modelado, caso seja percebido que sua performance não esteja aceitável (conforme definido no fluxograma da fig. 8). Está situação é mais provável ocorrer nos casos de ambientes virtuais de exteriores extensos.

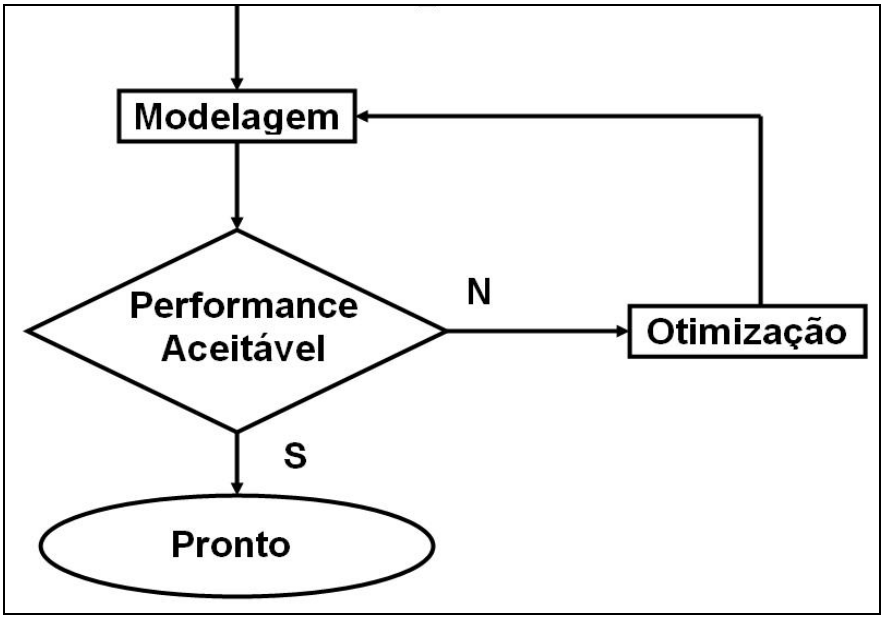

FIGURA 8. Algoritmo demonstrativo do processo de otimização .

3.1. AVALIAÇÃO DE EFICIÊNCIA

Há dois aspectos objetivos a considerar quanto a avaliação da eficiência de um mundo VRML:

- Tempo de carregamento (*download*);
- Velocidade de *rendering*;

O tempo de carregamento, ou *download,* depende diretamente do tamanho dos arquivos que compõem o ambiente, pois estes devem apresentar, dentro do possível, um tamanho reduzido.

A velocidade de *rendering* (medida em quadros por segundo – FPS, *frames per second*) tem dependências de vários outros fatores, que vão desde a complexidade da geometria, passando pela representação de texturas até a iluminação.

A dificuldade existente em avaliar os dois fatores é real e deve-se ao fato do usuário poder usar um leque variadíssimo de plataformas, com diferentes capacidades de *rendering* e velocidades de acesso à rede. Este leque pode ir desde um PC sem aceleração gráfica por *hardware* e com um *modem* a 14,4 Kbps até o mais avançado sistema gráfico, como uma *Onyx2* da *Silicon Graphics* com *InfiniteReality* e uma ligação direta à *Internet*.

Algumas diretrizes podem contribuir para o aumento da eficiência de um mundo e, desta forma, contrariar as referidas limitações. No entanto, o usuário de um sistema com poucos recursos encontrará sempre alguns problemas de desempenho na visualização e navegação de um mundo VRML.

3.2. TÉCNICAS DIRETAS DE OTIMIZAÇÃO

Agora a seguir, serão apresentadas técnicas diretas de otimização de ambientes virtuais em ordem de dificuldade e complexibilidade de implementação. São chamadas de diretas devido a sua aplicação ser individual.

3.2.1. Objetos e Texturas *Inline*

Quando uma cena VRML é complexa deve ser dividida em mundos menores, de modo a facilitar o *download*. Esta técnica permite reduzir o tamanho do arquivo VRML principal, normalmente aquela que não é visível no plano inicial, para dar lugar a uma referência, um apontador para um arquivo externo, diminuindo, assim, o tempo de *download* da vista inicial.

Ao estruturar um mundo complexo, uma boa técnica é construir um arquivo principal com um tamanho reduzido, para que o *download* seja rápido. O mesmo poderá conter *inlines*, ou seja, referências para arquivos externos que irão conter as cenas restantes que compõem o mundo.

Na figura 9 apresenta-se a organização espacial de alguns objetos representados em um Ambiente Extenso. Os objetos assinalados não fazem parte do arquivo principal, são objetos *inline*. Ou seja, os objetos assinalados são outros arquivos (MIRANDA, 1999).

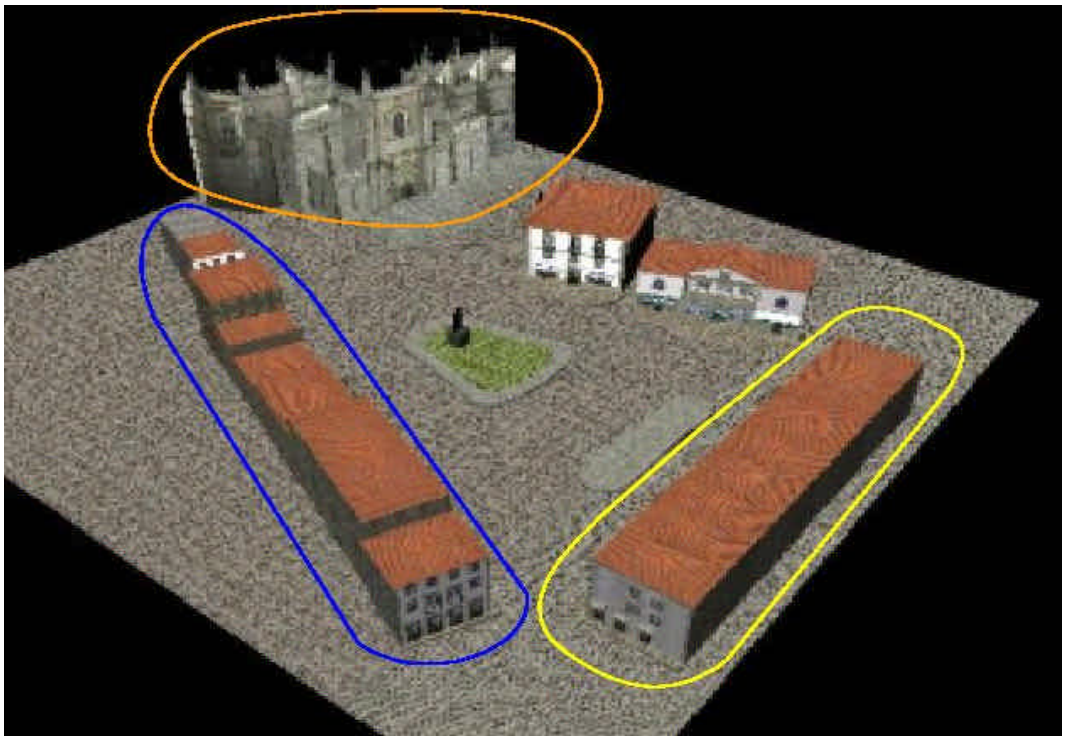

FIGURA 9. Organização espacial dos objetos.

Na figura 10 mostra um pequeno trecho de código demonstrando a sintaxe de um objeto *inline*. Neste exemplo, tirado da implementação do Cemitério Virtual (REDEL, 2004) mostra que os túmulos do mesmo foram modelados em outro arquivo “wrl” sendo chamados desta maneira no arquivo principal.

```
Inline {
    url "Tumulos.wrl"
}
```

FIGURA 10. Sintaxe de um objeto *Inline*.

3.2.2. Texturas

Texturas são imagens 2D que são aplicadas aos objetos. Texturas são capazes de acrescentar facilmente detalhes as formas (e portanto, realismo) sem muito esforço e modelagem geométrica.

Quando é aplicada uma textura a um cubo, conforme apresentado na figura 11, este passa a apresentar maior profundidade e realismo.

FIGURA 11. Exemplo de Cubo com uma Textura Definida.

Na figura 12 mostra como se insere uma textura em código VRML. O objeto onde se aplica a textura de mármore (arquivo *marble.jpg*) é um cubo de dimensões x=2, y=4 e z=3.

```
Shape {
        appearance Appearance {
        texture ImageTexture {
                url "marble.jpg" #Imagem que será incorporada ao objeto.
        }
        }
        geometry Box { size 2 4 3 } #Geometria e dimensões do objeto.
}
```

FIGURA 12. Sintaxe de Textura em VRML

3.2.3. Detecção de Colisões

O *browser* considera todos os objetos sólidos. Desta forma, não é possível passar através dos objetos, originando colisões (MIRANDA, 1999).

Em termos de recursos, é muito dispendioso calcular as colisões de todos os objetos que compõem a cena. Uma boa solução é informar o *browser* para que não faça este teste sobre aqueles objetos que não seja provável serem colididos, como o teto de um quarto ou um telhado de uma casa.

Deve-se portanto, verificar cuidadosamente quais objetos ou grupo de objetos pode-se aplicar este recurso. Isso afetará diretamente na performance de visualização (FPS – *Frames Per Second*).

Desta forma, há uma diminuição dos testes de colisão e um conseqüente aumento na eficiência (MIRANDA, 1999).

3.2.4. Condensação de Código

Existe um recurso da ferramenta VRMLpad, esta que será melhor explicada no Capítulo 5, que é de gerar uma versão "condensada" do arquivo “.wrl”. Este processo diminui sensivelmente o tamanho, conseqüentemente

aumentando a performance, pois elimina todos os comentários e espaços em branco colocados pela identação (que são importantes para visualização e entendimento, mas não fazem diferença para o *browser*).

3.2.5. *Billboard*

Ao ser aplicado a um objeto, o recurso *Billboard* cria um novo sistema de coordenadas e define um eixo de rotação para esse objeto. Esse sistema de coordenadas é automaticamente rodado sobre o eixo definido, de forma a que o objeto esteja sempre orientado para o ponto de visão e para que, dessa maneira, o usuário o veja sempre de frente (Ver fig. 13).

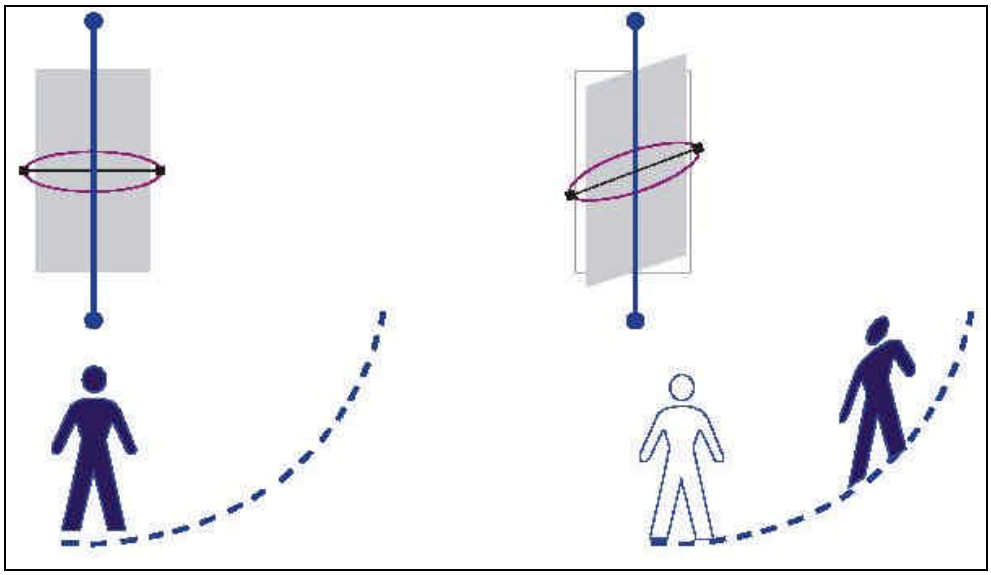

FIGURA 13. Ilusão tridimensional através de um objeto *Billboard*.

Se o usuário rodar em torno da superfície, esta acompanha a rotação e mantém-se sempre de frente para o usuário, iludindo a tridimensionalidade.

Esta técnica é muito eficiente para criar efeitos 3D usando superfícies planas bidimensionais. Um bom exemplo de aplicação é o caso de árvores. Neste caso, basta agrupar um nó *Billboard* com uma superfície 2D e atribuir uma textura com a imagem de uma árvore a essa superfície.

Os *Billboards* reduzem fortemente o número de polígonos, o que contribui

para um melhor desempenho.

A figura 14 mostra um pequeno trecho de código VRML da estrutura do nó *Billboard* representando uma imagem de árvore (Arvore.gif) sendo mapeada numa face (*IndexedFaceSet*).

```
Billboard {
        axisOfRotation 0 1 0
        children Shape {
                appearance Appearance {
                        texture ImageTexture {
                                url "img\Arvore.gif"
                        }
                }
                geometry IndexedFaceSet {
                        coord Coordinate {
                                point [
                                        -0.3 0.0 0.0,   0.3 0.0 0.0,
                                        0.3 0.6 0.0,  -0.3 0.6 0.0,
                                ]
                        }
                        coordIndex [ 0, 1, 2, 3 ]
                        solid FALSE
                }
        }
}
```

FIGURA 14. Sintaxe do nó *Billboard* em VRML.

3.2.6. *Bounding Box*

Muitos dos nós existentes em VRML incluem um campo de *Bounding Box* (bbox). Isto é usado geralmente através do agrupamento de nós para gerar uma orientação para o usuário no ambiente sobre o tamanho aproximado do grupo para selecionar otimizações. O tamanho *default* para um *Bounding Box* (-1,-1,-1), implica que o usuário não especificou o *Bounding Box* e que o *browser* precisa computar ou concluir o caso mais conservativo. Um valor de tamanho de *Bounding Box* (0,0,0) é válido e representa um ponto no espaço

(isto é, uma caixa infinitamente pequena).

Os campos "bboxCenter" (centro do *Bounding Box*) e "bboxSize" (tamanho do *Bounding Box*) podem ser usados para especificar a caixa máxima possível para os objetos dentro de um nó agrupador (isto é, Transformar). Estes campos são usados como orientação para melhorar certas operações como determinar se o grupo atende ou não as necessidades a serem desenhadas. Se o *Bounding Box* especificado é menor que o *Bounding Box* verdadeiro do grupo, os resultados serão indefinidos. O *Bounding Box* deverá ser grande o suficiente para conter completamente os efeitos de todos os nós de sons e luzes que são filhos deste grupo (CAREY e BELL, 1997).

Se o tamanho deste grupo mudar com o tempo devido a filhos animados, então o *Bounding Box* precisa ser grande o suficiente para conter todas as possíveis animações (movimentos). O *Bounding Box* deve tipicamente ser a união dos *Bounding Box* dos filhos do grupo, não deve incluir qualquer transformação feita pelo próprio grupo (isto é, o *Bounding Box* é definido pelo sistema de coordenada local do grupo).

A figura 15 mostra um nó agrupador e o seu *Bounding Box*. Nesta figura o nó agrupador contém 3 formatos: um cone, um cilindro e uma esfera. O tamanho do *Bounding Box* é escolhido para cercar as três geometrias completamente.

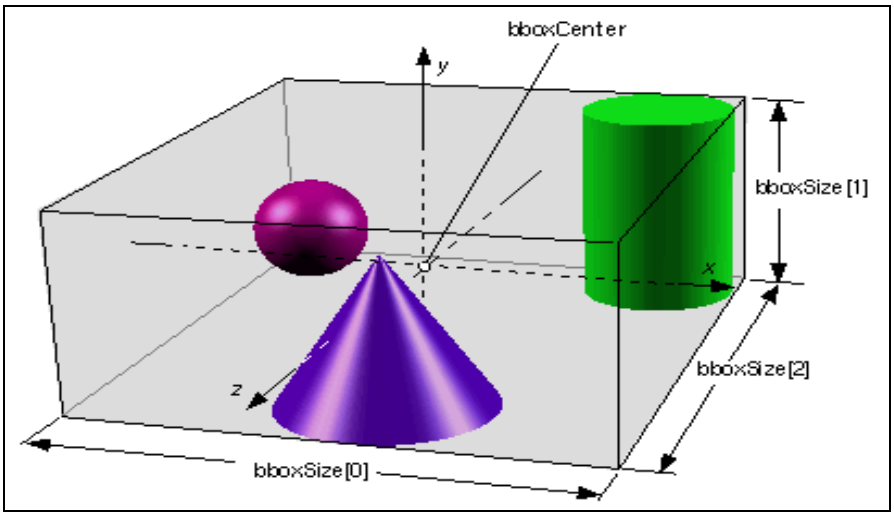

FIGURA 15. *Bounding Box* agrupando três diferentes tipos de geometria (CAREY e BELL, 1997).

Na figura 16 mostra um pequeno trecho de código demonstrando a sintaxe do *Bounding Box*.

```
Collision {
        children [
                Transform {
                        bboxSize 2 1 10
                        children [
                                Shape {
                                        geometry Box {
                                                size 2 1 .2
                                        }
                                        appearance Appearance {
                                                material Material {
                                                        ambientIntensity 0
                                                }
                                        }
                                }
                        ]
                }
        ]
}
```

FIGURA 16. Sintaxe do *Bounding Box.*

Para determinar os objetos envolvidos no cálculo de visibilidade, o *browser* executa um procedimento denominado *culling test*. Este teste consiste em comparar a *bounding box* de um objeto com o volume do campo de visão (Ver fig. 17). Se estes dois volumes não se intersectam, então o objeto não é visível neste ponto de visão, podendo ser ignorado durante o *rendering*. Sempre que houver deslocamento do ponto de visão ou movimento do objeto, o *browser* tem de fazer um novo teste (MIRANDA, 1999).

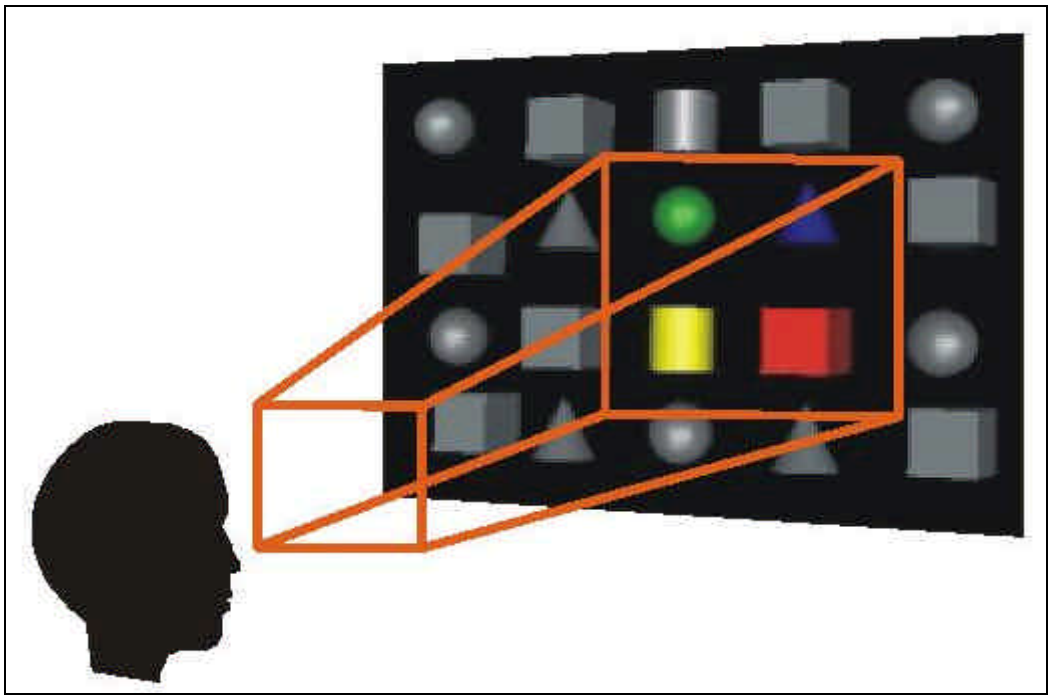

FIGURA 17. *Bounding Box* representando o volume do campo de visão (MIRANDA, 1999).

3.2.6.1. Outra Visão do *Bounding Box*

Pode-se utilizar o conceito de *Bounding Box* como se fosse um “campo de força”. Este conceito é utilizado para evitar com que o usuário, chegue perto de um objeto previamente texturizado e consiga perceber o mesmo proporcionando mais realismo.

Para isso, utilizamos um *box* comum que envolva todo o objeto em questão, onde o usuário não possa chegar muito próximo. Em seguida, aplicamos a este *box* a propriedade de colisão, fazendo com que o usuário não possa atravessar por este *box*, muito menos chegar perto dos objetos envolvidos pelo mesmo. Lembrando que este *box* deverá ser totalmente invisível para o usuário (Ver fig. 18).

```
DEF COLISAO Transform {
        children Shape {
                appearance Appearance {
                        material Material {
                                transparency 1 # Transparência total
                        }
                }
                geometry Box { # Box que envolve os outros objetos
                        size 10 10 10
                }
        }
}
```

FIGURA 18. Sintaxe da outra visão do conceito de *Bounding Box.*

3.2.7. Redução do Número de Polígonos-Simplificação da Geometria

O VRML é muito sensível ao número de polígonos. Por isso, quanto maior for o número de polígonos maior será o tamanho do arquivo VRML, e quantos mais polígonos forem visíveis a cada instante mais lento será o *rendering* da cena. Um bom método para melhorar, quer o tempo de *download*, quer a velocidade de *rendering*, é reduzir na medida do possível o número de polígonos na cena. Como regra geral, pode-se criar um efeito de superfícies mais complexas usando texturas com pouca resolução.

É boa regra que mundos planeados para serem vistos na *internet* por um PC não tenham mais do que, aproximadamente, 1000 triângulos visíveis a cada instante e não mais do que um par de milhares de triângulos totais na cena (HARTMAN, 1996).

Para que a simplificação da geometria aconteça de uma forma mais eficiente, podem-se citar dois aspectos fundamentais:

- Trocar o objeto em questão por outro mais simples (trocar por um objeto que possua menos faces);
- Trocar partes dos objetos por texturas.

3.2.8. LOD (*Level Of Detail*)

Quanto mais realista for um mundo VRML, mais tempo demorará o *browser* a desenhar esse mundo, o que acarreta uma perda na interatividade. Para manter um grande nível de realismo sem uma grande perda de interatividade é possível controlar o nível de detalhe com que o *browser* constrói os objetos. Tal é possível através de um recurso denominado Nível de Detalhe - *LOD.*

Esta característica implica que sejam criadas múltiplas versões de diferentes complexidades de um mesmo objeto, normalmente um modelo de grande detalhe, um de médio detalhe e outro de baixo detalhe. O *browser* seleciona o modelo que é visualizado em determinado instante, baseado na distância entre o ponto de visão e o objeto (Ver fig. 19).

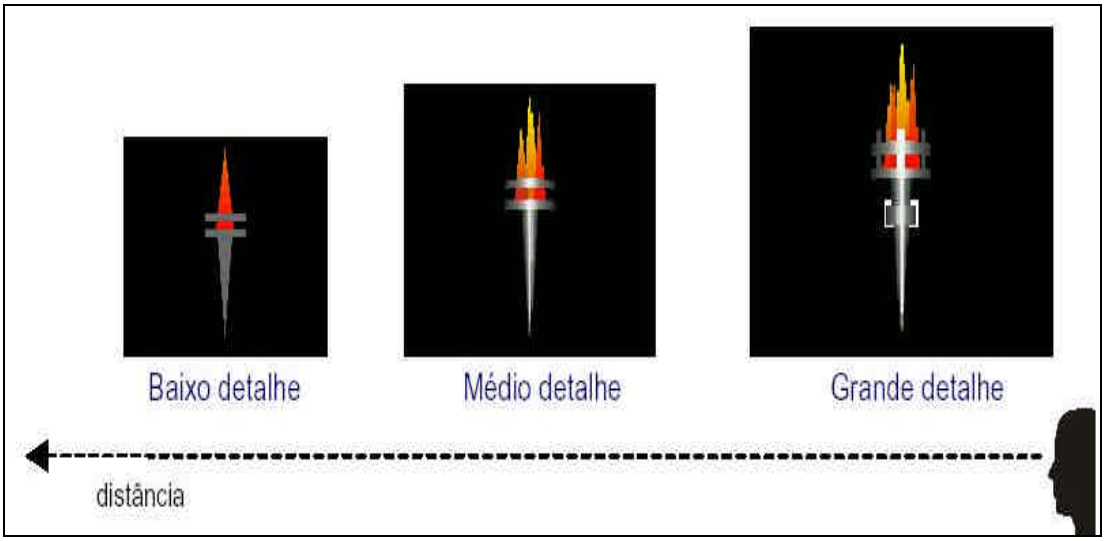

FIGURA 19. Três níveis de detalhe para o mesmo objeto.

No exemplo dado, o modelo que o *browser* vai desenhar depende da distância entre o ponto de visão e o objeto. A versão de grande detalhe é utilizada quando o ponto de visão está próximo do objeto, incluindo toda a minúcia necessária para fazer com que o mesmo pareça realista. Na versão de médio detalhe são eliminados alguns pormenores suplementares, de forma a

simplificar o objeto; esta versão é utilizada quando o ponto de visão está a meia distância. A versão de baixo detalhe é utilizada quando o ponto de visão está longe e por isso os objetos são o mais simplificados possível.

Um exemplo prático da aplicação da técnica de LOD para representar a geometria de uma determinada célula é detalhada na figura 20.

```
DEF HEXAGONO LOD {
        range [
        2 #Limite do nível 0 - 2 metros
        4 #Limite do nível 1 - 4 metros
        ]
        level [
        #nível 0 - menos que 2 metros.
        Inline {
                url "Nível_0.wrl" #Representação detalhada da
                        #referida célula.
        }
        #nível 1 - entre 2 a 4 metros.
        Inline {
                url "Nível_1.wrl" #Representação das imagens externas
                        #limítrofes da célula.

        }
        #nível 2 - maior que 4 metros.
        Inline {
                url "" #nada - seguramente fora dos campos de visão
           #(obscurecido pelas imagens de nível 1 das células
           #imediatamente adjacentes a célula onde o usuário
           #se encontra.
        }
        ]
}
```

FIGURA 20. Sintaxe do LOD em 3 níveis em VRML (REDEL, 2004).

Quando o usuário estiver a uma distância menor que dois metros do centro da célula, estará no nível 1, ou com detalhes aumentados ao máximo. A medida que o usuário se afasta do centro, ficando entre dois a quatro metros, o mesmo estará na célula adjacente. O nível de detalhe automaticamente diminuirá para 2 (dois) e será visualizada a imagem do submundo que o usuário saiu. Quando o usuário estiver a quatro metros ou mais da célula, o

usuário estará em uma célula não adjacente. Este fato torna desnecessária uma visualização do submundo representado pela célula inicial. O processo se repete para cada célula do ambiente virtual.

Para que um LOD colabore efetivamente com o processo de otimização de um ambiente virtual de exterior extenso, um objeto distante do observador neste ambiente não deve se representado com tanta precisão quando a um objeto que esteja mais próximo do visualizador, por mais detalhado que seja o modelo tridimensional utilizado. Por isto, podemos utilizar modelos tão mais simples quanto mais longe o objeto estiver do usuário, sem comprometer a qualidade visual da aplicação e poupando tempo de processamento do computador. Se observarmos as duas esferas renderizadas (Ver fig. 21), poderemos perceber uma diferença significativa entre os modelos. Já a diferença entre as duas esferas da parte inferior da figura é negligenciável, apesar de terem sido obtidos com os mesmos modelos 3D das duas primeiras esferas. O modelo de menor detalhamento necessita aproximadamente sete vezes menos tempo de processamento do que o modelo de maior resolução (MIRANDA, 1999).

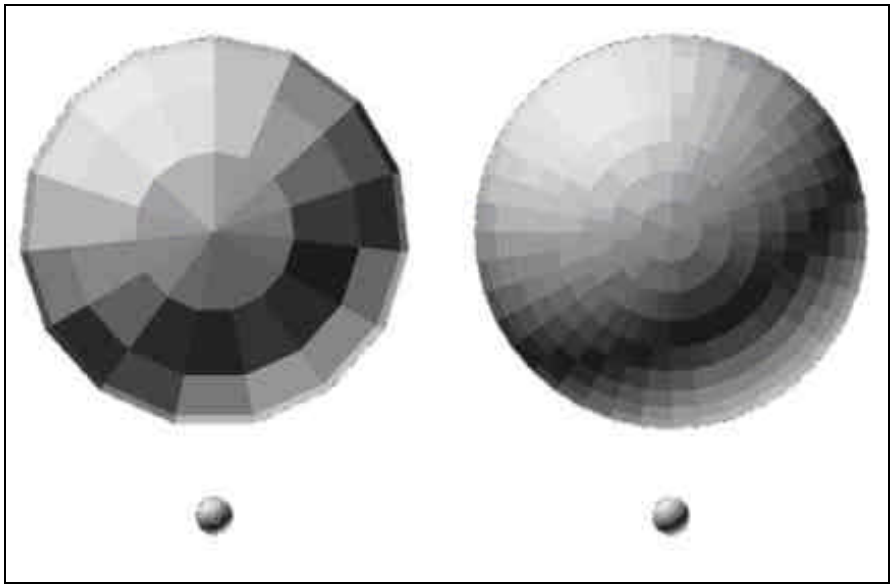

FIGURA 21. Duas esferas, uma com 15 e outra com 40 segmentos por paralelo.

3.2.9. Âncoras

Talvez uma das características mais interessante de VRML é a sua habilidade de *linkar* "mundos" VRML no WWW. Os *links* em VRML são ligações, ou âncoras, feitas em figuras específicas da cena. Qualquer figura descrita através de nós VRML pode ser uma âncora. Também pode-se usar *links* para unir cenas ou figuras que estão em arquivos diferentes em um único arquivo VRML. Esta técnica, conhecida como *inlining*, permite construir, por exemplo, uma sala usando as paredes de um arquivo VRML e a mobília de vários outros (ISO/IEC 14772, 2004).

O nó de agrupamento *Anchor* engloba um grupo de *shapes* e cria um *hyperlink* com outras mídias (página HTML, imagem, etc.), isto é, quando o usuário clica nestes *shapes* o navegador visualiza um novo arquivo, até mesmo um outro arquivo “WRL”.

A figura 22 mostra um pequeno trecho de código demonstrando a sintaxe do nó âncora, *linkando* para um WWW qualquer.

```
Anchor {
        url http://www.school.edu/vrml/someScene.wrl#OverView
        children  Shape { geometry Box {} }
}
```

FIGURA 22. Sintaxe do nó Âncora.

3.3. TÉCNICAS COMPOSTAS DE OTIMIZAÇÃO

Agora a seguir, serão apresentadas técnicas compostas de otimização de ambientes virtuais. São chamadas assim devido a sua aplicação ser composta de várias técnicas diretas.

3.3.1. Segmentação de Mundos Virtuais

Segmentar um ambiente virtual é utilizado de modo que o mesmo diminua sua complexidade geral e, então, possa ser executado em máquinas de baixo poder computacional e gráfico (REDEL, 2004). Segmentação é a utilização de duas técnicas diretas: a Textura e o LOD (*Level Of Detail* ou Nível de Detalhe). A implementação desta técnica é facilitada com o uso do conceito de LOD pois, através deste, é possível definir os limites de atuação de cada submundo ou célula/portal.

O algoritmo de célula/portal trabalha sobre uma cena organizada em salas (células), onde cada célula possui um conjunto de objetos renderizáveis e uma lista de portais, que são portas, janelas ou outras conexões com células vizinhas (SILVA et.al., 2003).

A primeira tarefa do algoritmo de célula/portal é procurar a célula onde o observador se localiza e processar graficamente os objetos que são visíveis para o observador (ignorando os objetos ocultos). No segundo passo, o algoritmo procura todos os portais visíveis da célula, processando as células adjacentes. A figura 23 demonstra o modelo célula/portal. Apenas os retângulos preenchidos, que estão no campo de visão do usuário, são processados graficamente.

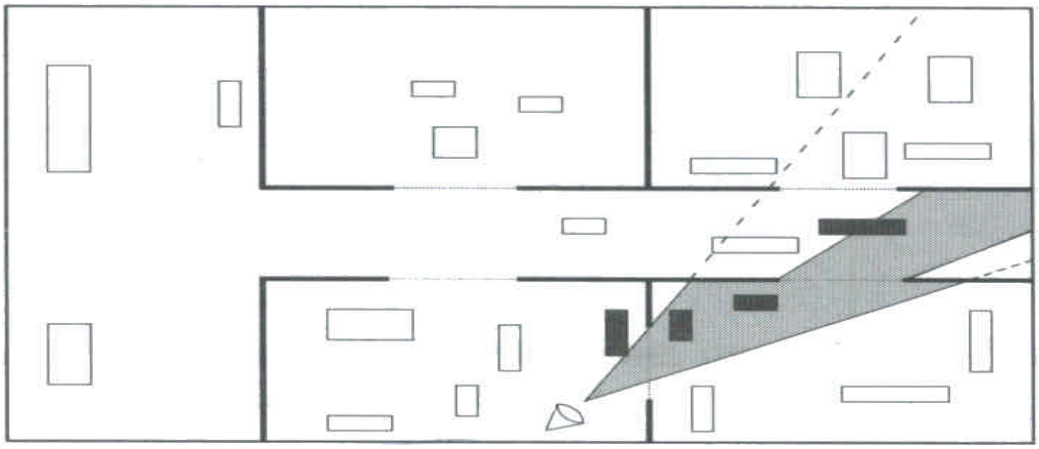

FIGURA 23. Modelo célula/portal (SILVA et. al., 2003)

Entretanto, a técnica de célula/portal não é aplicável a ambientes virtuais que não tenham pontos de transição (os portais – portas e janelas) explícitos, ou seja, ambientes abertos.

A segmentação de ambientes virtuais pode ser realizada entre dois métodos: recorte geográfico e granulação. Uma visão sistemática da segmentação de mundos virtuais é representada na figura 24.

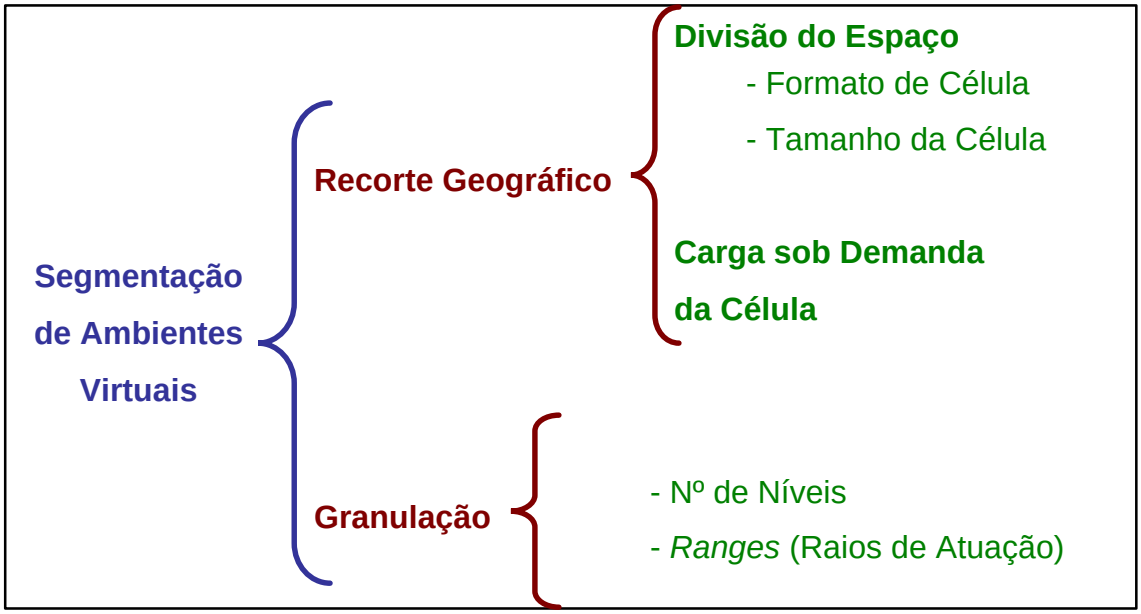

FIGURA 24. Técnica de Segmentação de Mundos Virtuais.

3.3.2. Embrulho

"Embrulhar" um objeto possuidor de várias geometrias de um ambiente virtual nada mais é do que envolvê-lo totalmente por um outro objeto menos complexo, possuidor de menos geometrias (*Bounding Box* por exemplo).

Para que este objeto mais externo, se aplica uma textura, ou melhor, uma fotografia do ambiente real com intuito de fornecer mais realismo ao usuário do ambiente.

Com objetivo de não carregar todas as geometrias do objeto no carregamento do ambiente virtual como um todo, é aplicado então o LOD contendo nos primeiros níveis do mesmo o embrulho propriamente dito. No

último nível, ficará o objeto como originalmente ele foi modelado rebuscado, rico em detalhes e geometrias.

3.3.3. Reuso de Código e/ou Geometria

Através do recurso *DEF/USE* e/ou do recurso de *PROTO* pode-se reutilizar uma parte da descrição de uma cena ou invocar um *script* padrão de geração de geometria.

Um componente de modelagem virtual é um conjunto de nós VRML que implementam a estrutura de um modelo virtual, podendo ser reutilizado em outras modelagens nas quais seja considerado adequado. Tais nós devem ser agrupados em um nó *PROTO* onde são definidos os campos que representam o estado persistente do modelo virtual. No nó *PROTO* também deve ser definida a interface do componente de modelagem virtual através de eventos de entrada (*eventIn*) e eventos de saída (*eventOut*). Os eventos definidos na interface implementam a mudança de estado do modelo virtual. Por exemplo: a interface de um componente de modelagem virtual determina que pode ser alterada a sua textura ou posição (FERREIRA et. al., 2004). Lembrando que este componente deve ser disponibilizado como um arquivo “.wrl” separado .

3.3.4. Ocultação

Alguns ambientes de modelagem e programação de realidade virtual permitem que se especifique um nó *fog* (nevoeiro) e as características do cone de visualização (plano de corte), dentre elas o plano de corte no limite visual da cena (*far culling plane*).

O nó *fog* provê um modo para simular efeitos atmosféricos misturando objetos e cores (especificado pelo campo *color*) baseado nas distâncias dos vários objetos do usuário. As distâncias são calculadas no espaço de coordenada do nó *fog*. O campo *visibilityRange* (Ver fig. 13) especifica a distância (no sistema de coordenada local) a qual objetos são totalmente

obscurecidos pelo nó *fog*. Objetos muito perto do usuário serão pouco misturados com a cor do nó *fog* e um visibilityRange de 0.0 incapacita ou desabilita a função do nó *fog* (CAREY e BELL, 1997).

O uso combinado destes pode diminuir e muito a quantidade de objetos visíveis pelo usuário, aproximando o *back culling plane* , permitindo um aumento no realismo para simular atmosferas sinistras, noturnas ou chuvosas.

3.4. VISÃO GERAL DAS TÉCNICAS

Uma outra forma de ver as técnicas apresentadas anteriormente é conforma a classificação apresentada na figura 25. Nesta vê-se como as técnicas se aplicam em relação a dois aspectos: a código e a geometria, as que são mais simples ou mais complexas respectivamente.

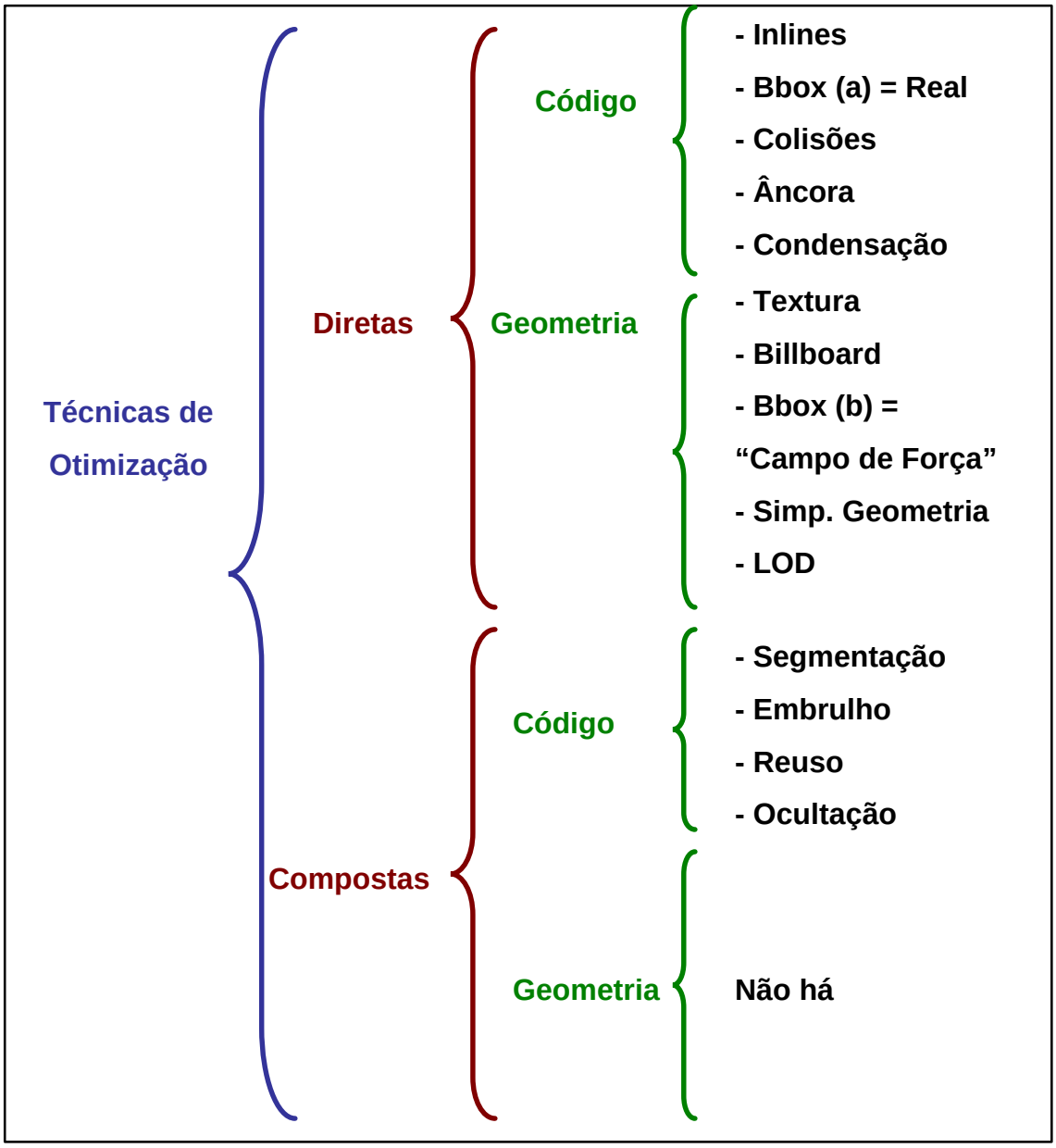

FIGURA 25. Técnicas de Otimização de Ambientes Virtuais.

4 CENTRO DE CIÊNCIAS TECNOLÓGICAS DA UDESC

O Centro de Ciências Tecnológicas – UDESC Joinville foi criado sob a denominação de Faculdade de Engenharia de Joinville (FEJ), pelo governo do Estado de Santa Catarina em 09 de outubro de 1956 (Ver Anexo A), através da Lei Nº 1520/56, que instituiu um curso de Engenharia, a ser implantado no interior do Estado. Foi a primeira tentativa da interiorização do ensino superior, tradicionalmente restrito às capitais dos estados (SCHULZ, 2004).

Joinville por ser o maior pólo industrial do Estado de Santa Catarina, constitui-se no local ideal para a concretização desse sonho. O primeiro vestibular foi realizado em julho de 1965, com apenas 09 candidatos, mas somente em 1° de agosto do mesmo ano que a instituição iniciou suas atividades com o curso de Engenharia de Operações (modalidade de Máquinas e Motores).

O Centro de Ciências Tecnológicas – UDESC Joinville conta hoje com uma estrutura que comporta os cursos de Graduação em Engenharia Civil, Engenharia Elétrica, Engenharia Mecânica, Engenharia de Produção e Sistemas, Tecnologia em Sistemas de Informação (Campus Joinville e São Bento do Sul), Bacharelado em Ciência da Computação Integral, Licenciatura Plena em Física, e, Tecnólogo Mecânico (modalidade Produção Industrial de Móveis), em São Bento do Sul.

A estrutura física do Centro de Ciências Tecnológicas é constituída de 11 (onze) blocos ou prédios e 1 (um) ginásio. Os blocos são numerados em forma de letras cujo intervalo é de A à K. Como ponto obrigatório para entrada de visitantes no campus, tem-se a guarita localizada na entrada do Centro de Ciências Tecnológicas (Ver fig. 26). O prédio da administração e diretoria do campus é referenciado como Bloco A (Ver fig. 27). O Departamento de Ciência da Computação se encontra no Bloco F.

FIGURA 26. Guarita situada na entrada do Campus.

FIGURA 27. Secretaria do CCT – UDESC.

4.1. CCT VIRTUAL – TRABALHO ANTERIOR

O trabalho anterior referente ao projeto de construção do campus virtual, representando o Centro de Ciências Tecnológicas, foi continuado por Alessandro Pinto Schulz (SCHULZ, 2004) sob um Trabalho de Conclusão de Curso intitulado “Modelagem de Exteriores Extensos: Estudo de Caso com “Campus Virtual do Centro de Ciências Tecnológicas de Joinville”.

Este trabalho teve como objetivo a modelagem de ambientes virtuais extensos. Como estudo de caso, utilizou-se o campus do CCT. Neste projeto, modelou-se todos os blocos do campus, incluindo a Biblioteca e o Ginásio, dando mais ênfase ao Bloco F, onde é situado o Departamento de Ciência da Computação.

Além desses, outros itens também foram modelados como placa de identificação do CCT – UDESC, totens indicando o bloco A e o estacionamento, identificação antiga placa da FEJ com letras em alto relevo, jardim do estacionamento e o próprio estacionamento. No entanto, este não possui o recurso de colisão com as paredes, permitindo erroneamente que o usuário “atravessasse” as paredes do campus. Este fato faz com que o ambiente virtual perca o realismo. Outra particularidade interessante é a inclusão de plantas no jardim virtual. Estas plantas, embora fictícias, acrescentam um grau de realismo maior.

O campus virtual foi implementado sem preocupações com performance (SCHULZ, 2004), somente com preocupações com o realismo.

As figuras 28 e 29 mostram o Campus Virtual modelado no trabalho anterior.

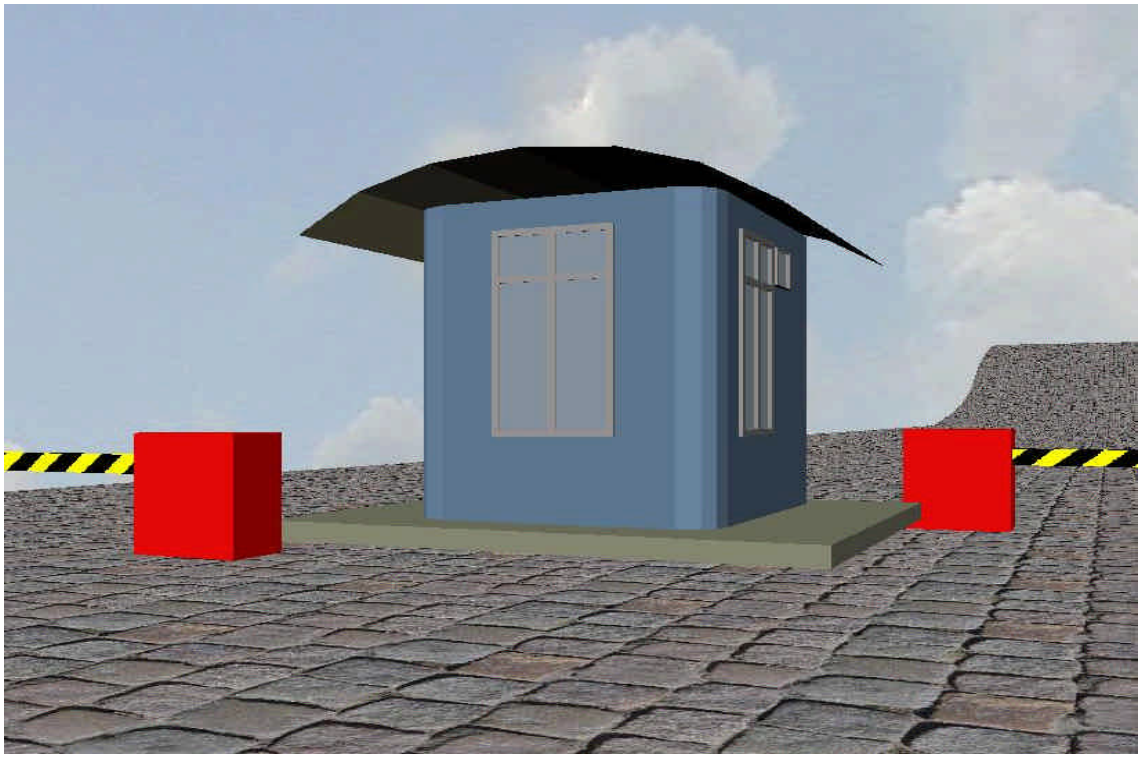

FIGURA 28. Visão da Guarita do Campus Virtual.

FIGURA 29. Visão da Secretaria (Bloco A) do Campus Virtual.

Fez-se um estudo deste projeto e foram detectados os seguintes problemas:

- Não havia a parte da vegetação ao lado da rua (acesso principal ao campus) o que dava uma idéia de inacabado e infinito, prejudicando bastante o realismo e principalmente a navegabilidade;
- A modelagem dos Blocos E e K estavam erradas (ausência de portas e janelas dos referidos blocos);
- Algumas das técnicas citadas no capítulo anterior não são utilizadas em nenhum ponto do Campus Virtual (*LOD* e *Bounding Box* por exemplo);
- Alguns blocos tiveram suas modelagens muito detalhadas desnecessariamente com excesso de realismo em partes quase desapercebidas pelo usuário (partes internas do bloco E);
- Devido a ausência de uma preocupação efetiva com a performance do Campus Virtual, o mesmo está quase que inavegável para usuários possuidores de conexão para *internet* e *hardware* limitado.

5 PROCEDIMENTOS

Otimizou-se o Ambiente CCT Virtual utilizando as duas ferramentas descritas anteriormente (*VRMLPad* e *Fireworks*) em conjunto e/ou isoladamente. Seguem-se as ferramentas utilizadas e as fases demandadas na elaboração do trabalho.

5.1. FERRAMENTAS

Nesta seção, serão apresentadas as ferramentas utilizadas no desenvolvimento e otimização do campus virtual.

5.1.1. VRML Pad 2.0

O *VRMLPad* é um editor de texto com sintaxe para a linguagem VRML. Produzido pela ParallelGraphics. O VRMLPad 2.0 possui diversos recursos que facilitam a programação e construção de cenas em VRML (PARALLEL GRAPHICS, 2002):

- Visualização de comandos que são adequados ao escopo da gramática da linguagem;
- Detecção em tempo real de erros de sintaxe;
- Texto com cores e sintaxe padrão;
- Visualização da hierarquia da cena que está sendo construída;
- Operações básicas sobre os recursos visuais (imagens);
- Mapa de roteamento entre os eventos de entrada e saída;
- Possibilita acrescentar e alterar os comandos do próprio VRMLPad;
- Possibilitar trabalhar com múltiplos documentos ao mesmo tempo;
- Visualiza a cena em vários *plug-ins* diferentes; e

- Organiza e otimiza a cena e suas dependências em formato de publicação na Internet.

O último recurso torna-se bastante interessante para grandes projetos que utilizam uma gama de pontos e polígonos de maneira acentuada. Tais Ambientes Virtuais são impossibilitados de serem publicados na Internet, devido ao seu tamanho resultante. O recurso do VRMLPad de publicação na Internet, compacta o arquivo principal e os dependentes, por eliminar os comentários e a identação do arquivo. Além disso, inclui as imagens *in-line* nos próprios arquivos o que reduz ainda mais a dimensão dos arquivos.

5.1.2. Macromedia Fireworks MX

Trata-se de uma ferramenta para criação e otimização de imagens da WEB através da combinação de vetores e *bitmaps* (MACROMEDIA, 1998). Possibilita otimizar imagens e visualizar como as imagens serão exportadas para comparar formatos e opções de paletas antes de tomar uma decisão final. As funcionalidades de maior interesse são duas: Varinha Mágica e Seleção em Laço. As funções são bastante semelhantes: selecionam uma área da imagem, podendo recortar, excluir e até tratá-la de maneira adequada. No entanto, a diferença resume-se ao critério de seleção. A Varinha Mágica seleciona uma área fechada da figura que possui a mesma tonalidade de cor ao passo que a função Seleção em Laço permite selecionar uma área em formato de polígono determinado pelo usuário. Trata-se de funções importantes, tendo em vista que as texturas serão utilizadas de maneira extensiva.

5.2. RECONHECIMENTO DO LOCAL

Inicialmente foi realizado um reconhecimento específico do CCT real, buscando as informações básicas do local. Embora, conheça-se bem o Campus, foram feitas fotografias com a intenção de armazenar detalhes como a dimensão, altura, largura e posicionamento dos elementos a serem

otimizados. Concluiu-se esta fase em 4 horas.

5.3. MODELAGEM DA VEGETAÇÃO

Nesta fase, verificou-se que o campus virtual anterior não possuía a vegetação ao lado esquerdo, anexo a entrada principal, estando em desconformidade com o campus real. Com isso, preocupou-se com as texturas chamadas replicáveis ou que são repetidas em um objeto virtual. A técnica utilizada na busca do aumento do realismo com a otimização foi a texturização do limite envolvente (Ver fig. 30).

As figuras 31 e 32 mostram o antes e o depois referente ao acesso principal ao campus virtual. Nesta etapa, foi gasta 1 hora.

FIGURA 30. Textura utilizada na modelagem da vegetação.

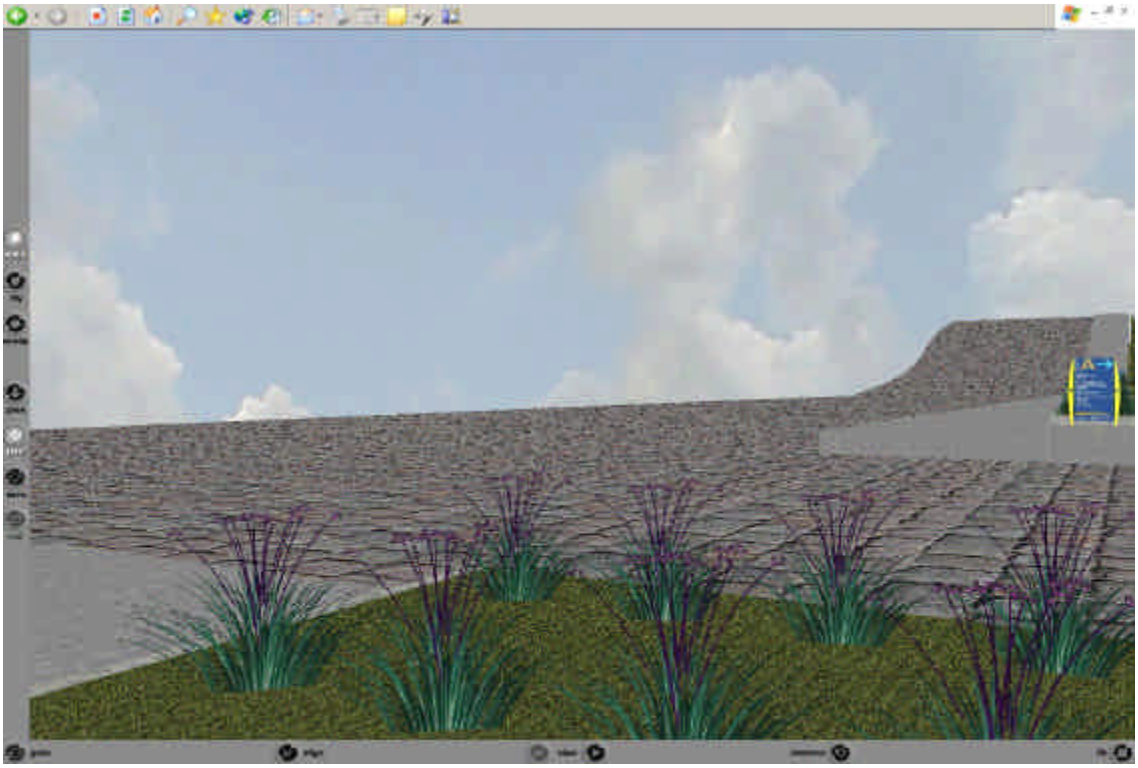

FIGURA 31. Campus antes da correção.

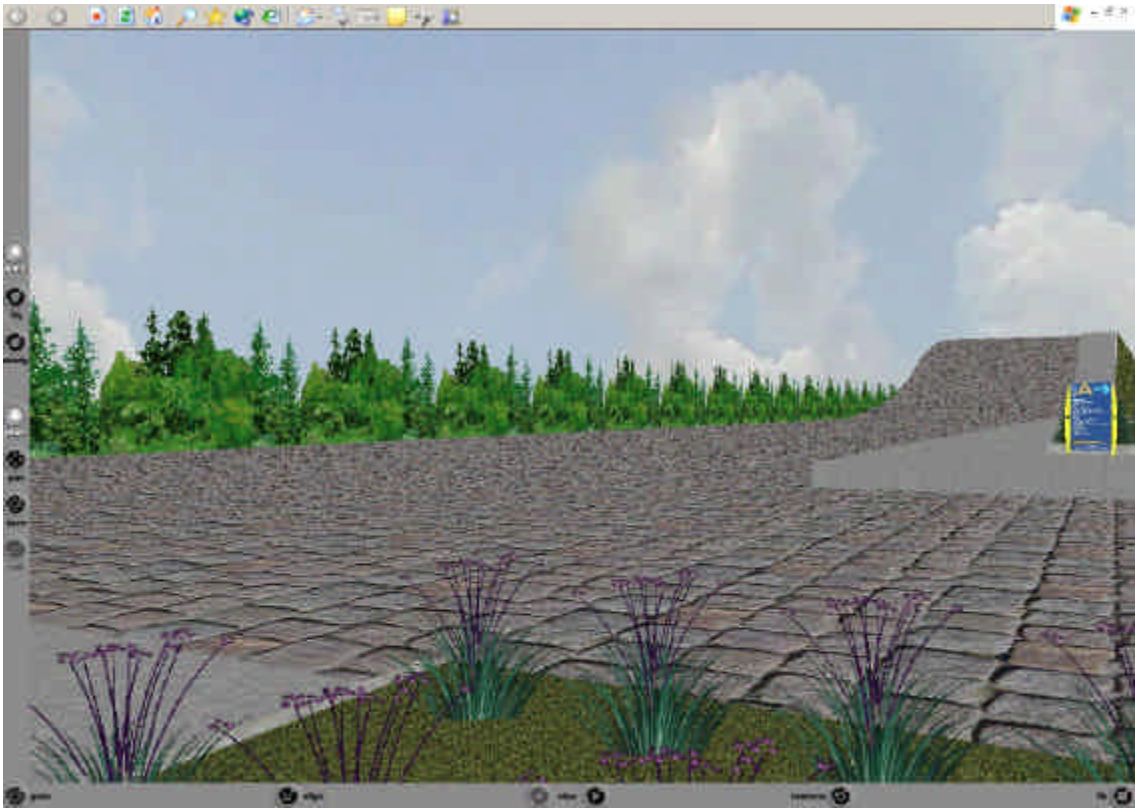

FIGURA 32. Campus após correção.

5.4. BLOCOS “B” E “D”

O próximo passo foi aumentar o realismo e performance nos blocos “B” e “D” do campus virtual. Para isso, foram tiradas fotos das entradas e saídas dos respectivos blocos, com intuito de fornecer ao usuário do sistema a sensação de profundidade dentre os corredores destes dois blocos.

Um processo importante para o aumento do realismo nestes casos foi a edição das fotos tiradas das entradas e saídas dos corredores dos blocos “B” e “D”. Em relação a performance, não podemos trabalhar com arquivos de fotos muito grandes, de alta resolução. Contrastando com isso, o compromisso com o realismo fica melhor evidenciado a partir destas.

Por isso, na edição destas fotos, foi utilizada uma resolução de *64x64 pixels*. Assim, conseguimos aumentar o realismo destes blocos sem afetar a performance. Nesta parte, após a edição das fotos tiradas, foi utilizada a técnica direta de texturização, respectivamente nas entradas e saídas dos corredores destes blocos.

Outro processo muito importante na modelagem dos blocos “B” e “D” foi a colocação de um *box* de transparência total (igual a 1) envolvendo cada bloco, conforme descrito na técnica de *Bounding Box* utilizada como um “campo de força”, com intuito de que o usuário colida com as fotos de entrada e saída dos blocos, não podendo assim caminhar entre os corredores.

O aumento da performance em termos navegacionais nestes blocos foi percebido devido ao grande aumento que o Campus Virtual sofreu em termos de FPS (*Frames Per Second*). Por exemplo na entrada do corredor do Bloco “B”, no campus virtual anterior marcava em torno de 29 FPS. Agora, depois das alterações, está marcando em torno de 70 FPS.

As figuras 33 e 34 mostram respectivamente o antes e o depois referente a entrada do bloco “B” do campus virtual. Este processo foi repetido na saída do bloco “B” e entrada e saída do bloco “D” com suas respectivas fotos reais. Nesta etapa, foram gastas 16 horas.

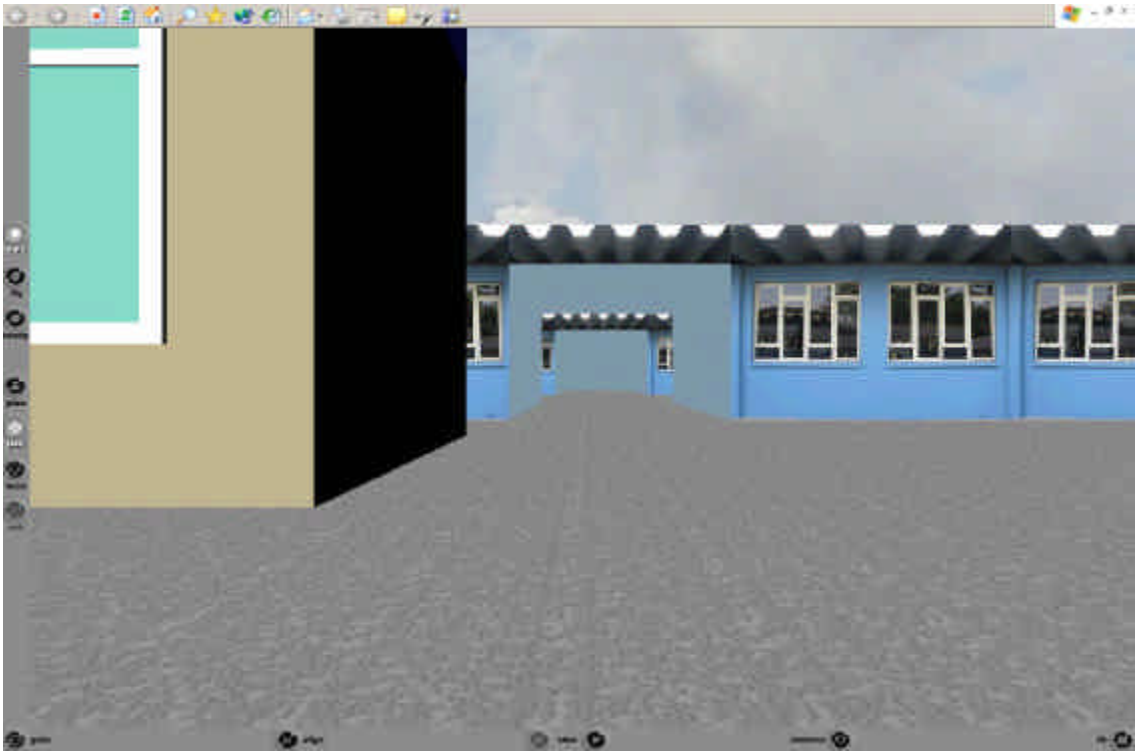

FIGURA 33. Antiga entrada do bloco “B”.

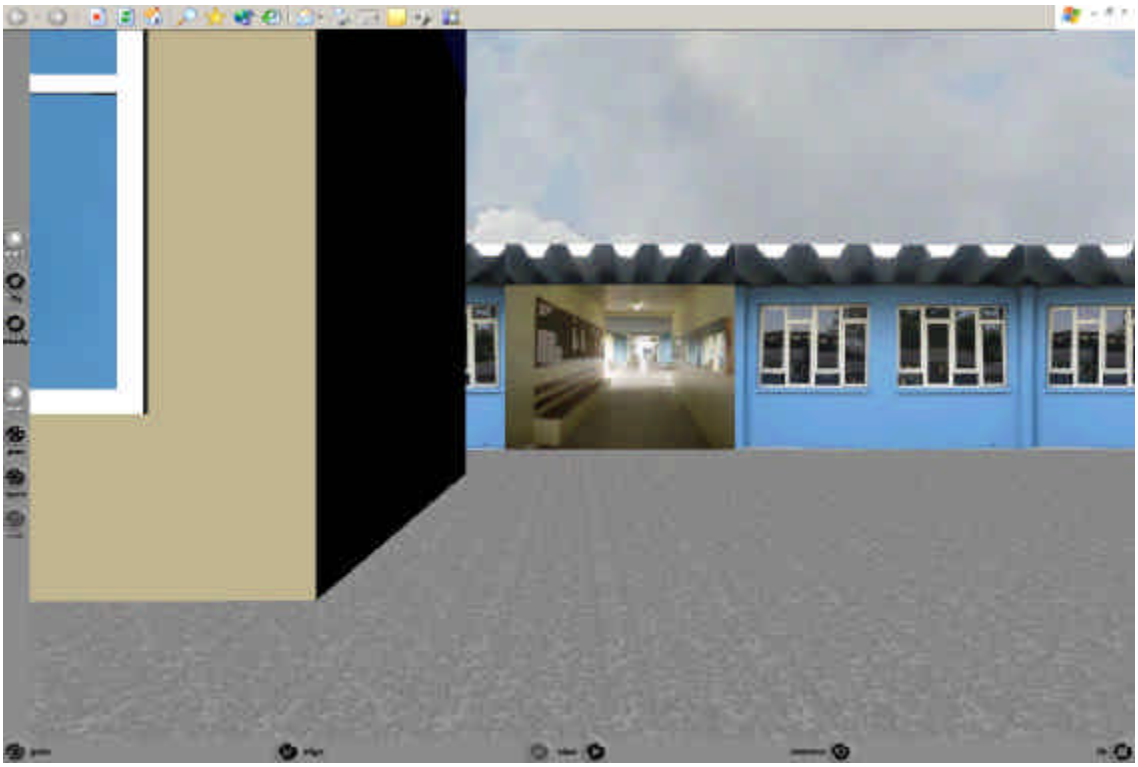

FIGURA 34. Nova entrada do bloco “B”.

5.5. BLOCO “E”

Nesta fase, o objetivo foi aumentar o realismo e performance no bloco “E” do campus virtual. Para isso, foram tiradas fotos das janelas e portas do respectivo bloco, pois no campus virtual antigo, estes objetos não existiam.

A edição das fotos no *Fireworks MX* foi tão importante quanto nos blocos “B” e “D”.

As técnicas implementadas neste bloco foram as mesmas implementadas anteriormente nos blocos “B” e “D”, que são as técnicas diretas de texturização da entrada do corredor interno mais portas e janelas e a aplicação de um *box* utilizando do conceito de *Bounding Box* como um “campo de força”.

O aumento da performance em termos navegacionais neste bloco foi percebido também devido ao grande aumento que o Campus Virtual sofreu em termos de FPS (*Frames per Second*). Por exemplo, bem na entrada do bloco “E”, no campus virtual anterior marcava em torno de 33 FPS. Agora, depois das alterações, está marcando em torno de 69 FPS.

As figuras 35, 36, 37 e 38 mostram respectivamente o antes e o depois referente ao bloco “E” do campus virtual vista de frente e de lado. Nesta etapa, foram gastas 8 horas.

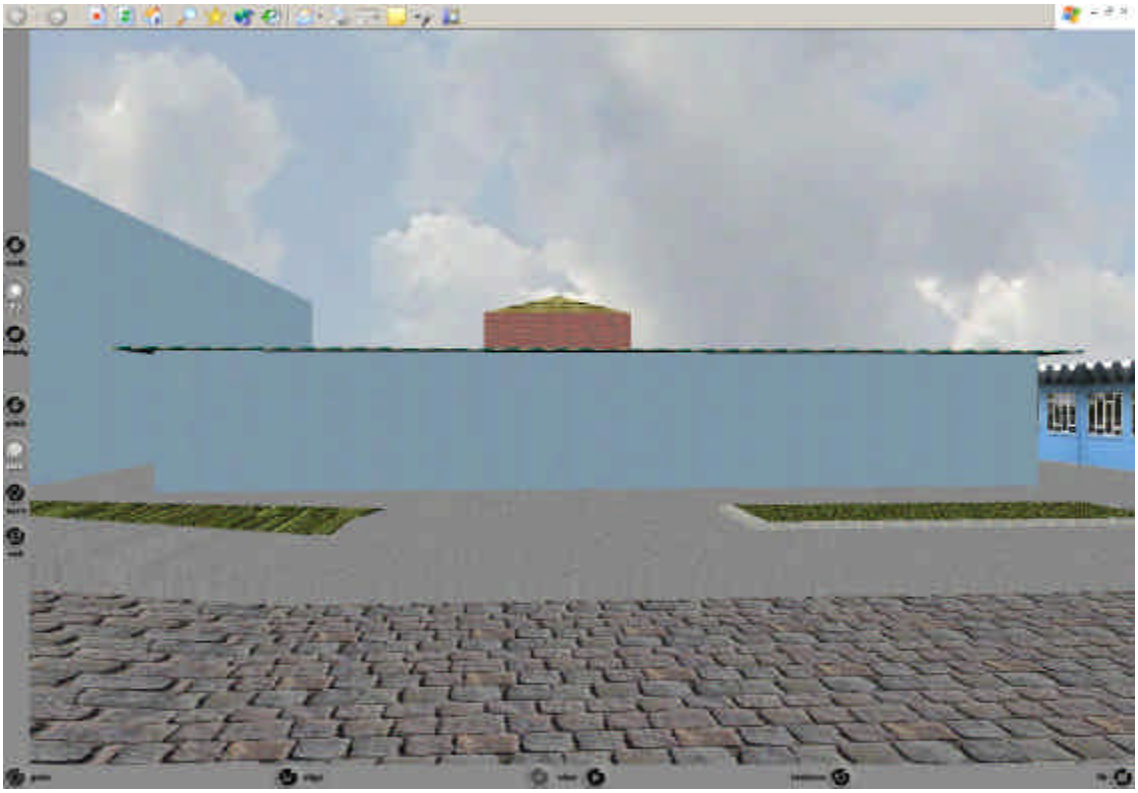

FIGURA 35. Frente antiga do bloco “E”.

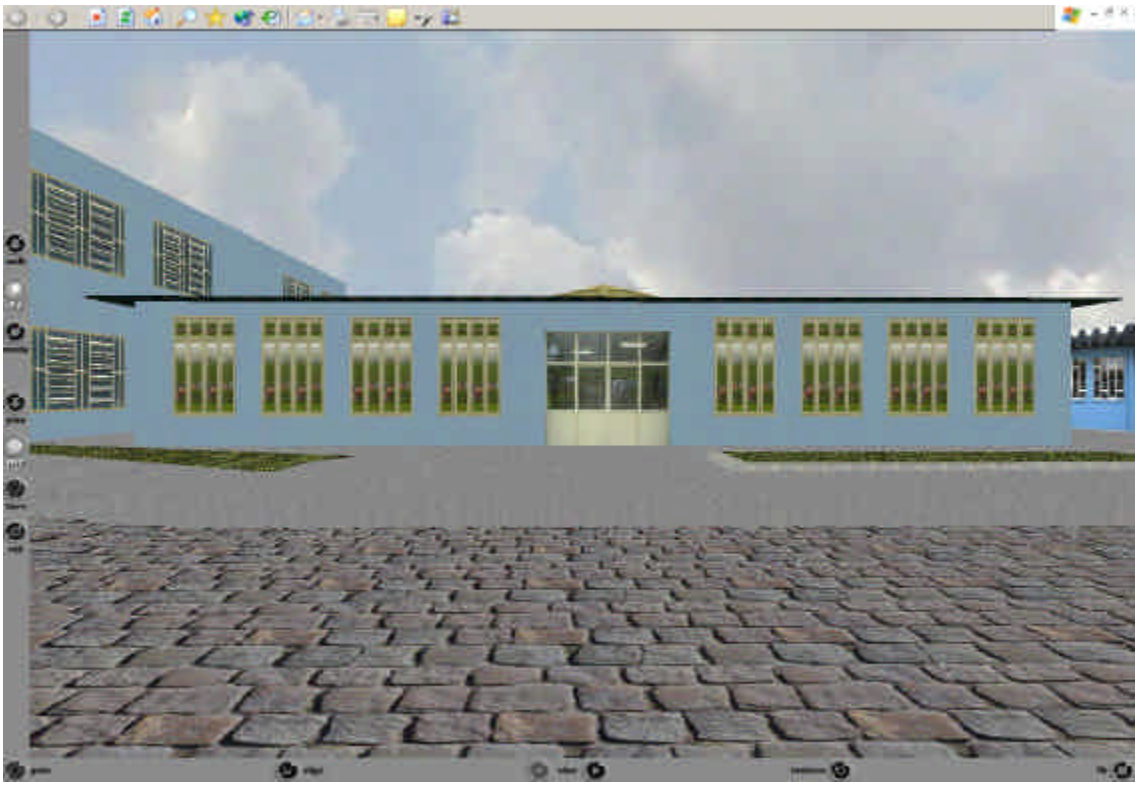

FIGURA 36. Frente nova do bloco “E”.

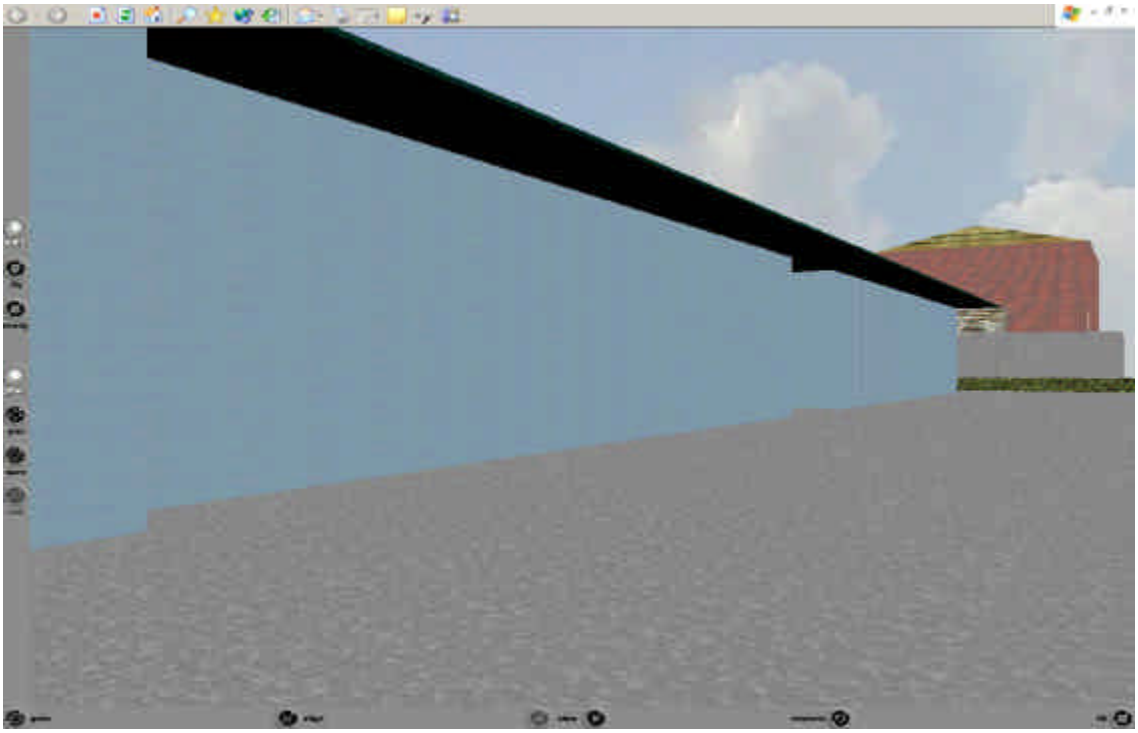

FIGURA 37. Lateral antiga do bloco “E”.

FIGURA 38. Lateral nova bloco “E”.

5.6. BLOCO “K”

O próximo passo foi aumentar o realismo e performance no bloco “K” do campus virtual. Para isso, foram tiradas fotos das janelas do respectivo bloco, pois no campus virtual antigo, estes objetos não existiam.

A técnica implementada neste bloco foi a mesma implementada anteriormente nos blocos “B”, “D” e “E”, que é a técnica diretas de texturização das janelas do referido bloco. A aplicação de um *box* utilizando do conceito de *Bounding Box* como um “campo de força” não foi necessária desta vez pois o bloco “K” virtual não possui nenhum tipo de entrada ou saída que devesse ser bloqueada, fazendo com que o usuário não possa navegá-la entre ela.

O aumento da performance em termos navegacionais neste bloco foi percebido também devido ao grande aumento que o Campus Virtual sofreu em termos de FPS (*Frames Per Second*) conforme já foi citado nos blocos “B”, “D” e “E”. Por exemplo, bem ao lado do bloco “K”, lado este em anexo à rua de acesso principal ao campus, no campus virtual anterior marcava em torno de 22 FPS. Agora, depois das alterações, está marcando em torno de 58 FPS.

As figuras 39 e 40 mostram respectivamente o antes e o depois referente ao bloco “K” do campus virtual vista de frente e de lado. Nesta etapa, foram gastas 4 horas.

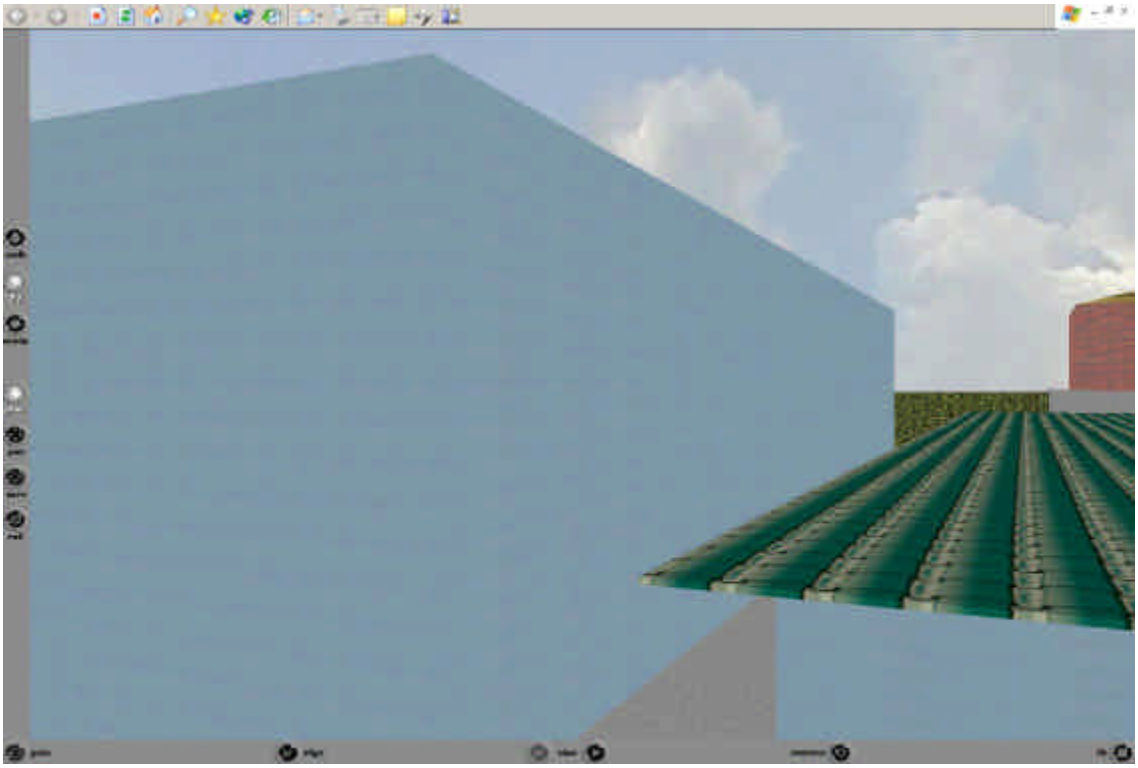

FIGURA 39. Lateral do bloco “K” antes.

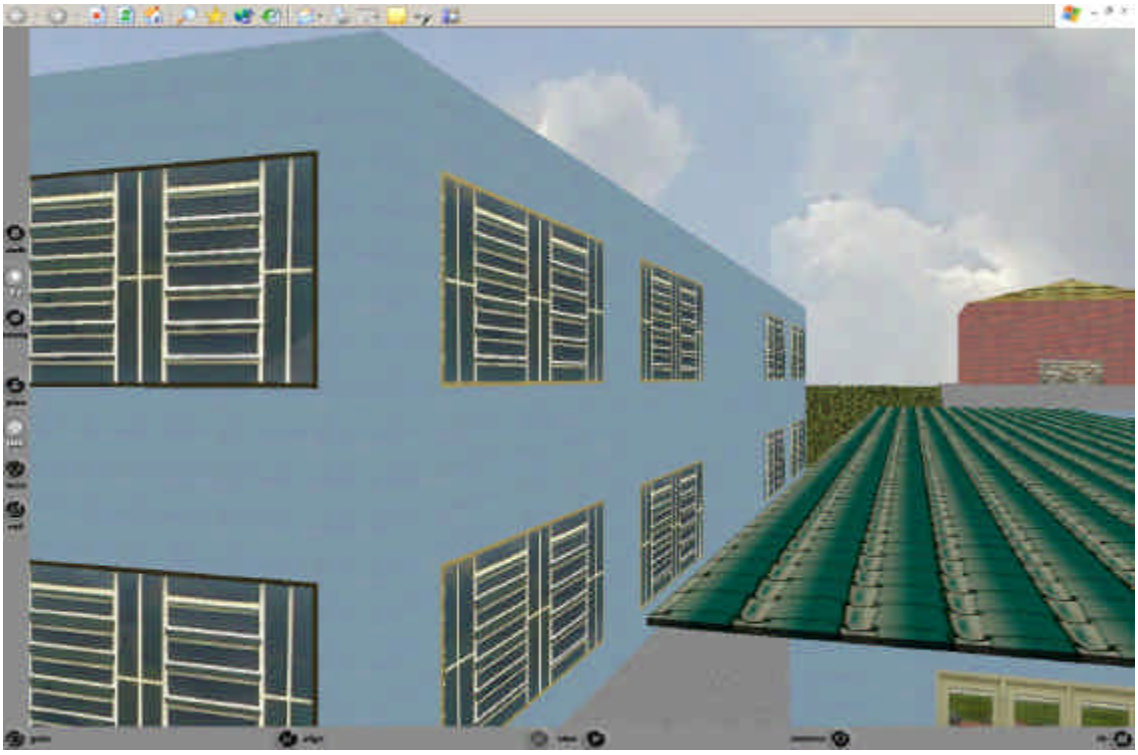

FIGURA 40. Lateral do bloco “K” depois.

5.7. BLOCO “F”

Depois de feitos vários testes, em termos de performance e navegabilidade, no campus virtual antigo percebeu-se que o bloco “F” era o objeto mais significativo neste ponto. Por isso, foi dada neste trabalho de conclusão de curso, uma atenção maior neste bloco comparado aos demais blocos.

Estes testes foram feitos de maneira que mostrassem ao usuário, durante a sua navegação e interação no campus virtual antigo, a quantidade de quadros por segundo no *plug-in* nas proximidades do bloco “F”.

O referido bloco, no trabalho de conclusão de curso anterior (SCHULZ, 2004) foi totalmente modelado na ferramenta **3DS MAX 6** (Ver fig. 41).

A seguir, serão discriminadas todas as técnicas, diretas e compostas, que foram utilizadas na otimização do bloco “F”.

FIGURA 41. Bloco “F” antes, totalmente modelado em 3DS MAX.

5.7.1. Texturização do Bloco “F”

Para o processo de texturização do bloco “F” foram tiradas fotos das janelas grandes e pequenas do respectivo bloco, com intuito de substituir as janelas já existentes, modeladas geometricamente com uma riqueza desnecessária de detalhes pelo aplicativo 3DS MAX. O tamanho da textura utilizada para cada janela do bloco “F” é de somente 3 Kbytes. As fotos foram editadas no Fireworks MX, diminuindo sua resolução para 64x64 *pixels,* salvando-as no formato JPG, com um tamanho final de 72 pontos para cada imagem/textura.

As janelas originais tinham tamanho de 36 pontos, num total de 2.592 pontos para todas as janelas do bloco “F”.

Com a substituição de todas as janelas do bloco “F”, foi notada já uma grande diferença de performance, oriunda da economia de pontos (2.592 para 72 pontos) e do tamanho total dos arquivos (de 1.250 Kbytes para 500 Kbytes), apesar de que este processo, aumentou a quantidade de arquivos (sendo adicionado 72 arquivos).

As figuras 42 e 43 mostram numa posição de trás do bloco “F” como uma visão de antes e depois do referido bloco. Nesta etapa foram gastas 16 horas.

FIGURA 42. Bloco “F” antes da substituição das janelas.

FIGURA 43. Bloco “F” depois da substituição das janelas.

5.7.2. Simplificação da Geometria no Bloco “F”

Depois de aplicada todas as texturas em todas as janelas do bloco “F”, através de testes navegacionais, foi notado um realismo desnecessário aplicado ao corrimão dos corredores do bloco “F”, modelado em forma de cilindros pelo **3DS MAX**.

Para o *browser,* é muito mais dispendioso o carregamento de vários cilindros, devido ao mesmo possuir muitas faces, do que o carregamento de vários cubos, sendo que um cubo possui apenas 6 (seis) faces.

Para não fugir do compromisso com o realismo e melhorar em termos de performance o bloco “F”, foi remodelado todos os corrimões de todos os corredores do referido bloco, trocando as superfícies cilíndricas dos corrimões por superfícies cúbicas.

Após esta alteração, foi percebido um significativo aumento da performance em termos navegacionais neste bloco, oriunda da sensível diminuição na quantidade de faces (2912 para 546 faces). Por exemplo, colocando o usuário do campus virtual bem de frente com o corrimão localizado a frente da sala F308, no campus virtual anterior marcava em torno de 29 FPS. No campus virtual atual, após estas alterações, foi registrada uma média de 60 FPS.

As figuras 44 e 45 mostram de uma posição de frente do corrimão localizado à frente da sala F308 bloco “F”, uma visão de antes e depois do referido bloco. Nesta etapa foram gastas 4 horas.

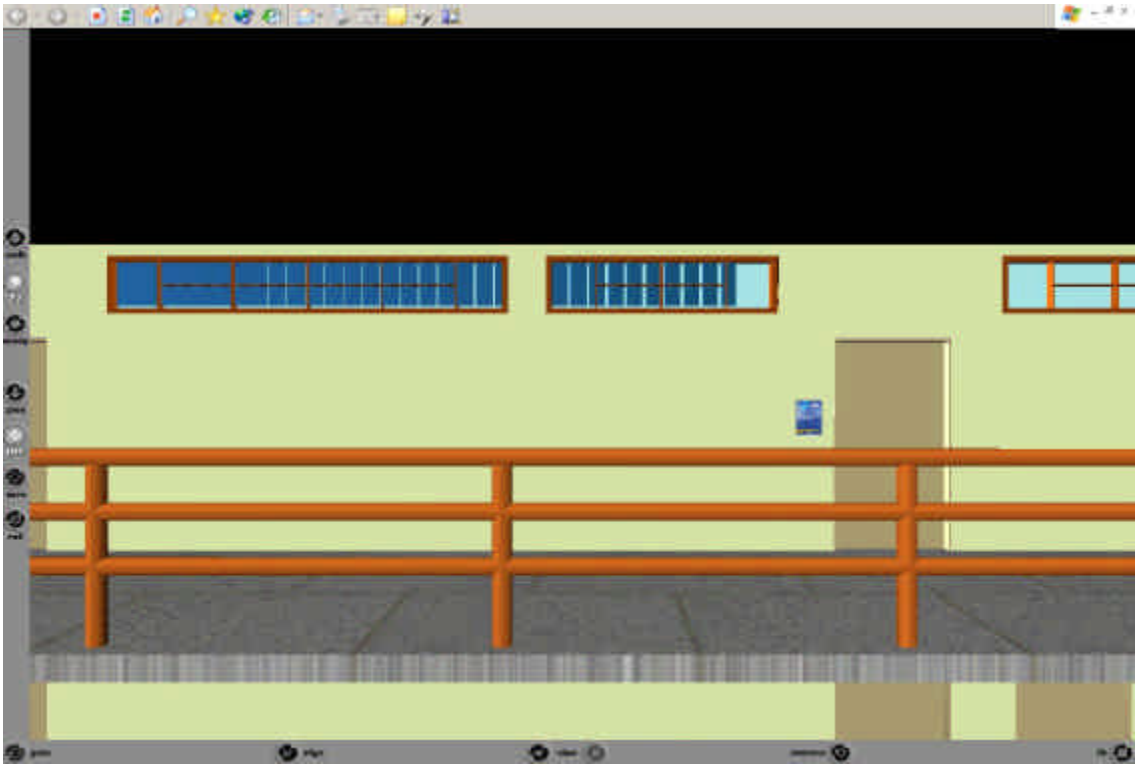

FIGURA 44. Bloco “F” antes da substituição dos corrimões.

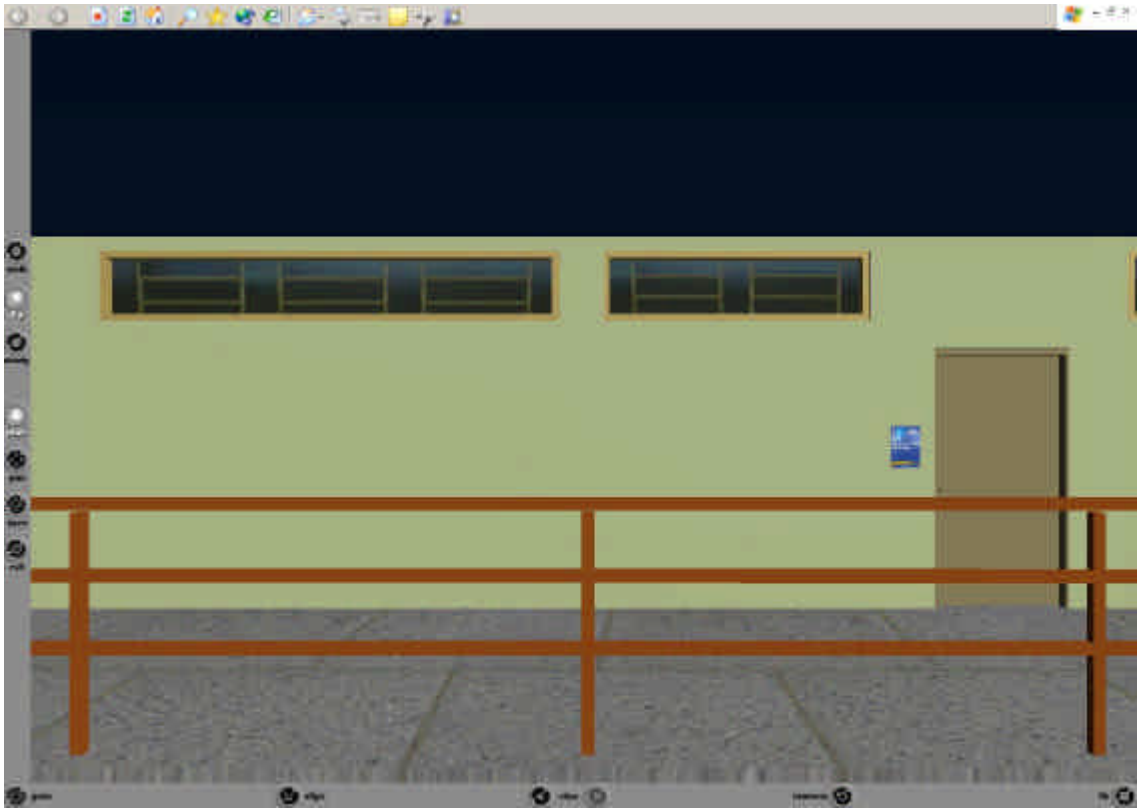

FIGURA 45. Bloco “F” depois da substituição dos corrimões.

5.7.3. “Embrulhando” o Bloco “F”

Como já foi estudado no capítulo 3 deste Trabalho de Conclusão de Curso, “Embrulho” é uma técnica composta de otimização de ambientes virtuais que utiliza as técnicas diretas *Bounding Box,* Textura e *LOD.*

Esta técnica foi aplicada ao bloco “F” devido à necessidade que o Campus Virtual antigo possuía de melhorar em termos de performance, pois a a maior parte deste problema provinha do referido bloco. Por isso, a grande necessidade desta outra customização.

Primeiramente, para uma melhor concepção desta técnica no bloco “F” , utilizando o conceito de *Bounding Box,* foi envolvido o bloco “F” por 3 (três) respectivos planos, um para cada face lateral e outro para a face frontal.

Após da aplicação destas faces, as mesmas foram texturizadas. Mas infelizmente com relação a face frontal, por intermédio de várias barreiras existentes ao redor do bloco (bloco “K”, caixa de água do bloco “E”, árvores) não foi possível tirar uma foto do perfil inteiro do bloco “F” para utilizar na texturização da referida face.

Então, ao invés de utilizar uma foto real do bloco “F”, foi usada uma captura de tela do perfil inteiro do bloco (Ver fig. 46).

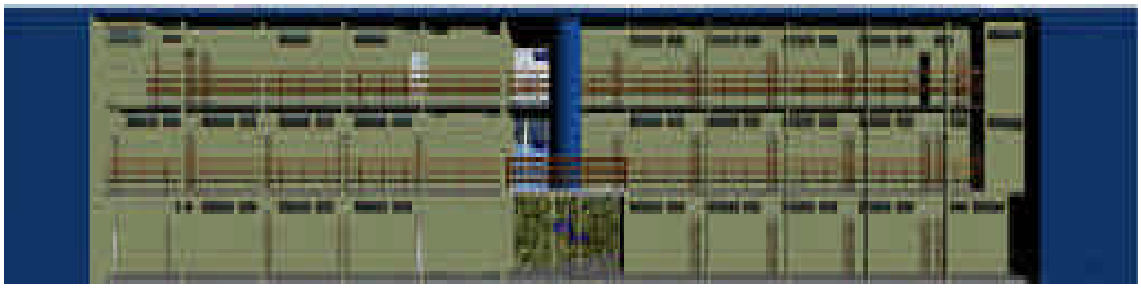

FIGURA 46. Captura de tela frontal do bloco “F”.

Para finalizar a implementação do embrulho no bloco “F”, o mesmo sofreu a última customização, que foi a aplicação do nó LOD em 2 (dois) níveis.

Para a concepção do nó LOD no bloco “F”, no primeiro nível foi utilizado os planos devidamente texturizados, para que o usuário não perceba a

diferença entre a foto e o bloco “F” modelado propriamente dito. Este nível é aplicado a todo o usuário que estiver posicionado num raio de 70 metros de distância do bloco “F”. Com relação ao último nível, nível mais próximo do bloco “F”, ficou o mesmo que já fora modelado anteriormente.

As figuras 47 e 48 mostram a visão dos dois níveis do nó LOD aplicado, um com o usuário situado num raio maior que 70 metros e o outro nível num raio menor que 70 metros de distância do Bloco “F”. Nesta fase foram gastas 16 horas.

FIGURA 47. Nó LOD com o usuário mais afastado.

FIGURA 48. Nó LOD com o usuário mais perto.

6 RESULTADOS

Nesta seção, será feita uma explanação geral dos resultados obtidos através das customizações realizadas no campus virtual e será listado também um conjunto condensado de dicas para que o leitor deste trabalho possa aplicá-las em seus ambientes virtuais.

6.1. REALISMO AUMENTADO

O resultado deste trabalho foi principalmente o aumento da performance do Campus Virtual, além do aumento do no realismo. Um dos fatores que contribuíram para que ocorresse relaciona-se com a inclusão de objetos no ambiente que, na versão anterior, sequer existiam, como por exemplo, as janelas e portas dos blocos “E” e “K”. Estes se apresentam com detalhes, como por exemplo com reflexos, que se aproximam e muito do mundo real.

Outro fator que determinou o aumento do realismo, foram as texturas que foram aplicadas em quase todos os blocos. Estas texturas são fotografias, editadas no Fireworks MX e exportadas para o formato JPG, que permitem que o usuário veja detalhes das portas e janelas inexistentes anteriormente (portas trabalhadas por exemplo).

As texturas que foram colocadas nas entradas e saídas dos corredores dos blocos “B” e “D” (Ver fig. 49 e 50), também são de suma importância para o aumento do realismo no campus virtual, pois as mesmas dão ao usuário que estiver navegando no ambiente uma sensação de “profundidade”.

No processo de aumento do realismo, teve-se sempre a preocupação com a performance quando foi utilizado a técnica de texturização.

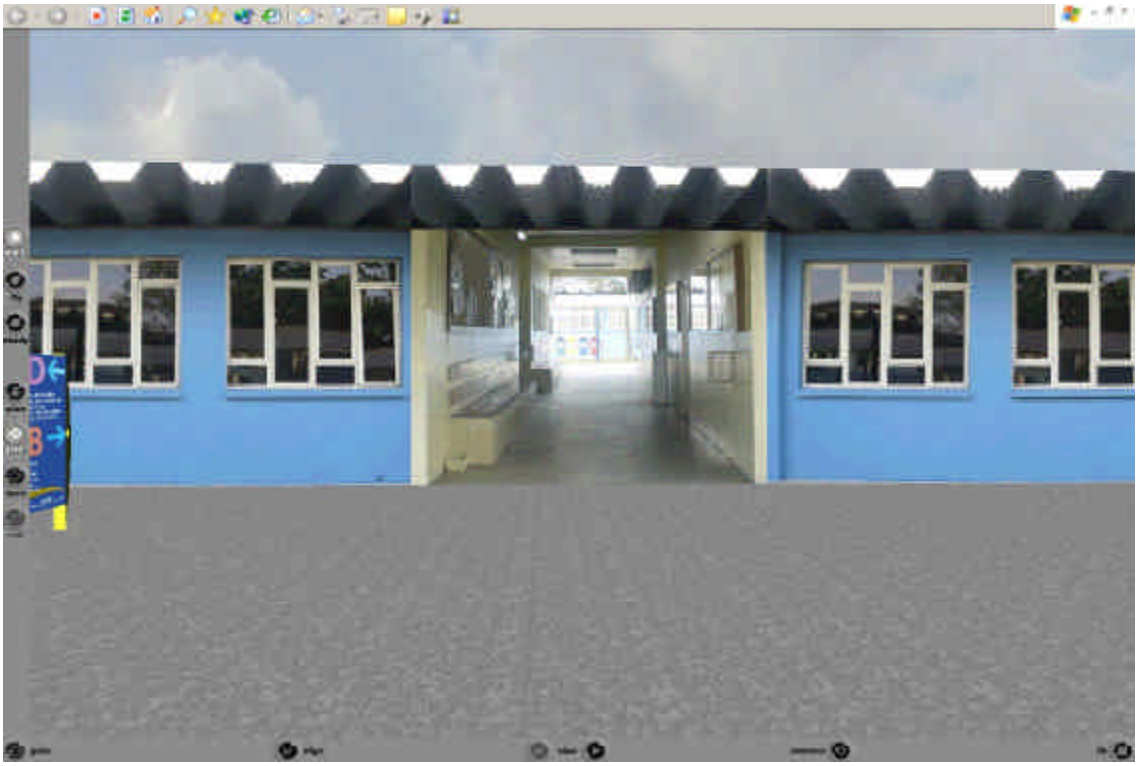

FIGURA 49. Entrada do bloco “D” atual.

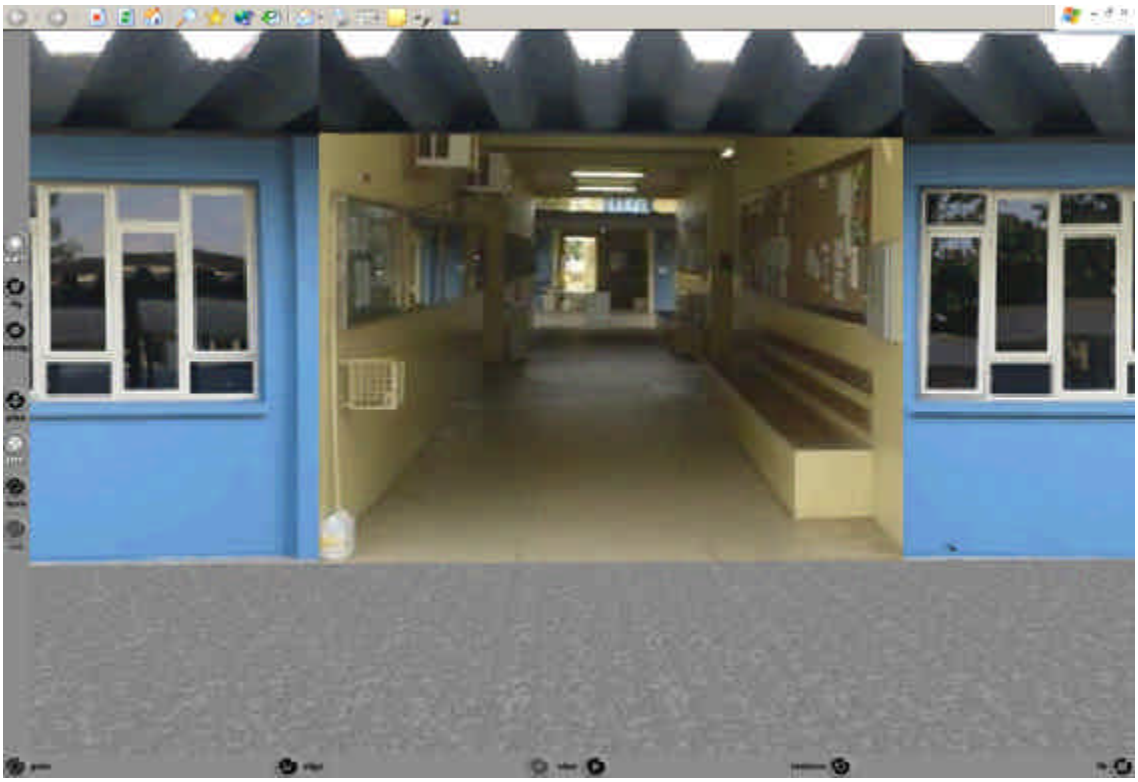

FIGURA 50. Saída do bloco “D” atual.

6.2. PERFORMANCE

Neste trabalho de conclusão de curso foi analisado a performance do novo campus virtual através de três variáveis: Tamanho dos Arquivos, Tempo de Carregamento e Visualização (FPS).

6.2.1. Tamanho dos Arquivos

Com toda a otimização realizada no campus virtual, notou-se que um arquivo em especial teve uma redução significativa no seu tamanho em Kbytes. Este que, no começo dos estudos e pesquisas para a realização deste TCC, era tido como o problema principal do campus, se transformou no foco principal da implementação deste ambiente.

A diminuição no tamanho do arquivo do bloco “F” detectada foi de 40%, pois no campus antigo o mesmo ocupava 1.250 Kbytes. Hoje, no campus virtual atual, o mesmo arquivo ocupa agora 500 Kbytes. Houve também uma diminuição significativa no tamanho de todos os arquivos “.WRL” juntos da aplicação. O rendimento calculado dessa vez foi de 55,56 %, pois os arquivos do campus antigo somados ocupavam 1.440 Kbytes e atualmente ocupam 800 kbytes.

Na figura 51 mostra com melhor clareza os números citados acima.

	Antes (Kbytes)	Depois (Kbytes)	Percentual (%)
Bloco “F”	1250	500	60.00
Todos	1440	800	44.45

FIGURA 51. Rendimento em termos de tamanho dos arquivos.

6.2.2. Tempo de Carregamento do Campus

Para medir o rendimento do novo campus virtual, foram obtidas cinco

amostras de cronometragem, calculando quanto tempo cada campus virtual demorava para carregar totalmente.

A figura 52 mostra os valores retirados da cronometragem do carregamento de cada campus virtual, o antigo e o novo.

Depois (s)	Antes (s)	Percentual (%)
2.18	4.24	51.42
2.25	4.44	50.68
2.21	4.07	54.23
1.98	4.73	41.86
1.91	4.26	44.84
Média = 2.106	Média = 4.384	Média = 48.606

FIGURA 52. Rendimento em termos de carregamento do campus.

Nesta tabela, mostra que o rendimento médio de carregamento do campus virtual novo em comparação ao antigo é de 48,606%.

6.2.3. Visualização

Para medir o rendimento do novo campus virtual, em termos de visualização e navegabilidade, escolheu-se 3 (três) posições fundamentais para que seja realizada esta medição em termos de quantidade de FPS (*Frames Per Second*). Estas posições são: posição inicial do usuário logo após o carregamento total do ambiente virtual, posição entre os blocos “B” e “D” e finalmente, logo a frente do bloco “F”.

A figura 53 mostra os valores retirados da demonstração da quantidade de FPS de cada posição citada acima.

	Quant. de FPS Antes	Quant. de FPS Depois
Posição Inicial	29	71
Entre Blocos “B” e “D”	13	44
Frente Bloco “F”	11	34

FIGURA 53. Rendimento em termos de visualização do campus.

Nesta tabela, demonstra que o rendimento em termos de visualização e navegabilidade foram ótimos, principalmente no ponto inicial do Campus Virtual. Essa diferença tão grande demonstrada na tabela acima, ocorre devido principalmente a aplicação de Técnica Composta Embrulho no bloco “F”. Com esta técnica aplicada ao bloco referido, o *browser* entende que, a priori, não há necessidade do carregamento total do bloco, inicialmente só carrega os planos devidamente texturizados.

6.3. RESUMO DAS IMPLEMENTAÇÕES

Na figura 54, mostra um resumo de todas as Técnicas Diretas que anteriormente já existiam e também as que foram acrescentadas agora no Campus Virtual neste Trabalho de Conclusão de Curso.

Técnicas	Já Tinha / Acrescentada	Locais no Campus Virtual
Inline	Já Tinha	Blocos A, B, D, E, F, Ginásio e Biblioteca
	Acrescentada	Bloco K
Texturas	Já Tinha	Blocos A, F
	Acrescentada	Blocos B, D, E e F
Det. Colisões	Já Tinha	Blocos A, F, Ginásio e Biblioteca
	Acrescentada	Blocos B, D e E
Billboard	Já Tinha	Árvore do Bloco A.
	Acrescentada	Nenhum Local
Bonding Box (a) = Real	Já Tinha	Nenhum Local
	Acrescentada	Nenhum Local
Bonding Box (b) = “Campo de Força”	Já Tinha	Nenhum Local
	Acrescentada	Blocos B, D e E
Simpl. Geometria	Já Tinha	Nenhum Local
	Acrescentada	Bloco F
LOD	Já Tinha	Nenhum Local
	Acrescentada	Bloco F
Âncora	Já Tinha	Nenhum Local
	Acrescentada	Nenhum Local

FIGURA 54. Técnicas já existentes e acrescentadas.

7 DIRETRIZES (DICAS DE IMPLEMENTAÇÃO)

Construir um ambiente VRML extenso e otimizado consiste em atravessar uma série de etapas ciclicamente. Estes ciclos devem seguir uma determinada ordem e ser repetidos sempre que necessário. Na prática, são descartadas várias etapas, especialmente após a primeira vez que o ciclo é percorrido. De acordo com MIRANDA, 1999, este planejamento pode envolver as seguintes etapas principais:

1. Definir qual o plano do projeto;
2. Criar objetos com base em primitivas 3D e com o menor número de polígonos possível;
3. Trabalhar na aparência dos objetos, adicionando para o efeito materiais e texturas (reduzidas) para criar o efeito de superfícies complexas;
4. Criar animações (não muito pesadas);
5. Atribuir algum realismo à cena virtual, através de luzes, nevoeiro, *backgrounds*, etc.
6. Adicionar ao modelo informação de navegação;
7. Testar o modelo em várias máquinas, de modo a poder ser avaliada a sua eficiência e se necessário voltar às etapas anteriores para otimização.
8. Colocar todos os arquivos utilizados na criação do modelo num servidor *web*, de modo a este ficar acessível na rede.

Como fora visto no capítulo 3, algumas das técnicas estudas tem como base a utilização de texturas. Um dos principais fatores a considerar na criação de uma textura é o seu tamanho. O número de *pixels* de uma textura pode afetar muitos aspectos de uma cena VRML, como, por exemplo, a eficiência do modelo. O primeiro e mais óbvio efeito que o tamanho de uma textura pode provocar num mundo VRML é o tempo de *download* da cena. Uma textura grande implica um arquivo VRML maior e, consequentemente, um maior tempo de *download*.

Uma outra grande questão reside na forma como as texturas são processadas pelo *browser* VRML. Segundo (BALLREICH, 1997), uma textura deve obedecer a algumas regras, de modo a não criar problemas quando é efetuado o *rendering* da cena:

- Ser o menor possível;
- Ser quadrada (64x64, 128x128);
- A sua resolução ser uma potência de dois (16, 32, 64, 128, 256).

Algumas considerações para produzir arquivos VRML pequenos com *renderings* rápidos são (MIRANDA, 1999):

- Medir frequentemente a eficiência do mundo em diferentes plataformas. Deve existir sempre um compromisso entre os conteúdos da cena e a eficiência;
- Usar o menor número de polígonos possíveis;
- Recorrer a textura, materiais e luzes para criar o efeito de superfícies mais complexas, sem usar muitos polígonos;
- Estruturar cuidadosamente a hierarquia dos objetos que compõem a cena, de forma a permitir que os *browser*s a visualizem rapidamente.
- Reutilizar texturas e sons sempre que possível, e com tamanhos o mais reduzidos possível;
- Aplicar as características do VRML que ajudam a aumentar a eficiência, como por exemplo, Níveis de Detalhe (LOD – *Level Of Detail*), *Billboards*, *Bounding Box, Inlines* e Detecção de Colisões.

Claramente, percebe-se que objetos mais distantes podem ter imagens menores e menos detalhadas ao passo que objetos mais próximos precisam de imagens mais detalhadas e maiores. Identificar e, quando possível, controlar o quanto o usuário chega perto de uma textura mapeada é um bom exercício de tentativa e erro que impacta diretamente nas performances de carregamento e de visualização do Ambiente Virtual.

O VRML possui uma série de outras características (citadas no capítulo 3) que, bem utilizadas, podem contribuir para o aumento da eficiência de um ambiente e contrariar, de algum modo, as ditas limitações. Contudo, o usuário

de um sistema com poucos recursos vai sempre encontrar alguns problemas para manter alguma *performance* na visualização de um mundo VRML.

7.1. DIRETRIZES PROPOSTAS

Após a pesquisa das técnicas de otimização e a implementação das mesmas no Campus Virtual, segue logo abaixo as diretrizes (dicas) propostas por este Trabalho de Conclusão de Curso para os leitores deste Trabalho de Conclusão de Curso, interessados em implementar Ambientes Virtuais do mesmo tipo que o Campus Virtual:

1. Criar objetos simples em termos de geometria e que sejam passíveis a serem reutilizados;
2. Sempre que possível, preferir texturas mapeadas ao invés de geometrias complexas, pois muitas vezes, o usuário nunca conseguirá diferenciar;
3. Usar intensivamente *Inlines*, Texturas, ou seja, todas as técnicas diretas ou compostas citada no capítulo 3;
4. Avaliar o uso do *Bounding Box* para conjuntos de objetos complexos;
5. Caso tenha-se que trabalhar com ambientes extensos e de interiores, a ancoragem é um recurso muito importante a se considerar.
6. Existe um recurso do *VRMLpad* que tem o intuito de gerar uma versão "condensada" do arquivo “.wrl”. Isso diminui sensivelmente o tamanho pois elimina todos os comentários e espaços em branco colocados pela identação (que são importantes para visualização e entendimento mas não fazem diferença para o *browser*).

8 CONCLUSÕES

Os recursos e técnicas apresentadas no capítulo 3, alguns simplesmente apresentados como comandos em algumas linguagens aqui são encaradas pelo ponto de vista do seu benefício, qual seja, o de otimizar a performance dos ambientes e isto se torna mais evidente para Ambientes Extensos.

O contexto das técnicas de otimização de ambientes virtuais pesquisadas neste trabalho de conclusão de curso é baseado totalmente em VRML 2.0 puro, ou seja, está se desconsiderando qualquer extensão de algum *plug-in* específico. Basicamente, resgataram-se comandos já existentes no VRML, que normalmente os desenvolvedores acabam menosprezando devido a necessidade de resultados rápidos.

Nem todas as técnicas foram experimentadas em termos de código neste trabalho. Algumas não puderam ser usadas, enquanto outras foram propositalmente deixadas de fora da experimentação (como a segmentação) devido às limitações do tempo.

A aplicação das técnicas efetivamente experimentadas, apesar de não representar todo o escopo de possibilidades, já foi o suficiente entretanto para produzir um Campus Virtual otimizado tanto no tempo de carregamento (dados em %) quanto na visualização (dados em %) bem como no seu realismo (o qual nem era o objetivo inicial do trabalho).

Todo e qualquer ambiente virtual tem um compromisso concorrente entre realisto, performance de carregamento e de visualização, onde cada uma, caminha para lados opostos (Ver fig. 55).

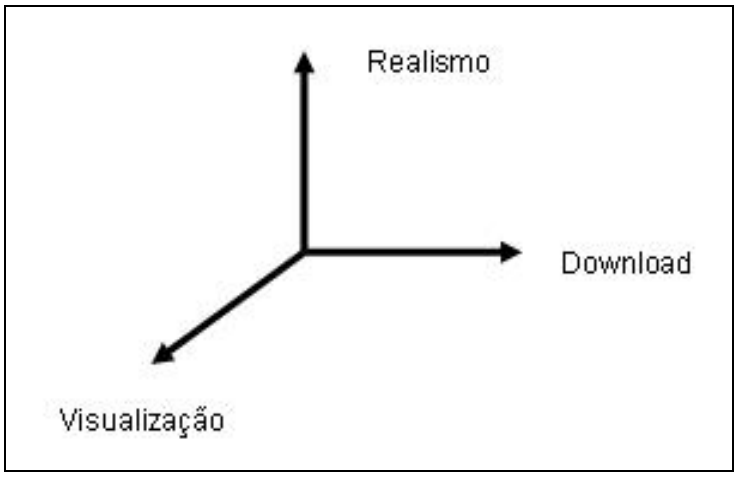

FIGURA 55. Compromissos dos ambientes virtuais.

Algumas dificuldades e problemas foram encontrados ao longo processo de otimização do CCT Virtual. Uma das dificuldades relaciona-se às fotografias dos blocos que pensou-se em utilizar como texturas. No entanto, as fotografias foram obtidas em partes e no mesmo sentido, porém, não na mesma posição. Este fato impossibilita de unir as várias fotos para formar uma textura única. Por isso, não foi modelado o LOD do bloco ”F” como era desejado, através de uma foto panorâmica do bloco. Como fora relatado no capítulo anterior, ao invés de utilizar uma fotografia real do bloco “F” no nível mais externo do nó LOD, foi utilizado uma captura de tela do bloco “F”virtual.

Um dos problemas de visualização resultantes do ambiente virtual refere-se à imensa quantidade de informações visuais, principalmente do bloco F. Notou-se que o Bloco F sem as partes modeladas internas das salas, obtém um rendimento em termos de performance ainda melhor. Tentou-se otimizar o bloco “F” neste sentido através da aplicação de Âncoras mas sem sucesso. Este é um problema que em trabalhos futuros deverá ser estudado.

Outra conclusão extraída da pesquisa e implementação deste Trabalho de Conclusão de Curso foi que, se tivesse tempo hábil, todos os objetos do Campus Virtual, anteriormente modelados através da ferramenta 3DS Max, seriam remodelados totalmente em VRML, deixando de lado assim, o excesso de geometrias e realismos (conforme foi modelado o bloco “F” na versão anterior do Campus Virtual), de vez em quando, desnecessários.

Acredita-se que este trabalho alcançou seus objetivos, geral e específicos, principalmente no que concerne à valorização da tarefa de otimização de ambientes virtuais extensos catalogando também uma série de dicas ou diretrizes que podem ser seguidas em futuras implementações de ambientes assemelhados.

8.1. TRABALHOS FUTUROS

Durante o processo de modelagem, identificou-se alguns requisitos para tornar o campus virtual com mais útil para os usuários. Estes estão identificados abaixo:

- Acesso a informações em banco de dados. Estas informações poderão ser de cunho acadêmico como médias dos alunos, históricos escolares, notas de provas, reservas de equipamentos audiovisuais (canhão e *notebook*), acesso ao sistema da biblioteca, agenda de eventos do auditório, entre outros.
- Disponibilizar informações históricas da instituição e suas dependências (departamentos, ginásio, biblioteca entre outros).
- Permitir acesso a notícias diárias sobre a instituição, bem como notas e avisos provenientes dos departamentos.
- Disponibilizar *outdoors* virtuais pelo ambiente com informações publicitárias como imobiliárias, bancos e parceiros da instituição.

Com a finalidade de tornar o ambiente mais agradável, segue abaixo algumas sugestões:

- Mais plantas: O ambiente virtual possui poucas plantas e árvores.
- Objetos animados: O campus virtual poderá ser incrementado com a inclusão de animações tridimensionais como telefone que balança quando toca, guia e bússola virtual.
- Uso mais intensivo de recurso de âncoras a fim de possibilitar a divisão do todo do modelo, entre partes externas e internas, quando possível.

REFERÊNCIAS

AMES, Andrea L.; NADEAU, David R.; MORELAND, John L. **VRML 2.0 Sourcebook**. 2nd ed. New York: John Wiley, 1997. 654 p.

CAREY, R.; BELL, G. **The Annotated VRML 2.0 reference manual.** Reading, Mass.: Addison-Wesley Developers Press, 1997. p. 501. ISBN 0201419742.

CORSEUIL, Eduardo; RAPOSO, Alberto; SILVA, Romano; PINTO, Márcio; WAGNER, Gustavo; GATTASS, Marcelo. **A VR Tool for the Visualization of CAD Models,** TeCGraf/PUC-Rio – Computer Graphics Group (2004).

DISCREET. **3DS MAX 6: Features & Benefits.** Disponível em http://www4.discreet.com/3dsmax/3dsmax.php?id=767. Acesso em: 3 julho 2004.

EASTGATE, Richard. **The Structured Development of Virtual Environments. Enhancing Functionality and Interactivity.** Tese de Doutorado, 2001.

FERREIRA, Glauber; LOUREIRO, Emerson; NOGUEIRA, Wallace; FERREIRA, André; ALMEIDA, Hyggo; FERRY, Alejandro. **Uma Abordagem Baseada em Componentes para a Construção de Edifícios Virtuais.** Universidade Federal de Campina Grande – Campina Grande, PB – 2004.

FRANCIS, G. A.; TAN, H.S. **Virtual Reality as a Training Instrument**. The Temasek Journal, Vol. 7. pp. 4 –15, 1999.

HARTMAN, J.; WERNECKE, J. **The VRML 2.0 handbook: building moving worlds on the web.** Reading, Mass.: Addison-Wesley, c1996. 412 p. ISBN 0201479443 (Broch.)

HOUNSELL, M. da S.; PIMENTEL, A. **On the Use of Virtual Reality to Teach Robotics**, 3rd International Conference on Engineering and Computer Education, São Paulo: 2003.

ISO/IEC 14772. **VRML97 Functional specification and VRML97 External Authoring Interface (EAI).** International Standard ISO/IEC 14772-

1:1997 and ISO/IEC 14772-2:2004.

KIRNER, C. Dr. **Sistemas de Realidade Virtual.** Disponível em http://www.dc.ufscar.br/~grv/tutrv/tutrv.htm. Acesso em: 4 julho 2004.

MIRANDA, José Carlos. **Urbanismo e Espaços Virtuais – Divulgação e Discussão na Comunidade.** Tese de Mestrado, Universidade do Porto, 1999.

MACROMEDIA. **Macromedia Fireworks: Guia do Currículo.** 1998. Disponível em http://www.eletricazine.hpg.ig.com.br/apostilas/internet/fireworks.pdf. Acesso em 2 julho 2004.

NETTO, A. V.; MACHADO, L. dos S.; OLIVEIRA, M. C. F. de. **Realidade virtual: fundamentos e aplicações**. Florianópolis: Visual Books, 2002. 94 p. ISBN 857502082X (broch.).

PARALLELGRAPHICS. **VRMLPad 2.0: General Overview.** 2002. Disponível em http://www.parallelgraphics.com/l2/bin/vrmlpad_features.pdf. Acesso em: 2 julho 2004.

PREECE, Jenny. **Human Computer Interaction**. Addison-Wesley, Essex, England, 1994.

REBELO, Irla B. **Proposta de uma Ferramenta de Verificação dos Procedimentos de Interação em Sistemas de Realidade Virtual,** Tese de Doutorado, Universidade Federal de Santa Catarina, 2004.

REDEL, Rubens. **Segmentação de Mundos Virtuais: Estudo de Caso com o Cemitério Virtual dos Imigrantes de Joinville,** Trabalho de Conclusão de Curso, Universidade do Estado de Santa Catarina, 2004.

RODRIGUES, Luciene; SEMENTILLE, Antônio; BREGA, José; RODELLO, Ildeberto. **Particionamento de Ambientes Virtuais de Grande Escala em células hexagonais associadas a grupos Multicast,** Programa de Pós-Graduação em Ciência da Computação (PPGCC), Centro Universitário Eurípides de Marília (UNIVEM) (2004).

SILVA, R. J. M; WAGNER, G. N.; RAPOSO, A. B.; GATTASS, M. **Experiência de Portais em Ambientes Arquitetônicos Virtuais**, IV Symposium on Virtual Reality, São Paulo: 2003.

SCHULZ, Alessandro P. **Modelagem de Exteriores Extensos: Estudo de Caso com o Campus Virtual do CCT de Joinville,** Trabalho de Conclusão de Curso, Universidade do Estado de Santa Catarina, 2004.

Székely, Gábor; Satava, Richard. **Virtual Realityin Medicine**. BMJ (www.bmj.com), v. 319, p. 1-4, 13 de novembro de 1999.

VINCE, J. **Essential virtual reality fast: how to understand the techniques and potential of virtual reality.** Berlin; Springer, 1998. 174 p. ISBN 1852330120.

WONG, R.; YEUNG, C.; KAN, C. **A Virtual Campus Kiosk**. Departamento de Arquitetura da Universidade de Hong Kong, 2002.

ANEXOS

ANEXO A: FOTO DA PLANTA BAIXA DO CCT JOINVILLE.