CAROLINE MAZETTO MENDES

VISUALIZAÇÃO 3D INTERATIVA APLICADA À PRESERVAÇÃO DIGITAL DE ACERVOS NATURAIS E CULTURAIS

CURITIBA

2010

CAROLINE MAZETTO MENDES

VISUALIZAÇÃO 3D INTERATIVA APLICADA À PRESERVAÇÃO DIGITAL DE ACERVOS NATURAIS E CULTURAIS

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.
Orientadora: Profa. Dra. Olga Regina Pereira Bellon
Co-Orientador: Prof. Dr. Luciano Silva

CURITIBA

2010

AGRADECIMENTOS

Aos professores Olga Regina Pereira Bellon e Luciano Silva, pela orientação e amizade. Agradeço a oportunidade de participar do Grupo de Pesquisa IMAGO.

Ao CNPq, FINEP, Fundação Araucária e CAPES, pelo financiamento deste mestrado.

A todos os colegas do IMAGO, principalmente ao Alexandre Vrubel e Beatriz Andrade por toda ajuda e contribuição para este trabalho.

Ao Dyego Rogher Drees, pelo apoio em todos os momentos. Agradeço o incentivo, a paciência e por todo conhecimento transmitido para a realização deste mestrado.

Aos meus pais, pelo apoio, compreensão e por terem investido com muito esforço na minha educação.

CONTEÚDO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE CÓDIGOS

LISTA DE ALGORITMOS

RESUMO

A visualização de modelos 3D de objetos de acervos naturais e culturais é uma importante ferramenta para a área de preservação digital, assim como para o desenvolvimento de aplicações científicas inovadoras e criação de museus virtuais interativos. A maioria dos projetos da área de computação, voltados a este tema, concentra os esforços no processo de digitalização do patrimônio cultural e em técnicas para aumentar o realismo na visualização dos modelos 3D. A visualização remota através da Internet dos modelos 3D gerados não tem recebido a atenção merecida.

Um dos motivos do pouco interesse se deve ao fato de que as tecnologias existentes para visualização de modelos 3D realistas na Internet não possuem as características necessárias para uma visualização eficiente. Este trabalho propõe a criação de um visualizador de modelos 3D realistas de objetos de acervos naturais e culturais, capaz de proporcionar alto grau de interatividade e usabilidade para os usuários. Com esta nova solução para visualização 3D em navegadores *web*, pode-se auxiliar atividades educacionais e tornar o estudo remoto mais atrativo e eficiente.

ABSTRACT

The realistic visualization of 3D models of natural and cultural assets is an important tool in the digital preservation, as well as for developing innovative scientific applications and creating interactive virtual museums. Most Computing projects dedicated to this theme focus on the cultural heritage digitalization process and on techniques to make the 3D models more realistic for visualization. However, the remote visualization of 3D models through the Internet has not received the attention it deserves.

One of the reasons for this low interest is that the existing technologies for realistic visualization of 3D models on the Internet do not have the necessary characteristics for an efficient visualization. This work proposes a new viewer for realistic 3D models of natural and cultural assets that will provide a high degree of interactivity and usability for the users. Introducing a new solution for 3D visualization in web browsers can help educational activities and make the remote study more attractive and efficient.

CAPÍTULO 1

INTRODUÇÃO

A preservação digital permite às gerações futuras compreender e contextualizar a história e a cultura dos seus povos [21]. A necessidade de garantir que o patrimônio cultural da humanidade seja preservado digitalmente tem motivado o desenvolvimento de novas tecnologias, fomentado projetos científicos e apoiado atividades de conservação e restauração, entre outras [22].

Um dos objetivos da preservação digital de artefatos é reunir o maior conjunto possível de dados sobre os mesmos antes da sua degradação. Uma vez obtidos e organizados, estes dados podem ser usados em diversas aplicações, como por exemplo, para estudos arqueológicos, ensino de história e cultura, ou mesmo para divulgação e disponibilização na Internet através de exposições virtuais [8, 13].

Nesta última década, as áreas de computação gráfica, processamento de imagens e visão computacional têm contribuído de forma significativa para o desenvolvimento de muitas dessas aplicações [30]. Neste contexto, sistemas para digitalização e visualização realista de modelos tridimensionais (3D) de objetos de acervos naturais e culturais têm se tornado uma ferramenta fundamental para suporte às atividades de pesquisa, educação, entretenimento e cultura.

Através dos sistemas de visualização, os artefatos podem ser acessados e explorados virtualmente, com alto nível de detalhes, reduzindo assim o risco de danos irreversíveis devido ao transporte ou manipulação física dos mesmos. Outras vantagens no uso destes sistemas é a possibilidade de investigar características dos artefatos com maior segurança e confiabilidade, como por exemplo, calcular o seu volume, dimensões, detectar áreas frágeis, rachaduras ou fendas [12].

Quando preservados digitalmente, os objetos podem ser representados por modelos 3D que consistem basicamente de uma superfície geométrica coberta por um mapa de textura [10]. A superfície geométrica dos objetos pode ser obtida por diferentes dispositivos [3], e entre estes os scanners 3D de triangulação laser são considerados mais precisos e adequados [28]. Estes scanners utilizam imagens de profundidade (range images [3]), adquiridas de diferentes vistas do objeto, que uma vez integradas fornecem informações sobre a sua geometria [33].

Câmeras digitais coloridas de alta resolução podem complementar o processo de modelagem 3D, de forma a gerar uma textura realista, preservando características como cor e reflectância do objeto [2]. É importante destacar que projetos recentes relacionados à preservação digital de artefatos [13, 22] têm explorado técnicas inovadoras para a criação de modelos 3D realistas. Entretanto, a visualização dos modelos 3D não recebe a atenção merecida, como será apresentado neste trabalho.

Para garantir um alto grau de interatividade com o usuário, os sistemas de visualização de modelos 3D precisam fornecer acesso rápido e prático, com recursos de usabilidade para atender o maior número de pessoas possível, e principalmente desempenhar com eficiência as suas funções. Neste sentido, quando um modelo 3D de um objeto está sendo visualizado interativamente pelo usuário, de tal forma que diferentes vistas (*i.e.* imagens) estejam sendo mostradas sucessivamente no sistema, é fundamental que este sistema proporcione uma interação realista com o objeto.

A taxa de quadros por segundo pode ser uma medida para calcular esta eficiência, pois quanto maior esta taxa, mais suave e natural é a sensação de movimento do modelo 3D durante a visualização. Em [4] foram realizados testes com uma aplicação de visualização 3D e os autores observaram a necessidade de pelo menos 6 quadros por segundo para possibilitar a navegação em mundos virtuais e 20 ou mais quadros por segundo para haver uma interatividade apropriada. Entretanto, a taxa ideal depende de cada aplicação, sendo importante que haja uma resposta em tempo real para cada ação do usuário.

A maioria dos projetos na área da preservação digital prioriza o processo de criação

dos modelos 3D e de geração de texturas enquanto a visualização dos acervos não recebe a atenção merecida. Um dos motivos para isso são as limitações de algumas ferramentas existentes para possibilitar a visualização 3D. O alto tempo de resposta para as ações do usuário, interfaces complexas e dificuldades para realizar as operações sobre os modelos 3D, como por exemplo, ampliar e reduzir, são exemplos de limitações.

Esta dissertação apresenta um estudo sobre a visualização de modelos 3D realistas de acervos naturais e culturais, assim como a disponibilização dos mesmos através da Internet. Com o objetivo de minimizar as limitações existentes na visualização 3D, foram desenvolvidas duas ferramentas: o *plugin* IMAGO e o IMAGO 3D Viewer.

No Capítulo 2 desta dissertação são apresentados os projetos relacionados à preservação digital e seus esforços para permitir a visualização 3D. Questões envolvendo a disponibilização de modelos 3D realistas através da Internet são discutidas no Capítulo 3. O aprimoramento de um visualizador 3D existente, o *plugin* IMAGO, é detalhado no Capítulo 4 e a criação de um novo visualizador 3D é apresentado no capítulo 5. O Capítulo 6 apresenta o Museu Virtual 3D do IMAGO, no qual foram incorporados os estudos e as ferramentas desenvolvidas neste trabalho.

CAPÍTULO 2

ESTADO DA ARTE

Na maioria dos projetos relacionados com preservação digital, as equipes desenvolvem seus próprios sistemas para visualização de modelos 3D [6, 14, 20]. Os sistemas de visualização podem integrar diferentes aplicativos, servidores, banco de dados e ferramentas computacionais para permitir visualização 3D. As ferramentas de visualização 3D, também chamadas de visualizadores 3D, podem permitir a visualização via Internet, porém são poucas as que possuem essa funcionalidade. Os visualizadores 3D podem ser *stand-alone* ou *plugins* para navegadores *web*.

Os visualizadores *stand-alone* são disponibilizados como aplicativos e precisam ser copiados e instalados no computador do usuário, enquanto visualizadores para navegadores *web* geralmente requerem a instalação de *plugins* que pode ser feita diretamente a partir do próprio navegador *web*. Um desafio desses visualizadores 3D para *web* é disponibilizar uma interface que seja intuitiva, proporcionando alto grau de interatividade, mesmo quando é exibida grande quantidade (volume) de dados.

A seguir são apresentadas as principais tecnologias que podem ser utilizadas para criação de visualizadores e exibição de conteúdo 3D na Internet. Também são mostrados os esforços realizados por alguns projetos na área de preservação digital para disponibilizar seus dados 3D na Internet e suas tecnologias alternativas para visualização 3D.

2.1 Tecnologias

Os projetos relacionados à preservação digital podem criar suas próprias ferramentas para permitir a visualização 3D na Internet ou podem utilizar soluções existentes. Os visualizadores 3D podem ser desenvolvidos utilizando a linguagem de programação C++ e

a biblioteca gráfica multi-plataforma Open Graphics Library (OpenGL). Outra biblioteca gráfica é a Microsoft DirectX, que permite o desenvolvimento de jogos para o sistema operacional Windows. Para a visualização em navegadores *web* podem ser utilizadas Application Programming Interfaces (APIs) para criação de *plugins*, como por exemplo a Netscape Plugin Application Programming Interface (NPAPI)[1], a qual possui cerca de 15 funções no total entre iniciar, criar e destruir instâncias de *plugin*, entre outras.

Outra maneira de criar visualizadores 3D é utilizar a linguagem Java e a API Java 3D[2], a qual é baseada nas bibliotecas OpenGL e DirectX. O Java 3D utiliza conceitos de grafos de cena, ou seja, os objetos 3D são criados e posicionados em um grafo de cena, que os combina em uma estrutura de árvore[3]. Outras APIs também podem ser utilizadas para o desenvolvimento de gráficos 3D em Java, como o Java OpenGL (JOGL)[4] que permite utilizar funções OpenGL sem abstrações, diferentemente do Java 3D. Através de Java Applets é possível extender as aplicações para navegadores *web*, sendo uma boa solução tanto para desenvolvedores quanto para os usuários, que na maioria das vezes possuem suporte a Java em seus computadores.

No ano de 2009, a Google apresentou uma API chamada O3D[5] para a criação de objetos 3D e exibição dos mesmos em navegadores *web*, através da instalação de um *plugin* desenvolvido para esse fim. O *plugin* possui versões para os navegadores *web* mais comuns, como por exemplo o Internet Explorer, Firefox, Chrome e Safari, e permite a interação com Javascript para ocasionar mudanças no conteúdo 3D em tempo real. Como toda tecnologia recente, a API está em fase de desenvolvimento e avaliação quanto ao seu potencial para criação de aplicativos.

Entre as tecnologias que possuem licença de *software* proprietário temos o Adobe Flash[6], que em conjunto com o componente PaperVision 3D permite criar e exibir

[1]https://developer.mozilla.org/en/Gecko_Plugin_API_Reference
[2]https://java3d.dev.java.net
[3]http://www.inf.pucrs.br/ manssour/Java3D/index.html
[4]https://jogl.dev.java.net
[5]http://code.google.com/intl/pt-BR/apis/o3d/
[6]http://www.adobe.com/flashplatform/

gráficos 3D em navegadores *web*. O DeMicron Wirefusion[7] e o Quest3D[8] são exemplos de *softwares* proprietários que também permitem a visualização 3D na Internet.

O Virtual Reality Modeling Language (VRML) é uma linguagem para modelagem geométrica que possibilita descrever objetos 3D e cenários 3D. É usada pela maioria dos *websites* para descrever os modelos 3D [35]. Desde o seu primeiro lançamento, no início de 1995, a VRML é atualmente utilizada na visualização científica de modelos arquitetônicos, simulações em 3D e até em desenhos animados [26].O conteúdo 3D criado em VRML pode ser exibido em navegadores *web* através do uso dos visualizadores VRML, os quais serão descritos na próxima seção.

Outra forma de visualização de conteúdo 3D em navegadores *web* é utilizar a tecnologia X3D - Extensible 3D[9]. O X3D pode ser considerado um melhoramento do VRML, incorporando os avanços dos recursos gráficos como por exemplo animações e multi-texturas. O X3D é baseado em Extensible Markup Language (XML), o que permite melhor interoperabilidade entre aplicações, como por exemplo comunicação em tempo real via rede. Visualizadores VRML mais recentes também são capazes de exibir conteúdo criado em X3D.

2.1.1 Visualizadores VRML

Para visualizar um modelo 3D em VRML é necessário um visualizador VRML, geralmente um *plugin* para um navegador *web*, como por exemplo o Octaga Player[10], Cosmo Player[11] e Cortona3D[12]. O visualizador VRML lê e interpreta o arquivo de descrição do mundo virtual (arquivo com extensão .wrl), o renderiza em uma janela e provê uma interface para visualização e interação. Como no Java 3D, um mundo VRML é um grafo hierárquico em forma de árvore e as hierarquias são criadas através de nós de agrupamento [9].

[7] http://www.demicron.com/
[8] http://quest3d.com/
[9] http://www.web3d.org/x3d
[10] http://www.octaga.com
[11] ovrt.nist.gov/cosmo
[12] www.cortona3d.com

Os visualizadores VRML existentes possuem diversas incompatibilidades de funcionamento e podem ser dependentes do navegador *web* e do sistema operacional [29]. Escolher o visualizador VRML mais adequado para uma determinada aplicação pode ser feito com base em estudos como em [36], o qual apresenta uma comparação em termos de qualidade visual do conteúdo 3D e de desempenho entre diferentes visualizadores VRML.

Em alguns casos, um determinado visualizador VRML pode ser recomendado, como por exemplo o Cortona3D, no projeto relatado em [6], porém é o usuário quem decide qual o visualizador deseja instalar. Além disso, o correto funcionamento dos *plugins* VRML nos navegadores *web* depende da configuração do computador do usuário. Apesar da linguagem VRML ser eficiente para a modelagem geométrica de cenários 3D, o fato de não possuir um visualizador padrão é uma limitação considerável.

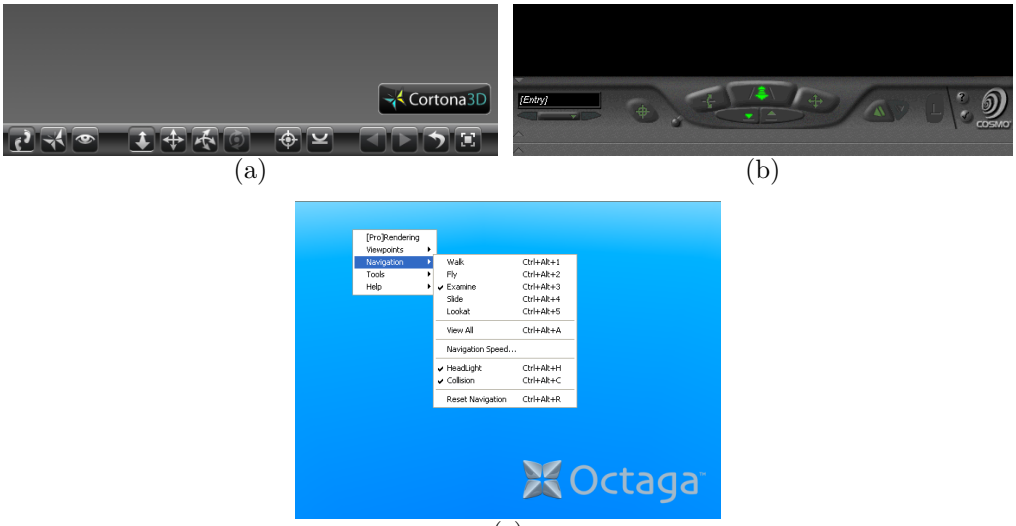
Nos experimentos realizados neste trabalho, diversos problemas nos visualizadores VRML foram encontrados, entre eles: comportamentos diferentes na visualização dos modelos 3D quando utilizados computadores com diferentes configuração de hardware ou sistema operacional; inoperabilidade repentina do sistema sem mensagem de erro; lentidão intermitente na visualização, mesmo com modelos 3D simples; alto consumo de memória e processamento; entre outros.

Para diversas aplicações, como por exemplo para pesquisas arqueológicas usando modelos 3D de artefatos, é importante que a visualização seja a mais realista possível e para tal são necessários modelos 3D com alta resolução, o que implica em arquivos VRML mais complexos. Os visualizadores VRML perdem desempenho a medida que a complexidade dos modelos 3D aumenta, influenciando negativamente na taxa de quadros por segundo durante o processo de visualização interativa dos mesmos.

Outro ponto negativo quanto ao VRML é que a interface de usuário para visualização 3D ainda não está padronizada. Mesmo os visualizadores VRML que cumprem o padrão ISO diferem tanto no número de opções de controle quanto na disposição dessas opções na tela [35]. Apesar da interface não ser padronizada, os visualizadores VRML sempre expõem uma interface para o usuário e, dessa maneira, os programadores precisam se

preocupar apenas com o conteúdo 3D que será exibido. Para os usuários, o fato de não existir uma interface padrão traz dificuldades na interação com os modelos 3D, já que é necessário se adequar à interface de cada visualizador VRML.

A Figura 2.1 mostra as diferentes interfaces de visualizadores VRML. O Cortona 3D (Figura 2.1a) apresenta uma barra de ferramentas com ícones logo abaixo da tela de exibição do modelo 3D e o Cosmo Player (Figura 2.1b) possui um painel de navegação. Já o Octaga Player (Figura 2.1c) em algumas versões, não dispõe uma interface visível, sendo necessário clicar com o botão esquerdo do *mouse* para exibir o menu com as opções para manipulação dos modelos 3D. Como é possível observar, alguns ícones presentes nas interfaces dos visualizadores possuem os mesmos símbolos para determinadas operações, porém os mesmos não são intuitivos para usuários iniciantes.



(a) (b) (c)

Figura 2.1: Exemplos de interfaces de visualizadores VRML em um navegador *web*: (a) Cortona3D, (b) Cosmo Player e (c) Octaga Player.

2.2 Projetos de Preservação Digital e a Visualização 3D

Os projetos relacionados à preservação digital que disponibilizam seus acervos através da Internet utilizam ferramentas de visualização de conteúdo 3D para fornecer interatividade aos usuários. Infelizmente, na maioria das vezes essa interatividade é comprometida devido às limitações dessas ferramentas.

No “The Great Buddha Project”[13], um trabalho pioneiro voltado à preservação e restauração de antigos objetos da cultura japonesa, foram desenvolvidos diversos estudos que contribuíram para o mapeamento e solução de problemas em muitas etapas do processo de preservação digital. O principal objetivo do projeto foi desenvolver um *software* para criar modelos 3D através da observação dos objetos reais. Os modelos 3D gerados foram utilizados para obtenção de informações e simulação da aparência original das esculturas. Neste projeto foram digitalizadas estátuas de grandes Buddha, como por exemplo Asuka, Kamakura e Nara Buddha, e faces da Biblioteca Sul do templo de Bayon no Cambodja. Entretanto, este projeto não provê meios para visualização dos modelos 3D preservados digitalmente através da Internet, o que seria interessante para fomentar atividades de ensino e pesquisa.

Ikeuchi *et al.* [27] apresentam um sistema para visualização de modelos 3D com grande volume de dados. Os usuários podem realizar a navegação em tempo real e acessar informações sobre os modelos 3D, podendo também associar informações a regiões específicas nesse modelo através da funcionalidade de selecionar regiões de um modelo 3D. Esse sistema foi utilizado para auxiliar a restauração do templo de Bayon (Figura 2.2), permitindo aos pesquisadores o armazenamento e gerenciamento das informações sobre os artefatos. Na implementação deste sistema foi utilizada a linguagem de programação C++ e a biblioteca gráfica OpenGL. É importante destacar que na seção de trabalhos futuros deste artigo os autores ressaltam que pretendiam estender o sistema desenvolvido para utilização na Internet, o que corrobora com os objetivos do presente trabalho.

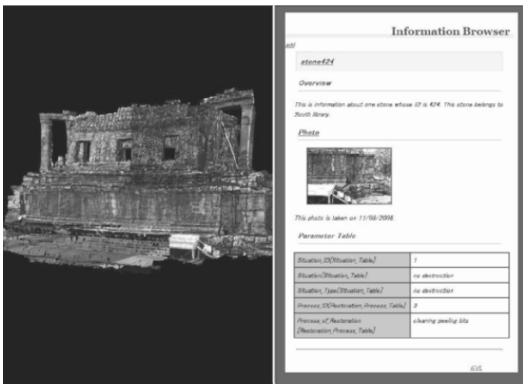
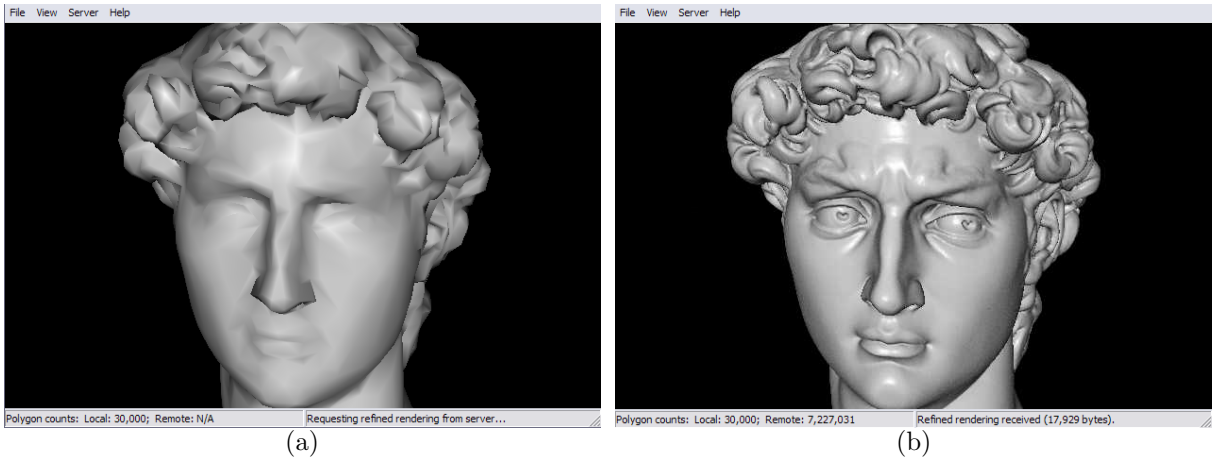


Figura 2.2: Exemplo da interface do sistema de visualização 3D desenvolvido para a Biblioteca Sul do Templo de Bayon [27].

No “The Digital Michelangelo Project”[22] foram digitalizadas grandes esculturas feitas por Michelangelo. Neste projeto os autores discutem também alguns problemas que podem ocorrer na disponibilização digital destas obras, como por exemplo a distribuição não autorizada dos modelos 3D, já que podem possuir direitos autorais. Outra preocupação foi a reconstrução física dos objetos que pode ser feita a partir da geometria e imagens dos modelos 3D. Para evitar esses problemas, técnicas de segurança são utilizadas no sistema de visualização proposto no projeto, como por exemplo armazenar modelos 3D de alta resolução apenas no servidor.

Neste projeto também foi desenvolvida uma ferramenta de visualização 3D *stand-alone* [20] que permite aos usuários interagir com modelos 3D de baixa resolução. Nesta ferramenta o usuário pode visualizar detalhes dos artefatos através da renderização remota e uma arquitetura cliente-servidor. Na renderização remota, imagens são obtidas a partir de modelos 3D de alta resolução renderizados em um servidor, as quais são enviadas via Internet para a visualização no computador do usuário (cliente), como pode ser observado na Figura 2.3. No entanto, não existe uma versão para realizar a visualização em navegadores *web*, o que tornaria o acesso mais apropriado e prático para os usuários.



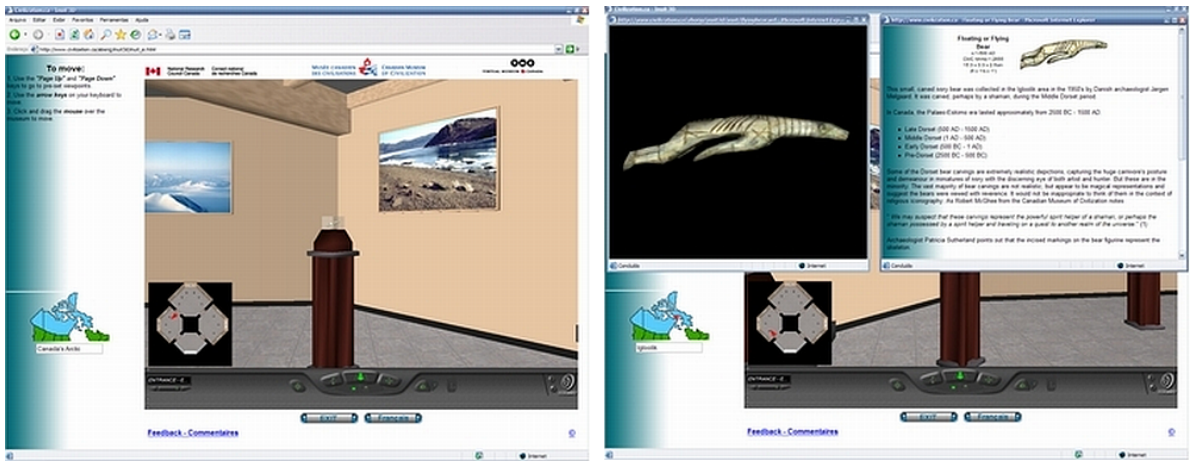
(a) (b)

Figura 2.3: Exemplo de visualização usando o sistema proposto no “The Digital Michelangelo Project”: (a) modelo 3D de baixa resolução renderizado localmente no cliente e (b) imagem do modelo 3D de alta resolução renderizado no servidor e transmitida ao cliente via Internet.

Em [14], exemplos do patrimônio cultural da República Checa são apresentados em um ambiente virtual. Modelos 3D de construções, monumentos e outros objetos foram

criados com o intuito de preservar a cultura da época e oferecer informações para os interessados em história e disciplinas relacionadas. O objetivo principal deste projeto foi criar modelos 3D de artefatos e utilizá-los para a reconstrução virtual de sítios arqueológicos, com o auxílio de informações obtidas em investigações realizadas por arqueólogos. No processo de reconstrução virtual, modelos VRML são disponibilizados para serem visualizados na *web* através de visualizadores VRML. Os autores ressaltam as desvantagens dos visualizadores de realidade virtual existentes, como por exemplo a necessidade de instalação de *plugins* e dificuldades no uso dessas ferramentas por usuários iniciantes.

O Inuit 3D é um projeto do Museu Canadense da Civilização [6] que permite que modelos 3D de obras de arte sejam visualizados através de um cenário virtual 3D na forma de construções representando um museu e seus arredores, como mostrado na Figura 2.4. O museu virtual desse projeto necessita de um visualizador VRML para exibição de modelos 3D em um navegador *web*. Os autores comentam a necessidade de utilizar outras tecnologias para visualização 3D, para evitar a necessidade de instalação de *plugins* e diminuir o tempo de transferência pela Internet dos arquivos VRML.



(a) (b)

Figura 2.4: Projeto Inuit 3D: (a) visualização do ambiente 3D, (b) exibição de um modelo 3D.

Outros trabalhos apresentam tecnologias alternativas para realizar a visualização 3D na Internet. Cheng e Basu [5] desenvolveram um sistema para visualização de modelos 3D e conteúdo digital utilizando a API Java 3D. Esta tecnologia pode ser utilizada em várias

aplicações, como por exemplo para a criação de museus virtuais. Além disso, ela permite que usuários interajam com imagens de baixa resolução em conjunto com imagens de alta resolução. Entretanto, o sistema de visualização proposto pelos autores não possui uma versão para navegadores *web*, o que poderia ser feito combinando o Java 3D com Java Applets, tornando o sistema mais acessível.

No projeto "Augmented Representation of Cultural Objects"(ARCO) [34] foi desenvolvida uma tecnologia para criar e gerenciar exposições de museu virtual para uso na *web*. O sistema consiste de três componentes principais: produção, gerenciamento e a visualização do conteúdo dos museus. Os usuários podem visualizar e interagir com modelos 3D de objetos culturais, que são exibidos com visualizadores VRML (Figura 2.5), podendo também utilizar realidade aumentada. Um estudo sobre a usabilidade desse sistema [16] mostrou que usuários finais e curadores de museus tiveram problemas com a navegação no sistema e consideraram a interface muito complexa, necessitando de instruções mais detalhadas para a sua utilização.

Figura 2.5: Exemplo de visualização de um artefato no projeto ARCO.

Através do estudo sobre as formas de disponibilização e visualização de modelos 3D nos projetos relacionados à preservação digital, pôde-se avaliar as principais tecnologias utilizadas e suas características. As limitações dessas tecnologias, como por exemplo interfaces complexas, baixo desempenho e baixo nível de interatividade na visualização dos modelos 3D realistas, motivaram o desenvolvimento das ferramentas propostas no presente trabalho.

CAPÍTULO 3

DISPONIBILIZAÇÃO DE MODELOS 3D NA INTERNET

Uma das etapas do processo de preservação digital é a disseminação da informação para as pessoas em geral. No que diz respeito aos objetos naturais e culturais, a visualização 3D transmite maior realismo e interação comparada a visualização 2D que faz uso apenas de imagens. A forma mais abrangente para divulgação das pesquisas e estudos da preservação digital é a disponibilização através da Internet, considerando o crescente número de computadores com acesso a Internet ao redor do mundo. Este Capítulo apresenta algumas considerações sobre o processo de disponibilização dos modelos 3D realistas para visualização através da Internet.

3.1 Segurança

Modelos 3D realistas, obtidos através da digitalização de alta resolução de patrimônios culturais, podem exigir proteção para evitar a violação dos direitos autorais. Medidas de segurança precisam ser consideradas na disponibilização dos modelos 3D, evitando assim a reconstrução física dos objetos ou a utilização não autorizada na Internet. Temos como exemplo o Projeto Michelangelo Digital [22] que digitalizou dez grandes estátuas de Michelangelo, como a estátua de Davi, as quais representam importantes obras do patrimônio artístico da Itália. Autoridades italianas permitem a distribuição dos modelos 3D criados, com baixo nível de detalhes, apenas para estudos e não para uso comercial [19].

Os sistemas de visualização podem controlar o acesso aos modelos 3D através da realização de cadastros e autenticação. O projeto do Museu Virtual 3D do IMAGO agrega essa característica importante [25], que consiste em disponibilizar modelos 3D com diferentes níveis de detalhes dependendo do perfil do usuário. Pessoas comuns, estudantes

ou curiosos podem visualizar os modelos 3D de baixa resolução com boa qualidade de detalhes, permitindo um primeiro contato com os objetos. Pesquisadores ou professores, após análise e confirmação das informações fornecidas no cadastro, podem adquirir uma autorização para acessar os modelos 3D com alta resolução, para fins de pesquisa e estudo remoto.

Uma abordagem para proteger modelos 3D consiste em gerar e inserir um sinal imperceptível, chamada de marca d'água, no modelo 3D [15]. A marca d'água pode carregar informações sobre o proprietário do modelo 3D ou autorizações para usuários e distribuidores. Os projetos de preservação digital com visualização na Internet também podem incorporar a inserção dinâmica de marcas d'água em seus sistemas de visualização. Informações como nome, data de acesso e endereço IP podem ser armazenadas no modelo 3D antes da visualização, ajudando assim a identificação dos usuários no caso de violação e disputas judiciais dos direitos autorais.

A renderização remota pode ser uma estratégia para proteger os modelos 3D [20], uma vez que os modelos de alta resolução não são enviados para o computador do usuário para a visualização. Os usuários interagem com modelos de baixa resolução e ao parar a movimentação, o servidor renderiza o modelo de alta resolução utilizando os mesmos ângulos e parâmetros de visualização do cliente, enviando essa imagem para ser visualizada pelo usuário. Assim, os modelos 3D com alta resolução ficam protegidos no servidor, possibilitando a incorporação de técnicas de segurança no sistema, como por exemplo renderizar o modelo com pequenas modificações nos ângulos de visualização, inserir ruído nas imagens ou limitar a visualização para determinadas regiões do modelo 3D, entre outras.

3.2 Perfil do usuário

Na visualização 3D para fins de pesquisa e estudo remoto é necessário considerar os diferentes perfis de usuários. Pessoas de outras áreas não relacionadas à informática

podem apresentar maiores dificuldades para visualizar os modelos 3D, como por exemplo profissionais da área de História, Antropologia, Museologia, Sociologia, entre outros. O processo de visualização 3D envolve desde a instalação dos aplicativos até a interação com o conteúdo 3D, operações estas que podem ser consideradas complexas para alguns usuários.

A configuração dos computadores pode variar para cada usuário, que pode possuir ou não os recursos necessários para permitir a visualização 3D, como por exemplo placas aceleradoras 3D que permitem maior desempenho gráfico para a aplicação. Também é preciso considerar os diferentes sistemas operacionais e tecnologias que precisam estar presentes nos computadores, assim como a velocidade de conexão da Internet. Quanto menos recursos forem necessários, mais fácil será o acesso aos modelos 3D.

3.3 Tecnologias

A visualização 3D na Internet está se tornando um atrativo no desenvolvimento de *websites* interativos. Cada vez mais existe o interesse por parte de desenvolvedores e usuários, incentivando a criação de novas tecnologias para esse fim. Sobre as tecnologias escolhidas para permitir a visualização 3D ou as utilizadas na sua criação, três aspectos principais precisam ser considerados: portabilidade, instalação e interface gráfica para o usuário.

A portabilidade é uma característica que permite o funcionamento dos visualizadores 3D nos diferentes sistemas operacionais. Essa característica influencia tanto os desenvolvedores quanto os usuários. Visualizadores desenvolvidos em Java e OpenGL são multi-plataforma, enquanto que o uso de outras tecnologias exigem a criação e disponibilização de versões para cada sistema operacional ou navegador *web*. Na falta de versões específicas, usuários podem ficar impedidos de visualizar os modelos 3D para realização de pesquisas e estudo remoto.

A instalação dos visualizadores 3D precisa ser um processo simples e transparente para não desmotivar os usuários. Programas instaladores auxiliam nessa tarefa, informando

todos os passos da instalação e comunicando a necessidade de outros recursos necessários. O ideal é que todos os recursos, como por exemplo bibliotecas que possam não estar presentes em alguns sistemas operacionais, sejam incorporados na criação do visualizador, evitando erros durante a sua execução.

Os visualizadores 3D precisam disponibilizar uma interface gráfica para o usuário. Quando os projetos optam por soluções existentes, como por exemplo visualizadores VRML, é necessário conhecer a interface dos mesmos para recomendar o mais adequado para os seus usuários. Desenvolver um visualizador próprio pode ser uma tarefa árdua em termos de codificação, mas possibilita a incorporação de critérios ergonômicos na criação da interface, proporcionando maior usabilidade.

3.4 Transmissão

Modelos 3D realistas geralmente necessitam de uma grande quantidade de dados para sua representação, resultando em arquivos de modelos 3D grandes que podem demorar para serem transmitidos pela Internet. Apesar do aumento das conexões de alta velocidade com o passar dos anos no Brasil[1], parte dos usuários da Internet possuem dificuldades para realizar *download* de arquivos muito grandes. A demora na transmissão pode ser um fator desmotivador na visualização 3D na Internet, por esse motivo algumas medidas devem ser incorporadas nos sistemas de visualização de modelos 3D realistas.

Para a preservação digital é desejável uma grande quantidade de dados precisos na representação da malha 3D do objeto. Porém, na visualização 3D na Internet, a simplificação pode ser essencial, gerando modelos 3D com menor resolução e preservando sua qualidade e precisão, além de permitir a transmissão de arquivos menores. Um exemplo de abordagem para simplificação de malhas desenvolvido por Hoope [11] gera resultados precisos com diferentes níveis de detalhes para cada objeto. Além disso, é possível gerar *normal maps* [18] dos modelos com maior nível de detalhes, proporcionando uma maior

[1] http://www.cisco.com/web/BR/barometro/barometro.html?sid=183333_3

sensação de relevo na visualização dos modelos 3D simplificados.

A compressão dos modelos 3D pode reduzir consideravelmente o tamanho dos arquivos, reduzindo também o tempo de transferência dos arquivos na Internet. Existem algoritmos para compressão 3D baseados em técnicas adaptadas da compressão 2D e também algoritmos que utilizam informações sobre a superfície 3D. Em [23] são apresentados os conceitos sobre compressão 3D assim como os algoritmos que podem ser utilizados, como por exemplo o EdgeBreaker, que considera informações da conectividade para codificação da malha 3D. A compressão também pode ser realizada utilizando o formato Zip[2], que fornece boas taxas de compressão para arquivos de modelos 3D no formato texto.

3.5 Interação com os Modelos 3D

A interação é uma característica que precisa ser considerada na visualização de modelos 3D, principalmente para fins educacionais. Com bom nível de interatividade, os estudos podem se tornar mais atraentes e facilitar o aprendizado. No nosso contexto, visualização interativa é proporcionada pelo visualizador 3D, capaz de detectar as entradas do usuário e realizar instantaneamente as operações sobre o modelo 3D. A interação pode ser feita através do uso de interfaces gráficas, dispositivos como *mouse* e teclado, ou até mesmo luvas de realidade virtual.

O nível de interatividade pode ser mensurado pelo usuário se suas ações correspondem a uma operação desejada. Operações como rotação, deslocamento, aproximar ou afastar o modelo 3D podem ser realizadas com o uso do *mouse* através de uma manipulação direta, que permite uma interação mais intuitiva para usuários iniciantes. As interfaces gráficas podem permitir uma interação mais fácil desde que os menus ou ícones possuam rótulos ou imagens intuitivas para o usuário. Geralmente os símbolos e suas operações correspondentes, consideradas padrões no ambiente de aplicações de realidade virtual, não são bem compreendidas pela maioria dos usuários.

[2]http://www.pkware.com/documents/casestudies/APPNOTE.TXT

O tempo de resposta para as ações do usuário também pode influenciar na sensação de interatividade. Modelos 3D com alta resolução podem representar demora no processo de renderização, ou seja, o computador terá que interpretar a grande quantidade de dados e criar uma imagem finalizada, que será vista pela perspectiva escolhida. Por esse motivo, os projetos precisam definir configurações mínimas necessárias para o funcionamento dos seus sistemas de visualização 3D.

A taxa de quadros por segundo (*frames per second - fps*) é outro aspecto que interfere na interação. O *fps* não possui um valor fixo e pode variar de forma considerável de acordo com modelo 3D e também pode ser limitado dependendo da configuração do computador, como o desempenho de placas de vídeo ou processadores. Apesar dos diversos valores sobre a taxa mínima ou máxima de *fps* que pode ser percebida pelo olho humano, o fato é que uma taxa muito baixa certamente irá prejudicar a interação causando um incômodo ao usuário.

Um bom *design* de interação deve ser considerado nos projetos que desejam fornecer uma boa visualização de modelos 3D. De acordo com [17], os objetivos do design de técnicas de interação nas aplicações de realidade virtual são: desempenho, usabilidade e utilidade. O desempenho significa o quão bem as atividades são executadas pelo o usuário, em termos de eficiência e produtividade. A usabilidade está relacionada com a facilidade de uso e aprendizado do usuário, e a utilidade mostra que a interação ajuda os usuários a atingir seus objetivos.

CAPÍTULO 4

PLUGIN IMAGO

O *plugin* IMAGO foi desenvolvido para permitir a visualização de modelos 3D através de um navegador *web*. O objetivo era apresentar uma nova ferramenta de visualização para o sistema de Museu Virtual 3D do IMAGO, porém essa versão foi desenvolvida para uso exclusivo e não permitia que outros projetos pudessem utilizá-la.

Como parte deste trabalho de mestrado foi desenvolvida uma nova versão que permite o uso por qualquer pessoa ou projeto. Museus podem disponibilizar de maneira segura e prática suas obras de arte em seus *websites*, tanto para divulgação quanto para tornar a visita aos museus mais atrativa. Artistas também podem apresentar a sua arte em seus *websites* pessoais, possibilitando ao seu público interagir com os objetos através da visualização de seus modelos 3D, assim como pesquisadores podem utilizar a visualização 3D para auxiliar os seus estudos e professores para suas atividades educacionais.

Para a criação do *plugin* IMAGO foi utilizada a NPAPI que permite desenvolver *plugins* para navegadores *web* da família de navegadores Mozilla/Firefox. A implementação também utiliza a linguagem C++ e a biblioteca gráfica OpenGL.

O *plugin* IMAGO está disponível em um *website*[1] e é compatível com navegadores *web* como Mozilla Firefox, Iceweasel, e Netscape, entre outros, nas plataformas Linux e Windows. A instalação em ambas as plataformas pode ser feita automaticamente com programas instaladores. Assim como em todas as aplicações gráficas 3D, o *plugin* IMAGO requer uma placa de vídeo com aceleração 3D para melhor desempenho.

[1] http://www.imago.ufpr.br/plugin

4.1 Arquitetura

O *plugin* IMAGO, além de possibilitar a visualização 3D em navegadores *web*, proporciona alta interatividade para os usuários, através de simples e intuitivas ações do *mouse* e do teclado. Também permite aos programadores personalizar a interface dependendo do seu público alvo com a utilização de um arquivo de configuração. A renderização de modelos 3D, tanto de baixa quanto de alta resolução, é feita mantendo uma boa taxa de quadros por segundo, proporcionando uma visualização interativa e realista.

4.1.1 Modo de uso

O *plugin* IMAGO aceita como entrada um arquivo no formato zip, formato de compactação muito difundido na Internet. O arquivo zip deverá conter obrigatoriamente um arquivo de configuração e pode conter um arquivo de um modelo 3D e um arquivo de textura. Esse arquivo zip deve ser especificado na chamada do *plugin* no arquivo HTML (Código 4.1.1) e deve estar localizado em um servidor *web* da aplicação.

```
1 <EMBED type = "application/x-imago-plugin"
2        name = "nome"
3        model = "nome_do_modelo.m"
4        src = "nome_do_arquivo.zip"
5        width = 800
6        height = 600 >
```

Código 4.1.1: *Chamada do* plugin *IMAGO no arquivo HTML.*

A chamada do *plugin* utiliza os seguintes parâmetros:

- **type:** especifica o tipo Multipurpose Internet Mail Extensions (MIME) do *plugin* para visualizar modelos 3D no navegador *web*.
- **name:** nome que o *plugin* será conhecido pela página no navegador *web*.
- **model:** se o modelo 3D está no arquivo zip de entrada, deve-se especificar aqui o nome do modelo com a extensão .m ou .m2. Caso obtenha o modelo 3D através de

um servidor de modelos 3D, esse parâmetro deve ser usado como requisição para o servidor.

- **src:** nome do arquivo zip de entrada que contém o arquivo de configuração e os arquivos necessários.
- **width e height:** largura e altura que a interface do *plugin* terá ao ser exibida no navegador *web*.

4.1.2 Configuração do Visualizador 3D

Com o uso de um arquivo de configuração é possível especificar de forma simples informações sobre o modelo 3D, a interface de visualização e os eventos relacionados à interface e ao modelo 3D. Como é mostrado no Código 4.1.2, o arquivo de configuração possui três seções: Menus, Events e Model.

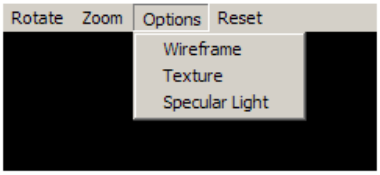
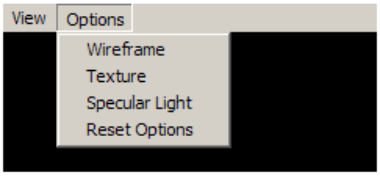
```
#Menus
View:Left,Right,Up,Down,Zoom In,Zoom Out,Reset Camera
Options:Wireframe,Texture,Specular Light,Reset Options
#Events
Left:left_event:mouse_device,keyboard_device
Right:right_event:mouse_device,keyboard_device
Up:up_event:mouse_device,keyboard_device
Down:down_event:mouse_device,keyboard_device
Reset Camera:resetcamera_event:mouse_device,keyboard_device
Zoom In:zoomin_event:mouse_device,keyboard_device
Zoom Out:zoomout_event:mouse_device,keyboard_device
Wireframe:wireframe_event:mouse_device,keyboard_device
Texture:texture_event:mouse_device,keyboard_device
Specular Light:specular_event:mouse_device,keyboard_device
Reset Options:resetoptions_event:mouse_device,keyboard_device
#Model
model=1
```

Código 4.1.2: *Exemplo do arquivo de configuração.*

Na configuração da interface são definidos os menus, os quais atualmente podem conter textos com quaisquer caracteres ocidentais, possibilitando sua internacionalização e o uso para diferentes grupos de usuários. Os menus são definidos no arquivo de configuração

após a tag "#Menus"e cada linha abaixo dela contém o rótulo do menu principal seguido pelos rótulos das opções. A Tabela 4.1 exemplifica duas maneiras distintas de montar os menus.

Tabela 4.1: Duas formas de configuração das opções no menu.

	**Rotate:** Left, Right, Up, Down **Zoom:** In, Out **Options:**Wireframe,Texture, Specular Light **Reset:** Camera, Options
	**View:** Left, Right, Up, Down, Zoom In, Zoom Out, Reset Camera **Options:**Wireframe,Texture,Specular Light, Reset Options

Na configuração dos eventos, os rótulos das opções descritas anteriormente são associados aos eventos pré-definidos disponíveis no *plugin* (todos os eventos pré-definidos são mostrados na Tabela 4.2). Os eventos podem ser acionados através do *mouse* (para clicar nas opções do menu), teclas de atalho ou outros dispositivos de entrada. A escolha dos dispositivos e a independência de idioma na escrita dos textos dos menus permitem maior flexibilidade à interface, se adaptando às necessidades da aplicação ou do usuário.

Tabela 4.2: Eventos disponíveis no *plugin* IMAGO.

Nome do Evento	**Descrição**
left_event	Evento para mover o modelo 3D para esquerda
right_event	Evento para mover o modelo 3D para direita
up_event	Evento para mover o modelo 3D para cima
down_event	Evento para mover o modelo 3D para baixo
resetcamera_event	Evento para deixar a câmera na posição inicial
zoomin_event	Evento para aproximar o modelo 3D da tela
zoomout_event	Evento para afastar o modelo 3D da tela
wireframe_event	Evento para exibir os triângulos do modelo 3D
texture_event	Evento para exibir o modelo com/sem textura
specular_event	Evento para alterar a intensidade da luz incidente no modelo 3D
resetoptions_event	Evento para zerar todas as opções, com exceção das opções de câmera

Os eventos são definidos após a tag "#Events"e cada linha contém um rótulo de uma opção definida na seção Menus, seguido de um evento pré-definido e seus dispositivos correspondentes. Por exemplo, a opção de rótulo "Wireframe"pode ser associada com o evento pré-definido "wireframe_event", habilitando ou desabilitando a exibição da malha de triângulos do modelo 3D.

O *plugin* IMAGO pode obter o modelo 3D pelo arquivo zip de entrada ou através do envio por um servidor de modelos 3D pela Internet. Para receber o modelo através do arquivo zip, inclui-se a linha "model = 1"após a tag "#Model". Para obter o modelo 3D através de um servidor, um exemplo de um servidor é disponibilizado no *website* do *plugin* IMAGO, contendo os comandos necessários para realizar a comunicação com o visualizador. Assim os programadores podem criar seu próprio servidor de modelos 3D utilizando outras linguagens e outros recursos.

A configuração que indica que o modelo 3D será obtido através de um servidor (Código 4.1.3) inclui, após a tag "#Model", a linha "model = 2"e especifica-se o endereço IP do servidor na Internet e o número da porta a ser utilizada para comunicação. Independente da configuração escolhida, as três tags das seções devem estar presentes no arquivo de configuração.

```
16 #Model
17 model=2
18 ip=200.1.1.1
19 port=9000
```

Código 4.1.3: *Exemplo da seção Model no arquivo de configuração que indica que o modelo 3D será obtido por um servidor no endereço IP e porta especificados.*

4.1.3 Arquivos de Modelo 3D e Textura

O arquivo do modelo 3D para visualização é representado no formato texto e pode ser gerado de forma simples. Neste arquivo são especificadas as informações mínimas sobre o objeto: as coordenadas 3D (X,Y,Z) de cada vértice do modelo, os vetores normais de cada vértice, o conjunto de faces, representadas por 3 vértices cada uma, e as propriedades de cor do objeto digitalizado. O arquivo de um modelo 3D que não utiliza textura tem a extensão ".m2", definida neste projeto. A informação de cor para cada vértice do modelo é incluída no conjunto das informações mínimas, como pode ser visto na Figura 4.1.

```
; number of wedges
; wedges (x, y, z, nx, ny, nz, r, g, b)
4
0.37 0.00 0.00 0.370 -0.929 -0.021 93 65 48
-3.00 0.00 30.00 0.432 -0.900 0.060 93 66 51
30.00 0.00 30.00 0.383 -0.922 0.060 81 60 47
30.00 0.00 0.00 0.383 -0.922 0.060 81 60 47
; number of faces
; faces
8
0 1 2
1 2 0
2 0 1
2 3 0
3 0 2
0 2 3
2 1 0
1 2 3
; material information
; diffuse color (r, g, b)
; specular color (r, g, b), specular exponent
255 255 255
32 32 32 8.000000
```

Figura 4.1: Exemplo de arquivo no formato m2.

Para modelos 3D com textura deve-se especificar as coordenadas uv (*i.e.* mapeamento das coordenadas 2D de textura para cada vértice do modelo 3D) de textura do modelo 3D e o nome do arquivo de imagem que contém a textura a ser utilizada, além das informações mínimas conforme descritas anteriormente. O arquivo de um modelo 3D com textura também é no formato texto e tem a extensão ".m", também definida neste projeto, como mostrado na Figura 4.2. Ele pode ser anexado no arquivo zip de entrada, juntamente com o arquivo da imagem de textura com extensão ".tga"(*i.e.* TARGA File Format).

```
; number of wedges
; wedges (x, y, z, nx, ny, nz, u, v)
5
-1.3335 -13.8592  8.6368  0.2914 -0.5051  0.8123  0.4250  0.5194
-4.3016  0.0062  9.1149  0.4135 -0.0000  0.9104  0.5006  0.5119
2.5789  5.4550  5.7859  0.3344  0.2856  0.8980  0.5335  0.5284
1.4717  8.1923  8.2289  0.3339  0.3439  0.8776  0.5479  0.5259
-0.1581  3.5878  3.6884 -0.5076  0.7589 -0.4077  0.7336  0.5665
; number of faces
; faces (w1, w2, w3)
7
0 1 2
3 4 1
4 3 0
1 0 3
3 4 6
4 3 1
1 2 3
; material information
; diffuse color (r, g, b)
; specular color (r, g, b), specular exponent
; texture file name
93 115 68
95 129 93 39.999996
texture.tga
```

Figura 4.2: Exemplo de arquivo no formato m.

4.1.4 Interação com os modelos 3D

A interface intuitiva e simples do *plugin* IMAGO torna a visualização mais agradável para o usuário, pois não exige esforço para interagir com os modelos 3D, sendo um atrativo para atividades educacionais. Operações básicas como rotacionar, movimentar, aproximar e afastar o modelo 3D são realizadas de forma prática, utilizando apenas o *mouse* com a manipulação direta (Tabela 4.3) ou com auxílio do menu e do teclado. Outras operações importantes como mostrar o modelo 3D em *wireframe*, com ou sem textura, aumentar ou diminuir o efeito especular na iluminação do modelo 3D e movimentar a origem da fonte de iluminação (Figura 4.3) também estão disponíveis neste visualizador, podendo ser acionadas através dos menus ou por teclas de atalho (Tabela 4.4) pelo teclado.

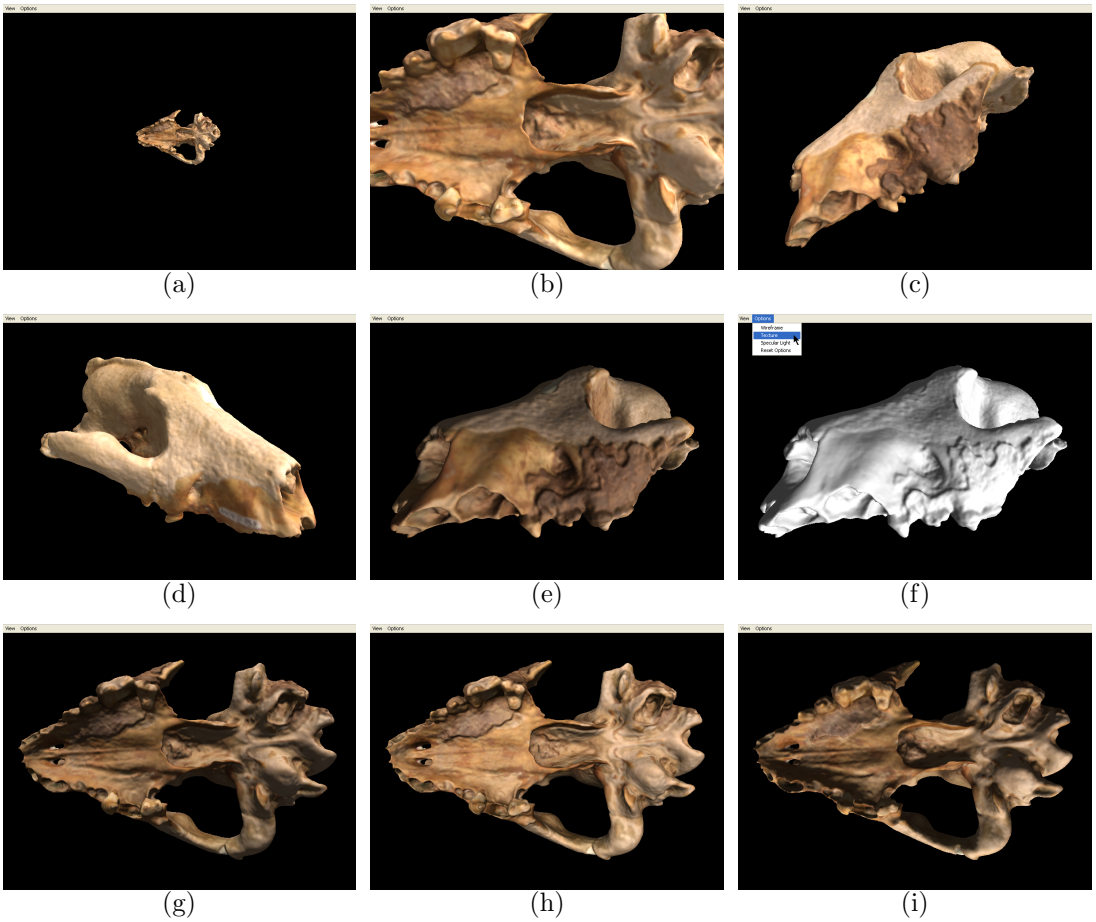


Figura 4.3: Exemplos de visualização do modelo 3D de um fóssil de um Protocyon, animal que habitou cavernas no Brasil durante o Pleistoceno (período compreendido entre 1.816.000 e 11.500 anos atrás). Fóssil pertencente ao Museu de Ciências Naturais da UFPR. Ações: (a) afastar, (b) aproximar, (c) e (d) mover o modelo 3D, (e) modelo 3D com textura, (f) modelo 3D sem textura, (g) - (i) movimentar a origem da fonte de iluminação.

Tabela 4.3: Maneiras para realizar a manipulação direta com o *mouse*.

Operação	**Utilização do *mouse***
Aproximar/Afastar (Zoom)	Scroll
Rotacionar	Botão esquerdo pressionado + arrastar do *mouse*
Deslocar	Botão direito pressionado + arrastar do *mouse*

Tabela 4.4: Teclas de atalho disponíveis para movimentar os modelos 3D.

Tecla de Atalho	**Operação**
+	Aproximar
–	Afastar
←	Rotacionar para esquerda
→	Rotacionar para direita
↑	Rotacionar para cima
↓	Rotacionar para baixo
C	Posicionar o modelo 3D na posição inicial
T	Remover/Colocar textura
W	Ativar/Desativar wireframe
S	Alterar a iluminação incidente no modelo 3D
R	Desfazer as opções escolhidas
Ctrl + Botão esquerdo do *mouse*	Movimentar a origem da fonte de iluminação

4.1.5 Funcionamento

O *plugin* IMAGO verifica se a entrada possui os arquivos obrigatórios e os necessários de acordo com a configuração escolhida. O arquivo de configuração é obrigatório e caso a obtenção do modelo 3D escolhido seja através do arquivo zip, o arquivo de modelo 3D deve estar incluso no arquivo zip. Se o modelo 3D usar textura, independente da forma de obtenção do mesmo, a textura deve estar dentro do arquivo zip.

De acordo com a configuração escolhida, os menus são montados para serem exibidos na tela e os eventos são associados às opções dos menus e aos dispositivos definidos. A configuração permite que sejam escolhidos:

- menus, eventos com *mouse* para utilização dos menus e movimentação do modelo 3D e uso de teclas de atalho com o teclado;
- menus e eventos apenas com *mouse* para utilização dos menus e movimentação do modelo 3D;
- sem menus, eventos com *mouse* para movimentação do modelo 3D e uso de teclas de atalho com o teclado;
- sem menu e eventos com *mouse* apenas para movimentação do modelo 3D.

O *plugin* utiliza a biblioteca gráfica OpenGL para renderização dos modelos 3D e dos menus, sendo que a versão Windows utiliza uma API do próprio sistema operacional para renderizar os menus, mantendo o mesmo padrão visual nas versões Linux e Windows. Para otimizar a renderização dos modelos 3D é verificado em tempo real se o computador do usuário possui placa de vídeo com aceleração gráfica 3D e suporte à extensão Vertex Buffer Array[2] (VBO) do OpenGL, e caso não tenha, utiliza-se a técnica de Triangle Strips [7].

A utilização do VBO para geometrias estáticas (sem animação) tem um ganho considerável no desempenho, já que com ele é possível enviar os dados uma única vez para a memória da placa de vídeo, evitando o excesso de transferência no barramento. Assim, os dados são gravados na placa de vídeo e altera-se apenas a posição da câmera quando ocorre a mudança na visualização do objeto.

Com Triangle Strips cada novo triângulo renderizado pode ser definido com adição de apenas um vértice, compartilhando dois vértices já definidos no triângulo anterior. Desta forma, pode-se reduzir a praticamente 1/3 a quantidade de vértices enviados e armazenados na placa de vídeo.

4.1.6 Desempenho

Para a realização dos testes nessa seção, foi utilizado um computador com a seguinte especificação: sistema operacional Windows XP Service Pack 2, processador Intel Core 2 Duo 6300 1.86Ghz, memória de 1 GB e placa de vídeo NVIDIA GeForce 7300SE. Os modelos 3D utilizados para estes experimentos estão disponíveis no *website* do IMAGO[3].

Arquivos de modelos 3D são bem compactados com o formato zip, reduzindo consideravelmente o tamanho do arquivo e consequentemente, diminuindo o tempo de transmissão para o computador do usuário. A Tabela 4.5 mostra a porcentagem de redução do tamanho dos arquivos zip com os modelos 3D no formato m2. Em geral, uma taxa de compressão em torno de 70% pode ser obtida para diferentes modelos 3D. Os arquivos

[2]http://www.opengl.org/registry/specs/ARB/vertex_buffer_object.txt
[3]www.imago.ufpr.br

zip foram gerados no Windows, utilizando o software Winrar com método de compressão normal. De forma equivalente, no Linux o comando zip pode ser utilizado. Os arquivos zip utilizados em todos os testes contêm um arquivo de configuração com tamanho de 802 *bytes* o qual obteve 64% de compressão. Na Tabela 4.6 podemos ver que arquivos zip com modelo 3D e textura também apresentam taxa de compressão similar às apresentadas na Tabela 4.5.

Tabela 4.5: Dados sobre a compressão dos modelos 3D utilizando zip.

Modelo 3D	**Número de Faces**	**Tamanho do arquivo m2 sem compressão**	**Tamanho do arquivo zip**	**Taxa de compressão do arquivo m2**
Alamito	20.892	820 Kb	246 Kb	70%
Besouro 1	263.010	11.070 Kb	2.924 Kb	74%
Besouro 2	162.201	6.550 Kb	1.778 Kb	73%
Cabeça	18.052	712 Kb	197 Kb	73%
Carybé	43.812	1.787 Kb	511 Kb	71%
Coruja	40.408	1.637 Kb	462 Kb	72%
Galo	142.774	6.056 Kb	1.741 Kb	72%
Protocyon	49.124	2.059 Kb	596 Kb	71%
São Francisco	21.336	830 Kb	247 Kb	70%
Stenzel	46.660	1.979 Kb	555 Kb	72%
Vaso	47.220	1.963 Kb	570 Kb	71%
Vitor	71.554	3.050 Kb	863 Kb	72%

Tabela 4.6: Dados sobre a compressão de modelos 3D com textura utilizando zip.

Modelo 3D	**Tamanho do arquivo m**	**Tamanho do arquivo de textura**	**Tamanho do arquivo zip**	**Taxa de compressão do arquivo m**	**Taxa de compressão do arquivo de textura**
Anta	6.9 Mb	3.0 Mb	3.1 Mb	68%	70%
Bote	3.4 Mb	3.0 Mb	1.9 Mb	68%	74%
Indias	4.9 Mb	3.0 Mb	2.7 Mb	68%	63%
Onça	4.4 Mb	3.0 Mb	2.2 Mb	68%	75%
Pato	2.5 Mb	3.0 Mb	1.6 Mb	68%	75%
Zoolito	6.1 Mb	3.0 Mb	2.6 Mb	69%	75%

A Tabela 4.7 mostra a taxa de quadros por segundo[4] do *plugin* IMAGO utilizando VBO e Triangle Strip. A Tabela 4.8 mostra os valores para três visualizadores VRML bastante utilizados: Cortona3D, Octaga Player e Cosmo Player. Comparando os resultados é possível observar a superioridade do *plugin* IMAGO em relação aos visualizadores VRML.

Além disso, pode-se verificar que a taxa de quadros por segundo se mantém aceitável com a variação na resolução dos modelos 3D. Modelos 3D de baixa resolução possuem alta taxa de quadros por segundo enquanto modelos 3D de alta resolução possuem uma taxa suficiente para proporcionar boa interatividade. A Tabela 4.9 mostra as comparações entre os modelos 3D, e os mesmos podem ser vistos na Figura 4.4.

[4]Testes realizados com o medidor de quadros por segundo do FRAPS (http://www.fraps.com)

Tabela 4.7: Taxa de quadros por segundo do *plugin* IMAGO.

Modelo 3D (Formato m2)	**Tamanho do Arquivo**	**Número de Faces (VBO)**	**FPS (VBO)**	**Número de Faces (Triangle Strip)**	**FPS (Triangle Strip)**
Alamito	15 Mb	342.520	75.0 fps	462.773	43.0 fps
Carybé	31 Mb	714.696	51.0 fps	969.356	22.6 fps
Galo	26 Mb	593.584	60.0 fps	818.192	26.0 fps
Profeta	64 Mb	1.409.939	28.5 fps	1.922.959	11.8 fps
Protocyon	39 Mb	873.746	43.0 fps	1.187.139	18.7 fps
Stenzel	30 Mb	666.900	54.0 fps	875.361	25.0 fps
Vaso	35 Mb	792.809	47.0 fps	1.074.908	20.3 fps
Vitor	54 Mb	1.210.630	32.5 fps	1.634.061	13.7 fps

Tabela 4.8: Taxa de quadros utilizando visualizadores VRML.

Modelo 3D (extensão wrl)	**Tamanho do Arquivo**	**Número de Faces**	**Cortona3D 6.0**	**Octaga Player 2.3.0.7**	**Cosmo Player 2.1.1**
Alamito	18 Mb	342.520	9.3 fps	12.06 fps	16.0 fps
Carybé	38 Mb	714.696	5.0 fps	5.97 fps	9.0 fps
Galo	31 Mb	593.584	5.87 fps	7.31 fps	10.7 fps
Profeta	76 Mb	1.409.939	2.69 fps	3.12 fps	4.6 fps
Protocyon	46 Mb	873.746	4.07 fps	5.0 fps	7.4 fps
Stenzel	35 Mb	666.900	5.27 fps	6.49 fps	9.6 fps
Vaso	42 Mb	792.809	4.53 fps	5.48 fps	8.1 fps
Vitor	65 Mb	1.210.630	3.25 fps	3.66 fps	5.3 fps

Tabela 4.9: Renderização de um modelo 3D com diferentes níveis de detalhes.

Modelo 3D	**Tamanho do arquivo**	**Número de Faces**	**FPS (VBO)**
São Francisco - Alta Resolução	13.113 Kb	306.716	86.0 fps
São Francisco - Média Resolução	3.102 Kb	75.540	175.0 fps
São Francisco - Baixa Resolução	830 Kb	21.336	220.0 fps

(a) (b) (c)

Figura 4.4: Exemplos de um modelo 3D com diferentes níveis de detalhes: (a) com 21.336 faces, (b) com 75.540 faces e (c) com 306.716 faces.

CAPÍTULO 5

IMAGO 3D VIEWER

Conforme apresentado no Capítulo 4, o *plugin* IMAGO foi desenvolvido para solucionar o problema do baixo nível de interatividade na visualização 3D com as tecnologias existentes. Primeiramente esse visualizador 3D estava incorporado no sistema de Museu Virtual 3D do IMAGO, não sendo possível ser utilizado por outras aplicações. Para solucionar essa limitação foi gerada uma versão que permitiu que outros projetos pudessem utilizar o visualizador 3D em suas aplicações.

Durante a realização das modificações para criar a nova versão, foi possível observar que determinadas soluções precisavam diferir de uma versão para outra (Linux e Windows). Alguns recursos eram disponíveis para a versão Windows e não para a versão Linux, por exemplo o tratamento dos eventos. Consequentemente, para manter as versões aparentemente iguais foi necessário utilizar tempo e esforço adicional na codificação.

Um problema existente no *plugin* IMAGO é a incompatibilidade com outros navegadores *web* mais utilizados, como o Internet Explorer, permitindo o funcionamento apenas para navegadores *web* da família Firefox. O fato de possuir duas versões, uma para Windows e outra para Linux, tornou complicada a atualização das versões. Além disso, para conseguir ampliar o acesso aos modelos 3D foi observado a necessidade de criar um visualizador que pudesse ser facilmente modificado para funcionar em dispositivos móveis, como celulares. Para solucionar esses problemas foi desenvolvido uma novo visualizador 3D portável com utilização da linguagem Java e a biblioteca OpenGL: o IMAGO 3D Viewer.

5.1 Java e JOGL

A linguagem de programação Java é geralmente escolhida para desenvolver aplicações na Internet por ser independente de *hardware* e por sua ampla disponibilidade em várias plataformas [31]. Há duas tecnologias Java que permitem a execução de aplicativos na

Internet: Java Applets e Java Web Start. As Applets são exibidas dentro do navegador *web* e geralmente são usadas para adicionar interatividade ou para proporcionar segurança às aplicações *web*. O Java Web Start permite fazer *download* e executar aplicativos a partir do navegador *web*, possui fácil instalação e garante que sempre seja executada a versão mais recente do aplicativo[1].

A linguagem Java originalmente não foi desenvolvida para fins gráficos e existem poucas bibliotecas para gráficos 3D na Internet. Existem algumas APIs (Java 3D, GL4JAVA, JOGL, LWGL, JGL) que auxiliam essa questão, porém nenhuma delas faz parte do pacote oficial do Java. Consequentemente, ainda é difícil construir programas 3D voltados para *web* que forneçam o mesmo realismo e interatividade encontrados em aplicações *stand-alone* [24]. O Java 3D também permite a criação de ambientes virtuais porém, apesar da Sun Microsystems Inc. tê-lo desenvolvido, ele não faz parte do pacote Java e deve ser instalado separadamente no computador do usuário.

Para o desenvolvimento desse visualizador optamos pela API JOGL[2], que é um "Java-OpenGL binding"e permite utilizar os comandos OpenGL. O OpenGL é acessado através do Java Native Interface (JNI), utilizando um mapeamento direto entre os métodos JOGL e as funções implementadas na linguagem C, o que permite bom desempenho na renderização de modelos 3D. Utilizando Applets Java é possível exibir os modelos 3D dentro do navegador *web*. O uso de Applets evitam a necessidade de instalação de *plugins*, sendo necessário apenas que o computador do usuário possua o Java Runtime Environment (JRE) instalado.

5.2 Arquitetura

O IMAGO 3D Viewer é uma ferramenta de visualização 3D que permite interagir com modelos 3D realistas de objetos naturais e culturais através de um navegador *web*. Com o uso de um arquivo de configuração é possível escolher opções sobre a interface e sobre os eventos disponíveis para a interação do usuário. Devido a sua característica de por-

[1] http://www.java.com/pt_BR/download/faq/java_webstart.xml
[2] https://jogl.dev.java.net

tabilidade, sua utilização não está restrita a sistemas operacionais ou navegadores *web* específicos, sendo necessário apenas que o computador do usuário permita o uso de Java e Applets.

Além das funcionalidades presentes no *plugin* IMAGO, o novo visualizador 3D oferece mais opções de interface, como por exemplo a utilização de uma barra de ferramentas contendo ícones para realizar operações sobre o modelo 3D e também uma barra de *status*, que indica quais operações o usuário está realizando. Assim como o *plugin* IMAGO, o visualizador utiliza um arquivo de configuração, um arquivo de modelo 3D e um arquivo de textura como apresentado no Capítulo 4.1.3.

5.2.1 Modo de Uso

Igual ao *plugin* IMAGO, o visualizador recebe como entrada um arquivo zip que contém o arquivo de modelo 3D, o arquivo de textura e um arquivo de configuração. Para exibir o visualizador dentro do navegador *web* é necessário incluir alguns comandos no arquivo HTML da página *web*. A tag APPLET (Código 5.2.1) permite especificar os arquivos JAR (arquivos da aplicação Java) e outros parâmetros necessários para permitir o correto funcionamento. O IMAGO3DViewer.jar é arquivo JAR principal e o jogl.jar e gluegen-rt.jar permitem a utilização do JOGL.

O recurso JNLPAppletLauncher permite que as Applets possam usar bibliotecas contendo código nativo, como extensões para Java 3D e JOGL. Para isso é necessário incluir o parâmetro "code"como mostra a linha 1 do Código 5.2.1, e especificar o arquivo applet-launcher.jar. O JNLPAppletLauncher usa um arquivo de extensão .jnlp para localizar os arquivos para uma determinada extensão, por esse motivo os parâmetros "jnlpNumExtensions"(linha 14) e "jnlpExtension1"(linha 15) estão presentes.

O parâmetro "codebase_lookup"(linha 8) é recomendado para melhorar a eficiência do carregamento da applet, para que os arquivos não possam ser buscados fora do servidor *web* de origem. Os parâmetros "subapplet.classname"(linha 9) e "subapplet.displayname"(linha 10) especificam o nome da classe principal do visualizador e o

nome de exibição da aplicação.

Applets que utilizam o OpenGL através da API JOGL podem apresentar problemas de robustez quando utilizadas no sistema operacional Windows. Isso pode ocorrer porque alguns *drivers* podem não renderizar na mesma janela a API Opengl e a API DirectDraw da Microsoft, utilizada pelo Java 2D nessa plataforma. Por esse motivo recomendamos a utilização dos parâmetros "noddraw.check"(linha 11) e "noddraw.check.silent"(linha 12) para desativar o DirectDraw.

Para evitar problemas de "piscar"de tela que podem ocorrer em algumas versões das atualizações do Java, utilizamos no parâmetro "java_arguments"(linha 16) com a opção para que a tela não seja apagada toda vez que é feito um desenho na tela. O parâmetro "model"(linha 18) especifica o nome do modelo 3D que será visualizado e parâmetro "src"(linha 19) indica o a localização do arquivo zip no servidor *web*.

```
1  <applet code="org.jdesktop.applet.util.JNLPAppletLauncher"
2      width=800
3      height=700
4      archive="http://www.imago.ufpr.br/Museu/java/IMAGO3DViewer.jar,
5               http://www.imago.ufpr.br/Museu/java/jars/applet-launcher.jar,
6               http://www.imago.ufpr.br/Museu/java/jars/jogl.jar,
7               http://www.imago.ufpr.br/Museu/java/jars/gluegen-rt.jar">
8      <param name="codebase_lookup" value="false">
9      <param name="subapplet.classname" value="viewer.Viewer">
10     <param name="subapplet.displayname" value="Museu Virtual 3D">
11     <param name="noddraw.check" value="true">
12     <param name="noddraw.check.silent" value="true">
13     <param name="progressbar" value="true">
14     <param name="jnlpNumExtensions" value="1">
15     <param name="jnlpExtension1" value="http://www.imago.ufpr.br/Museu/java/jogl.jnlp">
16     <param name="java_arguments" value="-Dsun.awt.noerasebackground=true">
17
18     <param name="model" value="alamito.m2">
19     <param name="src" value="http://www.imago.ufpr.br/Museu/java/novos/modelos/alamito.zip">
20 </applet>
```

Código 5.2.1: *Chamada do visualizador no arquivo HTML.*

5.2.2 Configuração do Visualizador 3D

Através de um arquivo de configuração é possível escolher algumas opções sobre a interface do visualizador. Assim como no *plugin* IMAGO, é permitida a utilização de menus,

teclado e *mouse* para realizar as operações sobre o modelo 3D. Para tornar a interface mais acessível, o visualizador também permite a utilização de uma barra de ferramentas que inclui ícones (acionados com o clicar do *mouse*) para cada operação sobre o modelo 3D e também uma barra de *status*.

Igualmente ao arquivo de configuração utilizado pelo *plugin* IMAGO, os menus são definidos após a tag "#Menus"(Código 5.2.2) e cada linha abaixo dela contém o rótulo do menu principal seguido pelos rótulos das opções.

Para a inclusão da barra de ferramentas e da barra de *status* foi necessário mudar a abordagem de configuração dos eventos. O rótulo para uma determinada opção significa mais do que apenas um texto que será exibido nos menus. Esse rótulo será utilizado também para descrever um determinado evento, ou seja, será o significado do evento associado. Dessa maneira o rótulo escolhido será o texto que será exibido nos menus, o texto que aparecerá na barra de *status* quando o usuário clicar sobre a opção do menu ou quando clicar sobre um ícone na barra de ferramentas, definidos para o evento.

Os eventos continuam sendo definidos após a tag "#Events"e cada linha contém um rótulo, seguido de um evento pré-definido e suas formas de acionamento, como por exemplo através de um ícone na barra de ferramentas ou os dispositivos *mouse* e teclado. No Código 5.2.2 podemos perceber as mudanças no arquivo de configuração, com a inclusão do texto "toolbar"após a especificação do evento.

O visualizador obtém o arquivo de modelo 3D apenas através do arquivo zip, diferentemente do *plugin* IMAGO que também permite conectar em um servidor para obter o modelo 3D, quando especificados o endereço IP e porta desse servidor. Testes realizados neste trabalho mostraram que podem ocorrer problemas nesta forma de obtenção, como por exemplo em ambientes de rede que utilizam *proxy* ou *firewall* bloqueando a conexão para determinadas portas. Como medida de segurança, as Applets não permitem a conexão em servidores diferentes dos quais foram geradas ou hospedadas. Por esses motivos, o visualizador por enquanto não permite a obtenção de modelos 3D através de servidores.

```
#Menus
View:Left,Right,Up,Down,Zoom In,Zoom Out,Reset Camera
Options:Wireframe,Texture,Specular Light,Reset Options
#Events
Left:left_event:toolbar,mouse_device,keyboard_device
Right:right_event:toolbar,mouse_device,keyboard_device
Up:up_event:toolbar,mouse_device,keyboard_device
Down:down_event:toolbar,mouse_device,keyboard_device
Reset Camera:resetcamera_event:toolbar,mouse_device,keyboard_device
Zoom In:zoomin_event:toolbar,mouse_device,keyboard_device
Zoom Out:zoomout_event:toolbar,mouse_device,keyboard_device
Wireframe:wireframe_event:toolbar,mouse_device,keyboard_device
Texture:texture_event:toolbar,mouse_device,keyboard_device
Specular Light:specular_event:toolbar,mouse_device,keyboard_device
Reset Options:resetoptions_event:toolbar,mouse_device,keyboard_device
#Model
model=1
```

Código 5.2.2: *Exemplo de arquivo de configuração para escolher a utilização da barra de ferramentas.*

5.2.3 Interação com os Modelos 3D

O visualizador permite realizar as mesmas operações de manipulação dos modelos 3D disponíveis no *plugin* IMAGO (Capítulo 4.1.4). A manipulação direta com o *mouse* e a utilização do teclado proporciona uma interação simplificada, não exigindo do usuário conhecimentos sobre navegação em ambientes de realidade virtual. A utilização de menus também pode facilitar a interação, desde que os textos dos rótulos escolhidos para as opções dos menus sejam auto explicativos.

Além da barra de menus, o visualizador permite a utilização de uma barra de ferramentas para realizar as operações sobre o modelo 3D. Os ícones da barra de ferramentas, os quais podem ser vistos na Figura 5.1, possuem desenhos intuitivos para cada operação, facilitando ainda mais o processo de visualização para usuários iniciantes. O visualizador também possibilita substituir os ícones disponíveis. Assim, os programadores podem criar seus próprios ícones como considerarem mais adequados para a sua aplicação.

Abaixo da tela de exibição do modelo 3D temos uma barra de *status* a qual imprime as ações do usuário. Quando uma determinação opção do menu ou da barra de ferramentas

é escolhida, um texto significando tal operação é exibido. Dessa maneira os usuários sabem as operações que estão realizando, e quanto mais claro for o texto escolhido pelo programador para determinada operação, mais fácil será a sua compreensão.

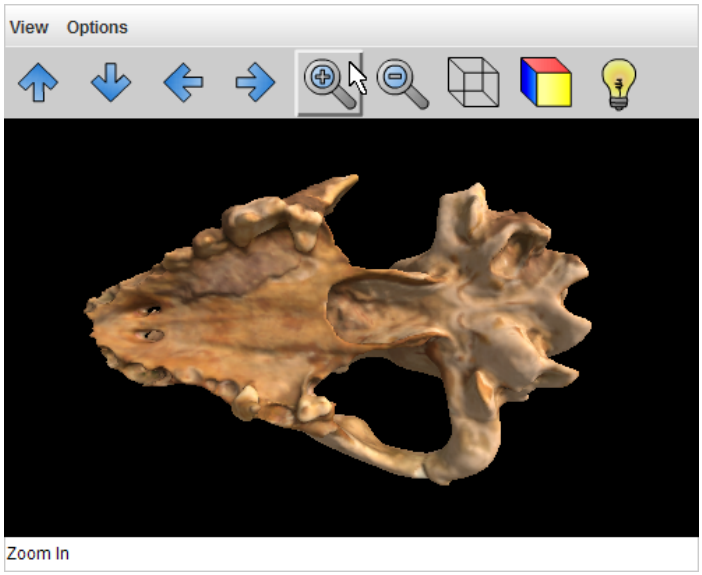


Figura 5.1: Interface do IMAGO 3D Viewer.

5.2.4 Funcionamento

O IMAGO 3D Viewer primeiramente verifica se o arquivo zip de entrada possui o arquivo de configuração e o arquivo de modelo 3D (obrigatórios) e no caso do modelo utilizar textura, também verifica se há um arquivo de textura. De acordo com a configuração escolhida, os menus são montados para serem exibidos na barra de menus e os ícones escolhidos são inseridos na barra de ferramentas. A configuração permite escolher os componentes de interação (teclado, menus e ícones) para cada operação (evento pré-definido) sobre o modelo 3D. A Tabela 5.1 mostra as maneiras possíveis de escolher os componentes de interação, podendo ser utilizado apenas um componente ou a combinação entre eles. Para quaisquer erros relacionados ao arquivo de entrada ou ao arquivo de configuração, mensagens são exibidas na tela para alertar os programadores.

A linguagem Java permite a criação de interfaces através de um conjunto de ferramentas para criação de ambientes gráficos chamado GUI (Grafical User Interface). As APIs mais conhecidas são a AWT (Abstract Window ToolKit) e SWING, esta que foi

Tabela 5.1: Possíveis maneiras de escolher os componentes de interação com o modelo 3D.

Combinações	Teclado	Menus	Ícones
1	✓		
2		✓	
3			✓
4	✓	✓	
5	✓		✓
6		✓	✓
7	✓	✓	✓

desenvolvida com o objetivo de resolver a maioria das deficiências da AWT.

A AWT utiliza a ambientação gráfica do sistema operacional, isso quer dizer que a aparência das interfaces é modificada dependendo da plataforma em que estão sendo executadas. Já a SWING utiliza uma ambientação gráfica do próprio Java, mantendo a consistência das interfaces independente do sistema operacional. Além disso, as interfaces desenvolvidas com SWING proporcionam um ambiente de desenvolvimento mais produtivo, por isso esta API foi utilizada para a criação da barra de menus, barra de ferramentas e barra de *status* do visualizador.

A tela de exibição do modelo é criada com a largura e altura desejados a partir dos valores fornecidos na tag APPLET na chamada do visualizador (Seção 5.2.1). Independente da dimensão escolhida, um cálculo é feito com as informações da geometria do modelo 3D para que o mesmo seja posicionado no centro da tela. Utilizamos o componente JOGL GLCanvas (compatível com a AWT) que nos permite definir a área de desenho do modelo 3D utilizando comandos OpenGL. O GLCanvas suporta aceleração de hardware e apresenta melhor desempenho em comparação ao componente SWING GLPanel.

A renderização do modelo 3D é feita utilizando comandos OpenGL através da API JOGL. Assim como o *plugin* IMAGO, o visualizador verifica em tempo real se o computador do usuário possui suporte à extensão VBO do OpenGL para fornecer um melhor desempenho. Quando não possível utilizar VBOs, faixas de triângulos (strips) do modelo 3D são geradas para utilizar a técnica de Triangle Strips.

5.2.5 Desempenho

Para tornar possível a comparação entre o IMAGO 3D Viewer e o *plugin* IMAGO, utilizamos o mesmo computador para a realização dos testes, com a seguinte especificação: sistema operacional Windows XP Service Pack 2, processador Intel Core 2 Duo 6300 1.86Ghz, memória de 1 GB e placa de vídeo NVIDIA GeForce 7300SE.

Um dos objetivos na criação do IMAGO 3D Viewer, além de manter as funcionalidades presentes no *plugin* IMAGO, era manter o bom desempenho na renderização dos modelos 3D realistas. Apesar do Java possuir a fama de ser mais lento em comparação ao C++, nossos resultados demonstraram não haver diferenças significativas no tempo de processamento das principais funções usadas nos dois visualizadores. Na Tabela 5.2 podemos observar que a taxa de quadros por segundo do IMAGO 3D Viewer apresenta bons valores, para os mesmos modelos 3D utilizados nos testes do *plugin* IMAGO (Capítulo 4.1.6).

Tabela 5.2: Taxa de quadros por segundo do IMAGO 3D Viewer.

Modelo 3D (Formato m2)	**Tamanho do Arquivo**	**Número de Faces (VBO)**	**FPS (VBO)**	**Número de Faces (Triangle Strip)**	**FPS (Triangle Strip)**
Alamito	15 Mb	342.520	75.0 fps	462.773	43.0 fps
Carybé	31 Mb	714.696	53.0 fps	969.356	22.0 fps
Galo	26 Mb	593.584	62.0 fps	818.192	25.6 fps
Profeta	64 Mb	1.409.939	29.0 fps	1.922.959	11.4 fps
Protocyon	39 Mb	873.746	44.0 fps	1.187.139	18.0 fps
Stenzel	30 Mb	666.900	56.0 fps	875.361	24.2 fps
Vaso	35 Mb	792.809	48.0 fps	1.074.908	20.0 fps
Vitor	54 Mb	1.210.630	33.0 fps	1.634.061	13.3 fps

Utilizando os mesmos modelos 3D e mantendo iguais o tamanho dos arquivos e quantidade de faces, reunimos na Tabela 5.3 os valores das taxas de quadro por segundo do IMAGO 3D Viewer apresentados na Tabela 5.2 com os valores do *plugin* IMAGO utilizando VBO e três visualizadores VRML (Cortona3D, Octaga Player e Cosmo Player). Nessa Tabela e na Figura 5.2 podemos ver que o IMAGO 3D Viewer possui valores muito próximos ao do *plugin* IMAGO, além da superioridade em relação aos visualizadores VRML nesse quesito.

Tabela 5.3: Comparação do IMAGO 3D Viewer com o *plugin* IMAGO e visualizadores VRML.

Modelo 3D	**IMAGO 3D Viewer**	***Plugin* IMAGO**	**Cortona3D 6.0**	**Octaga Player 2.3.0.7**	**Cosmo Player 2.1.1**
Alamito	75.0 fps	75.0 fps	9.3 fps	12.06 fps	16.0 fps
Carybé	53.0 fps	51.0 fps	5.0 fps	5.97 fps	9.0 fps
Galo	62.0 fps	60.0 fps	5.87 fps	7.31 fps	10.7 fps
Profeta	29.0 fps	28,5 fps	2.69 fps	3.12 fps	4.6 fps
Protocyon	44.0 fps	43.0 fps	4.07 fps	5.0 fps	7.4 fps
Stenzel	56.0 fps	54.0 fps	5.27 fps	6.49 fps	9.6 fps
Vaso	48.0 fps	47.0 fps	4.53 fps	5.48 fps	8.1 fps
Vitor	33.0 fps	32.5 fps	3.25 fps	3.66 fps	5.3 fps

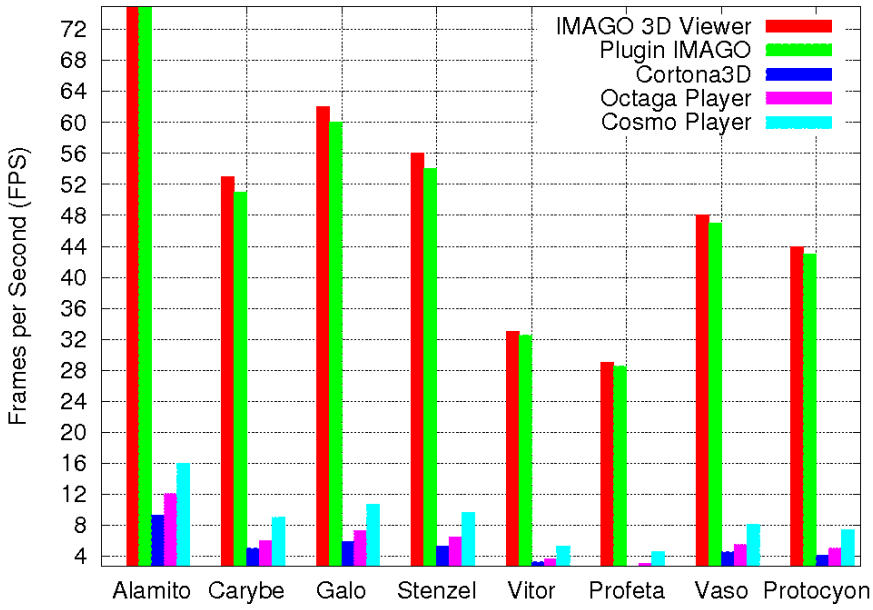


Figura 5.2: Comparação do IMAGO 3D Viewer com o *plugin* IMAGO e visualizadores VRML.

O IMAGO 3D Viewer foi testado nas plataformas Linux, Windows e Mac, nos navegadores *web* Firefox, Google Chrome e Internet Explorer. Foi verificado que determinadas versões de atualização do Java apresentam algumas diferenças em relação ao desenho da tela do visualizador. Os efeitos encontrados foram: piscamento da tela (efeito flicker), menus exibidos "por atrás"da tela de exibição dos modelos 3D, tela totalmente apagada após minimizar a janela do navegador *web*, entre outros.

Alguns desses efeitos são decorrentes da utilização de componentes AWT (*heavyweights*) em conjunto com componentes SWING (*lightweights*) na criação da interface. Isso acontece porque os componentes AWT são processados pelo sistema operacional, enquanto os componentes SWING são processados pelo Java, significando o uso de dois sistemas de pintura diferentes em uma mesma janela. Nesse caso os componentes do sis-

tema operacional sempre são priorizados, e uma das consequências são componentes AWT desenhados "por cima"dos componentes SWING. Algumas adaptações foram realizadas para minimizar esses problemas[3], porém recomenda-se que os usuários utilizem sempre a versão mais recente do Java.

Em relação ao consumo de memória das Applets, os testes mostraram que pode não ser possível abrir modelos 3D ou texturas muito grandes. Isso ocorre devido à pouca memória destinada para as Applets, porém é algo que pode ser corrigido aumentando o limite de memória na configuração do Java no computador do usuário. O passo a passo dessa operação pode ser encontrado na página *web* do visualizador 3D.

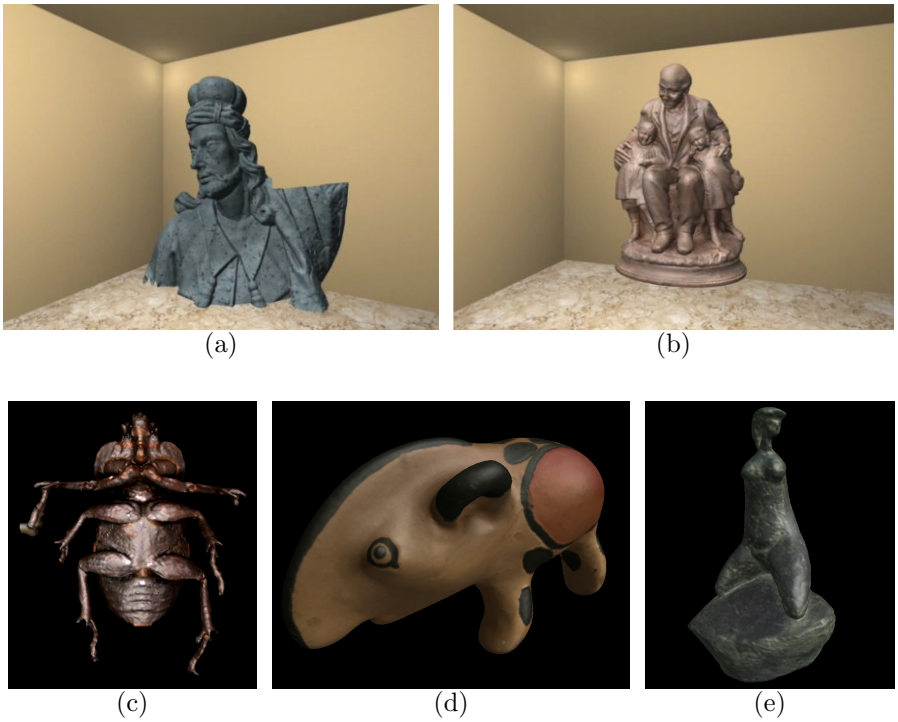
Um recurso bastante utilizado nos navegadores *web* é a utilização de abas, que permite visualizar várias páginas numa mesma sessão do navegador. Para melhor desempenho do visualizador 3D recomendamos evitar manter muitas abas abertas pois elas consomem boa parcela da memória destinada ao navegador, comprometendo a carga das Applets.

[3]http://java.sun.com/products/jfc/tsc/articles/mixing/

CAPÍTULO 6

CASO DE ESTUDO: MUSEU VIRTUAL 3D DO IMAGO

O projeto do Museu Virtual 3D do Grupo IMAGO possui um processo de preservação digital que envolve desde a aquisição de dados sobre os objetos até a disponibilização dos modelos 3D através da Internet. Neste projeto, foram preservados diferentes tipos de patrimônios pertencentes a acervos culturais e naturais, tais como: Museu de Belas Artes da UFMG (Figura 6.3a), Acervo da Reitoria (Figura 6.3b), Coleções Biológicas da UFPR (Figura 6.1c), Museu de Arqueologia e Etnologia da UFPR (Figura 6.1d), Museu Metropolitano de Arte Curitiba (Figura 6.1e) e Museu de Ciências Naturais da UFPR (Figura 6.2).



(a) (b) (c) (d) (e)

Figura 6.1: Exemplos de modelos 3D preservados: (a) Profeta Habacuc, feito pelo artista Aleijadinho, (b) Estátua presente no gabinete do reitor da UFPR, (c) Besouro, (d) Anta de cerâmica manufaturada por índios da tribo Wauja, (e) Obra do artista Carybé.

O objetivo desse Capítulo é apresentar o Museu Virtual 3D do IMAGO, no qual foram incorporados os estudos sobre a visualização interativa de modelos 3D realistas. A

Seção 6.2 descreve o processo de disponibilização dos modelos 3D, bem como a aplicação das ferramentas de visualização 3D desenvolvidas neste trabalho.

6.1 Geração dos Modelos 3D

A geração de modelos 3D é dividida em três etapas principais: aquisição de dados; reconstrução da geometria; e geração de texturas de alta resolução. Na etapa de aquisição, são obtidas informações sobre o objeto que serão utilizadas a longo de sua reconstrução digital. Utilizando um scanner a laser de alta resolução (Minolta Vivid 910) são obtidas imagens coloridas e imagens de profundidade, e através de uma câmera fotográfica profissional (Canon EOS5D) são capturadas imagens de alta resolução para a geração de texturas.

Na segunda etapa, as vistas (obtidas através das imagens de profundidade) são alinhadas e integradas para obter uma malha que corresponde ao modelo 3D do objeto. O processo completo de reconstrução desenvolvido pelo IMAGO é composto por várias etapas específicas, descritas em [32, 33]. Finalmente, na terceira etapa é gerada e aplicada uma textura de alta qualidade ao modelo 3D reconstruído [1, 2]. Na Figura 6.2 podemos ver as etapas do processo de preservação digital.

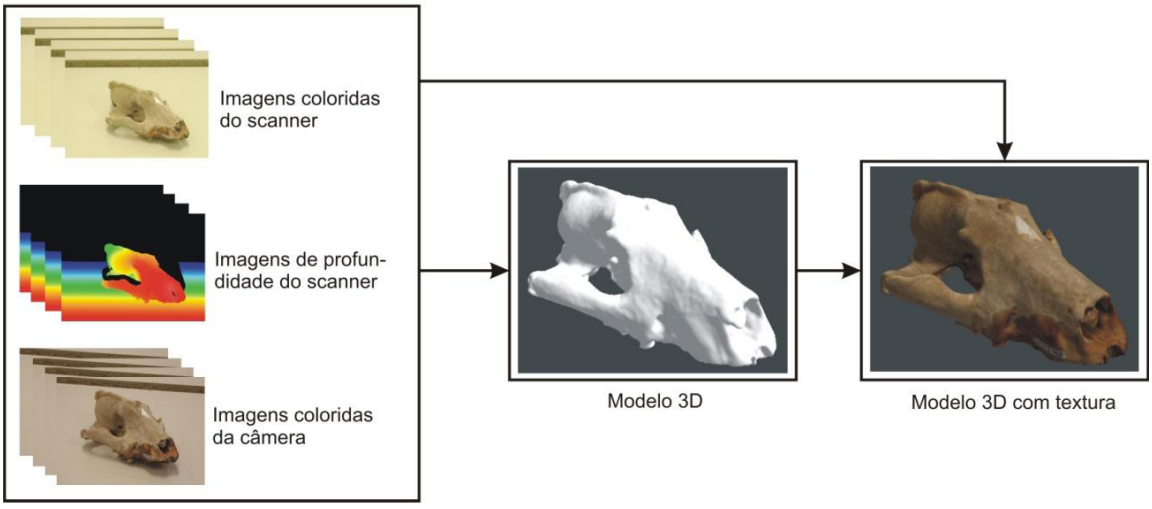
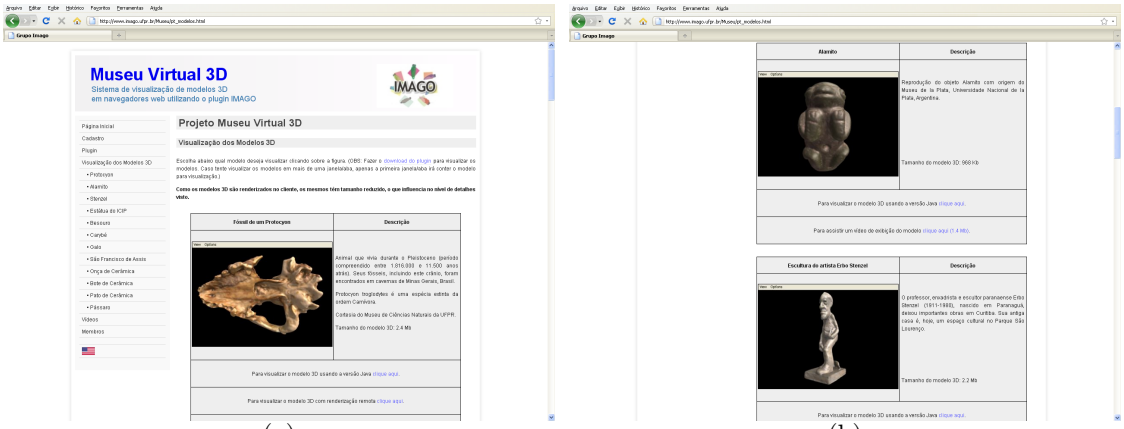


Figura 6.2: Ilustração do processo de preservação digital do fóssil de um Protocyon.

6.2 Disponibilização dos Modelos 3D

Após todas as fases para geração dos modelos 3D, os mesmos são disponibilizados para visualização. O Museu Virtual 3D[1] está disponível na Internet, sendo necessário efetuar um cadastro e *login*. A página *web* do Museu Virtual 3D (Figura 6.3) apresenta uma breve descrição sobre os objetos e também disponibiliza videos dos modelos 3D renderizados. A visualização é feita no navegador *web*, através da utilização do *plugin* IMAGO ou do IMAGO 3D Viewer dependendo da escolha do usuário.



(a) (b)

Figura 6.3: Páginas *web* do Museu Virtual 3D.

São disponibilizados modelos 3D de baixa resolução para a visualização, porém é permitido o acesso aos modelos 3D de alta resolução desde que comprovado o interesse para realização de estudo e pesquisa. Isso se deve ao fato de algumas obras possuírem direitos autorais, pela baixa capacidade computacional de alguns usuários para renderizar os modelos 3D de alta resolução e também para diminuir o tempo de transmissão na Internet.

A primeira forma de disponibilização dos modelos 3D adotada pelo projeto foi a utilização de um Sistema para Visualização de Museu Virtual 3D descrito em [25]. O sistema possui a arquitetura cliente-servidor (Figura 6.4), no qual os modelos 3D são armazenados em um banco de dados no servidor. A primeira versão do *plugin* IMAGO foi utilizada para permitir a visualização dos modelos 3D além de ser o cliente desse sistema.

[1] http://www.imago.ufpr.br/Museu

Com as informações sobre o servidor (endereço ip e porta) armazenadas no código do *plugin* IMAGO, o cliente faz uma requisição para o servidor o qual envia o modelo 3D. Os testes realizados nesse sistema apontaram falhas no processo de comunicação em ambientes de rede que utilizam *proxy* ou *firewall* bloqueando a conexão para a porta do servidor. Por esse motivo alguns usuários ficaram impedidos de visualizar os modelos 3D.

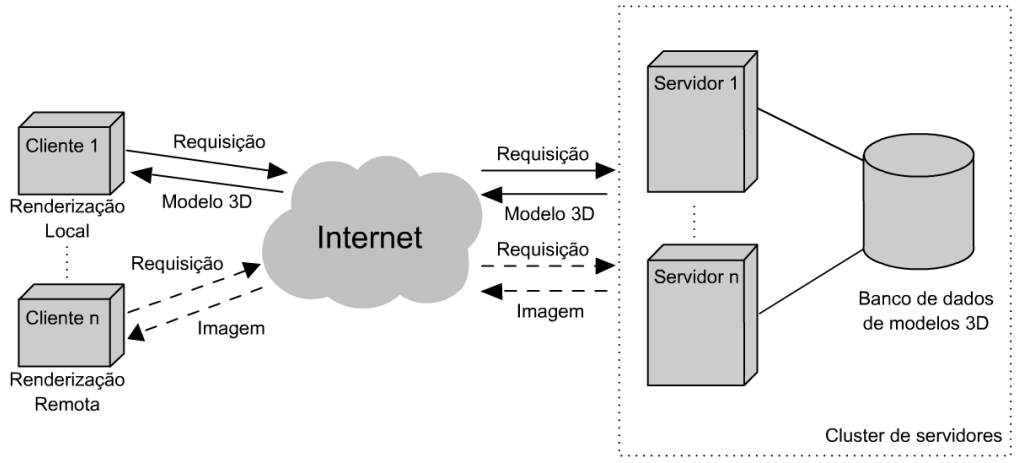


Figura 6.4: Sistema para visualização de Museu Virtual 3D do IMAGO.

As adaptações no *plugin* IMAGO, para permitir a sua utilização por qualquer pessoa ou projeto, tornaram fáceis as modificações no sistema. Com a utilização do arquivo de configuração, as mudanças na interface ou nas informações sobre o servidor podiam ser feitas sem a necessidade de codificação e compilação do visualizador 3D. Enviando o modelo 3D dentro do arquivo zip, os usuários não ficaram mais impedidos de realizar a visualização por falhas de comunicação com o servidor.

A característica mais impactante na inclusão do IMAGO 3D Viewer no Museu Virtual 3D foi a portabilidade. Na divulgação do projeto, foi possível observar que a maioria dos usuários escolhe a versão Java para iniciar a visualização automaticamente, sem ter a preocupação de qual navegador *web* está utilizando e sem a instalação de um *plugin*. A interface simples, com a utilização de ícones e o retorno das ações do usuário através da barra de status, tornou a visualização mais simplificada e interativa.

6.2.1 Sistema de Renderização Remota

A técnica de renderização remota dos modelos 3D pode tornar os museus virtuais mais seguros e confiáveis, ampliando a disponibilização através da Internet. Para esse fim, um sistema de renderização remota, similar ao apresentado em [20], está sendo incorporado ao Museu Virtual 3D. Os estudos sobre a técnica e o desenvolvimento desse sistema são feitos por um aluno de mestrado do projeto.

O sistema permite interagir com modelos 3D de baixa resolução e visualizar imagens do modelo 3D de alta resolução renderizado no servidor. Para permitir a visualização através dessa técnica, modificações foram feitas tanto no *plugin* IMAGO quanto no IMAGO 3D Viewer. Os visualizadores 3D enviam os parâmetros da visualização (coordenadas 3D, informações sobre textura, iluminação entre outros) para o servidor de renderização remota, recebem a resposta do servidor e exibem a imagem sobre o modelo 3D. A Figura 6.5 exemplifica o funcionamento desse sistema.

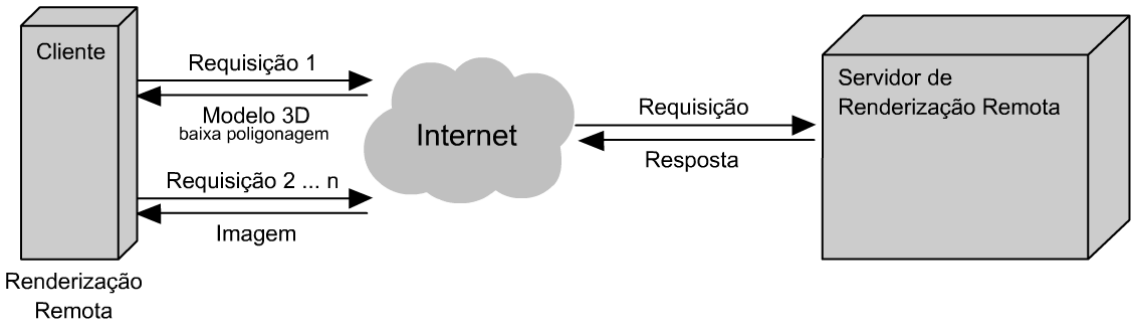


Figura 6.5: Sistema de Renderização Remota do Museu Virtual 3D.

Os testes realizados mostraram o bom funcionamento da técnica de renderização remota nos dois visualizadores 3D. As vantagens oferecidas pela linguagem Java, como por exemplo os recursos para realizar a comunicação na Internet e manipulação de imagens, influenciaram na escolha do IMAGO 3D Viewer para ser utilizado no Museu Virtual 3D com a técnica de renderização remota. Outro motivo se deve à dificuldade de manutenção de duas versões do *plugin* IMAGO (Linux e Windows), proporcionando uma queda de produtividade no desenvolvimento do sistema.

Com a finalização dos estudos sobre a renderização remota, o próximo passo será a disponibilização da visualização com renderização remota no IMAGO 3D Viewer. Dessa maneira, outros projetos relacionados à preservação digital de objetos naturais e culturais poderão criar seus próprios sistemas de renderização remota sem a necessidade de desenvolver um visualizador 3D para isso.

CAPÍTULO 7

CONCLUSÃO

Esta dissertação apresenta um estudo sobre a visualização interativa de modelos 3D realistas de objetos naturais e culturais, bem como a disponibilização dos mesmos através da Internet. Entre as contribuições desse trabalho têm-se:

- A descrição dos meios de visualização de modelos 3D de objetos naturais e culturais gerados nas etapas do processo de preservação digital;
- Síntese dos aspectos que devem ser considerados para a disponibilização de modelos 3D realistas na Internet;
- O desenvolvimento e avaliação de duas ferramentas (*plugin* IMAGO e IMAGO 3D Viewer) que permitem a visualização 3D interativa para serem utilizadas por projetos relacionados à preservação digital, museus virtuais 3D, entre outros.

Os projetos de preservação digital, em sua maioria, se dedicam à geração de modelos 3D e em técnicas para aumentar o realismo desses modelos. A visualização 3D não recebe a atenção merecida, pois os projetos tendem a procurar uma solução rápida, como por exemplo a utilização de visualizadores VRML. Desta forma, a visualização interativa fica prejudicada devido às limitações dessa tecnologia.

A criação de uma ferramenta inovadora para visualização de modelos 3D realistas de objetos naturais e culturais permite ampliar e garantir maior preservação das atividades de pesquisa e segurança aos objetos pelo acesso remoto. A qualidade visual dos modelos 3D em conjunto com a alta interatividade e a facilidade de uso são importantes para o usuário final e também para os desenvolvedores.

Para o desenvolvedor da aplicação que vai utilizar um visualizador 3D, escolher as opções da interface e as operações sobre modelos 3D, permite que sua aplicação seja

configurada de acordo com seus objetivos. Os usuários podem se beneficiar com essa escolha, que quando bem estruturada, certamente facilitará a interação para usuários. Assim, os projetos poderão ampliar a divulgação das suas pesquisas, não apenas para a comunidade científica mas para todas as pessoas, sem exigir habilidades das mesmas para realizar a visualização 3D.

A criação do *plugin* IMAGO permitiu a descoberta de vários fatores relevantes em relação a visualização interativa, os quais foram incorporados no desenvolvimento do IMAGO 3D Viewer. Os resultados obtidos na comparação dos dois visualizadores ajudaram a desmitificar a linguagem Java para desenvolvimento 3D, a qual mostrou ser tão adequada quanto a linguagem C/C++. Além disso, a característica de portabilidade da linguagem facilitou o acesso aos modelos 3D no Museu Virtual 3D, permitindo realizar a visualização na maioria dos navegadores *web* e sistemas operacionais.

A experiência na área da preservação digital nos mostrou a necessidade de garantir a segurança nos modelos 3D e permitir que vários tipos de usuários, que diferem quanto à capacidade de processamento de seus computadores, possam visualizar modelos 3D realistas com bom nível de interatividade. O fato de um visualizador 3D possuir meios alternativos para que a visualização ocorra, por exemplo a utilização da técnica de renderização remota, é uma característica inovadora que precisa ser incorporada nos demais visualizadores existentes.

Como trabalho futuro, pretende-se realizar testes de usabilidade do IMAGO 3D Viewer com usuários e profissionais da área de Antropologia, História e Museologia. O contato com esses profissionais permitirá a inclusão de novas funcionalidades que poderão auxiliar as atividades de pesquisa dessas áreas. Além disso, a integração com um sistema de renderização remota proporcionará maior acessibilidade ao Museu Virtual 3D do IMAGO, incluindo usuários que possuem computadores com recursos de hardware limitados. Dessa maneira, podemos facilitar o acesso à modelos 3D realistas de objetos naturais e culturais para um número maior de usuários.

BIBLIOGRAFIA

[1] Beatriz T. Andrade. Utilizando fotografias digitais de alta qualidade na geração de textura para modelos 3D: Uma abordagem prática na preservação digital de acervos culturais e naturais. Dissertação de Mestrado, Universidade Federal do Paraná, 2009.

[2] Beatriz T. Andrade, Olga R. P. Bellon, Luciano Silva, e Alexandre Vrubel. Enhancing color texture quality of 3D models for digital preservation of indigenous ceramic artworks. *Proceedings of IEEE International Conference on Computer Vision, Workshop on eHeritage and Digital Art Preservation*, 2009.

[3] Paul J. Besl. Active, optical range imaging sensors. *Machine Vision and Applications*, 1(2):127–152, 1988.

[4] Grigore C. Burdea e Philippe Coiffet. *Virtual Reality Technology, Second Edition with CD-ROM.* Wiley-IEEE, Junho de 2003.

[5] Irene Cheng e Anup Basu. Super high resolution 3D imaging and efficient visualization. *International Symposium on 3D Data Processing Visualization and Transmission*, páginas 186–189, 2002.

[6] Frank Corcoran, Jeffery Demaine, Louis-Guy Dicaire, Michel Picard, e John Taylor. Inuit 3D: An interactive virtual 3D web exhibition. David Bearman e Jennifer Trant, editors, *Museums and the Web 2002.*

[7] Francine Evans, Steven Skiena, e Amitabh Varshney. Optimizing triangle strips for fast rendering. *VIS '96: Proceedings of the 7th Conference on Visualization '96*, páginas 319–326, 1996.

[8] Henry M. Gladney. *Preserving Digital Information - Information Storage and Retrieval Journals.* Springer, 2007.

[9] Jed Hartman e Josie Wernecke. *The VRML 2.0 handbook: building moving worlds on the web.* 1996.

[10] Tim Hawkins, Jonathan Cohen, e Paul Debevec. A photometric approach to digitizing cultural artifacts. *VAST '01: Proceedings of the 2001 Conference on Virtual Reality, Archeology, and Cultural Heritage*, páginas 333–342, 2001.

[11] Hugues Hoppe. New quadric metric for simplifying meshes with appearance attributes. *VISUALIZATION '99: Proceedings of the 10th IEEE Visualization 1999 Conference (VIS '99)*, Washington, DC, USA, 1999. IEEE Computer Society.

[12] K. Ikeuchi e D. Miyazaki. *Digitally Archiving Cultural Objects*. Springer, 2008.

[13] Katsushi Ikeuchi, Takeshi Oishi, Jun Takamatsu, Ryusuke Sagawa, Atsushi Nakazawa, Ryo Kurazume, Ko Nishino, Mawo Kamakura, e Yasuhide Okamoto. The great buddha project: Digitally archiving, restoring, and analyzing cultural heritage objects. *International Journal of Computer Vision*, 75(1):189–208, Outubro de 2007.

[14] Zara Jiri e Slavik Pavel. Cultural heritage presentation in virtual environment: Czech experience. *DEXA '03: Proceedings of the 14th International Workshop on Database and Expert Systems Applications*, páginas 92–96, 2003.

[15] Wang Kai, Guillaume Lavoué, Florence Denis, e Atilla Baskurt. A comprehensive survey on three-dimensional mesh watermarking. *IEEE Transactions on Multimedia*, 10(8):1513–1527, 2008.

[16] Athanasis Karoulis, Stella Sylaiou, e Martin White. Combinatory usability evaluation of an educational virtual museum interface. *ICALT '06: Proceedings of the Sixth IEEE International Conference on Advanced Learning Technologies*, páginas 352–354, 2006.

[17] Judith Kelner e Veronica Teichrieb. *Realidade Virtual e Aumentada: Conceitos, Projeto e Aplicações*, capítulo Técnicas de Interação para Ambientes de Realidade Virtual e Aumentada, páginas 52–70. Sociedade Brasileira de Computação - SBC, 2007.

[18] Mark J. Kilgard. A practical and robust bump-mapping technique for today's gpus. *Advanced OpenGL Game Development, Proceedings*, 2000.

[19] David Koller e Marc Levoy. Protecting 3D graphics content. *Communications of the ACM*, 48(6):74–80, 2005.

[20] David Koller, Michael Turitzin, Marc Levoy, Marco Tarini, Giuseppe Croccia, Paolo Cignoni, e Roberto Scopigno. Protected interactive 3D graphics via remote rendering. *ACM Trans. Graph.*, 23(3):695–703, 2004.

[21] Kyong-Ho Lee, Oliver Slattery, Richang Lu, Xiao Tang, e Victor Mccrary. The state of the art and practice in digital preservation. *Journal of Research of the National Institute of Standards and Technology*, 107(1):93–106, Janeiro de 2002.

[22] Marc Levoy, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, e Duane Fulk. The Digital Michelangelo Project: 3D Scanning of Large Statues. *Proceedings of ACM SIGGRAPH 2000*, páginas 131–144, 2000.

[23] Thomas Lewiner. 3D compression: from a to zip a first complete example. *Revista de Informática Teórica e Aplicada*, 15(1):09–38, 2008.

[24] Francisco Luengo, Mariela Contreras, Aurely Leal, e Andres Iglesias. Interactive 3D graphics applications embedded in web pages. *CGIV '07: Proceedings of the Computer Graphics, Imaging and Visualisation*, páginas 434–440, 2007.

[25] Caroline M. Mendes, Dyego R. Drees, Olga R. P. Bellon, e Luciano Silva. Sistema para visualização de museu virtual 3D. V Workshop of Undergraduated Work, SIBGRAPI, 2007.

[26] David R. Nadeau. Tutorial: Building virtual worlds with VRML. *IEEE Computer Graphics and Applications*, 19(2):18–29, 1999.

[27] Yasuhide Okamoto, Takeshi Oishi, e Katsushi Ikeuchi. *Digitally Archiving Cultural Objects*, capítulo Editing, Retrieval, and Display System of Archeological Information on Large 3D Geometric Models, páginas 441–455. Springer, 2008.

[28] George Pavlidis, Anestis Koutsoudis, Fotis Arnaoutoglou, Vassilios Tsioukas, e Cristodoulos Chamzas. Methods for 3D digitization of cultural heritage. *Journal of Cultural Heritage*, 8(1):93–98, 2007.

[29] Randall M. Rohrer e Edward Swing. Web-based information visualization. *IEEE Computer Graphics and Applications*, 17(4):52–59, 1997.

[30] Luciano Silva, Olga R. P. Bellon, e Kim L. Boyer. Computer vision and graphics for heritage preservation and digital archaeology. *Revista de Informática Teórica e Aplicada*, 11(1):09–31, 2004.

[31] Sun Microsystem Inc. *The Source for Java(TM) Technology.*

[32] Alexandre Vrubel. Pipeline para reconstrução digital de objetos com scanners 3D de triangulação a laser : aplicação na preservação digital de acervos naturais e culturais. Dissertação de Mestrado, Universidade Federal do Paraná, 2008.

[33] Alexandre Vrubel, Olga R.P. Bellon, e Luciano Silva. A 3D reconstruction pipeline for digital preservation. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:2687–2694, 2009.

[34] Rafal Wojciechowski, Krzysztof Walczak, Martin White, e Wojciech Cellary. Building virtual and augmented reality museum exhibitions. *Web3D '04: Proceedings of the ninth international conference on 3D Web technology*, páginas 135–144, 2004.

[35] Jiri Zara. Virtual reality and cultural heritage on the web. *Proceedings of the 7th International Conference on Computer Graphics and Artificial Intelligence*, 2004.

[36] Jiri Zara e Jaroslav Krivanek. Graphics performance benchmarking based on VRML browsers. *Proceedings of the Virtual Reality International Conference*, páginas 111–120, 2001.

APÊNDICE A

TRIANGLE STRIPS

Em um Triangle Strip (faixa de triângulo), os três primeiros vértices definem o primeiro triângulo. O vértice seguinte, juntamente com os dois últimos vértices formam o segundo triângulo e assim sucessivamente. Por exemplo, considere os triângulos {0,1,2}, {1,2,3} e {2,3,4}, a faixa de triângulos será representada pela sequência {0,1,2,3,4} (Figura A.1).

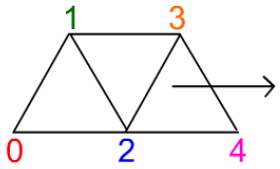


Figura A.1: Um Triangle Strip.

A criação de faixas a partir de um triângulo arbitrário é um problema NP-completo [7], portanto é necessário definir heurísticas para resolvê-lo. O algoritmo padrão para a criação de Triangle Strips possui os seguintes passos[1]:

1. Escolhe-se um triângulo inicial
2. Escolhe-se a direção para caminhar ao longo da faixa
3. Estende-se a faixa na direção escolhida até chegar em um triângulo sem conexões à diante
4. Volta para o passo 1 até que todos os triângulos tenham sido visitados

O ponto crucial do problema é definir qual o próximo triângulo que será incluído na sequência. A heurística utilizada neste trabalho atribui pesos para todos os triângulos da malha, sendo escolhido o triângulo candidato que possui o menor peso. O algoritmo pode ser descrito através do pseudo-código A.1.

Algoritmo A.1 Pseudo-código do algoritmo para criação de Triangle Strips.

```
1: Defina uma estrutura auxiliar para armazenar as faces adjacentes a cada face da malha
2: Defina uma estrutura auxiliar para armazenar as faces adjacentes a cada vértice da
   malha
3: Marque todas as faces das estruturas auxiliares como não visitadas
4: faceAtual ← Uma face que possua o maior número de faces adjacentes não visitadas
5: repita
6:    se faceAtual não possui faces adjacentes não visitadas então
7:       termine o strip atual
8:    senão
9:       se faceAtual possui apenas 1 face adjacente não visitada então
10:         inclua essa face adjacente no strip atual
11:      senão
12:         para cada face i adjacente a faceAtual faça
13:            Encontre os vértices v1 e v2 compartilhados
14:            pesoV1 ← número de faces adjacentes a v1 não visitadas
15:            pesoV2 ← número de faces adjacentes a v2 não visitadas
16:            pesoFace ← número de faces adjacentes a face i não visitadas
17:            pesoFaceTotal ← pesoFace + pesoV1 + pesoV2
18:            se pesoFaceTotal < melhorPeso então
19:               melhorPeso ← pesoFaceTotal
20:               proximaFace ← i
21:            fim se
22:         fim para
23:         inclua proximaFace no strip atual
24:      fim se
25:   fim se
26: até todas as faces serem visitadas
```

[1] http://www.codercorner.com/Strips.htm