

# **REALIDADE VIRTUAL E ENGENHARIA CIVIL: DETECÇÃO DE INTERFERÊNCIAS ENTRE PROJETOS DE EDIFICAÇÕES EM 3D**

Dissertação apresentada ao  
Programa de Pós-Graduação em  
Engenharia de Produção da  
Universidade Federal de Santa Catarina  
como requisito parcial para obtenção do grau  
de Mestre em Engenharia de Produção.

Orientador: Fabiano Luiz Santos Garcia, Dr.

Florianópolis  
2004

## **FICHA CATALOGRÁFICA**

C355r Castro, Tales Augusto Liguori de  
Realidade virtual e engenharia civil : detecção de interferências entre projetos de edificações em 3D / Tales Augusto Liguori de Castro; orientador Fabiano Luiz Santos Garcia. – Florianópolis, 2004.  
62 f. : il.

Dissertação (Mestrado) – Universidade Federal de Santa Catarina, Programa de Pós-Graduação em Engenharia de Produção, 2004.

Inclui bibliografia.

1. Realidade virtual. 2. Engenharia civil. 3. Projeto auxiliado por computador. 4. Detecção de colisão. I. Garcia, Fabiano Luiz Santos. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Produção. III. Título.

CDU: 624

Tales Augusto Liguori de Castro

REALIDADE VIRTUAL E ENGENHARIA CIVIL:  
DETECÇÃO DE INTERFERÊNCIAS ENTRE  
PROJETOS DE EDIFICAÇÕES EM 3D

Esta dissertação foi julgada e aprovada para a obtenção do grau de **Mestre em Engenharia de Produção** no **Programa de Pós-Graduação em Engenharia de Produção** da Universidade Federal de Santa Catarina.

Florianópolis, 20 de Julho de 2004.

---

Prof. Edson Pacheco Paladini, Dr.  
Coordenador do Programa

**BANCA EXAMINADORA:**

---

Prof. Fabiano Luiz Santos Garcia, Dr.  
(Orientador)

---

Prof. Marcelo da Silva Hounsell, Ph.D.

---

Prof. Alejandro Martins Rodriguez, Dr.

---

Prof. Onivaldo Rosa Júnior, MSc.

---

Rui Luiz Gonçalves, Esp.

## AGRADECIMENTOS

Aos meus pais, que, mesmo com a distância, sempre me apoiaram integralmente em mais este desafio;

Ao meu orientador, Fabiano, acima de tudo pela paciência e também pela disponibilidade durante toda a duração do mestrado;

À AltoQI, principalmente ao Rui Gonçalves e André Gustavo, pela parceria firmada e confiança depositada no meu trabalho;

Ao MOC, pela amizade e ensinamentos de programação, que foram fundamentais em situações críticas no decorrer do projeto;

À todos do LRV/UFSC que de alguma forma deram sua contribuição para este trabalho.

## RESUMO

Castro, Tales Augusto Liguori de. **Realidade Virtual e Engenharia Civil: Detecção de Interferências entre Projetos de Edificações em 3D**. 2004. 62 f. Dissertação (Mestrado em Engenharia de Produção) – Programa de Pós-Graduação em Engenharia de Produção, UFSC, Florianópolis, 2004.

A engenharia civil é campo muito fértil para o uso da Realidade Virtual e suas técnicas. A utilização de ferramentas CAD para a criação de projetos de edificações precisos facilita a geração de modelos 3D em ambientes virtuais. Os mundos virtuais permitem que os usuários visualizem com maior realismo, naveguem livremente e analisem cuidadosamente o modelo, tudo isso em tempo real.

Este trabalho visa desenvolver uma ferramenta colaborativa para os engenheiros que produzem os projetos de uma obra, que tem como principal característica integrar diversos projetos de uma mesma edificação em um ambiente 3D. Dessa forma, esta ferramenta auxiliará na identificação de eventuais problemas, analisando as interferências e apontando os resultados, possibilitando assim correções necessárias ainda na fase de projeto.

Esta ferramenta aborda a utilização de técnicas computacionais de detecção de colisão em ambientes 3D para o estudo de interferências entre os diferentes tipos de projetos – hidráulico, elétrico e estrutural - de uma mesma obra de engenharia civil, projetos estes provenientes de programas CAD.

**Palavras-chaves:** Realidade Virtual, Engenharia Civil, Detecção de Colisão

## ABSTRACT

Castro, Tales Augusto Liguori de. **Realidade Virtual e Engenharia Civil: Detecção de Interferências entre Projetos de Edificações em 3D**. 2004. 62 f. Dissertação (Mestrado em Engenharia de Produção) – Programa de Pós-Graduação em Engenharia de Produção, UFSC, Florianópolis, 2004.

The Civil Engineering is a very fertile field for the use of Virtual Reality and its techniques. The use of CAD tools for the creation of precise construction projects makes easy 3D models generation in virtual environments. The virtual worlds allow that the users visualize with a bigger realism, navigate freely and analyze the model carefully, in real time.

This work aims at to develop a collaborative tool for engineers who make building projects, that has as main feature integrate several projects of a building in a 3D environment. This way, this tool will assist in the identification of eventual problems, analyzing interferences and pointing the results, making possible required corrections still in project phase.

This tool approaches the use of collision detection computational techniques in 3D environments for the study of interference between different types of projects - hydraulic, electric and structural – of one same civil engineering building, these projects came from CAD softwares.

**Keywords:** Virtual Reality, Civil Engineering, Collision Detection

## Sumário

Resumo .....	4
Abstract .....	5
Lista de Abreviaturas .....	7
1. Introdução .....	8
1.1. Estrutura do Trabalho .....	9
1.2. Objetivos.....	10
1.2.1. Objetivo Geral .....	10
1.2.2. Objetivos Específicos.....	10
1.3. Justificativa .....	11
1.4. Parceria com a Iniciativa Privada .....	12
1.4.1. Histórico .....	12
1.4.2. Softwares .....	13
2. Realidade Virtual – RV .....	15
2.1. Conceitos.....	15
2.2. Sistemas de RV .....	16
2.3. LRV/UFSC.....	17
2.4. Evolução Histórica da RV .....	19
2.5. Técnicas Avançadas de Computação Gráfica Utilizadas em RV .....	21
2.6. Conclusão.....	26
3. Utilização de 3D em Projetos de Engenharia Civil .....	27
3.1. História do Software CAD.....	27
3.2. CAD e Realidade Virtual.....	29
3.3. Conclusão.....	31
4. Tecnologias Utilizadas .....	32
4.1. Delphi .....	32
4.2. OpenGL .....	32
4.3. GLScene.....	34
4.4. XML .....	35
4.5. Conclusão.....	37
5. Detecção de Colisão .....	38
5.1. Técnicas .....	38
5.2. Conclusão.....	45
6. O Software de Análise de Interferência.....	46
6.1. Módulo de Carregamento .....	49
6.2. Módulo de Verificação de Interferências.....	54
7. Conclusão .....	57
7.1. Trabalhos Futuros .....	58
8. Referências.....	59

## Lista de Abreviaturas

API – *Application Programming Interface*  
CAD – *Computer Aided Design*  
CG – *Computação Gráfica*  
CSS – *Cascading Style Sheets*  
HTML – *Hyper Text Markup Language*  
IDE – *Integrated Development Environment*  
LRV – *Laboratório de Realidade Virtual*  
MIP – *Multum In Parvo*  
PC – *Personal Computer*  
RAD – *Rapid Application Development*  
RV – *Realidade Virtual*  
SGML – *Standard Generalized Markup Language*  
UDESC – *Universidade do Estado de Santa Catarina*  
UFSCAR – *Universidade Federal de São Carlos*  
UFSC – *Universidade Federal de Santa Catarina*  
USP – *Universidade de São Paulo*  
VRML – *Virtual Reality Markup Language*  
XML – *Extensible Markup Language*  
XSL – *Extensible Style Sheets*



## 1. Introdução

No fim dos anos 80 e no início dos anos 90, a Realidade Virtual (RV) era uma área inexplorada em sistemas comerciais e todos tinham grande esperança na utilização desta tecnologia. Entretanto, os limites foram ofuscados pelas possibilidades e as visões foram baseadas mais em esperanças do que em fatos. Hoje, a situação é completamente diferente. Uma quantidade considerável de pesquisa e de desenvolvimento foi realizada e a RV, sem dúvida, está pronta para ser usada como ferramenta em quase todas as áreas do conhecimento humano. Como áreas de atuação mais freqüentes pode-se citar militar, arquitetura, medicina, educação, engenharia e entretenimento.

A interferência entre projetos de edificações durante a execução de uma obra é um problema constante encontrado por engenheiros e técnicos no momento da execução da obra, pois normalmente cada parte da obra é projetado por um engenheiro especialista em determinada área. A possibilidade de se prever com antecedência eventuais problemas de execução, pode garantir economia de tempo, mão-de-obra e custos, itens que podem ser bastante significativos dentro do cronograma e orçamento de um projeto.

O escopo deste trabalho é apresentar um sistema colaborativo que importe informações de alguns projetos de engenharia civil e verifique os pontos de interferência entre eles. Decidiu-se utilizar as técnicas de RV para trazer um maior realismo e melhor visualização da integração destes projetos e também uma opção de navegação (*walkthrough*) e visão espacial detalhada de cada interferência.

## 1.1. Estrutura do Trabalho

Este trabalho é composto de oito capítulos, incluindo as referências. A disposição dos capítulos foi colocada de maneira a apresentar conceitos e tecnologias importantes para que haja uma maior compreensão da complexidade no desenvolvimento de todo projeto da aplicação.

No Capítulo 1 faz-se uma breve introdução ao trabalho, seus objetivos e sua justificativa e enfatiza-se a importância da parceria com a iniciativa privada para o desenvolvimento deste projeto.

O Capítulo 2 enfoca a Realidade Virtual, trazendo conceitos de alguns autores e diferenciando sistemas imersivos de não-imersivos. Para um entendimento do momento atual da RV é apresentado o seu histórico evolutivo no tempo. É reservado um subitem para apresentar o Laboratório de Realidade Virtual (LRV) da UFSC e um pouco do seu trabalho na área. E para finalizar o capítulo, descrevem-se técnicas avançadas de computação gráfica (CG) que são utilizadas atualmente na RV e que dão um maior realismo aos ambientes virtuais, utilizando melhor os recursos computacionais disponíveis.

O Capítulo 3 apresenta um histórico dos softwares de CAD (*Computer Aided Design*) e em seguida apresenta uma comparação entre CAD e RV.

O Capítulo 4 descreve as quatro tecnologias que foram utilizadas no desenvolvimento deste trabalho. Para programação do software foram utilizadas a IDE Delphi, a biblioteca gráfica OpenGL e a biblioteca gráfica GLScene. E para importação dos projetos da construção civil usou-se XML para estruturar os dados.

O Capítulo 5 traz uma explanação sobre técnicas utilizadas na detecção de colisão entre objetos em ambientes tridimensionais como por exemplo: esfera e plano, *bounding box* e *octrees*.

O Capítulo 6 apresenta o protótipo desenvolvido. São mostrados os princípios de funcionamento, descrevendo seus módulos e algoritmos.

A conclusão do trabalho é feita no Capítulo 7, apontando ainda trabalhos futuros a serem implementados para esta aplicação.

O Capítulo 8 traz todas as referências utilizadas no decorrer deste trabalho.

## **1.2. Objetivos**

### **1.2.1. Objetivo Geral**

Implementar um sistema colaborativo com as funcionalidades de importar e visualizar projetos da construção civil originados a partir de softwares CAD em um ambiente 3D, onde o usuário possa navegar e interagir em tempo real com o ambiente, e por fim, analisar interferências geométricas entre estes projetos.

### **1.2.2. Objetivos Específicos**

Neste trabalho pretende-se apresentar os conceitos de Realidade Virtual, na visão de alguns autores, e a sua evolução cronológica.

A aplicação aqui implementada importará arquivos de projetos estruturais, hidráulicos e elétricos, gerados a partir dos softwares AltoQI Eberick, AltoQI Hydros e AltoQI Lumine respectivamente.

Especificar a técnica de detecção de colisão entre objetos tridimensionais mais adequada para este trabalho.

O resultado da análise das interferências deverá ser de fácil identificação para o usuário, apontando diretamente para o ponto onde objetos 3D de projetos distintos entram em contato. Uma navegação eficiente dentro do ambiente facilitará a exploração deste ponto pelo usuário, podendo visualizar de vários ângulos com uma maior ou menor proximidade.

O sistema visa ser uma ferramenta de colaboração entre os engenheiros envolvidos na obra, oferecendo integração e análise das partes que foram projetadas separadamente.

### 1.3. Justificativa

É indiscutível o interesse, a curiosidade e a fascinação que a área de Novas Tecnologias e a área de Computação Gráfica despertam em quem trabalha com computação. Essas são áreas que possuem um enorme campo e uma demanda contínua por pesquisas. A RV se situa hoje como uma tecnologia que vem ganhando cada vez mais terreno. O primeiro ponto de motivação para este trabalho foi a pesquisa e desenvolvimento de uma aplicação utilizando as mais diversas técnicas no campo da RV.

O segundo ponto parte da convicção de que toda pesquisa tem o seu papel a cumprir com a sociedade, não devendo de forma alguma ficar numa prateleira de biblioteca, sem utilização real. Cada pesquisa certamente visa melhorar, um pouco que seja, o mundo em que vivemos, devendo assim extrapolar o meio acadêmico. O fruto deste trabalho pretende tornar-se um produto comercializável, totalmente voltado para o mercado de software. Portanto, um dos objetivos foi fazer com que esse trabalho ultrapasse os limites da universidade, chegando à sociedade com status de aplicação real e vendável.

A parceria com a iniciativa privada veio proporcionar o terceiro ponto que era uma necessidade real de pesquisa e implementação. A empresa AltoQi, uma empresa de desenvolvimento de software de engenharia civil, apresentou a necessidade de integração e visualização de diferentes projetos de engenharia civil em um único software.

Felizmente foi possível a união dos três fatores expostos acima neste trabalho: uma aplicação real, voltada para o mercado de software, utilizando a Realidade Virtual como tecnologia principal.

## **1.4. Parceria com a Iniciativa Privada**

Muitos especialistas envolvidos na área de Pesquisa e Desenvolvimento destacam ser necessária a participação da iniciativa privada para inserir o Brasil em um panorama mais expressivo nos campos da Ciência e Tecnologia.

Sem dúvida, a pesquisa aqui desenvolvida só foi possível devido à colaboração, o apoio e financiamento que surgiu após parceria firmada entre empresa e universidade.

Será feita neste tópico a descrição da empresa AltoQi Tecnologia, a empresa que apoiou o desenvolvimento deste trabalho [9].

### **1.4.1. Histórico**

A AltoQi Tecnologia em Informática foi fundada em Florianópolis no ano de 1989, como projeto de seus quatro fundadores: Rui Luiz Gonçalves, José Carlos Pereira, Jano d'Araujo Coelho e Ricardo Eberhardt, profissionais de Ciências da Computação e Engenharia Civil. Nestes 15 anos de experiência, a empresa especializou-se no desenvolvimento de softwares para projetos de edificações.

A concepção do empreendimento teve início através do acompanhamento do trabalho de um profissional de Engenharia Civil no cálculo e detalhamento de vigas de um edifício em concreto armado. Assim, surgiu a idéia de desenvolver um software para agilizar esta etapa do projeto estrutural, o Proviga.

Desde os primeiros softwares desenvolvidos para o sistema operacional DOS, a AltoQi não parou a busca por novas tecnologias, o que culminou no lançamento da versão integrada dos sistemas para cálculo estrutural, em ambiente Windows, o AltoQi Eberick, sistema líder absoluto na tecnologia de cálculo estrutural em concreto armado. Este programa abriu novos caminhos para que a AltoQi expandisse sua atuação na área de edificação (AltoQi Hydros , AltoQi Lumine e QiCAD, por exemplo).

Hoje, são mais de 10 mil usuários dos softwares da AltoQi. Podem ser citados como clientes: Petrobrás, Eletrosul (Gerasul), Celesc, Instituto Militar de

Engenharia, principais instituições de ensino superior, Tribunais de Justiça, prefeituras municipais, entre outros. Para ratificar tal conquista, a AltoQi ganhou os prêmios FINEP de Inovação Tecnológica 99 e ASSESPRO 98, e foi RUNNER UP do Max Award na Fenasoft 98, prêmios importantes e consagrados no setor de tecnologia.

Há importantes parcerias com instituições de ensino superior, entidades de classe, conselhos regionais, institutos de engenharia e profissionais que são formadores de opiniões em seus universos de trabalho.

### **1.4.2. Softwares**

Nesta seção serão apresentados os softwares AltoQi Eberick, AltoQi Hydros e AltoQi Lumine, desenvolvidos pela AltoQi Tecnologia [9]. Vale salientar que estes foram os três softwares utilizados para gerar os projetos estruturais, hidráulico e elétrico que fazem parte do escopo deste projeto.

- **AltoQi Eberick**

Software que se aplica ao cálculo de edificações de concreto armado, com um ou mais pavimentos. Os elementos da estrutura calculados pelo programa são: lajes, vigas, pilares, blocos sobre estacas e sapatas.

O lançamento da estrutura é feito de forma gráfica, em um ambiente de CAD próprio, com possibilidade de visualização tridimensional da estrutura sendo modelada. Os esforços nos elementos são obtidos através de uma análise via pórtico espacial, sendo que, a partir disto, cada elemento pode ser dimensionado e detalhado.

O AltoQi Eberick conta também com módulos adicionais, que apresentam ferramentas complementares ao programa: o módulo Master, com recursos de consideração do efeito do vento e envoltória de esforços; o módulo Formas, para a geração de plantas de forma, de locação e cortes sobre a estrutura e o módulo Escadas, para o lançamento, dimensionamento e detalhamento de rampas, vigas inclinadas e escadas.

- AltoQi Hydros

Software para projeto de instalações hidro-sanitárias prediais, que permite o lançamento da tubulação do projeto como um todo, englobando seus vários pavimentos. Possui um ambiente de CAD integrado, cujo objetos inteligentes representam tubos e conexões. Com base na conectividade entre os elementos, identifica o fluxo, obtém dados de cálculo em cada trecho e sugere as peças mais adequadas a cada conexão. Permite o lançamento de detalhes hidráulicos (isométricos) e sanitários. Calcula vazões e perdas de carga em cada trecho da tubulação.

- AltoQi Lumine

Software para projeto de instalações elétricas prediais, contém sua própria base independente de CAD. Aplica-se aqui a proposta de trabalho característica da AltoQi, baseada em objetos inteligentes ao invés de simples desenhos. Com estes objetos que representam pontos e eletrodutos, o programa cria um modelo completo da tubulação e da fiação da edificação dentro do computador, sobre o qual é feito o dimensionamento elétrico e definem-se plantas e detalhes.

## 2. Realidade Virtual

Este capítulo abordará a Realidade Virtual como tecnologia, passando pelos conceitos, descrevendo as principais características na visão de vários autores. Será apresentada ainda a evolução dessa tecnologia com o passar dos anos. Finalmente, para melhor entendimento, exemplificaremos mostrando importantes pesquisas e aplicações em áreas específicas que utilizam a Realidade Virtual como ferramenta.

### 2.1. Conceitos

O termo “Realidade Virtual” é bastante abrangente. Alguns autores preferem utilizar outros termos como: “Realidade Artificial”, “Tecnologia de Simulação” ou “Cyberspace”; contudo pode-se afirmar que, independente da denominação, todas as definições têm como foco principal o usuário e a utilização de interfaces bidirecionais de interação com o ambiente em tempo real.

Serão apresentados a seguir alguns conceitos apresentados em ordem cronológica:

Isdale [16] no livro *The Silicon Mirage* conceitua da seguinte forma: A Realidade Virtual é um modo para os humanos visualizarem, manipularem e interagirem com computadores com dados extremamente complexos.

A Realidade Virtual, para Trauer [13], é uma tecnologia que possibilita sintetizar um mundo – um mundo tridimensional, tangível, palpável, audível e interativo – através de imagens e sensações geradas por computador.

Latta e Oberg [19] citam Realidade Virtual como uma avançada interface homem máquina que simula um ambiente realístico e permite que participantes interajam com ele: Realidade Virtual envolve a criação e experimentação de ambientes.

Pimentel e Teixeira [18] definem Realidade Virtual como o uso da alta tecnologia para convencer o usuário de que ele está em outra realidade - um novo



meio de “estar” e “tocar” em informações: Realidade Virtual é o local onde os humanos e computadores fazem contato.

Segundo Byrne [17], esta tecnologia é baseada em computadores e fornece a ilusão de se estar imerso em um espaço tridimensional com habilidade para interagir com este espaço 3D.

Segundo Luz [12], Realidade virtual é a utilização de artifícios para a reprodução da realidade, sendo que atualmente o meio mais utilizado é o digital, através do uso dos computadores. Para se criar a realidade virtual, é necessário aguçar o maior número de sentidos do usuário, sejam eles visual, auditivo, tátil, dentre outros. Assim, o usuário sente-se inserido, ou seja, imerso em um ambiente no qual pode interagir com objetos e outras pessoas.

## 2.2. Sistemas de RV

Alguns autores, como é o caso de Burdea e Coiffet [2], defendem que uma experiência de Realidade Virtual só será válida quando existirem estímulos relacionados à interação, imaginação e imersão, conforme a Figura 1. Portanto, este triângulo traz as características inerentes a qualquer sistema de RV.

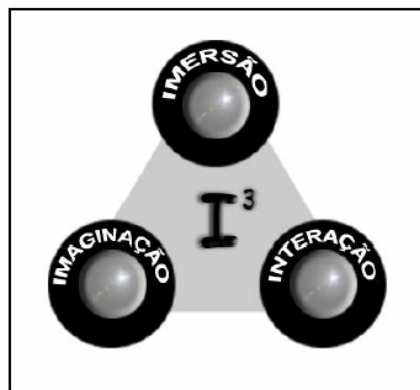


Figura 1 - Triângulo de RV: Imersão, Interação e Imaginação [2].

Já quanto a utilização de dispositivos em sistemas de RV, faz-se uma diferenciação quanto à imersão do usuário no sistema.

Há sistemas em que o usuário é estimulado de diversas formas, através de dispositivos específicos, passando a sensação de realmente estar dentro do

ambiente virtual. Os sistemas que utilizam tais dispositivos têm um maior grau de realismo e são chamados de sistemas imersivos.

Existem ainda sistemas onde a interação com o usuário dá-se de forma bem mais simplificada, sem a necessidade de dispositivos especiais, utilizando-se apenas dispositivos de um PC comum, como monitor, mouse e teclado. Esses sistemas mais simples são chamados de não-imersivos.

### **2.3. O LRV/UFSC**

No meio acadêmico brasileiro, diversas universidades de renome, podendo citar como exemplo a USP, a UFSCAR e a UDESC, possuem grupos que trabalham com o estudo e desenvolvimento de tecnologias de RV.

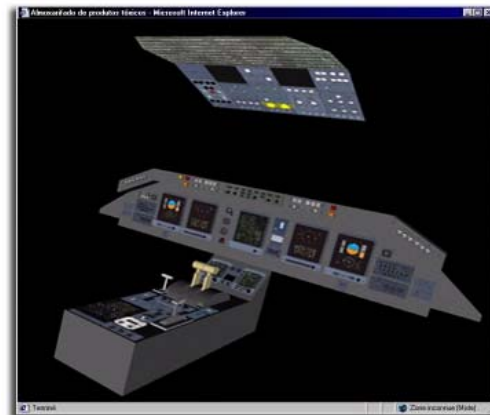
O Laboratório de Realidade Virtual (LRV) da Universidade Federal de Santa Catarina é um deles. Fundado em Julho de 1996, está subordinado ao Programa de Pós Graduação em Engenharia de Produção e Sistemas da UFSC [8].

A missão do LRV é desenvolver pesquisas, aplicações e produtos em Realidade Virtual, visando o domínio e desenvolvimento desta tecnologia beneficiando sociedade e indústria. As suas áreas de interesse são modelagem 3D, interfaces 3D homem/computador, comunicação de dados, linguagens de programação para RV, hardwares específicos para RV, Jogos e Computação Gráfica (CG) [8].

Desde a sua criação, em 1996, até hoje o LRV foi o berço de uma série de pesquisas, trabalhos e aplicações no campo da Realidade Virtual. Seguem abaixo três exemplos de trabalhos recentes desenvolvidos neste laboratório [8]:

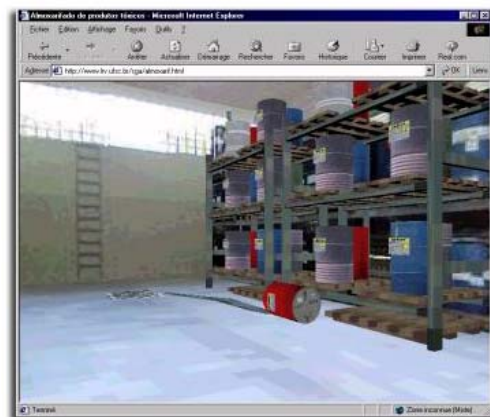
- ✓ **Virtual TrainingPit** – Um sistema desenvolvido para o treinamento de pilotos de aeronaves totalmente controlado por software e hardware em realidade virtual com substituição aos equipamentos convencionais.

Figura 2 [8].



- ✓ **SGA - Auditoria Ambiental** - Este projeto consiste na modelagem em VRML de ambientes virtuais interativos em 3D para simulação de processo de auditoria ambiental.

Figura 3 [8].



- ✓ **LRVChat3D** - Um ambiente virtual multiusuário que funciona como um *chat* para internet, tendo como pano de fundo um mundo tridimensional, onde os usuários são representados por personagens, e onde é possível a inclusão e manipulação de objetos.

Figura 4 [8].



## 2.4. Evolução histórica da RV

A evolução da Realidade Virtual se deu graças à criatividade e a perseverança de algumas pessoas notáveis que ao longo dos anos sonharam, pesquisaram e fizeram como que a RV se tornasse parte do nosso presente atual.

Conforme lembra Trauer [13], um dos marcos na história da RV foi o Link Trainer. O *Link Trainer* era um protótipo de simulador de vôo que foi patenteado em 1929. Este simulador utilizava sistemas articulados a ar para realizar vários movimentos em resposta aos controles. A sensação de movimento era satisfatória e, na época de seu desenvolvimento, os panoramas de janelas foram simulados, usando fotografias de aviões em pleno vôo e, posteriormente, fotografias de miniaturas. Era um protótipo que propiciava movimento e interatividade.

Um pouco mais tarde, na década de 50, temos outro marco que é o SENSORAMA, um equipamento mecânico desenvolvido por Morton Heilig. Luz [12] descreve o SENSORAMA de Heilig como “uma máquina de videogame que proporcionava ao usuário as sensações de andar de motocicleta pelas redondezas do Brooklin, Nova Iorque – EUA. A aplicação não era interativa, entretanto, ocorria o estímulo dos sentidos visual, auditivo, tátil e até o olfativo.”

Já nos anos 60, o Dr. Ivan Sutherland, fez o primeiro protótipo de capacete para ambientes virtuais. Este protótipo utilizava dois minitelevisores e nessa época, já possibilitava visualização estereoscópica, audição e posicionamento com movimentação da cabeça do usuário através de um sensor de movimento [20].

Em 1969, Myron Krueger, preferia utilizar a expressão “Realidade Artificial”, iniciou experiências com a interação homem-computador combinando a projeção de imagens geradas por computador e imagens de vídeo. Mais tarde, em 1982, Krueger publicaria seu livro *Artificial Reality*, onde expõe as suas experiências nesta área [13].

Em 1974, foram lançados no mercado os primeiros computadores pessoais. Logo em seguida, em 1977 foi patenteada a primeira luva para interação com computador [13].

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 [11].

Entre o fim da década de 80 e início dos anos 90 houve uma série de acontecimentos importantes para o desenvolvimento da Realidade Virtual. Trauer ressalta alguns acontecimentos [13]:

No fim de 1988, a IBM criou um grupo de pesquisa chamado *de Veridical Environment Department*. John Walker, da Autodesk, criador do software campeão de vendas AutoCAD, publicou um relatório interno chamado *Through the Looking Glass* e estimulou o projeto de espaço cibernético da firma.

Em 1989, a *AutoDesk* perdeu dois importantes pesquisadores que saíram para fundar a *Sense8 Corporation* e outros dois que se juntaram ao Laboratório de Tecnologia de Interface Humana, da Universidade de Washington.

A partir de 1989, a imprensa começou a divulgar a RV como uma nova e empolgante tecnologia. No fim de 1990, apareceram artigos não só em publicações científicas e de classe, mas, também, em publicações nacionais como *Rolling Stone*, *Forbes* e *The Wall Street Journal*. A primeira conferência profissional dedicada à RV aconteceu.

Em 1991, a *W. Industries*, de Leicester, Inglaterra, lançou um sistema completo de RV chamado *Virtuality*. Sua aplicação básica era para jogos de computador interpessoais e interativos, apesar de poder ser usado também, para desenho e outros trabalhos. Este sistema se tornou o primeiro produto a conquistar mercados, mundialmente”.

Em 1992, a *Silicon Graphics* introduz a biblioteca OpenGL sendo esse um grande passo para a Realidade Virtual em direção a plataforma comum: PC. [21]

A partir de meados dos anos 90, mais aplicações utilizando a RV começaram a aparecer devido ao avanço e barateamento do hardware, que até então era um obstáculo para que usuários comuns tivessem acesso a essa tecnologia fora dos centros de pesquisa ou através de altos investimentos.

Surgiram aplicações em diversas áreas como medicina, educação, química, física, entretenimento, arquitetura, etc.

Atualmente a RV tornou-se bastante acessível à população. Isto é comprovado pela introdução de ambientes virtuais no cotidiano das pessoas de forma despercebida. Devido a dois fatores foi possível que softwares de RV invadissem os micro computadores de lares e escritórios. O primeiro fator foi o aumento do poder de processamento dos microcomputadores comuns. A velocidade dos processadores, a quantidade e velocidade das memórias e também o poder de processamento das próprias placas de vídeo deram um salto extraordinário. O outro fator foi a criação de técnicas avançadas para melhorar a renderização dos modelos dentro do ambiente tridimensional, otimizando o uso de materiais, texturas e outros recursos, aumentando a veracidade. Há outras técnicas que vêm sendo aplicadas que dividem a carga de trabalho entre a CPU e *hardware* gráficos, fazendo com que alguns cálculos sejam executados diretamente por *hardware*, diminuindo o esforço exigido do processador do computador. Essas técnicas racionalizam o uso do poder de processamento disponível da máquina e algumas delas serão detalhadas no próximo tópico.

No campo do entretenimento a utilização da realidade virtual se dá de forma mais notória. Sem sombra de dúvida, a indústria de jogos em 3D é uma das que mais criam e utilizam tecnologia de ponta voltada para ambientes tridimensionais. Esse certamente é um dos motivos para abarcar um número tão grande de fãs e usuários caseiros.

## 2.5. Técnicas Avançadas de Computação Gráfica Utilizadas em RV

Para se chegar ao nível de desenvolvimento atual em sistemas RV, uma série de técnicas de computação gráfica (CG) foram criadas e introduzidas, buscando maior otimização na renderização de modelos e dos ambientes, melhor utilização do *hardware* disponível e visando sempre o aumento do realismo das cenas. Algumas dessas técnicas que são executadas em *hardware* gráfico são descritas e exemplificadas abaixo:

✓ **Texture Mapping** - O mapeamento de textura corresponde à projeção de uma imagem em 2D, digitalizada ou sintetizada, sobre uma superfície tridimensional. No mapeamento de textura procura-se minimizar a sua distorção através da curva geratriz da superfície, ou seja, deve haver certa coerência entre a curva e as coordenadas de textura. Como as coordenadas de texturas são normalizadas, isto é, estão no intervalo  $[0, 1]$ , uma das estratégias de minimizar a distorção é reparametrizar pelo comprimento de arco a curva poligonal que gera a superfície e depois tomar uma das coordenadas de textura do mesmo comprimento de cada aresta corresponde a poligonal, com isso teremos a não uniformidade das coordenadas de texturas obtidas durante a modelagem.

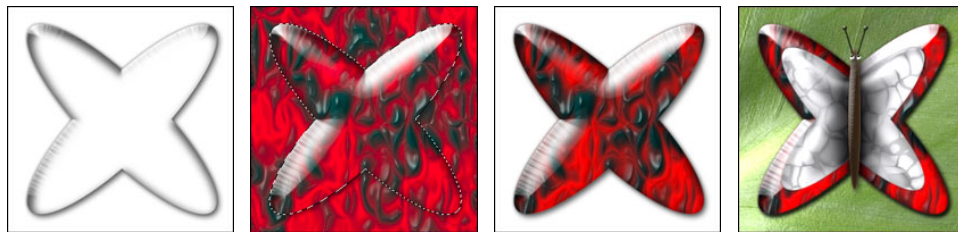


Figura 5: Criação de uma borboleta com mapeamento de textura [23].

✓ **Shaders** – Os *shaders* são programas que rodam exclusivamente em *hardwares* gráficos modernos. Eles permitem a representação realística em tempo real de objetos complexos como cabelos, roupas, vidros e materiais orgânicos (água), exigindo menos trabalho do que as técnicas anteriormente utilizadas.

Entretanto, existem diversos perfis para *shaders* que ditam quais hardwares gráficos podem executá-los. Os perfis consistem de diferentes conjuntos de instruções que possuem capacidades variantes. Eles são referenciados por números. Por exemplo, uma das versões anteriores do *vertex shader* é chamado *vertex shader 1.1* ou *VS\_1\_1* ou ainda *vs 1.1* para simplificar. *Shader profile* é uma das razões mais comuns que faz com que um *shader* não seja executado em seu hardware. Os *shaders* ainda se dividem em dois tipos: *Vertex Shaders* e *Pixel Shaders*. *Vertex Shaders* são subrotinas programáveis que realizam ações específicas em um vértice. *Vertex Shaders* programáveis no *hardware* de vídeo resultam em efeitos únicos com o mínimo uso da CPU. Já os *shaders* programáveis em hardware que manipulam *pixels* individuais, um de cada vez, são chamados de *Pixel Shaders*. O resultado é uma renderização perfeita com trabalho reduzido da CPU [27].

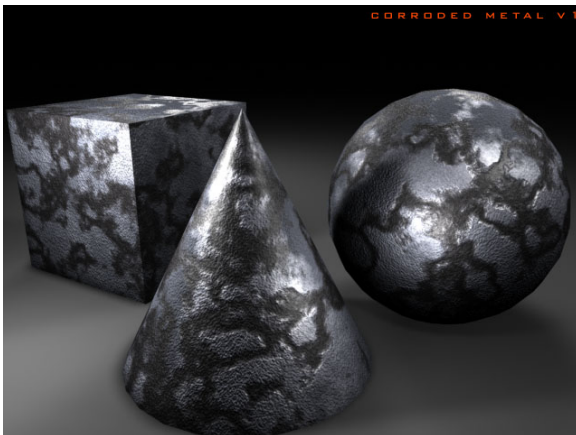


Figura 6: Objetos com efeito de metal corroído produzido com *shader* [22].

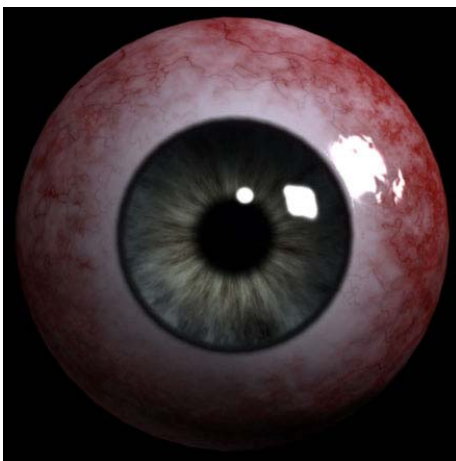


Figura 7: Olho humano produzido com a utilização de *shader*, no qual se tem controle sobre as veias e a vermelhidão do globo ocular [22].



✓ **Shadows** - Entenda-se aqui que uma sombra é uma área escura criada devido ao bloqueio da iluminação de um objeto. Foram desenvolvidos vários algoritmos para a criação de sombras em ambientes 3D. Existem desde técnicas simples como a colocação de uma “falsa sombra” no objeto (Figura 8); até outros métodos mais elaborados e com melhores resultados como a *Projeção de Vetores* e o *Shadows Z-Buffers*; chegando em algoritmos complexos como Volume de Sombra e até a combinação de métodos diferentes como a chamada *Reconstrução do Volume de Sombra* (Figura 9) que combina o *Shadows Z-Buffers* com o Volume de Sombra [26].

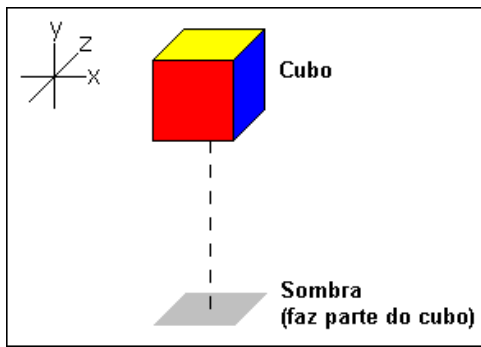


Figura 8: Exemplo uma “falsa sombra” [26].

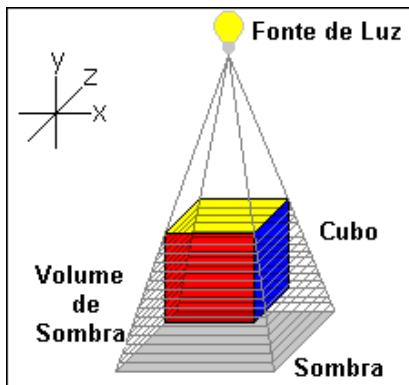


Figura 9: Exemplo de Volume de Sombra [26].

✓ **Bump Mapping** - O *bump mapping* é um mapeamento de textura que simula rugosidades na superfície através de uma perturbação na sua normal. Técnica que simula depressões numa superfície, através de perturbação angular variável da normal da superfície em pontos distintos. A normal modificada é usada no modelo de reflexão, produzindo assim diferenças de intensidade que simulam as depressões e ondulações na superfície.



Figura 10: Exemplo de *Bump Mapping* com textura metalizada e ponto de luz móvel, em seqüência [24].

✓ **Multitexturing** - Processo da adição de texturas múltiplas a um objeto durante a programação de um ambiente virtual. Elas podem estar uma em cima da outra como nos processos de *bump mapping* ou em conjunção umas com as outras para formar um modelo 3D extenso [27].

✓ **Environment Mapping** – Texturas mapeadas pelo ambiente podem refletir realisticamente outras texturas e objetos em volta de si. Faz uso de uma imagem do ambiente circundante a um objeto de modo que este a reflita. Por exemplo, se um avião cromado voa sobre um campo de areia, o cromo deve refletir a areia abaixo dele como se fosse um espelho. Não é propriamente um mapeamento de textura pois a imagem que aparece na textura varia de acordo com o ponto de observação no ambiente [27].

✓ **MipMapping** - MIP vem das iniciais de *Multum In Parvo* e significa muitas coisas num lugar pequeno. É uma técnica de pré-filtragem utilizada para melhorar a qualidade visual de imagens sintéticas. A técnica consiste em pré-filtrar a imagem, criando múltiplas cópias de um mapa de textura, todas derivadas da primeira, com resolução cada vez menor [25].

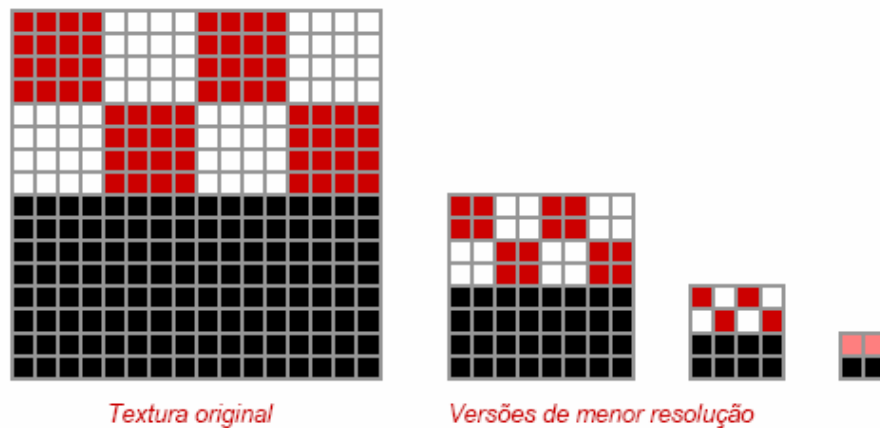


Figura 11: Exemplo de múltiplas cópias de mapa de textura [25].

## 2.6. Conclusão

Este capítulo foi inteiramente dedicado à Realidade Virtual, seus conceitos, sua história e técnicas que utiliza. Para apresentar conceitos sobre a RV recorreu-se a uma série de autores dentre os quais podemos citar Isdale, Latta, Oberg, Pimentel, Teixeira e Byrne. Dentro dos conceitos ainda foram explicitadas as diferenças entre sistemas imersivos e sistemas não-imersivos.

Devido a importância do Laboratório de Realidade Virtual (LRV) neste trabalho, foram apontadas a sua missão e as suas áreas de interesse, bem como alguns trabalhos desenvolvidos anteriormente por pesquisadores deste laboratório.

A evolução histórica da RV é de suma importância para situar o leitor do desenvolvimento da tecnologia e em seguida foram apresentadas diversas técnicas avançadas de CG que são largamente utilizadas para aumentar o realismo de ambientes tridimensionais sem perder desempenho, por otimizar a utilização de recursos computacionais disponíveis. Foram descritas as técnicas de *shaders*, *texture mapping*, *bump mapping* e sombra, entre outras.

### 3. Utilização de 3D em Projetos de Engenharia Civil

A engenharia civil define-se como o ramo da engenharia responsável pelo projeto e execução de obras e para isto prepara plantas, orienta as operações, determina especificações, controla os prazos, custos e os padrões de segurança necessários para o bom andamento da obra [1].

Antes da informática, os engenheiros civis eram obrigados a fazer tudo manualmente, utilizando prancheta, régua T e papel para fazer os cálculos e o desenho técnico de seus projetos. Em seguida, a informática passou a fazer a parte dos cálculos matemáticos, porém ainda não havia sido desenvolvido nada em relação a desenhos técnicos deixando bastante a desejar. Foi para suprir esta lacuna que foram desenvolvidos os primeiros *softwares* gráficos de engenharia, chamados de CAD (*Computer Aided Design*), que traduzindo para o português seria Projeto Assistido por Computador.

Atualmente há uma grande especialização por parte dos profissionais e alguns softwares também seguem essa linha. Isso quer dizer que numa mesma edificação, os projetos das diferentes áreas (estrutural, hidráulico elétrico, gás encanado, etc.) são feitos por pessoas diferentes em programas específicos e que em muitas vezes estão em escritórios diferentes, sem haver nenhum tipo de comunicação ou colaboração entre elas. A integração entre estes projetos só acontecerá na execução da obra.

#### 3.1. História do Software de CAD

Devido a larga utilização dos *softwares* de CAD pelas empresas de diversas áreas, a história do CAD será contada para que se tenha um melhor entendimento de como e porquê tamanho sucesso foi alcançado.

Concebido em 1957, o primeiro sistema de CAD foi realmente desenvolvido durante os anos 60. O Dr. Hanratty é largamente conhecido por suas contribuições pioneiras no campo de Projeto e Manufatura Assistido por Computador. Enquanto estava na General Motors, o Dr. Hanratty era co-designer do DAC (Projeto

Automatizado por Computador), a primeira produção de um sistema gráfico de manufatura interativo. Em 1971, o Dr. Hanratty fundou a Manufacturing e Consulting Services, Inc. (MCS), cujos produtos incluem Anvil-Express. No começo, a MCS forneceu o código para empresas como McDonnell Douglas (Unigraphics), Computervision (CADDS), AUTOTROL (AD380) e Control Data (CD-2000). Os analistas estimam que 70% de todos os sistemas de CAD/CAM 3D disponíveis hoje, têm suas raízes no código do original da MCS [29].

Os anos 70 foram dedicados em automatizar os desenhos em 2D. A chave aqui era colocar linhas e círculos na tela do computador e automatizar o processo usando uma interface de macro/programação. Durante este período, os operadores do CAD não tinham que ser somente bons desenhistas, mas eles também tiveram que ser bons programadores.

Algumas aplicações vieram da United Computing, da Intergraph e da IBM. A maioria destas aplicações utilizava números de ponto flutuante de precisão única (32 bits), mas naturalmente, era devido ao poder limitado do *hardware* da época. Embora o Unigraphics fosse um sistema de 3D CAD/CAM desde a versão inicial, não foi feita uma versão que utilizasse números de ponto flutuante de precisão dupla (64 bits) até 1979. [29]

Na metade dos anos 80, os principais avanços tecnológicos, incluindo a introdução de sistemas paramétricos de modelagem, permitiram que os produtos de software de CAD/CAM/CAE transformassem-se em uma parte integrante do processo de *design* de produto.

Uma das coisas mais importantes que aconteceram durante esta época foi o advento do computador pessoal e o começo da empresa Autodesk. John Walker fundou a Autodesk em 1982. Ele e seu pequeno grupo original de programadores iniciaram o desenvolvimento de cinco diferentes aplicações de automação para *desktop*. Fizeram isto com a noção de que uma das aplicações decolaria. O produto que vingou foi o AutoCAD, um pequeno programa gráfico 2D que permitiu que os fornecedores gráficos mostrassem o seu hardware de vídeo de última geração. O resultado líquido era um pequeno exército de vendedores de

*hardware* demonstrando o AutoCAD em cada feira de computação gráfica. Em 1987 a Autodesk tinha vendido 100.000 cópias do AutoCAD.

Nesse meio tempo, a empresa CADKEY entrou no mercado com o foco em aplicações 3D. Porém o 3D ainda era muito difícil de trabalhar em um PC. De qualquer maneira, AutoCAD controlou o mercado, direcionando seu desenvolvimento para adicionar a modelagem em *wireframe* 3D no AutoCAD versão 10 [29].

A modelagem de sólidos 3D em *software* comercial surgiu em 1988 com as primeiras cópias do software Pro/ENGINEER, que surgiu para ser um modelador extremamente robusto. Ainda 1988, a Unigraphics comprou a Shape Data Ltd. (desenvolvedores dos softwares Romulus, Romulus-D e Parasolid) e começou a comercializar o *kernel* do Parasolid como um produto *standalone*. O *kernel* do Parasolid permitia juntar várias superfícies e mostrar como um único sólido.

Sem dúvida, os anos 90 trouxeram progressos inacreditáveis nas potencialidades da modelagem em CAD. Em 1990, a Spatial Technologies trouxe a ACIS, uma *engine* comercial de modelagem de sólidos que fornecia uma base de dados comum que podia ser acessada e utilizada em múltiplos ambientes. A Autodesk começou a comprar empresas para ganhar o acesso às tecnologias. A aquisição da Micro Engineering Solutions em 1992 ajudou no lançamento do AutoSurf e depois da Woodburne em 1993, culminou no primeiro Designer de Modelagem Paramétrica AutoCAD da Autodesk. Em 1994, a Autodesk lança o AutoCAD 13 e chega ao número de 1.000.000 de cópias vendidas [29].

### **3.2. CAD e Realidade Virtual**

Esta parte do trabalho se concentra nas questões da integração CAD e RV. As diferenças entre sistemas de RV e o *software* de CAD nem sempre ficam bem claras. Muitas pessoas associam a RV com os jogos e o CAD, por outro lado, é associado com os desenhos arquitetônicos e de engenharia. A RV é freqüentemente referida a modelos mais realistas e o CAD aos modelos mais exatos e mais precisos [6].

É importante frisar que a distinção entre esses sistemas é o objetivo principal de cada um. Enquanto o CAD foi desenvolvido para criar modelos matematicamente precisos, em três dimensões, de objetos físicos do mundo real, os sistemas de RV priorizam a visualização e a interação do ambiente com o usuário. Dessa forma, o CAD e a RV são tecnologias complementares. Os modelos de RV podem ser extremamente realistas, mas os modelos do CAD também podem ser. Os modelos do CAD podem ser geometricamente exatos e precisos, mas os modelos de RV também podem ser. A distinção entre as duas tecnologias tem menos relação com que tipos dos modelos podem ser usados e mais relação com as melhores características dos programas [6].

O dilema crucial é o tipo dos dados manipulados em ambos os sistemas: de um lado, os sistemas de CAD operam sobre os dados semanticamente ricos, onde os conjuntos, as peças, as características ou as entidades topológicas podem ser criados e modificados. Os sistemas de RV, por outro lado, operam principalmente sobre triângulos simples que contém apenas informações geométricas [7].

A indústria da construção civil está lentamente adotando o uso de técnicas modernas de computação para vários propósitos diferentes. Atualmente, os desenhos em 2D são quase sempre produzidos em algum tipo de sistema CAD. Poucas empresas ou projetos empregam modelos 3D e menos ainda utilizam RV para melhorar as comunicações entre as partes envolvidas no processo de construção. Entretanto, há indicação de que um ganho de eficiência de até 30% pode ser alcançado com a utilização de RV e outras técnicas para melhorar as comunicações no processo de construção. Outra vantagem é que o investimento necessário para criar um modelo RV é pequeno comparado ao impacto que este, proporcionando análises e correções prévias do projeto, pode ter na qualidade final e no resultado financeiro ao construir uma estrutura grande e complexa [28].

No intuito de auxiliar o engenheiro que tem que lidar com modelos 3D complexos, os sistemas de CAD comerciais podem ser melhorados com uma interface usuário adicional, alternativa baseada na tecnologia de RV [7]. As animações produzidas em uma aplicação de RV são de baixa qualidade comparadas com aquelas criadas em programas de visualização profissional,

como 3D Studio Max <sup>tm</sup>. Por outro lado, exigem um tempo de renderização consideravelmente menor. Para a construção civil e seus objetivos, essas figuras e animações de baixa qualidade são suficientes para ilustrar o *layout* e a estrutura de uma edificação [28].

## **Conclusão**

A evolução dos programas CAD mostra com nitidez o desenvolvimento de uma linha de programas baseado em paradigmas 2D e, anos depois, a incorporação de características e funcionalidades 3D. Essa cadeia evolutiva constante certamente foi a chave para o sucesso do CAD.

Confrontando a RV com o CAD faz-se uma distinção clara da finalidade de cada um. Os programas CAD foram criados para construir modelos. Os programas de RV são os mais adequados para visualizar e interagir com modelos. Embora cada um adote algumas das potencialidades do outro, nenhum pretende replicar todas as funções do outro. Dessa maneira as forças de ambos os sistemas podem ser unidas na criação de mundos virtuais bons, exatos e precisos [6].



## 4. Tecnologias Utilizadas

Após ter sido contextualizada a aplicação de tecnologias 3D em projetos de construção civil e a apresentação da teoria de RV, será apresentada a seguir uma abordagem descritiva das tecnologias que foram utilizadas no desenvolvimento deste trabalho tanto na parte de programação quanto na parte de manipulação e comunicação de dados.

### 4.1. Delphi

O Delphi é um IDE (*Integrated Development Environment*), ou seja, um ambiente integrado de desenvolvimento comercializado pela empresa Borland Software Corporation para o desenvolvimento de programas utilizando a linguagem ObjectPascal. A linguagem ObjectPascal como o nome já sugestiona é uma linguagem de programação de 4ª geração, compilada, totalmente orientada a objetos [31].

O Delphi também se encaixa na definição de RAD (*Rapid Application Development*). Os RAD's são ambientes gráficos que ajudam o programador no desenvolvimento de sistemas, através de várias ferramentas, como por exemplo, a medida que os componentes são selecionados, o próprio RAD escreve o código para o programador e inclui, se necessário, as classes e bibliotecas das quais depende o componente, diminuindo, dessa forma, drasticamente o tempo de implementação [31].

A versão do Delphi que foi utilizada para o desenvolvimento deste projeto foi o Delphi 7 [31].

### 4.2. OpenGL

OpenGL é uma biblioteca de rotinas gráficas de modelagem, manipulação de objetos e exibição tridimensional que permite a criação de aplicações que usam Computação Gráfica. Seus recursos permitem ao usuário criar objetos gráficos

com qualidade, de modo rápido, além de incluir recursos avançados de animação, tratamento de imagens e texturas, é possível ainda, ter visualização em vários ângulos [21].

A biblioteca OpenGL foi introduzida em 1992 pela Silicon Graphics, no intuito de conceber uma API (Interface de Programação de Aplicações) gráfica independente de dispositivos de exibição. Com isto, seria estabelecida uma ponte entre o processo de modelagem geométrica de objetos, situadas em um nível de abstração mais elevado, e as rotinas de exibição e de processamento de imagens implementadas em dispositivos (*hardware*) e sistemas operacionais específicos. As função utilizada pelo OpenGL para desenhar um ponto na tela, por exemplo, possui os mesmos nome e parâmetros em todos os sistemas operacionais nos quais a OpenGL foi implementada, e produz o mesmo efeito de exibição em cada um destes sistemas [21].

Diante das funcionalidades providas pela OpenGL, tal biblioteca tem se tornado um padrão amplamente utilizado na indústria de desenvolvimento de aplicações. Esta biblioteca tem sido adotada também pela facilidade de aprendizado, pela estabilidade das rotinas e pelos resultados visuais consistentes para qualquer sistema de exibição concordante com este padrão. Diversos jogos, aplicações científicos e comerciais têm utilizado a OpenGL como ferramenta de apresentação de recursos visuais, principalmente com a adoção deste padrão por parte dos fabricantes de placas de vídeo destinadas aos consumidores domésticos.

Entre os recursos gráficos disponíveis pela OpenGL, podem ser destacados os seguintes [21]:

- Modos de desenho de pontos;
- Ajuste de largura de linhas;
- Aplicação de transparência;
- Ativação/desativação de serrilhamento (*aliasing*);
- Mapeamento de superfícies com textura;
- Seleção de janela de desenho;
- Manipulação de fontes/tipos de iluminação e sombreamento;

- Transformação de sistemas de coordenadas;
- Transformações em perspectiva;
- Combinação de imagens (*blending*);

As implementações do OpenGL geralmente provêem bibliotecas auxiliares, tais como a biblioteca GLU (*GL Utilities*), utilizada para realizar tarefas comuns, tais como manipulação de matrizes, geração de superfícies e construção de objetos por composição.

As especificações da OpenGL não descrevem as interações entre OpenGL e o sistema de janelas utilizado (Windows, XWindow, etc.). Assim, tarefas comuns em uma aplicação, tais como criar janelas gráficas, gerenciar eventos provenientes de mouse e teclado, e apresentação de menus ficam a cargo de bibliotecas próprias de cada sistema operacional [21].

### 4.3. GLScene

Várias bibliotecas gráficas estão à disposição no mercado para o desenvolvimento de aplicações. Este trabalho foi baseado na biblioteca gráfica chamada GLScene. A GLScene é distribuída ao público com código aberto sob a Mozilla Public Licence. Seu uso é gratuito tanto para aplicações comerciais ou não-comerciais, sendo solicitado apenas mencionar a sua utilização no software.

De acordo com a definição encontrada na própria página oficial do GLScene [3]: “A *GLScene* é uma biblioteca 3D baseada em OpenGL. Ela fornece componentes visuais e objetos permitindo a renderização de cenas 3D de maneira fácil e poderosa.”

Dentre muitas características da biblioteca, as principais são: estrutura hierárquica dos objetos; gerenciamento interativo da cena; objetos de câmera e luz que podem ser utilizados em qualquer lugar na hierarquia dos objetos da cena; biblioteca de materiais para compartilhar e reusar materiais; suporte a luzes dos tipos *ambient*, *diffuse*, *emission*, *specular* e *shininess*; suporte para formatos de textura do OpenGL, inclusive as compactadas (DXT, S3TC, etc.); utiliza o driver

OpenGL do hardware automaticamente, se estiver disponível, múltiplos visualizadores para uma ou mais cenas, troca de ponto de vista de maneira facilitada através da seleção de câmera; suporte para neblina de profundidade de visualização; suporte para *full-screen* com mudança dinâmica de resolução; animação de esqueletos (múltiplos ossos por vértice); física dinâmica: inércia, aceleração e aplicação de força, funções e utilidades de geometria otimizadas (vetores, *quaternions* e matrizes...); manipulação e otimização de malhas; determinação precisa da velocidade dos quadros [3].

#### 4.4. XML

O termo XML quer dizer *Extensible Markup Language* e a função do XML é prover uma representação estruturada dos dados. Essa representação do XML já se mostrou ser amplamente implementável e fácil de ser desenvolvida [14].

Implementações industriais na linguagem SGML (*Standard Generalized Markup Language*) mostraram a qualidade intrínseca e a força industrial do formato estruturado em árvore dos documentos XML [14].

O XML, assim como o HTML, é um subconjunto derivado do SGML, o qual é otimizado para distribuição através da *web*, e é definido pelo Word Wide Web Consortium (W3C), assegurando que os dados estruturados serão uniformes e independentes de aplicações e fornecedores. O XML faz uso de *tags* (palavras encapsuladas por sinais '<' e '>') e atributos (definidos com `name="value"`), mas enquanto o HTML especifica cada sentido para as *tags* e atributos (e freqüentemente a maneira pela qual o texto entre eles será exibido em um navegador), o XML usa as *tags* somente para delimitar trechos de dados, e deixa a interpretação do dado a ser realizada completamente para a aplicação que o está lendo. Já as regras de formatação para documentos XML são muito mais rígidas do que para documentos HTML. Uma *tag* esquecida ou um atributo sem aspas torna o documento inutilizável, enquanto que no HTML isso é tolerado. As especificações oficiais do XML determinam que as aplicações não podem tentar

adivinhar o que está errado em um arquivo (no HTML isso acontece), mas sim devem parar de interpretá-lo e reportar o erro [14].

A mais importante característica do XML se resume em separar a interface com o usuário (apresentação) dos dados estruturados. O HTML especifica como o documento deve ser apresentado na tela por um navegador. Já o XML define o conteúdo do documento. Por exemplo, em HTML são utilizadas *tags* para definir tamanho e cor de fonte, assim como formatação de parágrafo. No XML você utiliza as *tags* para descrever os dados, como por exemplo: *tags* de assunto, título, autor, conteúdo, referências, datas, etc.

O XML provê um padrão que pode codificar o conteúdo, as semânticas e as esquematizações para uma grande variedade de aplicações desde simples até as mais complexas, dentre elas:

- Um simples documento;
- Um registro estruturado tal como uma ordem de compra de produtos;
- Um objeto com métodos e dados como objetos Java ou controles ActiveX;
- Um registro de dados. Um exemplo seria o resultado de uma consulta a bancos de dados;
- Apresentação gráfica, como interface de aplicações de usuário;
- Entidades e tipos de esquema padrões.

Uma característica importante é que uma vez tendo sido recebido o dado pelo cliente, tal dado pode ser manipulado, editado e visualizado sem a necessidade de reacionar o servidor. Dessa forma, os servidores têm menor sobrecarga, reduzindo a necessidade de computação e reduzindo também a requisição de banda passante para as comunicações entre cliente e servidor.

O XML é considerado de grande importância na *Internet* e em grandes *intranets* porque provê a capacidade de interoperação dos computadores por ter um padrão flexível e aberto e independente de dispositivo. As aplicações podem ser construídas e atualizadas mais rapidamente e também permitem múltiplas formas de visualização dos dados estruturados.

O XML ainda conta com recursos tais como folhas de estilo definidas com *Extensible Style Language* (XSL) e *Cascading Style Sheets* (CSS) para a apresentação de dados em um navegador. O XML separa os dados da apresentação e processo, o que permite visualizar e processar o dado como se quiser, utilizando diferentes folhas de estilo e aplicações.

Essa separação dos dados da apresentação permite a integração dos dados de diversas fontes. Informações de consumidores, compras, ordens de compra, resultados de busca, pagamentos, catálogos, etc. podem ser convertidas para XML no *middle-tier* (espécie de servidor), permitindo que os dados pudessem ser trocados *on-line* tão facilmente como as páginas HTML mostram dados hoje em dia. Dessa forma, os dados em XML podem ser distribuídos através da rede para os clientes que desejarem [14].

#### **4.5. Conclusão**

Este capítulo destinou-se a enumerar e descrever as tecnologias utilizadas no desenvolvimento do sistema apresentado neste trabalho.

Dentre essas tecnologias, foram utilizadas na parte de programação do sistema a linguagem Object Pascal através do Delphi para desenvolvimento do sistema e as bibliotecas gráficas OpenGL e GLScene, que aceleraram o processo de desenvolvimento gráfico, agregando funções específicas de ambientes virtuais

Finalizando o capítulo, o padrão XML é descrito, deixando claro o porquê da sua escolha para a estrutura de dados para importação/exportação entre as diferentes aplicações envolvidas.

## 5. Detecção de Colisão

Os objetos que compõe os ambientes virtuais normalmente colidem com outros objetos ou com a própria câmera do usuário. Essa interação ocorre quando dois ou mais objetos tentam ocupar o mesmo espaço ao mesmo tempo no ambiente. O processo pelo qual esses eventos são determinados é chamado de Detecção de Colisão.

Algumas aplicações necessitam determinar a ocorrência de colisão entre objetos que estão em movimento, outras necessitam identificar o volume da intersecção entre objetos estáticos. Os objetos que possuem formas geométricas complexas podem ter suas formas simplificadas por volumes que limitam os objetos para diminuir o esforço computacional durante os cálculos da detecção de colisão.

Serão descritas a seguir algumas técnicas de detecção de colisão, destacando a *Bounding Box* e *Octrees* que podem ser aplicáveis a objetos estáticos, um dos objetivos deste trabalho.

### 5.1. Técnicas

O estudo de técnicas de detecção de colisão em Realidade Virtual considera qualquer método de prevenção de interpenetração entre dois poliedros, que se movimentam e giram com o tempo. Estas técnicas são aplicadas para simular, em Ambientes Virtuais, o fenômeno físico que rege que dois corpos não podem ocupar o mesmo lugar no espaço [5].

As técnicas criadas a partir das pesquisas realizadas nesta área são inúmeras. Cada uma tem características próprias, com vantagens e desvantagens, e com situações específicas para a sua utilização. Ainda não foi desenvolvida uma técnica única para todas as situações. Para melhor exemplificação, segue uma breve descrição de algumas técnicas:

- **Esfera e Plano Único** – Analisa os vetores da velocidade e do raio da esfera e normal do plano para achar o possível ponto de colisão da esfera (Figura 12). Se a distância entre a origem da esfera e o plano for menor que o raio, o plano invadiu a esfera ou se o ponto de colisão da esfera estiver no plano, há então colisão. Suponhamos que o centro de uma esfera de raio  $r$  que se move do ponto  $P_1$  no tempo  $t=0$  para o ponto no  $P_2$  no tempo  $t=1$  e que queremos determinar quando ela colide com o plano  $L$ . Assumimos que inicialmente não há intersecção com o plano e que o ponto  $P_1$  está no lado positivo do plano, uma vez que o lado negativo representa o interior de alguma estrutura. A posição  $P(t)$  do centro da esfera no tempo  $t$  é dada pela equação:

**(e.1)**  $P(t)=P_1 + tV$ ; onde  $V$  é a velocidade da esfera

**(e.2)**  $V=P_2 - P_1$ ;

A colisão ocorre quando a equação

**(e.3)**  $L' \cdot P(t) = 0$ ;

onde  $L'$  é paralelo ao plano  $L$  e foi deslocado pela distância do raio  $r$  na direção da sua normal.

Substituindo a equação (e.1) por  $P(t)$  temos:

**(e.4)**  $L' \cdot P_1 + t(L' \cdot V) = 0$

Isolando o tempo, a equação fica:

**(e.5)**  $t = - \frac{L' \cdot P_1}{L' \cdot V}$

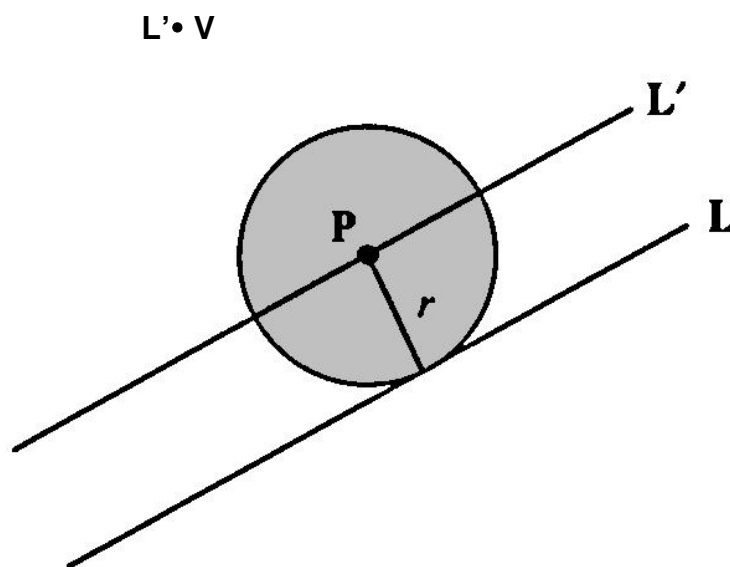


Figura 12: Esfera e Plano Único [15].



Se a esfera está se movendo paralelamente ao plano, não ocorre nenhuma colisão, se não a esfera colide com o plano no tempo  $t$ . Esta técnica tem um ótimo realismo como vantagem. A desvantagem é um alto esforço computacional [15].

- **Esfera e Polígono** – Analisa o vetor da velocidade e do raio da esfera e normal dos planos para achar o possível ponto de colisão da esfera com o ponto mais próximo dos planos que formam o polígono. Uma vez encontrado o ponto de colisão da esfera, aplica-se a mesma condição da Esfera e plano único. Também tem um ótimo realismo como vantagem. Também exige um altíssimo esforço computacional [15].
- **Cubo e Plano Único** – É possível determinar quando um cubo colide com um plano (Figura 13) utilizando um método similar ao utilizado para detectar colisão entre esfera e plano. A diferença é que devemos utilizar o raio efetivo do cubo para essa verificação, uma vez que o cubo pode colidir com um plano em mais de um ponto. Suponha um cubo que os lados têm o tamanho e a orientação dados pelos vetores  $\mathbf{R}$ ,  $\mathbf{S}$  e  $\mathbf{T}$ . O raio efetivo ( $r_{\text{eff}}$ ) do cubo em relação ao plano que tem a normal  $\mathbf{N}$  é dado pela seguinte equação:

$$(e.6) \quad r_{\text{eff}} = \frac{1}{2} (|\mathbf{R} \cdot \mathbf{N}| + |\mathbf{S} \cdot \mathbf{N}| + |\mathbf{T} \cdot \mathbf{N}|)$$

Dado  $\mathbf{Q}_1$  a posição do centro do cubo no  $t = 0$  e  $\mathbf{Q}_2$  a posição no  $t = 1$ , então a posição  $\mathbf{Q}(t)$  do cubo segue a equação:

$$(e.7) \quad \mathbf{Q}(t) = \mathbf{Q}_1 + t\mathbf{V}; \text{ onde } \mathbf{V} \text{ é a velocidade do cubo}$$

$$(e.8) \quad \mathbf{V} = \mathbf{Q}_2 - \mathbf{Q}_1$$

Para achar a intersecção com o plano  $\mathbf{L}$  calculamos:

$$(e.9) \quad t = - \frac{\mathbf{L}' \cdot \mathbf{Q}_1}{\mathbf{L}' \cdot \mathbf{V}}$$

$$\mathbf{L}' \cdot \mathbf{V}$$

O ponto de contato ( $\mathbf{C}$ ) entre o cubo e o plano é encontrado pela equação:

$$(e.10) \quad \mathbf{C} = \mathbf{Q}(t) - \frac{1}{2} [ \text{sgn}(\mathbf{R} \cdot \mathbf{N})\mathbf{R} + \text{sgn}(\mathbf{S} \cdot \mathbf{N})\mathbf{S} + \text{sgn}(\mathbf{T} \cdot \mathbf{N})\mathbf{T} ] ; \text{ onde}$$

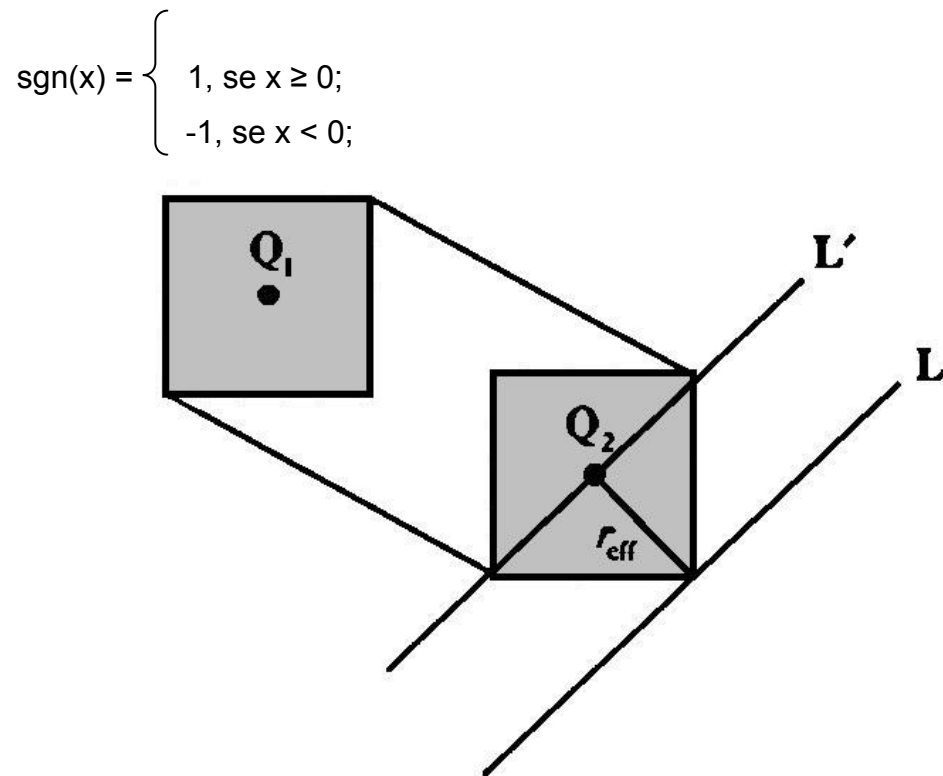


Figura 13: Cubo e Plano [15].

- **Bounding Sphere** – Uma Bounding Sphere é uma esfera que circunda todo o objeto (Figura 14). O cálculo para determinação da colisão é bastante otimizado, pois basea-se no centro e no raio da esfera. Porém este cálculo pode ser bastante impreciso dependendo da complexidade da forma do objeto. Um exemplo são os objetos longos e finos deixam uma enorme área vazia dentro da esfera, aumentando a margem de erro [30].



Figura XX: Bounding Sphere [30].

- **Axis Aligned Bounding Box** – O Axis Aligned Bounding Box também é um polígono retangular que envolve completamente os limites do objeto (Figura 15), porém este polígono é alinhado com os eixos do sistema de coordenadas local. Como vantagem tem-se a simplicidade e um dos melhores desempenhos do lado computacional. A desvantagem é que dependendo da forma do objeto, sugere colisões inexistentes [30].



Figura 15: Exemplo de AABB [30].

- **Oriented Bounding Box** – Um Oriented Bounding Box (OBB) é um polígono retangular com uma orientação arbitrária no espaço. Este polígono envolve completamente os limites do objeto e é orientado de acordo com a orientação do objeto envolvido (Figura 16). Além de ter as mesmas vantagens e desvantagens do AABB, o OBB apresenta uma característica de envolver o objeto com maior exatidão, pois os dois têm a mesma orientação [30].



Figura 16: Exemplo de OBB [30].

- **Octrees** - A Octree um método de particionamento de espaço que consiste de uma estrutura de dados hierárquica que divide recursivamente o volume de um cubo (Figura 16), que é um Bounding Box do objeto a ser testado, chamado de nó raiz ou célula raiz em oito cubos menores, conhecidos como octantes, até

um determinado nível de precisão. Cada octante pode estar em uma das três situações distintas: ou completamente contido no sólido; ou completamente fora do sólido; ou parcialmente contido no sólido. Os octantes que estiverem parcialmente contidos no sólido são divididos recursivamente em oito filhos até que o nível de subdivisão seja alcançado ou até que somente existam octantes dentro ou fora do sólido. A vantagem é poder representar com facilidade sólidos de diversas formas, tanto côncavos quanto convexos. Outra vantagem é a performance que é otimizada, pois muitas regiões do objeto que não estão na iminência de colidir são eliminadas. A desvantagem é que a representação do sólido original nunca será fiel, será sempre uma aproximação da sua forma [4]. Um exemplo de particionamento de espaço é mostrado nas Figuras 18 e 19. Na primeira, o espaço total que contém dois objetos esféricos é particionado e somente são subdivididos os octantes que contém os objetos. Na segunda, os octantes continuam sendo subdivididos para melhor representar o espaço ocupado pelas esferas.

**Célula Raiz**

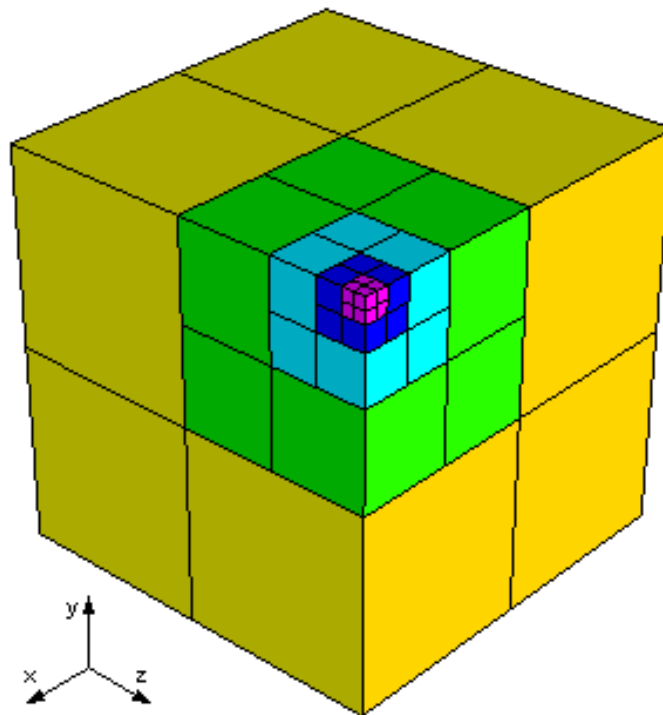
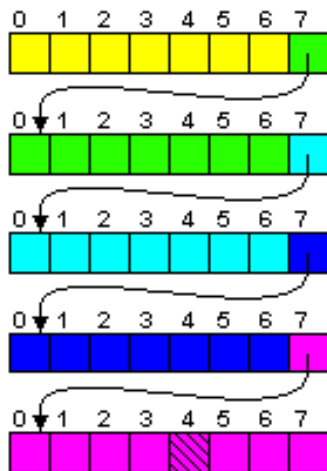


Figura 17: A célula raiz (amarela) dividida recursivamente em octantes.

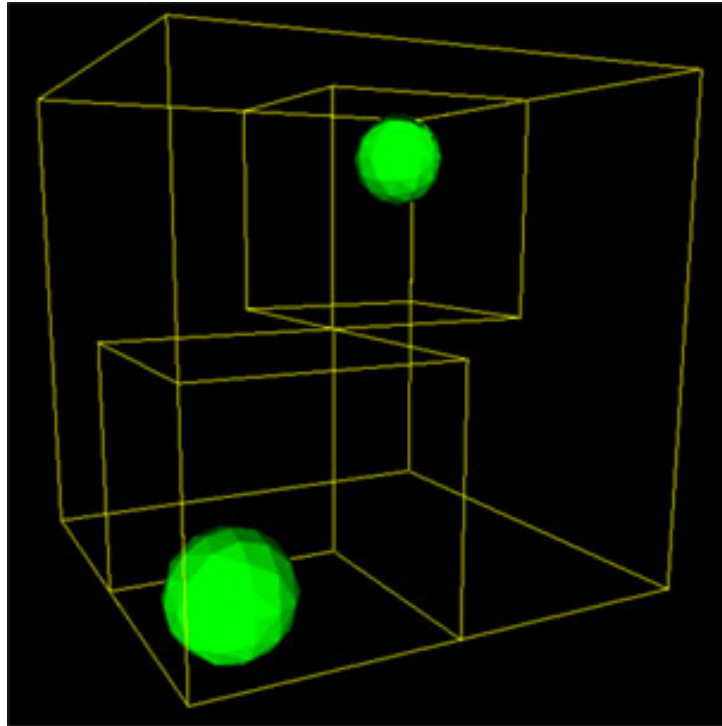


Figura 18: Espaço abrangendo 2 esferas, particionado com octrees.

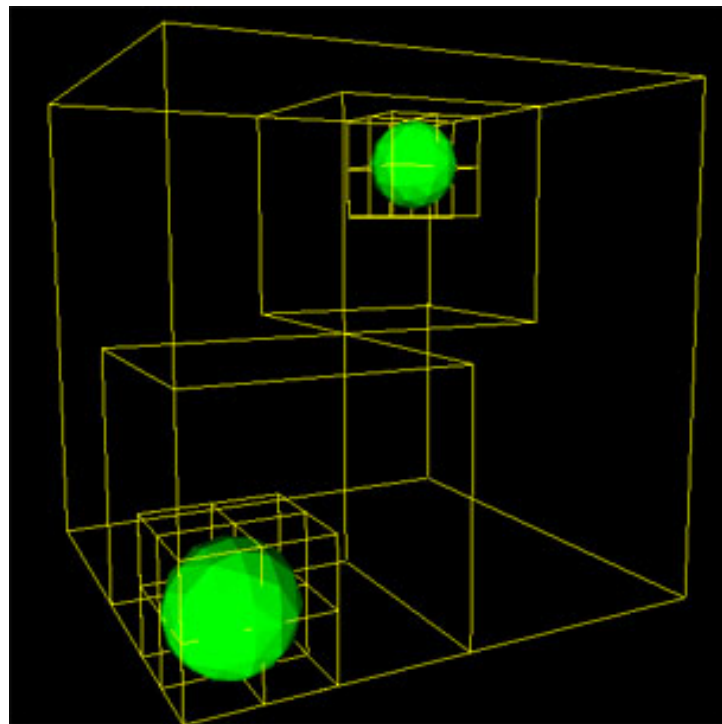


Figura 19: Mesmo espaço novamente particionado, agora com um nível abaixo de octantes.

## 5.2. Conclusão

Objetivou-se neste capítulo conceituar o que é a detecção de colisão, que, de forma resumida, é a identificação da existência de contato entre objetos em um ambiente. E por fim, fez-se uma explanação sobre diversas técnicas de detecção de colisão.

Esfera e Plano Único, Esfera e Planos Diversos, Esfera e Polígono, Cubo e Plano Único, Bounding Box e Octrees todas essas técnicas foram citadas de forma simples e objetiva.

## 6. O Software de Análise de Interferência

O sistema proposto consiste em integrar os diversos tipos de projetos da construção civil como, o projeto estrutural, o projeto hidráulico e o projeto elétrico de uma mesma obra de engenharia (uma casa, um edifício, etc.) em um ambiente 3D, e, a partir daí, verificar as interferências entre eles. O termo interferência aqui é utilizado para identificar o volume em que se detecta a colisão entre objetos geométricos 3D contidos em projetos diferentes. Pode-se verificar se a tubulação do projeto hidráulico não está situada dentro da viga do projeto estrutural ou ainda se uma caixa de passagem do projeto elétrico está dentro de um pilar. O sistema, como todo sistema de RV, ainda proporciona que o usuário navegue pelo ambiente e interaja com os objetos do projeto, alterando a cor e visibilidade de cada objeto ou de um conjunto de objetos, sendo possível que o usuário obtenha mais informações sobre o objeto com o clique do mouse. A utilização do software e sua navegação são detalhadas no Apêndice 1.

A correção de erros de projeto ainda na fase inicial da obra diminui consideravelmente situações inesperadas e custos desnecessários com mão-de-obra e materiais. A análise das interferências dará aos engenheiros responsáveis pela obra uma visão geral de como se comportará a estrutura da edificação junto com os demais projetos.

Atualmente softwares de engenharia civil utilizados para gerar os projetos, na sua maioria já dispõem de módulos de visualização 3D. Esta funcionalidade tornou um pouco mais fácil levar os dados dos projetos para dentro de ambientes virtuais. Porém, o fato dos projetos estrutural, elétrico e hidráulico serem gerados por programas diferentes faz a tarefa de integrá-los um pouco mais complexa, Este é o caso do Eberick, do Hidros e Lumine da AltoQi Tecnologia, descritos no Capítulo 1, que são softwares totalmente distintos, não havendo comunicação entre si.

A solução adotada foi criar um arquivo padrão para que fosse utilizado tanto na exportação, quanto na importação dos projetos. Em cada um dos softwares - Eberick, Hidros e Lumine – a empresa AltoQi criou um módulo de exportação dos

dados de cada objeto do projeto para um arquivo no formato XML. Já do lado do software de análise de interferência foi implementado um módulo de leitura dos dados e carregamento de cada objeto dentro ambiente virtual.

Chamado de Software de Análise de Interferências em Projetos de Edificações (AIPE), o software foi desenhado com uma interface simples e objetiva, com o intuito de valorizar a visualização da integração dos projetos e a navegação no ambiente, facilitando assim o objetivo final que é a identificação das interferências na obra.

A Figura 20 traz um fluxograma simplificado das tarefas executadas na durante a detecção de interferências utilizando o software AIPE. Em primeiro lugar, cada software da AltoQI exporta os dados de cada projeto (estrutural, hidráulico ou elétrico) para um arquivo no formato XML que em seguida será lido pelo AIPE. A partir dos dados lidos do XML, cada objeto é carregado no ambiente. Após os projetos estarem carregados, faz-se a verificação das interferências entre projetos distintos, isto é, o estrutural versus o hidráulico, o hidráulico versus o elétrico. A cada interferência encontrada uma lista é incrementada com os nomes dos objetos que estão se interceptando. Ao final, o resultado contendo todas as interferências encontradas é exibido para o usuário.

Durante a análise e implementação do sistema foram identificadas duas partes como as mais importantes: o Módulo de Carregamento e o Módulo de Verificação de Interferências, ambos descritos nos capítulos seguintes.



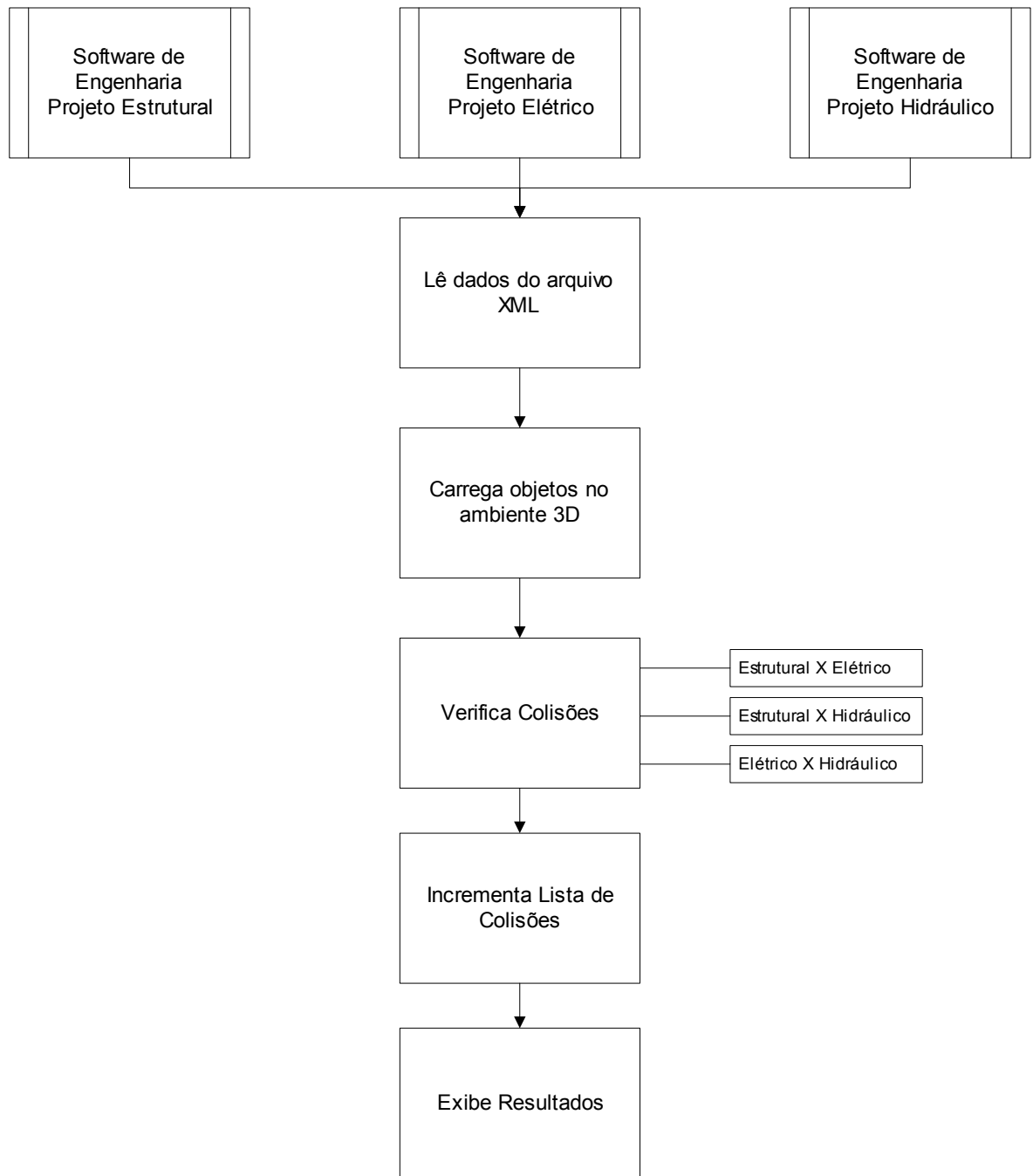


Figura 20: Fluxo simplificado das tarefas executadas pelo software AIPE

## 6.1. Módulo de Carregamento

Uma das partes mais importantes da aplicação é o módulo de carregamento dos projetos, pois este tem a finalidade de ler os dados e posicionar os objetos de forma adequada no espaço. Apesar da biblioteca GLScene já ter uma série de classes para objetos geométricos pré-definidos, como cubos, cilindros e esferas, é importante também ressaltar que uma única classe é utilizada neste projeto para carregar todos os tipos de objetos. Esta classe é chamada de TGLFreeForm e a sua característica é conter uma ou várias malhas. Independente do número total de malhas todas elas serão tratadas como um objeto único. Portanto, cada objeto traz consigo no XML, os dados de cada ponto que compõe a sua malha, sendo possível dessa forma montar todos os triângulos e todas as faces deste objeto.

Por causa das diferenças dos objetos de cada projeto e pelas características dos próprios softwares de origem do projeto, o módulo de carregamento faz diferenciação entre dois grupos, tratando-os de formas distintas. No primeiro grupo estão os projetos estruturais e no segundo grupo estão os projetos elétricos e projetos hidráulicos.

No grupo dos projetos estruturais, as *tags* utilizadas na estrutura hierárquica dos dados que representam os objetos são descritas abaixo:

**<Projeto nome="" tipo="">** A *tag* principal onde agrupa os objetos dentro de si e tem dois atributos o nome do projeto e o tipo do projeto.

**<Pavimentos>** Esta *tag* agrupa todos os pavimentos da obra.

**<Pavimento nome="">** Abre um pavimento específico da construção, agrupando os demais tipos de objetos pertencentes a este pavimento

**<Fundações>** Esta *tag* indica que o pavimento é térreo e que possui fundações no projeto.

**<Sapata nome="">** A sapata é um dos objetos que podem ser encontrados nas fundações da obra e possui um único atributo que é seu nome.

**<Bloco nome="">** O bloco é o segundo tipo de objeto que pode ser encontrado nas fundações de uma obra. Também tem o nome com único atributo.

**<Vigas>** Esta *tag* agrupa todas as vigas correspondentes a um pavimento.  
**<Viga nome=" ">** Especifica uma viga indicando no atributo seu nome.  
**<Lajes>** Esta *tag* agrupa todas as lajes correspondentes a um pavimento.  
**<Laje nome=" ">** Especifica uma laje indicando seu nome no atributo.  
**<Pilares>** Agrupa todas os pilares correspondentes a um pavimento.  
**<Pilar nome=" ">** Especifica uma pilar indicando seu nome no atributo.  
**<Rampa nome=" ">** Esta *tag* indica uma escada no projeto trazendo seu nome como único atributo.

Ainda na hierarquia, as *tags* <Sapata>, <Bloco>, <Viga>, <Laje>, <Pilar> e <Rampa> são os objetos propriamente ditos e são os que serão visualizados dentro do ambiente. Por isso, trazem consigo as informações sobre sua geometria. Sendo elas:

**<Poligono1>** Esta *tag* traz as informações de uma das faces do objeto (superior ou inferior), podendo ter quantos pontos forem necessários.

**<Poligono2>** Esta *tag* traz as informações da face oposta da indicada no <Polígono1>, conseqüentemente possuindo o mesmo número de pontos.

**<Ponto x=" " y=" " z=" ">** Esta *tag* é a que tem as informações de cada ponto dentro dos polígonos.

No grupo dos projetos hidráulicos e elétricos, a hierarquia dos objetos é um pouco mais complexa, podendo cada objeto ser formado por várias partes e vêm dispostos na seguinte estrutura em XML (*tags*):

**<Projeto nome=" " tipo=" ">** A *tag* principal onde agrupa os objetos dentro de si e tem dois atributos o nome do projeto e o tipo do projeto.

**<Linha>** Traz as informações de uma linha da planta baixa de um pavimento.

**<Grupo>** Indica o grupo a que a linha faz parte.

**<Pavimento>** Indica o pavimento da linha.

**<Ponto x=" " y=" " z=" ">** Sempre aparece duas vezes seguidas indicando o ponto inicial e final da linha.

**<Objeto nome=" ">** Agrupa as informações de um objeto que pode ser composto por várias partes. Por exemplo, uma esfera e um cilindro, dois cilindros, uma caixa e um cilindro, etc. O atributo único é o nome.

**<Box3D>** É uma caixa que pode fazer parte de um objeto.

**<Grupo>** Indica o grupo a que o Box3D faz parte.

**<Pavimento>** Indica o pavimento do Box3D.

**<Poligono1>** Esta *tag* traz as informações de uma das faces do Box3D (superior ou inferior), podendo ter quantos pontos forem necessários.

**<Poligono2>** Esta *tag* traz as informações da face oposta da indicada no <Polígono1>, conseqüentemente possuindo o mesmo número de pontos.

**<Ponto x=" y=" z=" ">** Esta *tag* é a que tem as informações de cada ponto dentro dos polígonos.

**<Cilindro>** Agrupa informações de uma parte cilíndrica de um objeto. Em um projeto elétrico é chamado de “Duto”, já no projeto hidráulico é chamado “Tubo”. Um algoritmo monta o cilindro por extrusão de um círculo.

**<Grupo>** Indica o grupo a que o cilindro faz parte.

**<Pavimento>** Indica o pavimento do cilindro.

**<Origem x=" y=" z=" "/>** É a posição de origem do círculo.

**<Direcao x=" y=" z=" "/>** É o vetor de direção de extrusão do círculo.

**<Raio>** É o raio do círculo.

**<Altura>** É o comprimento do cilindro.

**<Esfera>** Agrupa as informações de uma parte esférica de um objeto. A sua geometria é feita através de um algoritmo que recebe um círculo e monta todos os pontos da esfera.

**<Grupo>** Indica o grupo a que a esfera faz parte.

**<Pavimento>** Indica o pavimento da esfera.

**<Origem x=" y=" z=" "/>** É a posição de origem do círculo.

**<Raio>** É o raio do círculo.

A Figura 21 apresenta a visão inicial que o usuário tem após seguir todos os passos para criação de uma nova obra e inserção dos projetos nesta obra, passos esses descritos no Apêndice 1. No lado esquerdo são mostrados os projetos que foram carregados com ícones diferentes identificando o tipo do projeto, seguido do respectivo nome do projeto. Ao centro, o usuário tem uma visão externa de toda a estrutura da obra, podendo identificar os objetos. Navegando no ambiente com o teclado ou com o mouse, conforme descrito no Apêndice 1, o usuário pode chegar até o centro da estrutura, como é mostrado na Figura 22. Dessa forma poderá analisar muito mais de perto cada objeto que foi carregado no ambiente.

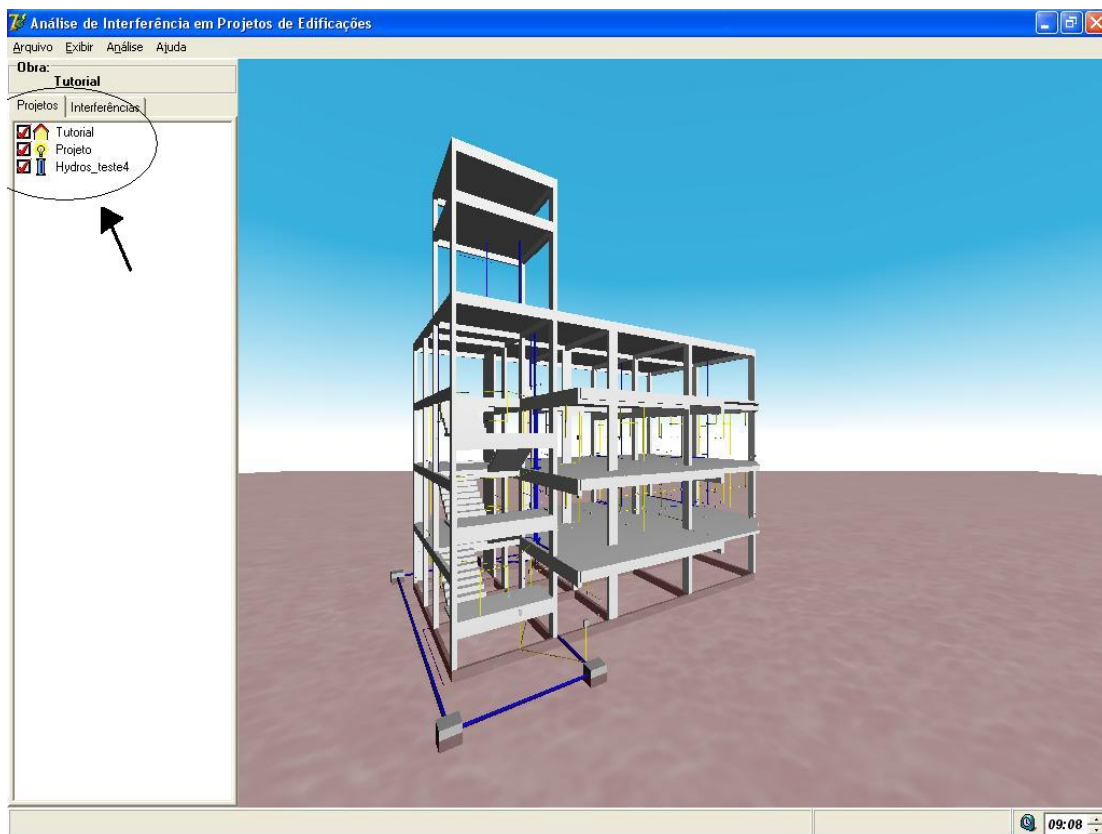


Figura 21: Vista Externa dos projetos estrutural, elétrico e hidráulico importados pelo módulo de carregamento.



Figura 22: Vista Interna dos projetos estrutural, elétrico e hidráulico importados pelo módulo de carregamento.

## 6.2. Módulo de Verificação de Interferências

O módulo de verificação de interferência, certamente, é o mais importante da aplicação. É nele que vai se executar o que o programa se propõe a fazer.

Como foi visto no capítulo anterior, os cálculos para a verificação de colisão em ambientes virtuais não são simples e muitas vezes exigem bastante processamento da máquina. Portanto, a maior preocupação no desenvolvimento desse módulo se deu na obrigatoriedade de manter a performance da análise, ou seja, a rotina de verificação de interferência tinha que ser executada no menor tempo possível, não podendo ser extremamente demorada, independentemente do número de objetos que existissem no ambiente.

Na maioria dos programas que detectam colisão isto é feito normalmente de duas maneiras: ou entre um único objeto e vários outros - como por exemplo no caso de verificação de colisão da câmera ou avatar com a geometria do próprio ambiente; ou poucos objetos específicos entre si - como por exemplo em ambientes multiusuários. Diferentemente destas duas maneiras apresentadas, no sistema AIPE a quantidade de objetos que podem colidir é enorme. Em um projeto de uma residência pequena poderão existir centenas de objetos, porém este número poderá chegar aos milhares no caso de grandes edifícios.

Buscando a melhor performance, neste sistema utilizou-se a detecção de colisão através da construção de octrees dos objetos. O particionamento do espaço em cubos otimiza bastante os cálculos na execução desta rotina. A escolha da classe `TGLFreeForm` da biblioteca `GLScene` como a classe padrão para todos objetos do projeto facilitou a implementação de octrees. Esta classe possui uma série de métodos para manipular octrees.

A detecção de colisões acontece em duas fases distintas no protótipo. Na primeira fase, a rotina somente verifica quais objetos estão colidindo e monta uma lista com os nomes.

Uma função recursiva faz a função de filtro. Com o intuito de diminuir o número de verificações e cálculos, esta função envia para a detecção de

interferências somente projetos diferentes, pois não faz sentido verificar interferências de objetos de um mesmo projeto.

Em seguida, outra função recursiva pega cada objeto da lista de objetos do primeiro projeto e confronta com cada objeto da lista do segundo projeto, procurando por interferências utilizando octrees. O processo funciona da seguinte forma: Primeiro a função verifica se as octrees dos dois objetos (de projetos distintos) já estão construídas. Se alguma não estiver, então constrói. Na sequência, de posse das duas octrees, executa a verificação de interferências. Confirmando a interferência, cria-se um ponteiro de referência para cada objeto.

Um passo importante ao fim da verificação das interferências é fazer com que as octrees de todos os objetos sejam destruídas. Não fazê-lo implica em deixá-las todas carregadas em memória, o que seria um desperdício de memória, podendo ocasionar lentidão ou até uma falha de todo sistema.

Por fim, um *checklist* na interface do usuário (Figura 23) é preenchido com os ponteiros que referenciam os objetos em que há interferência.

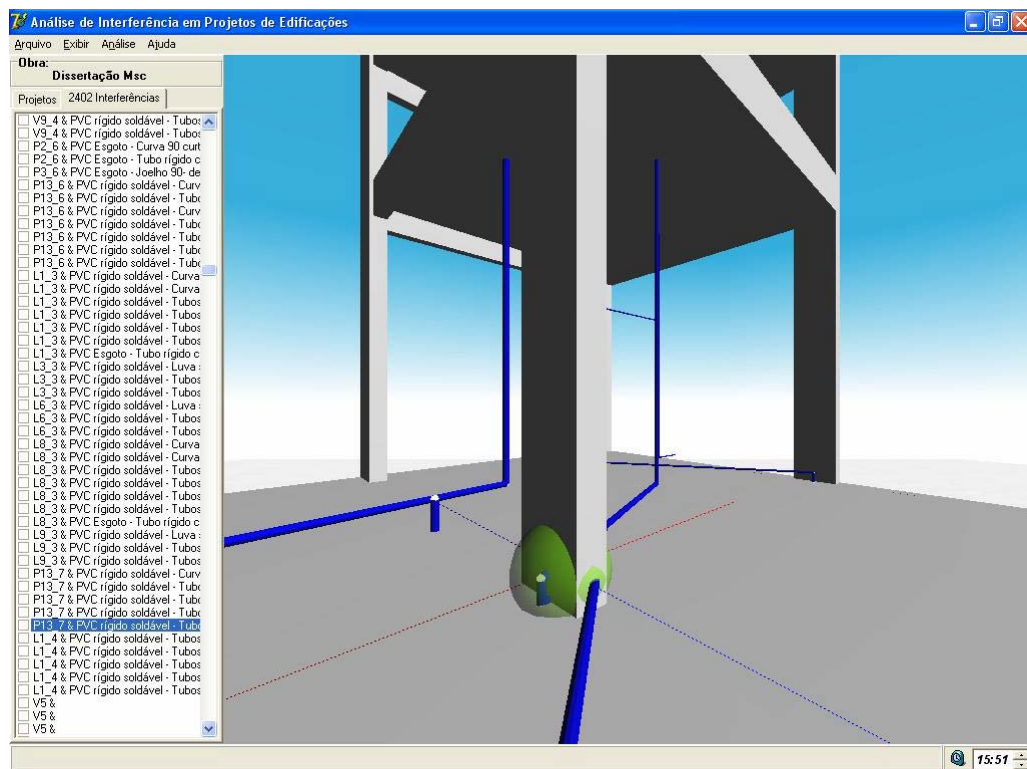


Figura 23: Lista de resultados da análise de interferências à esquerda. À direita, a intersecção dos objetos é marcada por uma esfera translúcida.



No segundo momento outra rotina detecta a colisão entre dois objetos específicos e indica o ponto exato da intersecção.

Um duplo clique em qualquer uma das linhas do *checklist* que contém o resultado da verificação de interferência inicia esta segunda verificação. Quando este evento acontece, a aplicação recupera os objetos referenciados na linha que foi clicada, através de ponteiros que apontam para cada um deles. Em seguida, constrói novamente a octree de cada um (pois esta foi destruída no final da primeira verificação), verifica novamente a interferência entre os objetos. O resultado da verificação não serão nomes e ponteiros, desta vez será um conjunto de pontos. Estes são os pontos que determinam exatamente o volume da intersecção dos dois objetos. Para uma melhor visualização, uma esfera semi-transparente é colocada ao redor desta intersecção. A câmera se desloca até um ponto próximo da nova esfera (intersecção) e o focaliza em sua direção, fazendo com que o usuário verifique diretamente a interferência escolhida. A Figura 24 apresenta uma situação em que o usuário executou um duplo-clique na lista de interferências, na linha que está selecionada, e a câmera foi deslocada automaticamente até esta intersecção.

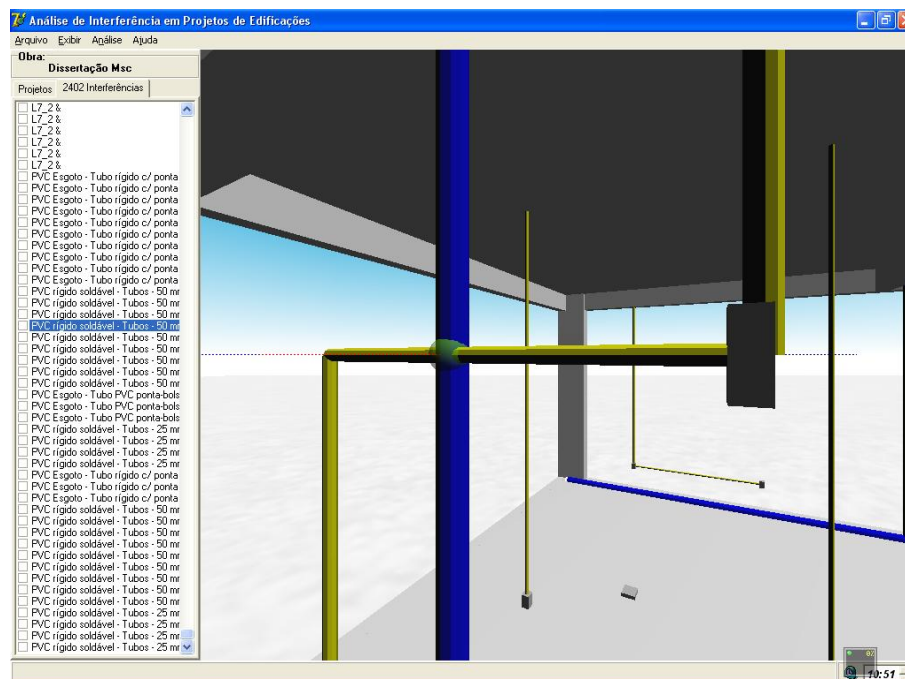


Figura 24: Colisão entre um tubo do projeto hidráulico (em azul) e um duto do projeto elétrico (em amarelo).

## 7. Conclusão

A engenharia civil mostrou inúmeras oportunidades para aplicação de software utilizando a RV. As técnicas desta nova tecnologia oferecem benefícios potenciais consideráveis para todos os estágios do processo da construção civil, desde planejamento inicial do projeto conceitual à gerência das operações.

O software apresentado demonstrou a aplicabilidade do sistema de detecção de colisão entre projetos estruturais, hidráulicos e elétricos e se mostrou ser bastante útil para a correção de problemas ainda em fase de projeto.

Outro ponto que é extremamente importante é a interatividade do usuário com o sistema. Somente um sistema de RV pode proporcionar aos engenheiros, que são os responsáveis pelos projetos, uma melhor visualização com liberdade de rotação e translação nos três eixos (X, Y e Z), facilitando identificação e análise dos problemas que uma obra possa apresentar.

Do ponto de vista das técnicas de detecção de colisão, a opção por construir as octrees de todos os objetos e a partir deste ponto verificar o contato entre eles apresentou duas grandes vantagens: a performance, pois os cálculos são baseados sempre em cubos, e uma maior facilidade para identificar o volume da intersecção entre os objetos.

Infelizmente a RV permanece ainda como um recurso subutilizado para a indústria de construção civil, embora algumas empresas estejam adotando lentamente a tecnologia e descobrindo sua potencialidade e eficácia para a área. Certamente as suas técnicas podem ser aplicadas na construção com tanto sucesso quanto ocorreu com as indústrias mecânicas, químicas, da aviação e cinematográficas.

## 7.1. Trabalhos Futuros

Durante a implementação do software, diversas novas funcionalidades foram identificadas e/ou sugeridas. Alguns pontos que podem ser desenvolvidos na continuação deste projeto:

1. Implementar a verificação de colisão por segmentos da obra. Por exemplo, verificar colisão apenas de um pavimento e não de toda obra como é feito atualmente.
2. Colocar a opção de colisão da câmera do usuário com o ambiente durante a navegação, evitando passar “por dentro” dos objetos como viga, pilar ou dutos. Esta característica exige bastante processamento pois essa verificação é feita a cada passo do movimento dentro do ambiente e deve poder ser desabilitada dependendo da necessidade do usuário. Porém sem dúvida dá um maior realismo ao ambiente.
3. Outra implementação que traz um maior realismo é a utilização de sombras no ambiente, a escolha do método será de grande importância, pois este deverá ser executado de forma bastante otimizada
4. por causa da quantidade de objetos que são criados por cada projeto.
5. A versão atual somente importa arquivos de projetos estrutural, hidráulico ou elétrico. A intenção é fazer com que o software importe os mais variados tipos de projetos da construção civil como por exemplo projetos de gás natural, projeto contra incêndio, etc.
6. Criação de um módulo, diretamente dentro do sistema ou como um plug-in, para fazer com que o sistema importe arquivos de formato padrão (DXF, DWG, etc.) e não somente arquivos em XML de formato proprietário.

## 8. Referências

- [1] ENCICLOPÉDIA DIGITAL MASTER ON-LINE. Disponível em <<http://www.encyclopedia.com.br/med2000/pedia98a/prof57n6.htm>> Acesso em 28.07.2004.
- [2] BURDEA, G.; COIFFET, P. Virtual Reality Technology. New York, NY: John Wiley & Sons, 1994.
- [3] GLSCENE – OPEN SOLUTION FOR DELPHI. Disponível em <<http://www.glscene.org>> Acesso em 15.06.2004.
- [4] NASCIMENTO, H. P.; Detecção de Colisão entre Objetos Convexos com Características Geométricas Mutantes, Proceedings of VI Symposium on Virtual Reality, Ribeirão Preto, SP, 2003, p 48-59.
- [5] STRADIOTTO, César Ramirez Kejelin. Biblioteca Para Criação De Jogos Utilizando Geração Pseudo-Randômica Paramétrica, Realidade Virtual, Inteligência Artificial e Redes De Computadores, Dissertação de Mestrado, Florianópolis, SC, 2002, p 44.
- [6] Virtual Reality and CAD - Complementary or Competing? CSA Newsletter, Vol. XII, No. 3, Inverno, 2000.
- [7] VAHL, Matthias, LUKAS, Uwe von. Integration Of Virtual Reality And Cad Based On Omg's Cad Services Interface, Ecec 2003, 10th European Concurrent Engineering Conference, Plymouth, United Kingdom, Abril, 2003.
- [8] LABORATÓRIO DE REALIDADE VIRTUAL DA UFSC. Disponível em <<http://www.lrv.ufsc.br>> Acesso em 15.06.2004.
- [9] ALTOQI – TECNOLOGIA EM INFORMÁTICA. Disponível em <<http://www.altoqi.com.br/>> Acesso em 16.06.2004.
- [10] ROSA JUNIOR, Onivaldo. LRVCHAT3D, desenvolvimento de um ambiente virtual tridimensional multiusuário para Internet, Dissertação de Mestrado, Florianópolis, SC, 2003.
- [11] REBELO, Irla Bocianoski. Realidade Virtual Aplicada À Arquitetura e Urbanismo: Representação, Simulação e Avaliação De Projetos, Dissertação de Mestrado, Florianópolis, SC, 1999.
- [12] LUZ, Rodolfo Pinto da. Proposta de Especificação de uma Plataforma de Desenvolvimento de Ambientes Virtuais de Baixo Custo, Dissertação de Mestrado, Florianópolis, SC., 1997, p.12.
- [13] TRAUER, Eduardo. Concepção De Feiras Virtuais Como Instrumento De Marketing Interativo, Dissertação de Mestrado, Florianópolis, SC., 1998.
- [14] TUTORIAL XML. Disponível em <[http://www.gta.ufrj.br/grad/00\\_1/miguel/link3.htm](http://www.gta.ufrj.br/grad/00_1/miguel/link3.htm)> Acesso em 20.07.2004.
- [15] LENGYEL, Eric. Mathematics for 3D Game Programming And Computer Graphics. Hingham, Massachussets: Charles River Media, Inc, 2002.

- [16] ISDALE, Jerry. What is virtual reality? IEEE Virtual Reality 2003, Los Angeles, CA, EUA, Março, 2003.
- [17] BYRNE, Christine M. The Use of Virtual Reality as Educational Tool. A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor Philosophy at University of Washington. Washington: University of Washington, 1996.
- [18] PIMENTEL, K.; TEIXEIRA, K. Virtual reality – through the new looking glass. 2. Ed. New York : MacGraw-Hill, 1995.
- [19] LATTA, J. N.; OBERG, D. J. A conceptual virtual reality model. IEEE Computer Graphics & Applications, 14: p. 23-29, Jan. 1994.
- [20] SUTHERLAND, Ivan. The Ultimate Display. NY: IFIP, p.506-508, 1965.
- [21] OPENGL. Disponível em <<http://www.opengl.org/>> Acesso em 25.07.2004.
- [22] MAYA SHADERS. Disponível em <<http://www.highend3d.com/maya/shaders>> Acesso em 02.08.2004.
- [23] BioRUST Design Community. Disponível em <[http://biorust.com/index.php?page=tutorial\\_detail&tutid=10](http://biorust.com/index.php?page=tutorial_detail&tutid=10)> Acesso em 02.08.2004.
- [24] BUMP MAP APPLET. Disponível em <<http://users.interfriends.net/maurid/BumpMapping.htm#>> Acesso em 02.08.2004.
- [25] CÔRREA, Renata; GOMES, Silmara Pedretti. Mapa de Textura: Mip-Mapping, Monografia apresentada à Faculdade de Engenharia Elétrica e da Computação da Universidade de Campinas, Departamento de Engenharia da Computação e Automação Industrial. Campinas, SP, 2004.
- [26] UNIDEV. Disponível em <<http://www.unidev.com.br/artigos/realtimeshadowshibrid000.asp?id=232>> Acesso em 23.07.2004.
- [27] DICIONÁRIO GAMES BRASIL. Disponível em <<http://www.gamesbrasil.com.br/dicionario.php>> Acesso em 23.07.2004.
- [28] WOKSEPP, Stefan, TULLBERG, Odd. VR modeling in Building Construction – Some experiences and directions. The 20th CIB W78 Conference on Information Technology in Construction, Waiheke Island, Auckland, New Zealand, Abril 2003.
- [29] PILLERS, Michelle. MCAD Renaissance of the 90's - A Report on the State of the Art. CADesign Magazine, Março, 1998.
- [30] CHANG, Allen Y. A Survey of Geometric Data Structures for Ray Tracing. Technical Report, Department of Computer and Information Science, Polytechnic University, Outubro, 2001.
- [31] BORLAND. Disponível em <<http://www.borland.com/>> Acesso em 30.06.2004.

## 9. Apêndices

### 9.1. Apêndice 1

- **Adicionando uma nova obra**

Para adicionar uma nova obra execute os seguintes passos:

1. No menu principal clique em Arquivo;
2. Clique na opção Nova Obra...;
3. Na caixa de diálogo selecione o diretório e digite o nome da nova obra;
4. Clique no botão Save.

Obs.: O arquivo da obra será salvo com a extensão (.XML).

- **Abrindo uma obra**

Para abrir uma obra já existente:

1. No menu principal clique em Arquivo;
2. No submenu Abrir, clique em Obra...;
3. Na caixa de diálogo selecione o diretório e o arquivo (.XML) da obra que deseja abrir;
4. Clique no botão Open.

Obs.: Pode-se também abrir uma obra existente clicando diretamente da lista das obras que foram abertas recentemente.

- **Inserindo projetos**

Para inserir um projeto em uma que está aberta:

1. No menu principal clique em Arquivo ;
2. Clique em Inserir Projeto...;
3. Na caixa de diálogo selecione o diretório e o arquivo (.XML) do projeto que deseja inserir;
4. Clique no botão Open.

- **Navegação no ambiente**

Os seguintes comandos são utilizados para navegação no ambiente 3D:

1. As setas Up e Down são utilizadas para mover a câmera para frente e para trás no plano horizontal;

2. As setas Right e Left são utilizadas para virar a câmera à direita e à esquerda no plano horizontal. Esses comandos forem executados conjuntamente com a tecla Alt pressionada, o movimento da câmera será andar à direita e à esquerda, sem girar;
3. As teclas PageUp e PageDown são utilizadas para movimentar para cima e para baixo no plano vertical;
4. Todos os movimentos da câmera são executados com uma velocidade padrão.  
Para aumentar a velocidade, execute os movimentos com a tecla Shift pressionada;  
Para diminuir a velocidade, execute os movimentos com a tecla Control pressionada;
5. As teclas Home e End são utilizadas para “olhar” para cima e para baixo;
6. Pressionando juntamente as teclas Control+Home, a câmera volta para a posição inicial;
7. A câmera também poderá ser movimentada em todas as direções no plano horizontal através do mouse. Basta manter o botão esquerdo pressionado e mover o mouse na direção desejada. A velocidade sempre será padrão na movimentação com o mouse.