COPPE/UFRJ

CONSTRUÇÃO DE UM VISUALIZADOR PARA SIMULAÇÕES COMPUTACIONAIS AMBIENTAIS

Renan Leser de Medeiros

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Civil, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Civil.

Orientadores: Luiz Landau
Luiz Paulo de Freitas Assad

Rio de Janeiro
Agosto de 2010

CONSTRUÇÃO DE UM VISUALIZADOR PARA SIMULAÇÕES COMPUTACIONAIS AMBIENTAIS

Renan Leser de Medeiros

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA CIVIL.

Examinada por:

Prof. Luiz Landau, D. Sc.

Dr. Luiz Paulo de Freitas Assad, D. Sc.

Prof. Gerson Gomes Cunha, D. Sc.

Prof. José Luis Drummond Alves, D. Sc.

Prof. Marcus Vinicius Dohmann Brandão, D. Sc.

RIO DE JANEIRO, RJ - BRASIL

AGOSTO DE 2010

Medeiros, Renan Leser de

Construção de um Visualizador para Simulações Computacionais Ambientais / Renan Leser de Medeiros – Rio de Janeiro: UFRJ/COPPE, 2010.

XI, 90 p.: il.; 29,7 cm.

Orientadores: Luiz Landau

Luiz Paulo de Freitas Assad

Dissertação (mestrado) – UFRJ/ COPPE/ Programa de Engenharia Civil, 2010.

Referencias Bibliográficas: p. 86-90.

1. Visualização Científica. 2. Modelagem Computacional Ambiental. 3. Computação Gráfica. I. Landau, Luiz, et al. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Civil. III. Título

"Believe you can, believe you can't, either way you're right"

(Acredite que você pode. Ou não. De qualquer forma, no final você estará certo.)

Sebastien Legrain

Dedico este trabalho à minha mãe, Isaura, por ter sempre incentivado meus estudos, principalmente o ingresso no mestrado, à Carol, minha namorada e seus pais Celso e Lúcia pelo apoio incondicional nos momentos difíceis e à minha avó Nair e meu irmão Rafael, pelos bons conselhos.

Agradeço ao meu orientador Luiz Landau por acreditar na minha pessoa, pela oportunidade de trabalhar, estudar, crescer profissionalmente e desenvolver este trabalho no âmbito do LAMCE. Serei eternamente grato.

Agradeço ao meu orientador do NUMA, Luiz Paulo Assad, pela amizade, orientação e paciência ao longo da elaboração deste trabalho.

Agradeço ao professor Gerson Gomes Cunha, por todas as aulas ministradas e conhecimento compartilhado, que tornou este trabalho possível.

Agradeço mais uma vez a minha namorada, Carolina, por ter sempre me incentivado a não desistir dos desafios que surgiram.

Agradeço novamente a minha mãe, Isaura, por ter acreditado e investido na minha formação acadêmica.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

CONSTRUÇÃO DE UM VISUALIZADOR PARA SIMULAÇÕES COMPUTACIONAIS AMBIENTAIS

Renan Leser de Medeiros

Agosto / 2010

Orientadores: Luiz Landau
Luiz Paulo de Freitas Assad

Programa: Engenharia Civil

O crescente desenvolvimento das atividades ligadas à Indústria do Petróleo e Gás no Brasil vem demandando um maior nível de detalhamento na análise e gerenciamento de resultados gerados por modelos computacionais ambientais. Tal demanda exige o acesso a um enorme volume de dados em plataformas computacionais que facilitem a interação com o usuário final responsável pela tomada de decisões. Este trabalho tem como propósito o desenvolvimento de uma metodologia para criação de gráficos tridimensionais interativos de visualização científica com capacidade de armazenamento de campos de variáveis prognósticas obtidas a partir da aplicação de modelos computacionais ambientais, atmosféricos, hidrodinâmicos - marinhos e fluviais - e de dispersão de poluentes, em diferentes escalas espaciais e temporais.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

DEVELOPMENT OF AN ENVIRONMENTAL COMPUTATIONAL SIMULATIONS VISUALIZER

Renan Leser de Medeiros

August / 2010

Advisors: Luiz Landau
Luiz Paulo de Freitas Assad

Department: Civil Engineering

The urging developments of activities from the Oil and Gas Industry from Brazil have been demanding a higher level of detail in the interpretation of space and time distributions of prognostic variables from environment computational models. This demand requires the access to a huge volume of data in such a way that make it easier the interaction between the final user responsible to decision making and the data. The following work has the aim to develop a methodology for interactive tridimensional figures built from scientific data, able to store prognostic variable fields obtained from environmental, atmospheric, hydrodynamic - marine and fluvial - and oil dispersion computational models, with different time variations and scale.

ÍNDICE

CAPÍTULO 1
INTRODUÇÃO

1.1 – Introdução

O crescente desenvolvimento das atividades ligadas à Indústria do Petróleo e Gás no Brasil trouxe consigo a necessidade da redução de riscos sócio-econômicos-ambientais ocasionados por estas atividades. Neste sentido, projetos de pesquisa e desenvolvimento multidisciplinares fomentados pela indústria vêm surgindo com o objetivo de suprir demandas em diversas áreas do conhecimento. Um exemplo desses é o projeto PIATAM (Projeto Inteligência Sócio Ambiental Estratégica da Indústria do Petróleo e Gás na Amazônia), desenvolvido através da parceria entre a Financiadora de Estudos e Projetos (FINEP) e a PETROBRAS. Este projeto conta com diversos grupos de pesquisa, nas mais variadas áreas do conhecimento, como doenças tropicais, imunologia, sócio-economia, solos, flora, modelagem ambiental, dentre outros. Uma das áreas de conhecimento fundamentais para o gerenciamento ambiental e de riscos associados às atividades da indústria de petróleo e gás natural é a modelagem ambiental. Pode-se pensar que o desenvolvimento e aplicação de modelos que representam a dinâmica ambiental de uma determinada região possam ser aplicados de duas formas básicas: operacional e estratégica, onde um dos desafios é a criação de mecanismos que possam ser tanto utilizados em estudos ambientais, quanto em programas de atendimento a situações emergenciais de vazamento de óleo na atracação e transbordo de navios, encalhes de balsas e etc., fazendo previsões numéricas da situação atmosférica (principalmente regime de ventos), da hidrodinâmica do rio, da formação de ondas na região costeira e da dispersão de constituintes no meio aquoso [1]. Estas previsões, geradas por modelos computacionais ambientais, demandam um alto nível de detalhamento na sua visualização e interpretação. Tal demanda exige o acesso a um enorme volume de dados em plataformas computacionais que facilitem a interação com o usuário final responsável pela tomada de decisões.

O presente trabalho visa reunir, em um ambiente interativo tridimensional, resultados oriundos de modelos computacionais ambientais gerados no âmbito de projetos apoiados tecnicamente pelo NUMA (Núcleo de Modelagem Ambiental do Laboratório de Métodos Computacionais em Engenharia da UFRJ), como: o próprio PIATAM (acima citado) e o PIATAM-MAR (Potenciais Impactos do Transporte de Petróleo e derivados na zona costeira amazônica). Tais projetos têm como área de interesse regiões costeiras marinhas e fluviais da Amazônia brasileira. Dentro dos objetivos gerais dos projetos citados, as

atividades do NUMA, têm como objetivo básico a implementação de modelos computacionais ambientais (hidrodinâmicos fluviais e marinhos, atmosféricos e de dispersão de poluentes) para apoiar o gerenciamento ambiental voltado para as atividades da indústria do petróleo na Amazônia. Um ponto a ser destacado é a utilização de diferentes escalas espaciais e temporais na realização da modelagem dos dados. Como por exemplo, na modelagem hidrodinâmica marinha. Esta atividade vem sendo desenvolvida com o objetivo de simular a hidrodinâmica de regiões da costa norte do Brasil, considerando-se a influência de processos marinhos e atmosféricos de diferentes escalas espaciais e temporais na dinâmica local. A modelagem realizada deve, portanto, contemplar variabilidades espaciais e temporais existentes na circulação oceânica sobre a Plataforma Continental Amazônica (PCA), na vazão dos rios, nas marés e na circulação atmosférica. A estratégia utilizada envolve a utilização de três diferentes modelos hidrodinâmicos. Para a modelagem da dinâmica marinha na região de interesse vem sendo utilizado um modelo oceânico global, que fornece condições de contorno para um modelo oceânico de escala regional. Este, por sua vez, fornece condições de contorno para o modelo de maior resolução espacial e suficientemente grande para representar os principais aspectos dinâmicos da circulação costeira nas regiões de interesse [1]. A ferramenta computacional escolhida para pré-visualização dos dados gerados foi o Paraview. Tal escolha é justificada, por exemplo, pela abertura do código computacional, imensa gama de formatos de entrada e saída dos dados, facilidade na confecção de animações e pela possibilidade de utilização de processamento paralelo para sua execução.

Através deste trabalho, o usuário poderá interagir com os dados de uma forma muito mais interativa, devido à facilidade e riqueza em detalhes que uma visualização tridimensional apresenta.

Neste trabalho serão apresentados resultados obtidos por simulações computacionais usando o modelo de circulação geral dos oceanos chamado Modular Ocean Model [2] , o Princeton Ocean Model [3] e o NICOIL [4].

1.2 - Definição do Problema

As ferramentas existentes de visualização científica, como Paraview e Ensight não possuem formatos de saída que permitam a visualização de seus dados fora de seus programas base. Ambos os programas citados exportam em VRML, mas apenas modelos estáticos, sem variação temporal e sem deformação de malha.

Neste contexto, o problema surge em como oferecer ferramentas que facilitem a visualização em três dimensões destes dados, viabilizando resultados de modelos computacionais ambientais complexos, com campos escalares e vetoriais, variáveis no

tempo, dentro de um ambiente tridimensional interativo e portátil. Com base nesses requisitos, o presente trabalho foi proposto.

1.3 - Motivação

O NUMA possui um acervo com várias simulações de modelos computacionais hidrodinâmicos - atmosféricos, oceânicos ou fluviais, porém, até o início da elaboração deste trabalho, não possuía visualizações tridimensionais destas simulações.

Esta dissertação surgiu inicialmente como forma de agregar este conhecimento ao Núcleo, gerando novas perspectivas de visualização para os resultados gerados, como por exemplo o ângulo de visão do fundo do Rio Solimões, próximo à cidade de Coari, na Amazônia, mostrando a trajetória que uma mancha de óleo simulada poderia trilhar no caso de um acidente - informação crucial para a tomada de decisão no processo de Gestão Ambiental e contenção da mancha de óleo.

Uma vez que diversos modelos tridimensionais foram criados, surgiu a necessidade de montar um ambiente onde os usuários pudessem acessar e visualizar esses dados. Tal necessidade foi fundamental para a definição do escopo deste trabalho. Este trabalho cumpre também com o papel de normalizar o acesso ao conhecimento das diferentes áreas em estudo : modelagem hidrodinâmica fluvial e marinha, atmosférica e de dispersão de poluentes.

1.4 - Objetivo

O objetivo desta dissertação consiste em descrever a criação e funcionamento de um visualizador tridimensional de dados científicos para acesso e visualização de campos prognósticos tridimensionais de variáveis ambientais obtidas a partir da aplicação de modelos computacionais ambientais.

1.5 - Objetivo Específico

Construir um ambiente interativo tridimensional que compile todas as visualizações executadas pelo NUMA, tanto no âmbito de projetos como o PIATAM e o PIATAM-MAR quanto as simulações destinadas a outras áreas geográficas, na escala global e regional.

Este trabalho prevê construir uma metodologia, que possa ser sempre aplicada a fim de manter esse ambiente tridimensional atualizável pelos usuários do Núcleo (e por outros membros da comunidade científica que se interessarem em contribuir para com o visualizador) - deixando claro que deverão seguir a mesma metodologia para formatação

dos dados ou encaminhá-los ao NUMA para que o pesquisador responsável faça a conversão.

1.6 - Relevância e Justificativa

Em Netto et al. [5], é citado o uso de um ambiente virtual como interface tridimesional para manipulação de dados em redes de distribuição de energia elétrica, com a finalidade de auxiliar a "tomada de decisão" em um sistema para redução de perda e facilitar a interpretação (avaliação) das soluções propostas pelo sistema. O INFOPAE [6] da Petrobrás é um sistema de Gerenciamento Ambiental bidimensional, baseado em séries de dados relacionais, utilizando questionários e modelos de dados para gerar uma solução ágil para acidentes ambientais.

Hoje, sistemas de simulação e prevenção tridimensionais já são uma realidade em diversos setores (de produção). Eles reduzem riscos, oferecem treinamento de baixo custo e auxiliam nos processos de tomada de decisão[7]. O presente trabalho tem como intenção levar os simuladores ambientais um passo adiante ao inserir o processo de interação usuário-máquina de uma forma imersiva e interativa.

Apesar de já existirem programas específicos para modelagem e visualização de dados científicos, o foco deste trabalho é na saída de dados destes programas - como transportar e visualizar estes dados *fora* dos programas específicos, sem necessidade de instalação de plugins, mas mantendo a interação e a visualização de dados transientes.

1.7 - A Concepção do Projeto

O processo de construção deste visualizador teve início com a idéia de armazenar resultados de modelos em escala espacial regional e global em uma única plataforma de visualização tridimensional. Diferentes testes e versões foram criadas e avaliadas até a construção da versão final aqui descrita.

1.8 - Composição do Sistema

Este sistema foi criado seguindo um processo de conversão, tratamento e manipulação de resultados obtidos de modelos em escala espacial regional e global, utilizando-se das últimas técnicas de Realidade Virtual e Computação Gráfica disponíveis, a serem descritas nesta dissertação.

1.9 - Atividades

O referido trabalho consiste na criação de uma metodologia para visualização científica tridimensional que contenha dados de modelagem global e regional, seguindo as seguintes atividades:

- Pesquisa dos melhores formatos de Visualização Científica dentro do escopo do projeto e dos dados gerados;
- Modelagem tridimensional dos dados;
- Conversão de arquivos para leitura em programas de modelagem 3D;
- Preparação para exibição Real-Time em placas de vídeo;
- Criação de interface gráfica e estudo de interação homem-máquina.

1.9.1 - Requisitos Básicos

O Visualizador deve proporcionar:

- Visualização tridimensional de dados científicos transientes e não-transientes;
- Facilidade de Interação;
- Uma base de dados para todas as simulações produzidas pelo NUMA.

1.9.2 - Resultados Esperados

Espera-se que no ato da conclusão desta dissertação, o produto gerado ofereça:

- Facilidade de interação, visualização e interpretação dos dados comparada a visualização bidimensional;
- Disponibilização Online das visualizações desenvolvidas;
- Fundamentação de uma metodologia para que outros pesquisadores alimentem o ambiente ou contribuam com seus dados para com o mesmo.

1.10 - Organização da Dissertação

O conteúdo deste trabalho está dividido e exemplificado nos 5 capítulos abaixo:

CAPÍTULO 1 - Introdução - apresenta os objetivos do presente trabalho;

CAPÍTULO 2 - Estado da Arte - análise prévia dos programas de visualização científica e iniciativas semelhantes a este trabalho;

CAPÍTULO 3 - Testes Preliminares - exemplifica as técnicas e iniciativas realizadas no início deste trabalho;

CAPÍTULO 4 - Ferramentas e Dispositivos - todo o conhecimento e programas necessários para a construção da interface;

CAPÍTULO 5 - Construção do Visualizador - todos os passos da metodologia utilizada;

CAPÍTULO 6 - Resultados - todos os resultados de testes de campo e finalização da Interface.

CAPÍTULO 7 - Conclusão -conclusão sobre o trabalho realizado e implementações futuras.

CAPÍTULO 2
ESTADO DA ARTE

2.1– Introdução

Neste capítulo serão descritas iniciativas semelhante ao trabalho proposto nesta dissertação.

2.2– Paraview e Ensight

Os dois principais softwares presentes no mercado de visualização científica são o Paraview e o Ensight.

O Paraview, como será descrito com mais detalhes nos próximos capítulos, trata-se de um programa open-source, multi-plataforma para análise e visualização de dados. Ele permite o tratamento dos dados utilizando-se de diversos filtros, além de permitir o processamento em paralelo, algo extremamente necessário quando se trata de um grande volume de dados para se processar.

Já o Ensight é um software de visualização científica comercial, onde o foco é a separação dos dados em diversos pacotes, para fins de processamento paralelo. No Ensight, toda a informação é associada em partes, e quase todas as ações são aplicadas à essas partes.

Cada parte consiste em nós e elementos (elementos são um conjunto de nós conectados em uma forma geométrica particular). Cada nó, que é compartilhado pelos elementos adjacentes é definido por sua coordenada de localização no modelo.

No Ensight, é possível especificar informações adicionais para cada parte. Esses atributos dizem ao programa como mostrar cada parte e como essa parte responde aos comandos do programa. Dentre estes atributos, podemos destacar: coloração, visibilidade, modo de exibição, elementos 3D e etc [8].

Assim como o Paraview, o Ensight importa diversos formatos, mas o grande ganho em processamento particionado se dá quando os arquivos são escritos no formato padrão do Ensight, o Ensight Gold.

Esse arquivo, inclusive, pode ser carregado em outros visualizadores, como o próprio Paraview, para fins de processamento em paralelo.

Porém, o grande gargalo de ambos os programas é a exportação de dados.

O Paraview exporta nos seguintes formatos: .POV; .VRML; .X3D; .X3DB.

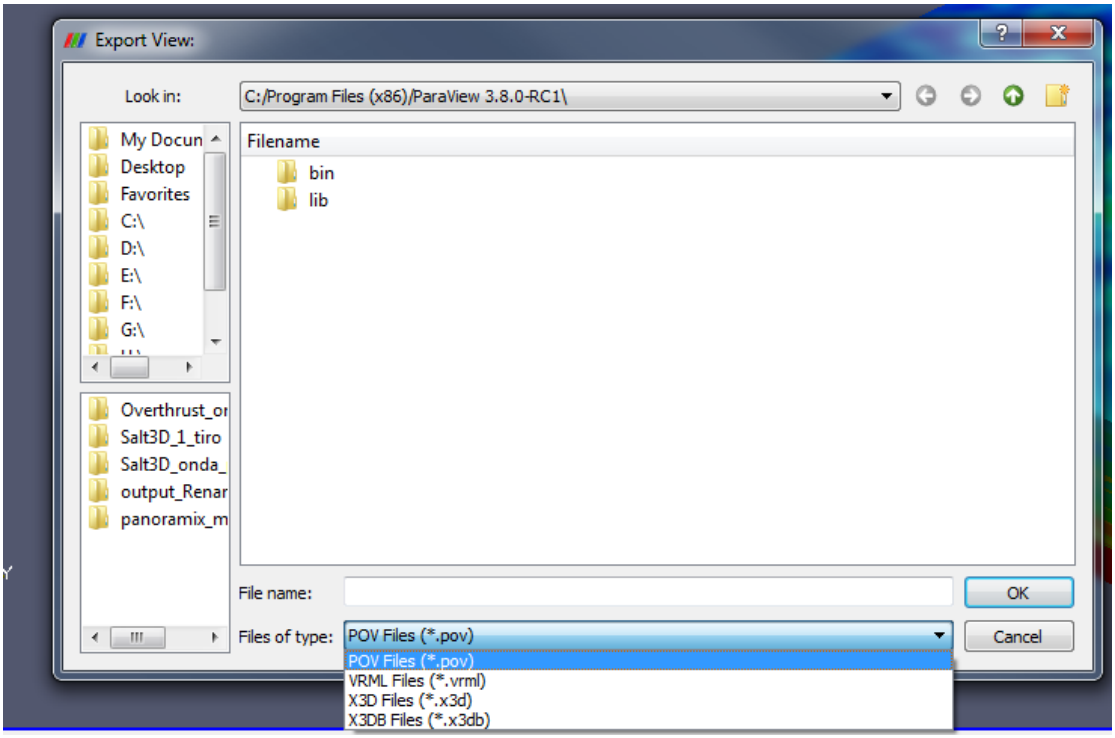

Figura 1 - Tela de exportação do Paraview.

O VRML é um dos formatos que exporta os dados de vertex color dos modelos, porém, só exporta dados não-transientes (que não variam no tempo) , ou em outras palavras, nenhum dado animado é exportado, como variações de temperatura e velocidade.

A exportação em VRML é bastante difundida, principalmente levando-se em conta o software .PDF3D, que permite embutir esses dados em VRML em arquivos .PDF [9].

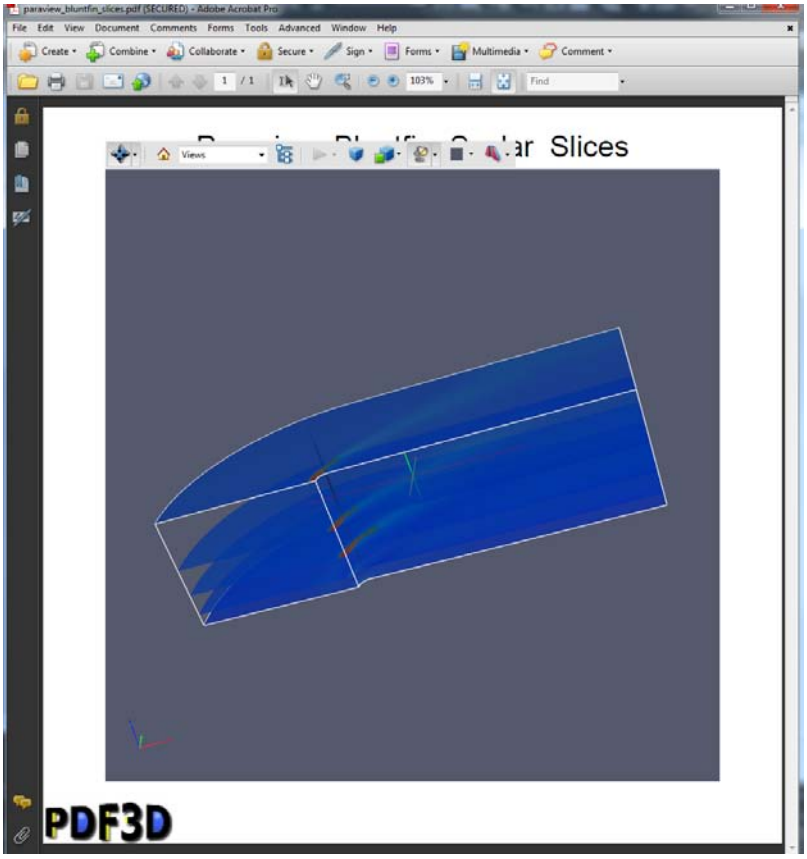

Figura 2 - Modelo 3D exportado do Paraview em funcionamento dentro de um arquivo .PDF.

Entretanto, como os arquivos em VRML não carregam informações transientes, o PDF3D também não as visualiza, permitindo apenas a exibição de malhas estáticas.

O Ensight também sofre com problema semelhante. Ele exporta nos seguintes formatos: CASE, VRML, EXODUS II, FLATFILE, HDF 5.0, STL

Porém, assim como no Paraview, nenhum desses arquivos carrega as informações transientes dos modelos, seja alterações de malha ou de coloração.

Uma solução que a equipe do Ensight desenvolveu, foi o Enlighten: o Enlighten é uma espécie de player do Ensight. Uma vez que você tenha todos os seus dados visualizados e carregados, você pode exportá-los para o Enlighten. Assim, quando você executar o arquivo, ele já estará todo pré-configurado e com seus dados transientes.

Contudo, os arquivos exportados para o Enlighten ficam muito grandes, com aproximadamente 2 GB (dois gigabytes), além de também requisitar a instalação do software Enlighten para a exibição dos arquivos.

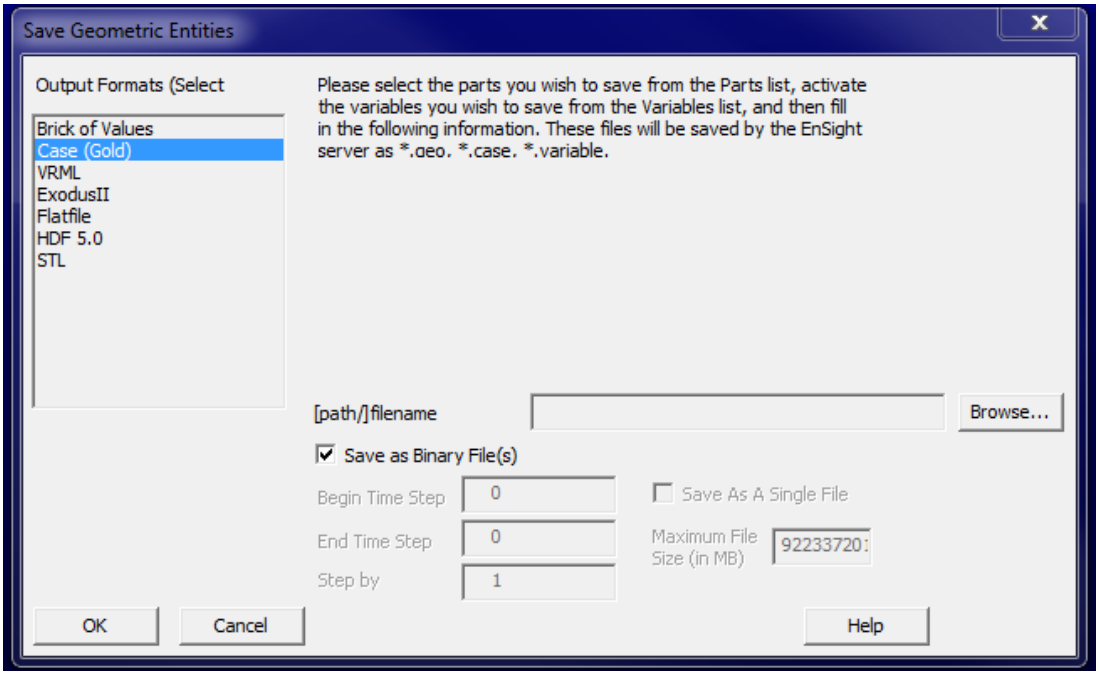

Figura 3 - Tela de exportação do Ensight.

Em síntese, ambos os programas não oferecem opções viáveis de exportação dos dados visualizados, de forma interativa e com dados transientes, que não necessitem da instalação de visualizadores e players.

Foi com base nesta ausência de opções que o escopo deste trabalho foi definido: "criar um visualizador que tenha baixo custo computacional, tamanho de arquivo reduzido, que não requeira a instalação de outros programas para funcionar; que contenha dados transientes em 3D e seja interativo."

2.3 - Visualização Interativa na Internet

Uma solução largamente explorada é a visualização diretamente na internet. Alguns sistemas oferecem interação, outros apenas visualização 3D, mas a maioria conta com o seguinte requisito básico: o servidor faz o cálculo e processamento dos dados e o cliente apenas recebe as imagens e transmite as ações do usuário.

Um exemplo é descrito em ANG *et al*. [10] : eles desenvolveram uma ferramenta de visualização volumétrica para uso geral em diversos sistemas, integrada com o browser Mosaic como sistema de visualização. A ferramenta, chamada "VIS" usa um conjunto de placas gráficas dedicadas para gerar modelos 3D a partir de dados volumétricos, distribuindo as tarefas entre as placas disponíveis. Para permitir a distribuição da informação para os clientes remotos, , eles criaram plugins para o Mosaic suportar dados

volumétricos e definiram um canal de comunicação entre o browser e o VIS. Isso permitiu inserir visualizações tridimensionais interativas dentre de páginas HTML. A partir do browser, clientes remotos mandam informações que executam tarefas de visualização para o ambiente do VIS e o resultado é encaminhado para a página HTML.

Outro trabalho semelhante na internet foi feito por LIU *et al* [11], que implementou um sistema de visualização interativa distribuída chamado "Discover" (Distributed Interactive Scientific Computing and Visualization Environment). Este sistema adotou também a arquitetura cliente-servidor e consiste em uma máquina virtual, que inclui um ou mais computadores e um conjunto de processadores que provêem a máquina virtual. O sistema oferece funções para ambos os clientes e servidores, entretanto o cliente realiza apenas operações básicas de manipulação de dados. Foi um sistema voltado para aplicações médicas, que suportava visualização cooperativa, com usuários posicionados em diferentes clientes, sendo capazes de interagir com uma única janela de visualização.

Um trabalho brasileiro é descrito em ALVES *et al* [12], onde seu produto, chamado *"VisWeb",* usa JAVA applets (software executado no contexto de outro programa, como por exemplo, o plugin Flash Player executando dentro do browser Firefox) para oferecer recursos de visualização na internet. Neste sistema, o servidor provê os applets e o processamento é feito totalmente no cliente, que também guarda os dados, como ilustrado na figura abaixo.

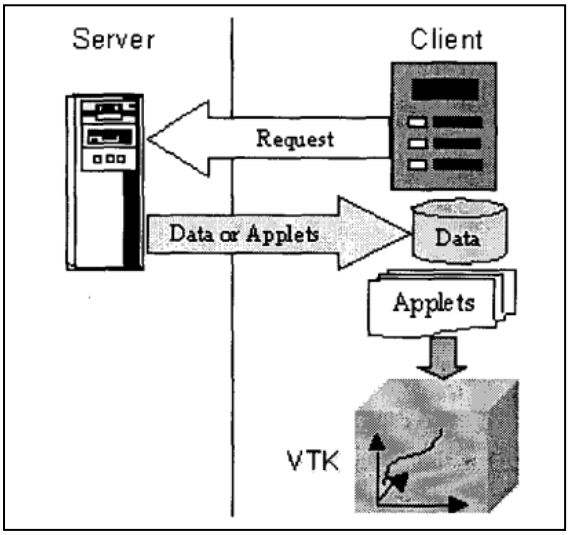

Figura 4 - Arquitetura do VisWeb

Os applets do Java fazem o interfaceamento do VisWeb com o VTK Toolkit, para implementar um conjunto básico de técnicas de visualização escalares e vetoriais.

Como mostrado na figura, o VTK é executado no cliente. Mas para isso, o cliente deve ter instalado em sua máquina algumas bibliotecas de dados do VTK para que os applets do JAVA permitam que o VisWeb execute as visualizações. Neste projeto, todas as funções de visualização são implementadas usando classes do VTK, e todas as atividades de rendering são implementadas em algoritmos controlados pelo VTK e pelo OpenGL. O JAVA apenas provê instruções para controle de eventos na interface do usuário e ativa as classes do VTK, usando o interfaceamento criado entre eles.

Em síntese, esse sistema requer a instalação de bibliotecas do VTK, além do plugin do JAVA. Além disso, um arquivo de permissão tem que ser baixado para que o servidor possa ter acesso aos dados no disco local do cliente.

Uma outra solução interessante foi proposta por BETHEL *et al* [13] - usando o QuickTime player ou uma página da internet, usuários remotos podem navegar livremente através de imagens pré-renderizadas de visualizações 3D transientes(com variação temporal).

Através desse trabalho, eles tentaram desenvolver um mecanismo que fizesse um equilíbrio entre custo computacional e funcionalidade. Mandar imagens via internet ao invés de dados é um método mais inteligente em termos de usabilidade (pouca ou nenhuma instalação e configuração são necessárias no lado do cliente) e escalabilidade, em termos de acessar remotamente a visualização de resultados obtidos através de grandes volumes de dados.

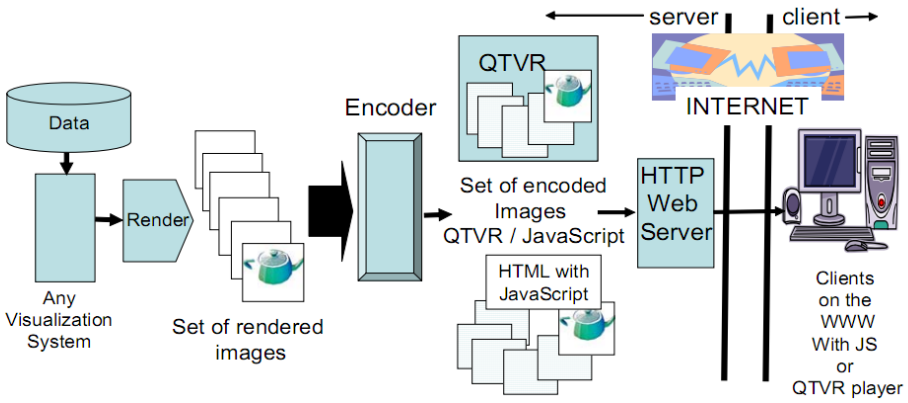

Figura 5 - Fluxograma do método desenvolvido por Chen.

A experiência da interação em 3D com uma cena dinâmica é obtida através da visualização de imagens pré-renderizadas de diferentes ângulos de visão e diferentes espaços temporais. Essas imagens podem ser renderizadas por qualquer visualizador ou sistema de render, pois este método é largamente utilizado em diversos campos (panoramas e passeios virtuais com Quicktime VR). Durante a interação por parte do

cliente, seu visualizador, seja um Quicktime VR player ou no código em JavaScript desenvolvido pela equipe, permite que o usuário possa navegar através do modelo, com variação temporal, representados pela composição de imagens pré-renderizadas.

Entretanto, ambas as soluções desenvolvidas nesse trabalho apresentam alguns entraves. Ao usar o Quicktime VR, todo o arquivo tem que ser baixado da internet, antes que se possa visualizar qualquer imagem. Ele funciona bem para visualizações curtas com poucos dados, contudo, quando as visualizações aumentam, o tamanho do arquivo a ser baixado cresce enormemente, aumentando muito o tempo de download do arquivo. Já na solução em JavaScript, as imagens são carregadas dinamicamente no cache do computador, conforme surge a necessidade. Mas, a longo prazo, quando se muda totalmente de ambiente ou visualização, o frame rate fica muito baixo, pois as imagens tem que ser baixadas inteiramente do servidor. Além disso, um problema encontrado em ambas as soluções é que, devido ao fato das imagens que compõem a "simulação 3D" serem pré-renderizadas, ao dar zoom nas visualizações, a resolução da imagem em destaque continua a mesma, resultando em degradação da qualidade visual. Uma solução seria usar imagens de alta resolução para compor os gráficos 3D, mas isso aumentaria demais a tamanho final do Quicktime VR Object Movie. Outra limitação é que este sistema não oferece a possibilidade de navegação livre. Se as imagens pré-renderizadas são feitas com determinada angulação de câmera, os usuários não poderão mudá-lo de forma a "caminhar" pelo gráfico tridimensional.

Em 2008, CHEN, *et al* [14] atualizou o sistema, desta vez com imagens pré-renderizadas em diferentes resoluções, que são transmitidas ao cliente conforme sua necessidade. Além disso, é possível navegar através de diferentes ângulos de visão, diferentes parâmetros de visualização e rendering. Nesta atualização, foi desenvolvido um sistema de streaming de imagens multi-resolução, em blocos (da mesma forma que funciona o Google Earth), que utiliza eficientemente a capacidade da rede disponível, enquanto provê ao usuário a resolução máxima que ele pode perceber de um determinado ponto de vista. Entretanto, apesar das atualizações, a navegação entre diferentes ângulos ainda é comandada através de ângulos pré-renderizados, o que impede a "navegação livre" pela visualização tridimensional.

Em BURIOL [15], é descrito um sistema de para visualização de campos elétricos e magnéticos em subestações de energia, onde as simulações foram geradas com VTK, a subestação de energia foi modelada em CAD (Computer Aid Design) e ambos os dados foram congregados em um ambiente tridimensional interativo, através do uso do VRML (que pode ser visualizado online).

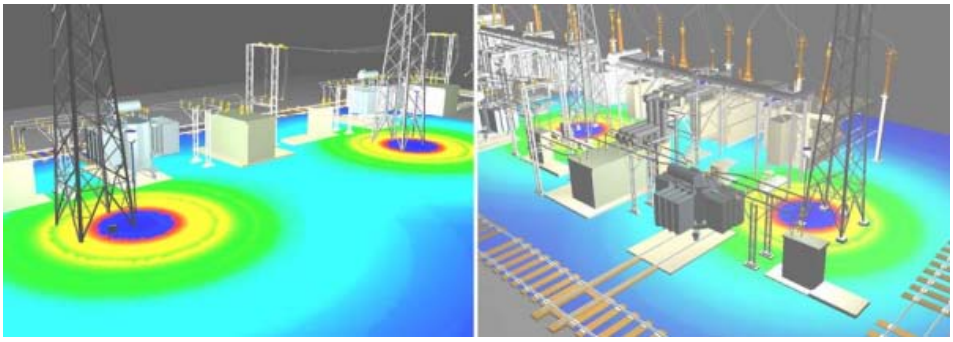

Figura 6 - Visualização de campos de luminância em conjunto com o modelo da subestação, dentro do ambiente VRML

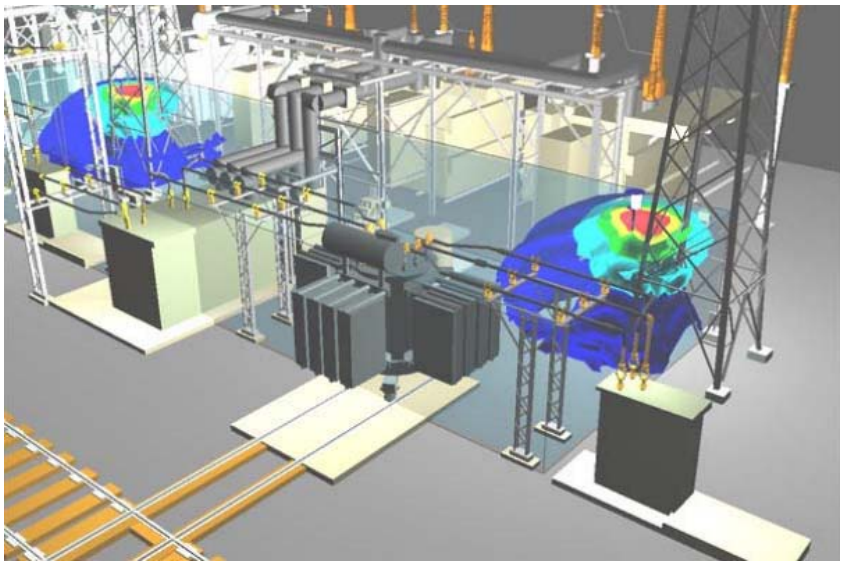

Figura 7 - Visualização de isosuperfícies e do modelo CAD em VRML

No sistema de Buriol, a navegação é livre, porém apresenta o mesmo problema de outras visualizações realizadas em VRML: os dados de visualização científica são estáticos, não transientes - sem variação temporal.

CAPÍTULO 3

TESTES PRELIMINARES

3.1 – O Google Earth como repositório de dados científicos

A empresa de busca eletrônica na Web, Google, oferece ao mercado o Google Earth (GE) [16], um sistema para navegação em qualquer região do planeta. Todas as regiões da Terra estão cobertas por mosaicos de imagens de satélites de diferentes resoluções (Landsat, Ikonos e Quickbird). A maioria dos mosaicos foi construída com imagens Landsat, resolução espacial de 30m. Alguns lugares contam com recobrimento de imagens com alta resolução espacial, de 1m ou 0,67m. O GE usa também imagens topográficas, geradas pela missão Shuttle Radar Topography Mission (SRTM). As aplicações do GE servem para fins comerciais ou científicos. O sistema funciona nos sistemas operacionais mais comuns, Windows, Mac OS e Linux.

O GE está fundamentado numa variante da Geography Markup Language (GML), a Keyhole Markup Language (KML). A codificação das imagens e vetores estão descritos nessa linguagem de codificação para compartilhar dados espaciais na Web.

A GML é uma iniciativa do consórcio OpenGIS e tem por base a OpenGIS Abstract Specification - OAS (www.opengis.org). As iniciativas OpenGIS e GML oferecem um padrão, livre do vendedor, para codificar qualquer tipo de primitiva geográfica bem como qualquer método para mostrá-lo (Miranda, 2002) [17]. O objetivo da GML é o *transporte* e *armazenamento* da informação geográfica, incluídos a geometria e propriedades do atributo geográfico (OGC, 2002). A GML e, conseqüentemente, a KML, não é uma linguagem para o *desenho* das primitivas geográficas que ela codifica. Os procedimentos de visualização (desenhos) dessas primitivas ficam a cargo de rotinas que devem ser desenvolvidas por quem implementa a GML. No caso da KML, a empresa Keyhole Inc. Images (www.keyhole.com) - empresa especializada em mapeamento digital - foi comprada em 2004, pelo Google.

Uma vez que os dados cartográficos estejam disponibilizados no formato da KML, eles podem ser disseminados pela Web, pois a versão cliente do GE passa a funcionar como um servidor de mapas na Web.

Seguindo esta linha, diversos pesquisadores e universidades ao redor do mundo vem utilizando o Google Earth para disseminar seus resultados de modelos científicos e modelos tridimensionais de edifícios. A própria adição do passeio pelo fundo do oceano na versão 5 do GE foi construída através de dados de batimetria fornecidos por pesquisadores.

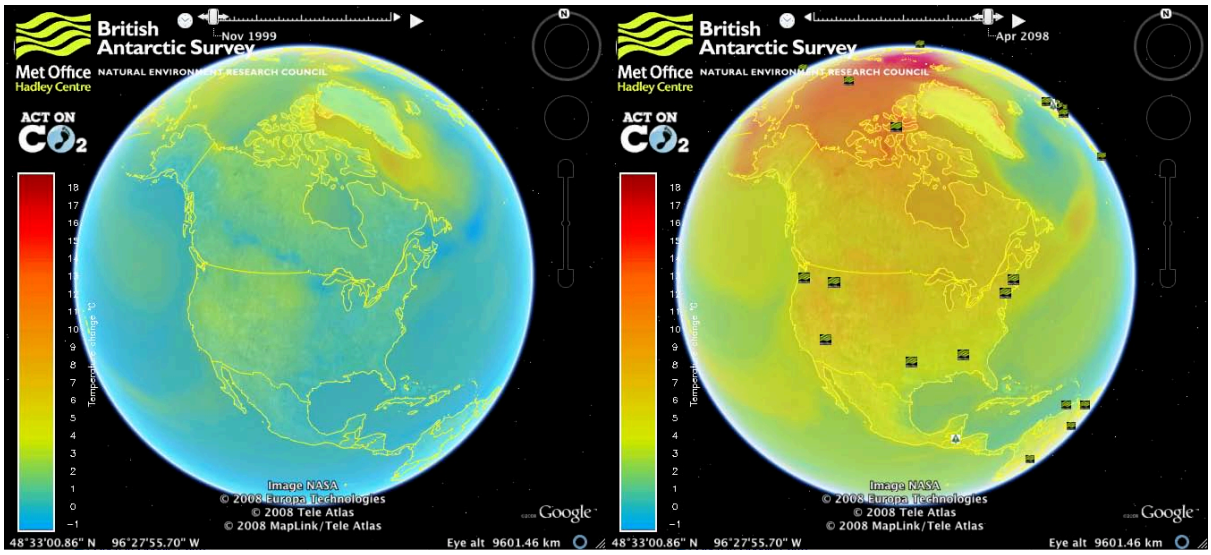

Figura 8 - Prognóstico de Variação Climática na Terra a partir de Novembro de 1999 até Outubro de 2099, publicado no Google Earth por pesquisadores britânicos em 2008.

A princípio, decidiu-se estudar um meio de se utlizar o Google Earth como repositório de resultados de modelos ambientais e dados observados utilizados pelo NUMA/LAMCE. As imagens eram georeferenciadas pelos membros da equipe e exportadas no formato .KML, para ser inserido no programa. Contudo, havia uma dificuldade em se inserir dados transientes (variantes no tempo) e em três dimensões. Mesmo com o estudo de importadores e exportadores de dados em .KML para o software 3D Studio Max, o resultado não foi satisfatório. O dado em 3D causava instabilidade no Google Earth (devido a alta resolução espacial e temporal dos dados) e houve extrema dificuldade em georeferenciar os dados (para estes testes foi utilizado o campo de batimetria fluvial da região amazônica). Outro problema encontrado foi que os dados processados pelo NUMA, como as batimetrias do Rio Solimões (AM) e da região da Bacia de Santos, no litoral do Rio de Janeiro, ficavam invisíveis quando inseridos no Google Earth, uma vez que o software não dispõe dos dados de profundidade nessas regiões. Para que eles ficassem visíveis na interface do programa, seria necessário enviá-los para a equipe do Google e esperar que estes avaliassem a possibilidade de inseri-los na malha geométrica original do "planeta Terra" desenvolvido por eles. Tal fato demandaria um tempo não mensurável, o que vai contra a idealização da interface, que é a de oferecer resultados ágeis para situações de emergência, além de retirar a propriedade intelectual do LAMCE sobre o material produzido.

Também se apresentou a necessidade de se criar um sistema próprio de visualização, sem necessidade de plataformas pré-existentes, que suportasse os dados acima descritos (variações no tempo e geometrias tridimensionais). Sendo assim, a idéia de construir a plataforma de visualização científica com base no Google Earth foi abandonada

e partiu-se para o estudo de softwares que pudessem trazer a tona uma plataforma inteiramente nova produzida pelo NUMA.

3.2 – O Papervision3D

Seguindo a linha de publicação de conteúdo na internet, foi estudado o Papervision3D.

O Papervision3D [18] é uma biblioteca 3D para o software Adobe Flash, que permite a criação e visualização de conteúdo tridimensional diretamente na internet, através do Flash Player Plugin. Ele utiliza a linguagem Action Script 3 para montagem e programação, possui um bom sistema de física e permite até aplicações em Realidade Aumentada.

Uma das grandes vantagens de se utilizar o Papervision3D é a facilidade de pulverização de seus resultados, pois ele utiliza o Flash Player, o plugin mais difundido na internet, seja para visualização de vídeos no Youtube, animações ou mesmo jogos. Como a maioria dos sites de internet requerem o flash player para exibição de seu conteúdo (barra de notícias, por exemplo), ele já é presente na maioria dos lares ligados na rede mundial de computadores. Logo, com conteúdo desenvolvido em Papervision3D, o usuário provavelmente não teria que baixar qualquer outro plugin, tendo acesso direto ao conteúdo científico tridimensional.

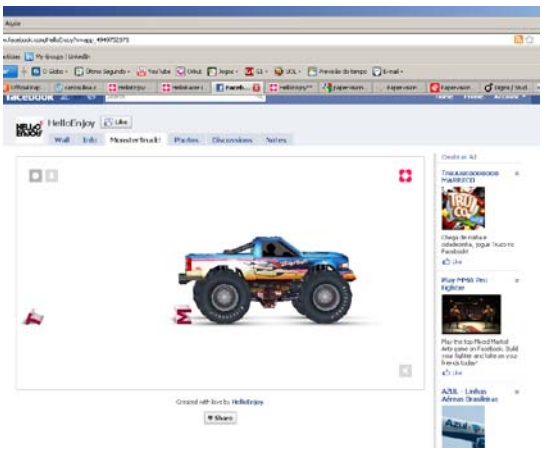

Figura 9 - Aplicativo 3D, executado no browser através do Papervision3D.

Todavia, vários testes foram realizados e o Papervision3D também não trouxe bons resultados: a quantidade de polígonos que ele permite ainda é extremamente baixa, assim como a resolução das texturas. Além disso, todo o interfaceamento para o programa tem que ser programado em Action Script 3, linguagem complexa e desconhecida pelo

pesquisador. Outro problema encontrado foi que o plugin exige grande quantidade de processamento em CPU (processador) e baixo FPS (Frames por Segundo), tornando sua visualização lenta e não-funcional.

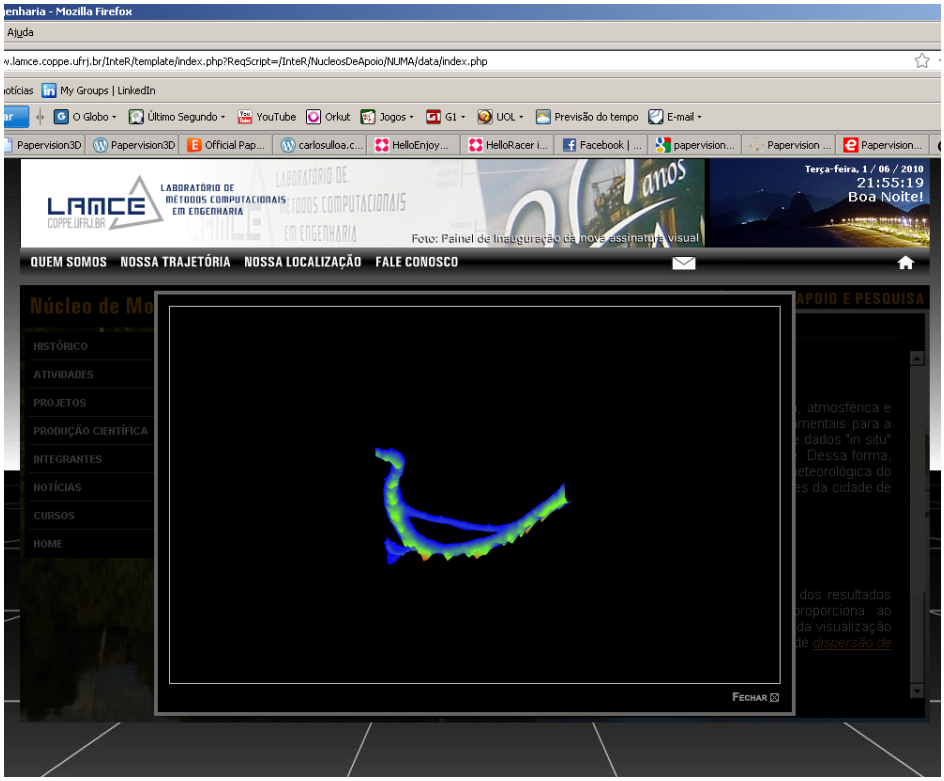

Figura 10 - Aplicativo desenvolvido com Papervision3D, visualização científica tridimensional de trecho da calha do Rio Solimões, disponível no site do NUMA/LAMCE para fins de teste.

3.3 – Unity3D

O Unity3D [19] é uma ferramenta de desenvolvimento multi-plataforma, sendo muito utilizada para criação de games para internet, Nintendo Wii, iPhone e, mais recentemente, iPad. O Unity tem se popularizado devido ao sucesso de aparelhos como o iPhone, que permite que o usuário compre jogos a um baixo custo na AppStore, loja virtual da empresa Apple, e execute-os no próprio celular em qualquer lugar do mundo. Essa novidade alavancou muito a criação de aplicativos e jogos para dispositivos móveis, inclusive no Brasil é o nicho onde mais se investe tratando-se de jogos: jogos para dispositivos móveis

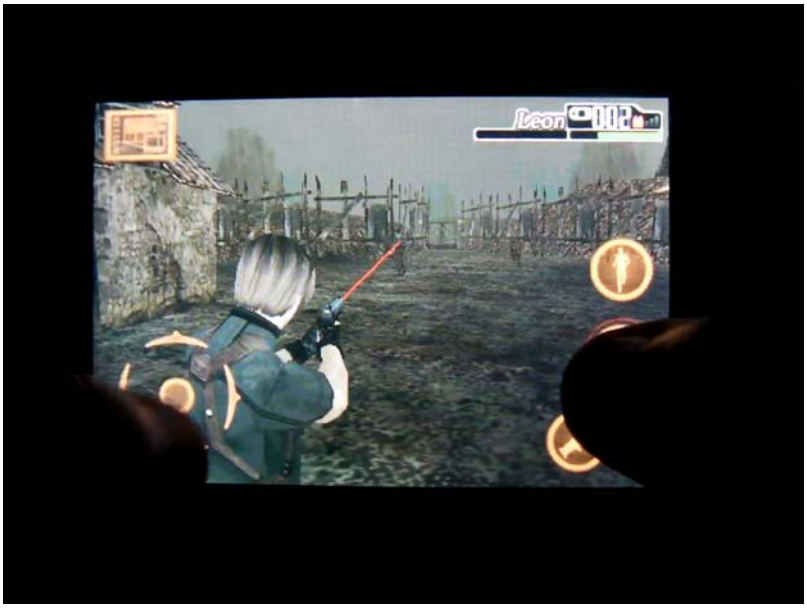

Figura 11a – Jogo Resident Evil 4, da empresa CAPCOM, sendo executado no iPhone.

Figura 11b – Jogo Street Fighter 4, da empresa CAPCOM, sendo executado no iPhone.

O Unity3D despertou o interesse do pesquisador na medida em que os modelos científicos tridimensionais poderiam ser visualizados e manipulados na tela de um celular, o que facilitaria muito a tomada de decisão *in situ* (no local) de um provável acidente, porém, o celular iPhone, da Apple, ainda é um artigo de luxo para a maior parte da população

brasileira Mesmo os integrantes das classes média e alta ainda relutam em adquirir tal aparelho. Outro fato desanimador foi que o Unity3D requer programação em Javascript, outra linguagem de programação desconhecida pelo pesquisador. Uma vez que o tempo de desenvolvimento do visualizador era curto (dois anos), foi necessário encontrar uma engine de desenvolvimento na qual o pesquisador pudesse trabalhar livremente, sem entraves de programação. O Unity3D permanece como uma boa opção para trabalhos futuros que permitam maior tempo de desenvolvimento e equipes multidisciplinares (design e programação).

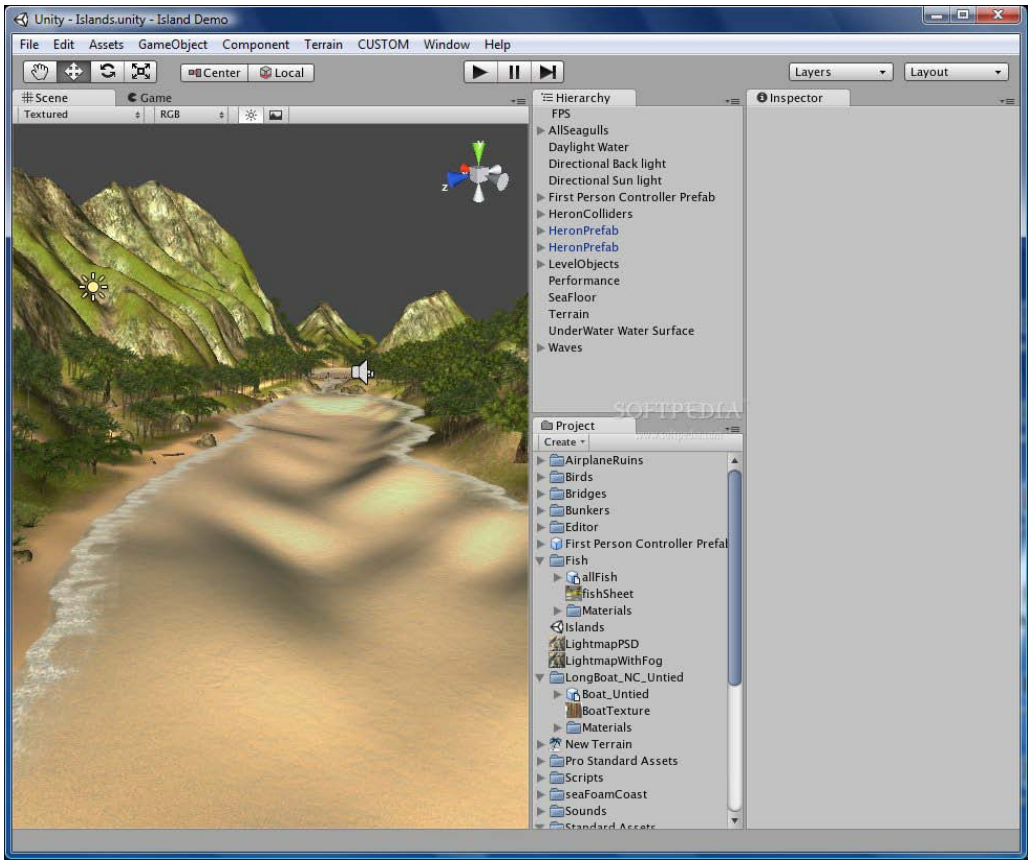

Figura 12 – Tela do software Unity3D, com gráfico 3D, á esquerda e programação à direita.

3.4 – Quest3D

O software Quest3D [20] é uma ferramenta de desenvolvimento de aplicativos tridimensionais que se destaca pelo fato de não cobrar nenhum conhecimento prévio em programação, pois conta com um sistema de Programação Visual, também chamado de "Building Blocks". Além disso ele também permite a “Object Oriented Development” ou Desenvolvimento Orientado ao Objeto. Qualquer profissional seja da área de Computação Gráfica ou não pode utilizar o software e obter bons resultados.

Ao adquirir uma licença do Quest3D, o usuário ganha uma licença comercial para produzir e comercializar seus projetos. O Quest não cobra royalties sobre os projetos desenvolvidos e o usuário pode publicá-los em arquivos executáveis (.EXE) e em páginas da internet (é necessário a instalação do Quest3D Web Viewer). Além disso, o Quest suporta diversos tipos de dispositivos, incluindo mouse, teclado, telas touch-screen, joysticks, Wiimote, Luvas 3D, Motion Trackers, Monitores 3D como o Philips WOWvx e Render em Stereo para uso em CAVES (Cavernas de Projeção Digital).

Outra vantagem animadora do Quest3D é a alta qualidade dos gráficos que o programa pode alcançar. Ele conta com vários recursos avançados como suporte a shaders (técnica que dá qualidade visual aos modelos tais como Normal Mapping, Relief Mapping), High Dynamic Range rendering (HDR - técnica que simula a adaptação do olho humano a ambientes com pouca ou muita luz), Nature Painting (técnica para desenvolver grandes cenários em tempo real), Cloth Rendering (simulação de tecidos), Crowd Rendering (simulação de multidões), Path Finding (definição de caminhos a serem percorridos por câmeras, avatares ou objetos), Water Rendering (o Quest3d já vem com um sistema embutido para simulação de água, tanto em pequena quanto em larga escala, com controle de ondas, etc), Particle Simulation (Simulação de Partículas, como fumaça, fogo, etc), Animation Blending (técnica muito usada em personagens - este sistema faz uma mistura de diferentes animações criadas para um personagem, como andar e correr, permitindo uma transição mais suave em tempo real), GUI rendering (técnica para criação de interface gráfica como botões, barras, etc), Stereo Rendering (técnica para visualização 3d com óculos polarizados ou Anaglifo, que são os óculos vermelhos e azuis, podendo ser utilizados em qualquer monitor ou TV), Weather System (esse sistema já vem pronto no Quest3d, ele simula a passagem do dia em tempo real, inclusive com chuva); Três sistemas de Física : ODE, Newton e o Physics, da NVIDIA. O Quest3d oferece suporte à rede, permitindo aplicações e Games Multiplayer em redes locais e online. Também oferece acesso a banco de dados MySql, ODBC e suporta diferentes arquivos de áudio e vídeo.

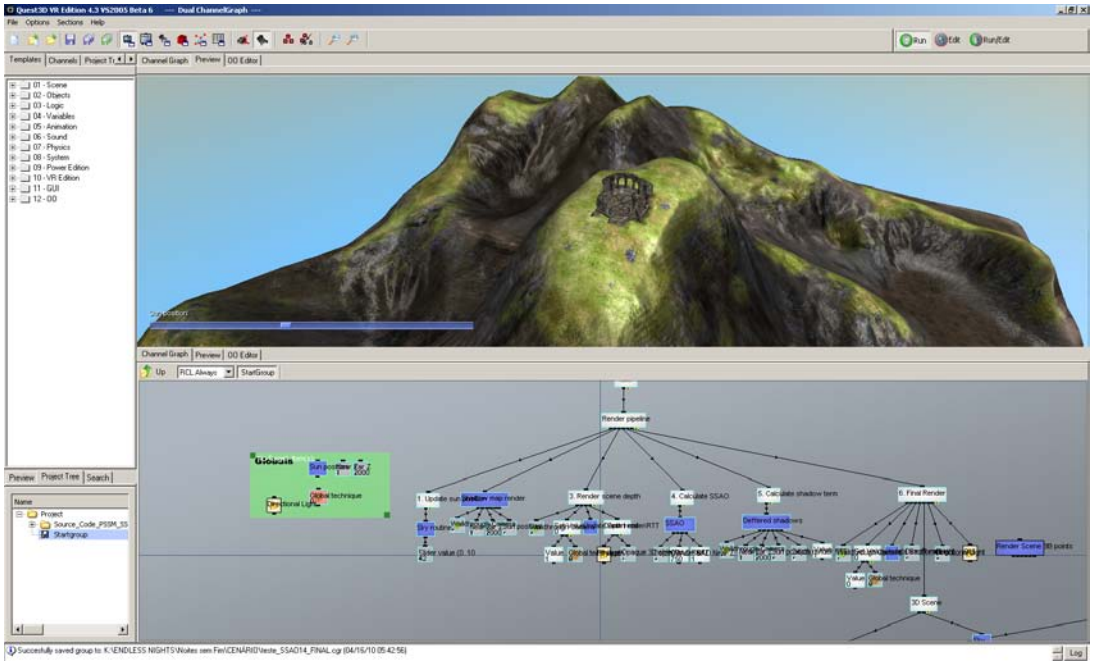

Figura 13 – Tela do software Quest3D, gráfico tridimensional na tela superior e programação na tela inferior.

Em resumo, devido ao fato de não requerer conhecimento prévio de programação, permitir gráficos tridimensionais de alta qualidade, oferecer diversos dispositivos de entrada e saída de dados e publicação online, o Quet3D foi o software escolhido para realização desta dissertação de mestrado.

CAPÍTULO 4
FERRAMENTAS E DISPOSITIVOS

4.1 – Introdução

Neste capítulo são apresentados todos os softwares, formatos de arquivo e hardwares usados na realização deste trabalho.

4.2 – Modelagem Computacional Ambiental

Modelagem Computacional é um método de investigação de conhecimento multidisciplinar que trata da aplicação de modelos matemáticos à análise, compreensão e estudo da fenomenologia de problemas complexos em áreas abrangentes como as engenharias, ciências exatas, biológicas, humanas, economia e ciências ambientais [21].

Alguns exemplos são os modelos de implantação de indústrias e simulação de impacto ambiental determinada pela implantação de sistemas de produção. Simulação, análise, modelagem e projeção temporal e espacial do fluxo das substâncias ou materiais envolvidos nas emissões industriais e no transporte destas no ambiente, das taxas de acumulação nas áreas de influência e projeção dos efeitos sobre as populações afetadas. Simulação de variação de correntes marítimas transporte de poluentes via atmosfera ou meio hidrodinâmico.

O NUMA/LAMCE implementa e aplica modelos que representam a dinâmica marinha, fluvial e atmosférica. Tais modelos são aplicados em diferentes escalas espaciais (global, regional e local) e temporais. Além disso, o grupo também desenvolve e aplica modelos de dispersão de poluentes em ambientes aquáticos. É importante ressaltar que os modelos citados geram resultados com características comuns como: a tridimensionalidade espacial e a variação no tempo.

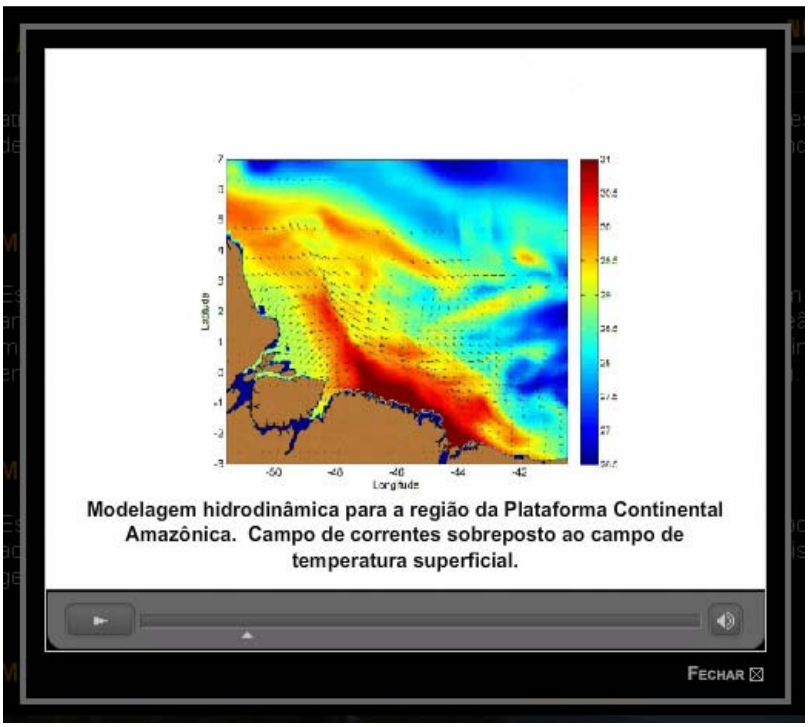

Figura 14 – Modelagem hidrodinâmica disponibilizada no site do NUMA/LAMCE

4.3 – Linguagem Fortran

A família de linguagens de programação conhecida globalmente como **Fortran** foi desenvolvida a partir da década de 1950 e continua a ser usada hoje em dia. O nome é uma abreviação da expressão "IBM Mathematical **FOR**mula **TRAN**slation System".

As versões iniciais da linguagem eram conhecidas como FORTRAN, mas o uso de letras maiúsculas foi ignorado em versões recentes da linguagem começando a partir do Fortran 90. Os padrões oficiais da linguagem referem-se a ela atualmente como "Fortran".

A linguagem Fortran é principalmente usada em Ciência da Computação e Análise Numérica. Apesar de ter sido inicialmente uma linguagem de programação procedural, versões recentes de Fortran possuem características que permitem suportar programação orientada por objetos [22].

O Fortran permite a criação de programas que primam pela velocidade de execução. Daí reside seu uso em aplicações científicas computacionalmente intensivas como meteorologia, oceanografia, física, astronomia, geofísica, engenharia, economia etc.

No âmbito deste trabalho, uma pequena rotina foi desenvolvida em Fortran, pelo pesquisador Manlio Mano, integrante do NUMA, para agilizar a formatação dos dados em .VTK (ver apêndice)

.

4.4 – VTK – Visualization Toolkit

O Visualization Toolkit (VTK) é uma biblioteca de desenvolvimento *open-source*[1], usado em computação gráfica em 3D, processamento de imagens e visualização. O VTK consiste em um conjunto de classes C++ e várias camadas de interface interpretadas icluindo Tcl/Tk, Java e Python. A Kitware, cuja equipe criou e continua a desenvolver o VTK oferece suporte profissional e consultoria para aqueles interessados em desenvolver aplicações com o programa. O VTK suporta uma grande variedade de modificadores incluindo: escalares, vetores, tensores, texturas, e métodos volumétricos; e também técnicas de modelagem avançada como: modelagem implícita, redução de polígonos, suavização de malha, cortes, contornos, e triangulação de malha. O VTK tem um pacote extenso para visualização de informação, possui um conjunto de interações em 3D, suporta processamento em paralelo e faz integração com vários bancos de dados em pacotes GUI (pacotes de interface gráfica) como o Qt e Tk. O VTK é um software multi-plataforma, funciona em Linux, Windows e Unix [23].

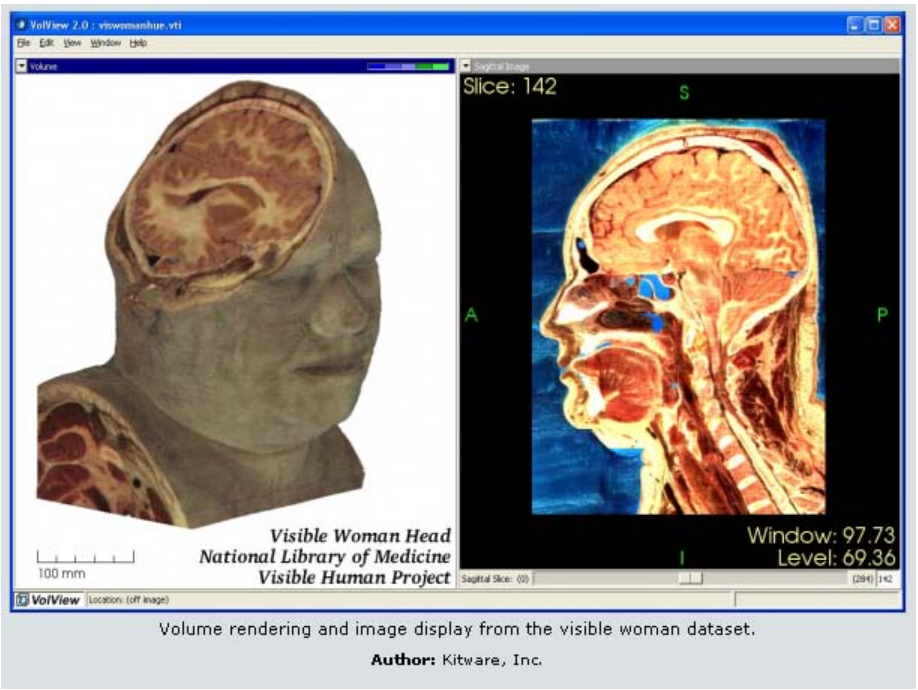

Figura 15 – Render volumétrico e imagem de corte de um corpo feminino usando .VTK como linguagem de visualização tridimensional.

[1] -Software Livre.

4.5 – Kitware Paraview

O software Paraview, da empresa Kitware é um programa open-source, multi-plataforma para análise e visualização de dados. Usuários do Paraview podem rapidamente criar visualizações para analisar seus dados utilizando técnicas qualitativas e quantitativas. A exploração dos dados pode ser interativa em 3D ou programada usando a capacidade de processamento em seqüência do programa.

O Paraview foi criado para analisar dados extensos usando recursos de memória distribuída. Ele pode ser executado em supercomputadores para analisar dados em escala de terabytes, assim como pode ser usado em laptops para dados menores.

O Paraview funciona em sistemas com processamento em paralelo para diversos processadores e em sistemas com apenas um processador. Ele funciona plenamente em sistemas Windows, Mac OS X, Linux, IBM Blue Gene, Cray Xt3, em vários computadores que utilizam Unix, clusters e supercomputadores. No seu interior, o Paraview usa o VTK (Visualization Toolkit) como processador de dados e motor gráfico e usa uma interface gráfica escrita em Qt® [24].

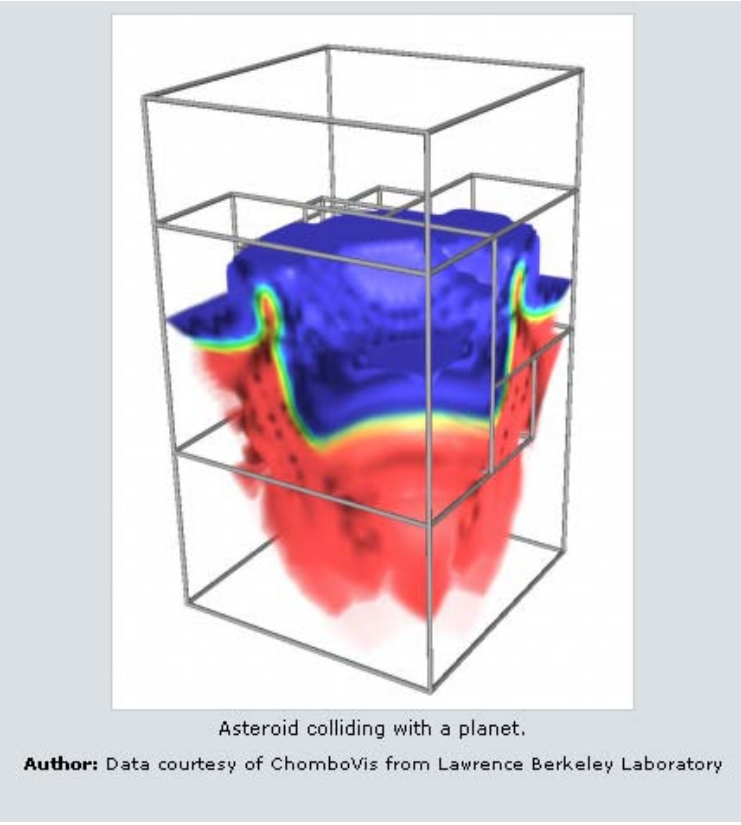

Figura 16 – Visualização científica tridimensional de um asteróide se chocando com um planeta, visualizada no Paraview.

4.6 – VRML

VRML (**V**irtual **R**eality **M**odeling **L**anguage) é um padrão de formato de arquivo para realidade virtual, utilizado tanto para a internet como para ambientes desktop. Por meio desta linguagem, escrita em modo texto, é possível criar objetos (malhas poligonais) tridimensionais podendo definir cor, transparência, brilho, textura (associando-a a um bitmap). Os objetos podem ser formas básicas, como esferas, cubos, elipsóides, hexaedros, cones, cilindros, ou formas criadas pelo próprio programador, como as extrusões.

Além dos objetos, também é possível acrescentar interatividade a estes por meio de sensores, podendo assim deslocá-los de posição, acrescentar luz, produzir um som quando o objeto é clicado ou quando o avatar (personagem 3D) simplesmente se aproxima dele, e abrir um arquivo ou página da Web, ou ainda outra página em VRML, quando o objeto é acionado.

Não é necessário um software específico para a criação de arquivos VRML (embora existam), uma vez que os objetos podem ser todos criados em modo texto. Usualmente a extensão de arquivo para este formato é o .WRL.

Suplantado pela norma X3D, que integra a linguagem VRML com XML e estende as capacidades de modelação e interação da norma VRML97 [25].

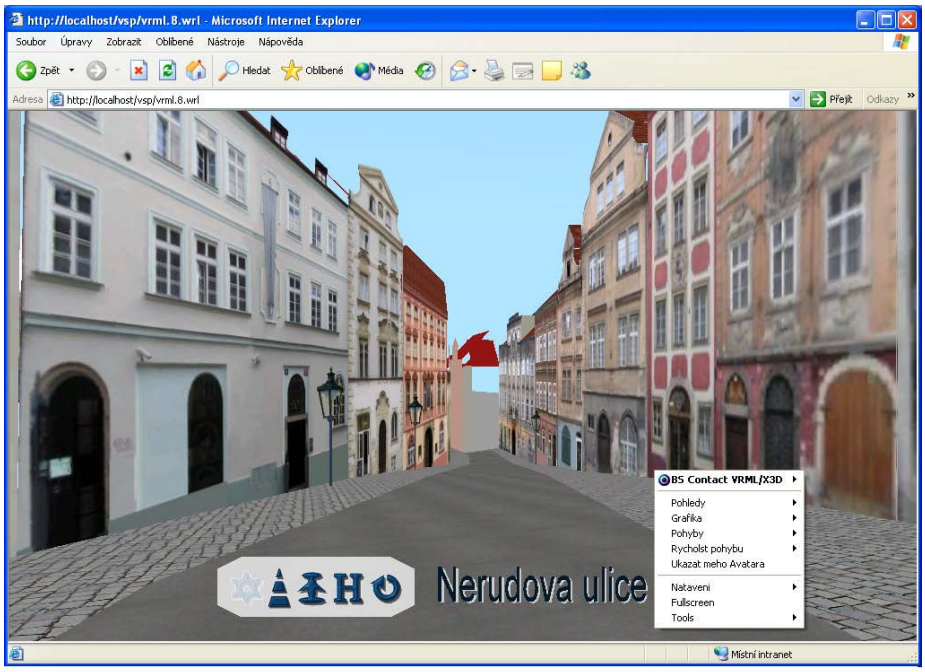

Figura 17 – Arquivo VRML sendo executado no browser Internet Explorer.

4.7 – Deep Exploration

O software Deep Exploration, da empresa Right Hemisphere é uma poderosa ferramenta para visualização e conversão de arquivos em 2D, 3D, áudio e vídeo. O programa suporta diversos formatos populares e particularmente permite a conversão de formatos 3D entre si, incluindo animação. O programa possui uma tela onde os arquivos podem ser visto em 3D, com todas as suas layers de hierarquia e animação. Ele possui duas versões – Standard e CAD edition. A segunda suporta mais formatos, assim como possui uma gama maior de ferramentas para arquivos em 3D. Ele possui ferramentas para otimização de modelos, remoção de objetos escondidos ou invisíveis e muito mais. Além disso, você pode visualizar os arquivos e realizar manipulações sem necessariamente afetar o arquivo original, como mover objetos, apagar layers (camadas), realizar renders (imagens finalizadas), executar animações e etc [26].

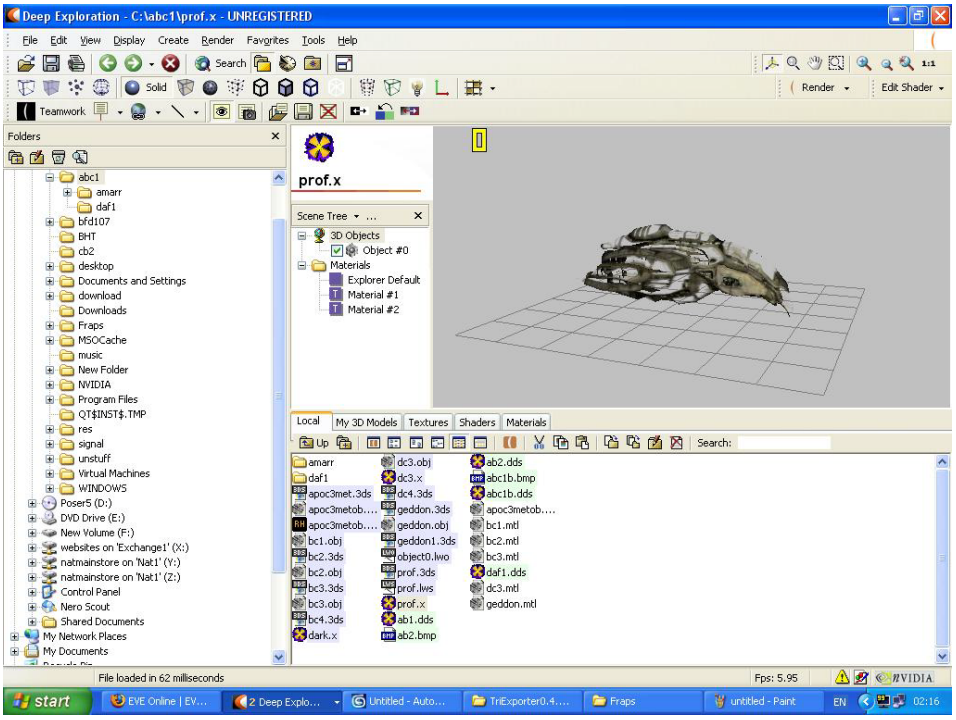

Figura 18 – Tela do software Deep Exploration.

4.8 – 3Ds Max

3Ds Max [27] ou como era anteriormente conhecido *3D Studio Max* é um programa de modelagem tridimensional que permite o rendering (criação de imagens realistas) de imagens estáticas e vídeos. Sendo utilizado em produções de filmes de animação, criação de personagens para jogos em 3D, vinhetas e comerciais para TV, maquetes eletrônicas e na criação de qualquer espaço virtual. Produzido pela Autodesk, o 3Ds Max se destaca dos demais softwares pela versatilidade, diversas aplicações e plugins, que funcionam como extensores do programa. Atualmente é um dos softwares líderes do mercado de Computação Gráfica, ao lado do Maya, também pertencente à Autodesk.

Suas principais características são as seguintes:

Modelagem – O programa oferece ferramentas de modelagem poligonal, modelagem via NURBS, primitivas e diversos plugins que adicionam e expandem as ferramentas básicas de modelagem.

Texturização – Na versão mais recente, a 2011, você pode pintar mapas diretamente na viewport e visualizar o render final através do software Quicksilver, que faz a pré-visualização do render final diretamente na placa gráfica; possui editor de materiais utilizando nós para facilitar a visualização, etc.

Animação – Além do tão conhecido biped, do Character Studio (um esqueleto humano pré-pronto que facilita a construção de animações) as novas versões do 3Ds Max trazem o CAT – Character Animation Toolkit embutido, uma poderosa ferramenta de animação de personagens com diversos controles e níveis de hierarquia.

Iluminação e Render – O programa possui todas as primitivas de iluminação: spot, omni, skylight, direct lights, assim como simulação física de todas as propriedades das luzes como caustics, ambient occlusion, dentre outros. Para fins de rendering, o 3Ds Max conta com a ferramenta básica de render Scanline e o tão utilizado Mental Ray.

Efeitos – O 3Ds Max possui simuladores nativos para cálculos de explosão, física avançada com o Reactor, fogos de artfício, vento, chuva e pode ser expandido por plugins específicos e mais avançados como o FumeFx, Dreamscape, dentre outros.

Programação via Maxscript – O software também conta com uma linguagem de programação própria, o Maxscript, utilizado para agilizar grandes tarefas e otimizar operações repetitivas.

Em suma, trata-se de um software completo para todo e qualquer tipo de produção gráfica tridimensional.

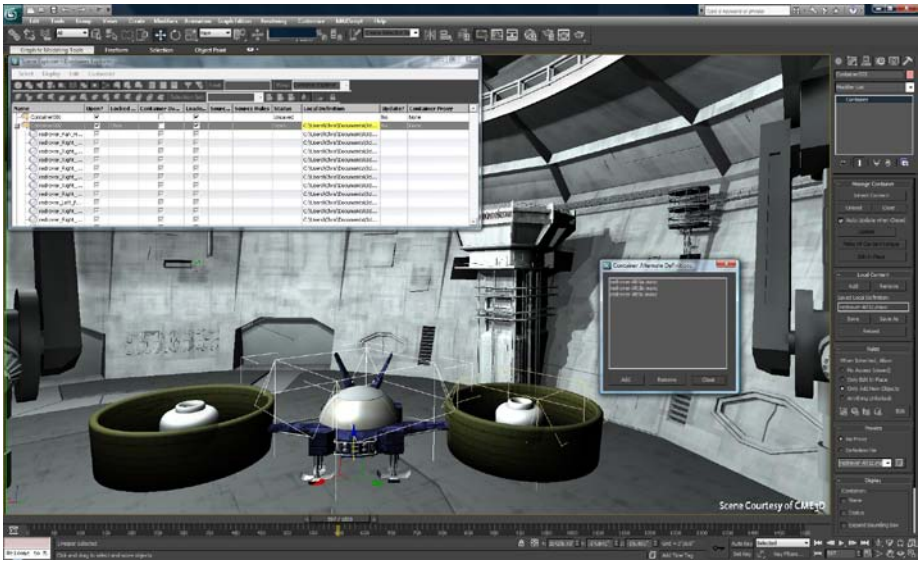

Figura 19 – Tela do software 3Ds Max 2011.

4.9 – ZBrush

ZBrush [28] é um software de escultura e pintura digital que revolucionou a indústria de Computação Gráfica com suas poderosas ferramentas e métodos de trabalho. Ele possui uma interface intuitiva, com um arsenal de ferramentas que foram desenvolvidas com foco na usabilidade. Sua capacidade de escultura digital com bilhões de polígonos permite que os artistas criem livremente, sem restrições, tendo somente a imaginação como freio.

O ZBrush oferece ferramentas que permitem um simples sketch (esboço) feito em 2D possa ser transformado em um conceito completo em 3D, através das excelentes ferramentas de modelagem e escultura orgânica e do seus sistema de rendering com luzes dinâmicas e efeitos atmosféricos. Além disso, graças a facilidade de exportação de formatos, o usuário pode esculpir seu modelo tridimensionalmente e no final do processo enviá-lo direto para impressão 3D e prototipagem.

Devido ao seu excelente processamento em software, os usuários podem esculpir e pintar modelos 3D com milhões de polígonos, sem necessariamente terem que se

preocupar em adquirir placas de vídeo poderosas e caras. Foi por este motivo que o ZBrush conquistou tanto artistas entusiastas, como grande estúdios de cinema e jogos.

Figura 20 – Escultura e pintura digital no ZBrush realizadas pelo artista Rodrigue Palier.

4.10 – Adobe After Effects

O software After Effects [29], da empresa Adobe Systems é um programa de criação de gráficos com movimento e efeitos visuais. Ele é extensamente usado em pós-produção de vídeo, filmes, DVDs e animações.

After Effects usa um sistema de camadas dispostas em uma **linha do tempo** (*timeline*) para criar composição de vídeo e animações como arquivos de vídeo. Propriedades tais como posição e opacidade podem ser controladas independentemente para cada camada, e cada camada pode ter vários efeitos aplicados. Frequentemente chamado de “Photoshop do vídeo”, por sua flexibilidade e por permitir aos compositores alterem o vídeo vendo os ajustes, como o Photoshop faz com imagens.

Embora o After Effects possa criar imagens próprias, ele é geralmente utilizado com materiais compostos a partir de outras fontes para fazer animações gráficas. Por exemplo, com a foto de uma nave espacial e a imagem de estrelas no fundo, o After Effects poderia ser utilizado para colocar a nave em uma layer acima das estrelas e animá-la de modo que simule um vôo.

Figura 21 – Tela do software After Effects.

4.11 – Adobe Photoshop

Adobe Photoshop [30] é um software caracterizado como editor de imagens bidimensionais do tipo raster - em pixel - (possuindo ainda algumas capacidades de edição típicas dos editores vetoriais) desenvolvido pela Adobe Systems. É considerado o líder no mercado dos editores de imagem. O Photoshop é amplamente utilizado pelos profissionais da área de computação gráfica, tanto para trabalhos digitais quanto impressos.

Sua mais recente versão é apelidada como Adobe Photoshop CS5 (sigla cujo significado é Creative Suite 5, correspondente à décima segunda edição desde seu lançamento), disponível para os sistemas operativos Microsoft Windows e Mac OS X. Pode ser rodado também no Linux, através da camada de compatibilidade Wine. Algumas versões anteriores foram lançadas também para IRIX, mas o suporte a esta versão foi descontinuado após a versão 3.0.

Apesar de ter sido concebido para edição de imagens para impressão em papel, o Photoshop está a ser cada vez mais usado também para produzir imagens destinadas à Internet. Até a versão 9.0(CS2) o programa, o Adobe ImageReady, muito semelhante ao Photoshop, que era utilizado em conjunto para a edição e criação de imagens e animações para a internet. A partir da versão 10(CS3), os recursos do Adobe ImageReady estão incluídos dentro do próprio Photoshop.

O Photoshop também suporta edição com outros tipos de programas da Adobe, especializados em determinadas áreas: o já referido Adobe ImageReady (edição de imagens para a web), Adobe InDesign (edição de texto) Adobe Illustrator (edição de gráficos vectoriais), Adobe Premiere (edição de vídeo não-linear), Adobe After Effects (edição de efeitos especiais em vídeo) e o Adobe Encore DVD (edição destinada a DVDs). Os formatos de arquivos nativos do Photoshop (PSD ou PDD) podem ser usados entre estes programas. A título de exemplo, o Photoshop CS5 permite fazer elementos da interface gráfica de DVDs (menus e botões), desde que dispostos separadamente no ficheiro original (PSD ou PDD) por camadas (layers) agrupadas por ordem específica, de forma que, ao ser importado pelo Adobe Encore DVD, este consiga criar a edição para DVD com esses elementos.

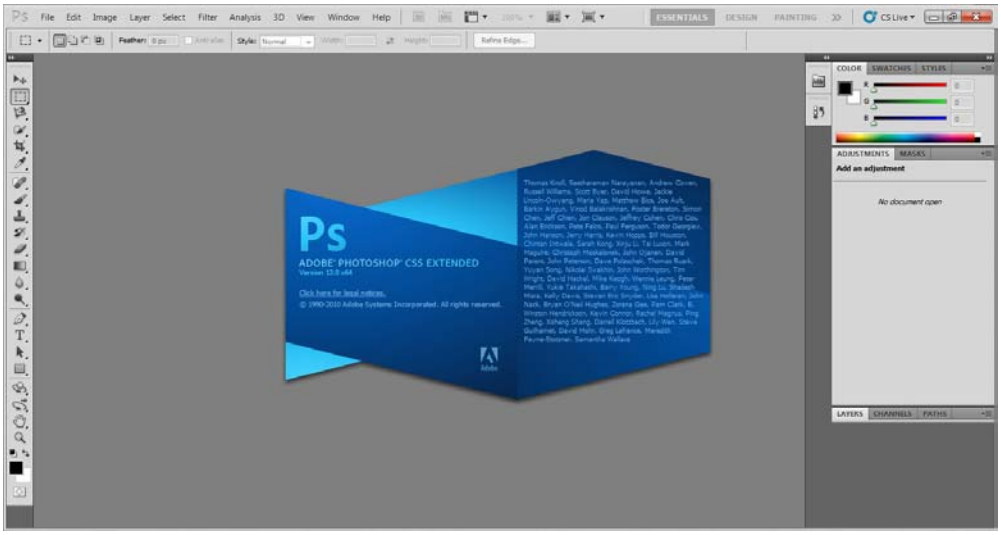

Figura 22 – Tela do software Adobe Photoshop CS5.

4.12 – Formato COLLADA

COLLADA significa **COLLA**borative **D**esign **A**ctivity, ou Atividade Colaborativa de Design. Trata-se de um formato de intercâmbio para arquivos 3D. COLLADA é controlado por um consórcio sem fins lucrativos chamado Khronos Group [31].

COLLADA define um sistema aberto de XML para intercâmbio de arquivos digitais entre vários softwares de produção tridimensional. Arquivos COLLADA são escritos em XML, usualmente identificados pela extensão .dae (**d**igital **a**sset **e**xchange).

Originalmente criado pela Sony Computer Entertainment, ele se tornou propriedade do Khronos Group, um consórcio industrial financiado pelos seus membros, que agora compartilha o copyright do formato com a Sony. Várias empresas dedicadas à Computação

Gráfica colaboraram com a Sony nos primórdios do formato COLLADA, para criar uma ferramenta que abrangeria a maior audiência possível. O COLLADA continua a evoluir através dos esforços dos contribuintes do Khronos Group. Colaboradores nessa fase inicial incluíram a Alias Systems Corporation, Criterion Software, Autodeks, Inc., e a Avid Technology. Dezenas de estúdios de jogos e engines adotaram o formato.

4.13 – Quest3D

Como já dito anteriormente (tópico 2.4), o Quest3D foi o programa de desenvolvimento adotado para a criação deste trabalho. Maiores especificações podem ser encontradas na descrição anterior.

4.14 – Periféricos Utilizados

A seguir serão listados os periféricos utilizados nas versões de teste deste trabalho.

4.14.1 – TV 42'' PANASONIC TOUCH SCREEN

Este hardware foi utilizado no primeiro teste de campo do visualizador que será descrito em detalhes mais a frente neste texto.

Trata-se de uma TV de quarenta e duas polegadas, com um sistema de touch screen embutido frente à tela. Este sistema suporta apenas um toque, chamado *One Touch*, a tela mais comum de touch screen. Ele simula o uso do mouse, através do dedo ou de uma caneta, interpretando um clique, dois cliques e arrastar. Sistemas mais avançados de toque permitem duas ou mais interações na mesma tela ao mesmo tempo, sendo chamados de *Multi Touch*.

Figura 23 a – Foto do modelo TH-42PV80LB da Panasonic, mesmo modelo utilizado nos testes de campo deste trabalho.

Figura 23 b – Foto do painel traseiro do mesmo modelo, mostrando as entradas de áudio e vídeo.

4.14.1.1 – Especificação Técnica

Tabela 1 – Especificações da TV TOUCH SCREEN TH-42PV80LB

			TH-42PV80LB	TH-50PV80LB
Fonte de alimentação			110 - 220 V ~ 50 / 60 Hz	
Consumo de alimentação	Uso médio		340 W	519 W
	Condição em espera		0,7 W	0,5 W
Painel de Plasma	Proporção de formato		16:9	
	Tamanho visível da tela (nº de pixels)		106 cm (diagonal)	127 cm (diagonal)
			922 mm (L) × 518 mm (A)	1.106 mm (L) × 622 mm (A)
			786.432 (1.024 (L) × 768 (A)) [3.072 × 768 pontos]	1.049.088 (1.366 (L) × 768 (A)) [4.098 × 768 pontos]
Som	Alto-falante		2 unidades de 160 mm × 42 mm, 8 Ω	
	Saída de áudio		20 W (10 W + 10 W), 10% THD	
	Fones de ouvido		1 mini-tomada estéreo M3 (3,5 mm)	
Sinais do computador			VGA, SVGA, XGA SXGA ····· (comprimido) Freqüência de varredura horizontal 31 - 69 kHz Freqüência de varredura vertical 59 - 86 Hz	
Nome do sistema de recepção / banda			1. PAL-M 2. PAL-N 3. NTSC	Recepção de transmissões e reprodução do videocassete ou DVD
Canais de recepção (TV comum)			**VHF BAND** 2-13 (NTSC M USA)	**UHF BAND** 14-69 (NTSC M USA) **CATV** 1-125 (USA CATV)
Antena - traseira			VHF / UHF	
Condições de operação			**Temperatura** : 0 °C - 40 °C **Umidade** : 20 % - 80 % RH (sem condensação)	
Terminais de conexões	Entrada AV1	ÁUDIO Esq. – Dir.	2 pinos tipo RCA	0,5 V[rms]
		VÍDEO	1 pino tipo RCA	1,0 V[p-p] (75 Ω)
		Componente	Y P_B/C_B, P_R/C_R	1,0 V[p-p] (incluindo sincronização) ±0,35 V[p-p]
	Entrada AV2	ÁUDIO Esq. – Dir.	2 pinos tipo RCA	0,5 V[rms]
		VÍDEO	1 pino tipo RCA	1,0 V[p-p] (75 Ω)
		Componente	Y P_B/C_B, P_R/C_R	1,0 V[p-p] (incluindo sincronização) ±0,35 V[p-p]
	Entrada AV3	ÁUDIO Esq. – Dir.	2 pinos tipo RCA	0,5 V[rms]
		VÍDEO	1 pino tipo RCA	1,0 V[p-p] (75 Ω)
		S VÍDEO	Mini DIN de 4 pinos	Y:1,0 V[p-p] (75 Ω) C:0,286 V[p-p] (75 Ω)
	Entrada AV4	ÁUDIO Esq. – Dir.	2 pinos tipo RCA	0,5 V[rms]
		VÍDEO	1 pino tipo RCA	1,0 V[p-p] (75 Ω)
		S VÍDEO	Mini DIN de 4 pinos	Y:1,0 V[p-p] (75 Ω) C:0,286 V[p-p] (75 Ω)
	Outros	Entrada HDMI1/2/3	Conectores do tipo A	●Este televisor é compatível com a função "HDAVI Control 3".
		Entrada do computador	Alta densidade Sub-D 15 pinos	R, G, B/0,7 V[p-p] (75 Ω) HD, VD/TTL Nível 2,0 – 5,0 V[p-p] (alta impedância)
		Ranhura do cartão	1 ranhura do cartão SD	
	Saída do monitor	ÁUDIO Esq. – Dir.	2 pinos tipo RCA	0,5 V[rms] (alta impedância)
		VÍDEO	1 pino tipo RCA	1,0 V[p-p] (75 Ω)
Dimensões (L × A × P)			1.020 mm × 727 mm × 327 mm (com o pedestal) 1.020 mm × 679 mm × 95 mm (só o televisor)	1.210 mm × 844 mm × 387 mm (com o pedestal) 1.210 mm × 790,5 mm × 95 mm (só o televisor)
Peso			28,0 kg Líquido (com o pedestal) 26,0 kg Líquido (só o televisor)	38,0 kg Líquido (com o pedestal) 36,0 kg Líquido (só o televisor)

4.14.2 – Controle sem fio para PS2 LEADERSHIP

Este controle ou *joystick* foi utilizado na segunda fase de testes do visualizador. Trata-se de um controle sem fio que pode ser usado tanto no console Playstation 2, quanto no PC através de um adaptador USB.

Figura 24 – Controle sem fio para PS2 e PC.

4.14.2.1– Especificações Técnicas

- Compatível: PlayStation2 e PC (Através de cabo adaptador);
- No PC Interface USB (Plug & Play);
- Alimentação do Joypad: 4 pilhas AAA (Não incluídas);
- Alcance máximo de 5 metros do Receptor para o Joypad;
- 2 LEDs indicadores: Vermelho = Analógico e Verde = Turbo;
- Botões: 13 Botões e 3 Controles de Direção;
- Chave Liga/Desliga Energia e Vibration indepentes.

Dimensões:

- Joypad: L=165 / C=100 / A=65 mm;
- Receptor: L=65 / C=44 / A=12 mm;
- Compr. do Cabo: 28,5 cm.

Requerimentos no PC

- 1 Porta USB disponível;
- Windows® 98SE/Me/2000/XP.

CAPÍTULO 5
CONSTRUÇÃO DO VISUALIZADOR

5.1 – Introdução

Neste capítulo serão descritos todos os processos e técnicas utilizados para construção do visualizador de dados científicos em três dimensões. Todo o processo passou por reformulações em diferentes fases do desenvolvimento do trabalho, porém aqui será apresentada a metodologia final utilizada.

5.2 – Resultados de Modelos Computacionais Ambientais

A primeira fase da metodologia de construção deste visualizador começa a partir dos resultados de Modelos Computacionais Ambientais gerados pela equipe do NUMA/LAMCE.

Os dados foram obtidos no formato ascii (.DAT) a partir da execução do modelo de circulação geral dos oceanos Modular Ocean Model, versão 4.0. Esses dados representam médias mensais de campos espaciais superficiais de temperatura do mar (global) e da componente zonal de velocidade (recorte da costa norte e nordeste do Brasil e região oceânica adjacente). Os dados batimétricos utilizados para a confecção da grade são oriundos do Southampton Oceanography Center. Essa batimetria (global) é o resultado da composição de vários produtos. Entre 72º S e 72º N, a versão 6.2 do produto levantado através de sensoriamento remoto de SMITH & SANDWELL [32] foi mapeado de uma projeção mercator original para uma grade georeferenciada (Latitude X Longitude) com 2 minutos de resolução espacial. Ao norte de 72º N, uma versão da International Bathymetric Chart of the Oceans, descrita por ASSAD, *et al*. [33] foi usada, enquanto ao sul de 72º S o produto batimétrico ETOPO 5 [34] foi o escolhido.

Para os dados de topografia da Região Amazônica presentes no visualizador, foi utilizada a base de dados SRTM - Shuttle Radar Topographic Mission, da Nasa, com resolução espacial de 90 metros. Para a construção da batimetria do Rio Solimões, foram utilizadas as seguintes cartas náuticas da Marinha do Brasil: HS-B1 e HS-B2, para a área do rio entre Coari e Lauro Sodré, com resolução espacial de 50 metros. Outros modelos utilizados neste visualizador foram o Princeton Ocean Model, NICOIL (desenvolvido pelo LAMMA - MEHDI, *et al*) [35], e o ETOPO 2 [36].

Com essa fase concluída, os modelos descrito em .DAT (DOS Basic) são reescritos para o formato .VTK.

5.3 – Conversão para a linguagem VTK (Visualization Tool Kit)

Em um momento inicial, a conversão dos dados em .DAT para .VTK foi feita manualmente. Porém, a partir do momento que o volume de dados aumentou significativamente, foi gerado uma rotina em linguagem FORTRAN para agilizar a conversão dos arquivos (ver apêndice). Apesar de ser um formato antigo, o VTK é de fácil compreensão para usuários não-programadores e permite a conversão dos dados georeferenciados, o que era uma exigência primordial na construção da interface.

Para a visualização dos dados utilizados neste trabalho, foi utilizado a formatação *Legacy* do .VTK. O Legacy é o formato mais simples, que pode ser escrito manualmente e permite a leitura de diversos passos de tempo através de uma série de arquivos numerados sequencialmente. Abaixo segue a formatação do arquivo Legacy do VTK:

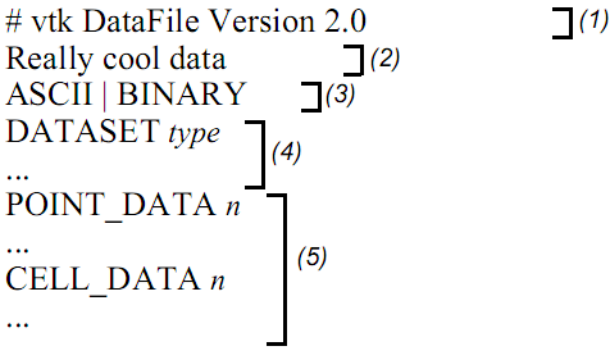

Figura 25 - Representação das cinco partes que compõem o arquivo .VTK

Parte 1 : Header ou Cabeçário

Parte 2 : Título (No máximo 256 caracteres)

Parte 3 : Tipo de dado, seja ascii ou binário

Parte 4: Geometria/topologia. Onde *type* pode ser um dos seguintes:

STRUTCTURED_POINTS

STRUCTURED_GRID

UNSTRUCTURED_GRID

POLYDATA

RECTILINEAR_GRID

FIELD

Parte 5: Atributos dos dados. O número de itens *n* de cada tipo deve ser o mesmo que o número de pontos (points) ou células (cells) do dado. Se type for FIELD, point and cell data devem ser omitidos.

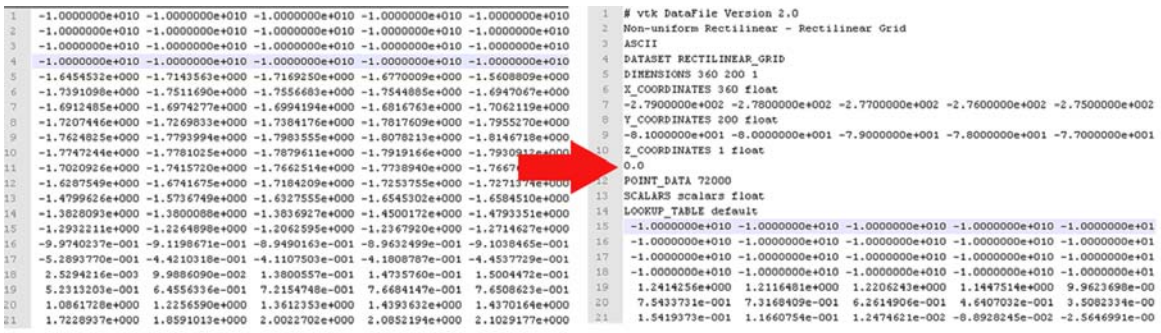

Figura 26 - Conversão dos dados em ascii (.DAT) para .VTK

Para todos os modelos visualizados no aplicativo, foi utilizado o formato RECTILINEAR_GRID, com uma, duas e três dimensões.

5.4 – Entrada no Programa Paraview

Uma vez que o arquivo .VTK é importado dentro do Paraview, os dados são visualizados e tratados através de uma coleção de filtros própria do programa como: Extract Surface, Warp Scalar, Texture Map, etc.

Cada instante da amostra de dados compõe um arquivo .VTK separado. Ao importar no Paraview, usa-se o formato *legacy* do .VTK onde cada instante compõe um arquivo único com os instantes reconhecidos automaticamente como *timesteps* dentro da timeline do programa.

Configurados todos os parâmetros de tabela de cores, filtros, opacidade e legendas, podem ser geradas figuras, animações ou exportação dos dados.

Desde o início do processo de visualização dos dados, a seguinte pipeline foi adotada para a maioria dos dados:

- Importação do arquivo *legacy*;
- Escolha da variação cromática desejada (cores para representar a variação dos dados);
- Aplicação do filtro Extract Surface (que permite a posterior manipulação 3D do dado);

- Aplicação do filtro Warp Scalar (surface elevation based on scalar values) - esse filtro permite que se extraia o volume 3D a partir do dado escalar presente no dado, como por exemplo a batimetria.

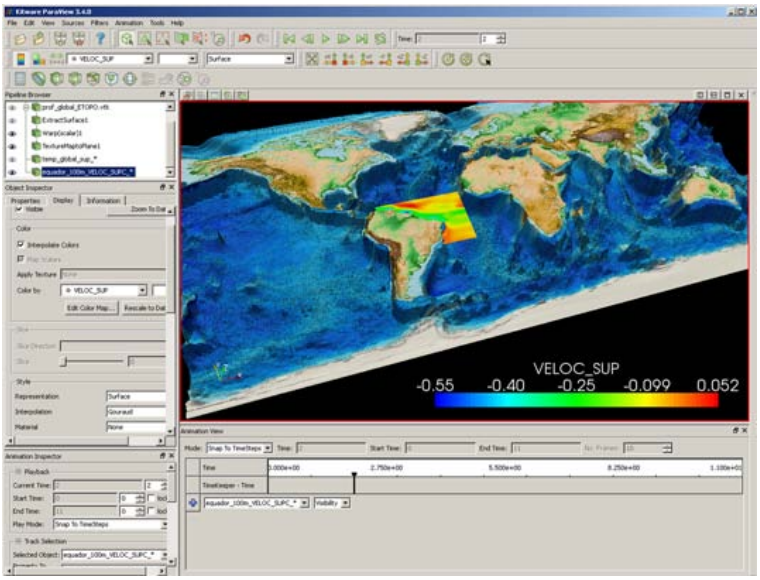

Figura 27 - Tela do Paraview com os dados de batimetria e velocidade superficial visualizados e tratados.

Após todo o tratamento dos dados, o arquivo é exportado no formato .WRL, que contém os dados de geometria, texturas e coordenadas X,Y e Z do dado. Em versões anteriores da metodologia deste trabalho, o formato .PLY (*Poligonal File Format*) era utilizado, porém foi abolido pois não conseguia exportar as texturas dos dados para outras plataformas tridimensionais. Já o .WRL atendeu a todos os requisitos esperados, apesar de se tratar de um formato antigo.

5.5 – Criação de Texturas Animadas

Apesar do Paraview visualizar os dados transientes (que variam no tempo) dinamicamente, uma vez exportados, a animação dos dados não acompanha o formato .WRL.

Esta foi uma grande dificuldade para este projeto. Pensou-se em manipular as animações dos dados via shader (dispositivo de programação para placas de vídeo), mas carecia-se do conhecimento e tempo para tanto.

A solução encontrada foi exportar cada passo de tempo, de cada dado, como uma textura em separado, e animá-las em softwares específicos, como o Adobe After Effects,

gerando um vídeo altamente compactado, para que seja visualizado em real-time, dentro do aplicativo 3D. O formato de saída dos vídeos foi o .AVI, devido as possibilidades de compressão que ele oferece e também por ser o formato padrão de vídeo para sistemas Windows, garantindo a compatibilidade do vídeo.

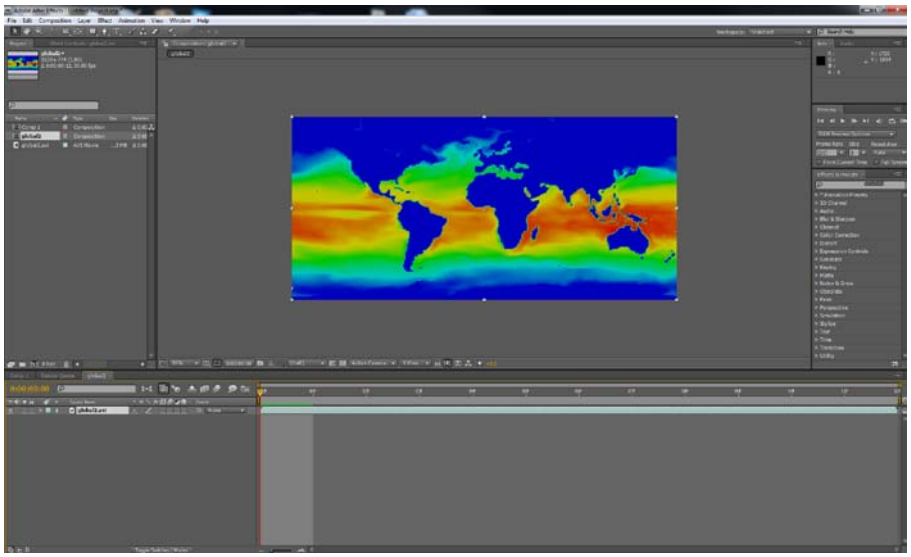

Figura 28 - Textura animada sendo gerada no software After Effects.

5.6 – Entrada no programa Deep Exploration

Ao ter o dado exportado em VRML do software Paraview, teoricamente este deveria ser importado sem problemas para o software 3Ds Max, usado para compilar todos os dados do visualizador.

Porém, por motivos que fogem ao conhecimento do pesquisador, nenhum importador nativo de VRML das versões 9.0 e 2009 do 3Ds Max conseguiram com sucesso importar o VRML extraído do Paraview.

Importar esses dados no 3Ds Max era essencial, pois permitiria que os dados estáticos (não transientes) pudessem ser trabalhados em conjunto com os dados variantes no tempo (transientes).

A solução encontrada foi procurar um software para intercâmbio de dados tridimensionais e devido ao conhecimento prévio do pesquisador, o software Deep Exploration foi utilizado para resolver o problema em questão.

O Deep Exploration conseguiu visualizar os dados em VRML extraídos do Paraview, porém com uma alteração de 90º em um dos eixos do dado. Após uma pequena

correção, o dado foi novamente exportado em VRML, que pôde ser importado com sucesso no software 3Ds Max.

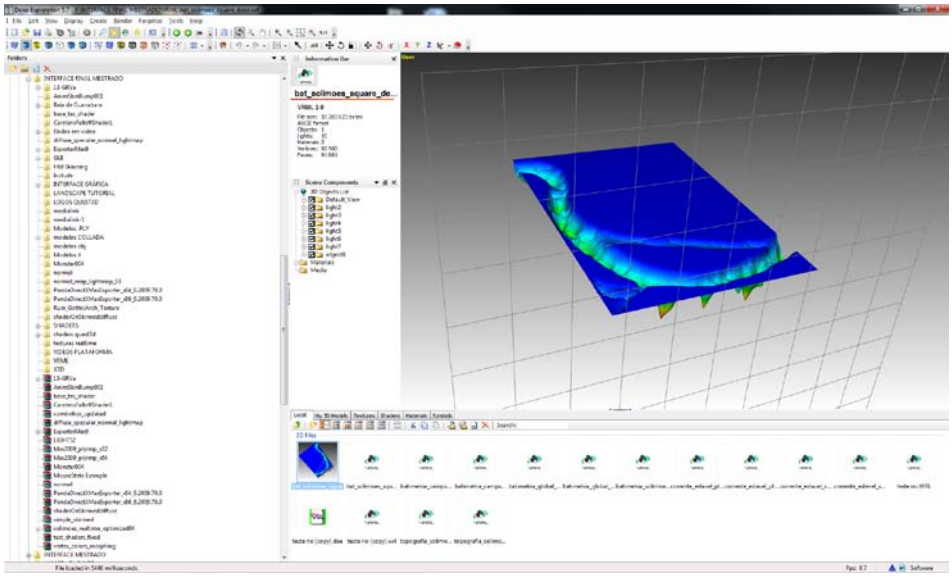

Figura 29 - Batimetria do Rio Solimões extraída do Paraview visualizada no software Deep Exploration

5.7 – Configuração no 3Ds Max

Assim que todos os dados foram importados no 3Ds Max, realizou-se a inserção dos dados transientes através do mapeamento das texturas animadas

Isto foi possível seguindo a metodologia a seguir:

- Os planos 3D referentes aos dados transientes foram exportados para o 3Ds Max;
- Usando o mapeamento de texturas Planar, dentro do Modificador UVW Map do 3Ds Max, foi possível mapear o vídeo com a textura animada diretamente no plano 3D a que este pertence, obtendo excelente acuidade com o dado visualizado no Paraview. O dado pode ser visualizado na própria tela do 3Ds Max, da mesma forma que é visualizado no Paraview.

Além disso, foi mapeada uma textura planetária, usando a fonte ETOPO 5 para "ilustrar" a batimetria global. Entretanto, devido ao fato da Batimetria em uso no aplicativo ser diferente do ETOPO, ocorrem pequenas alterações nas coordenadas da textura. Isso

está ligado às diferenças de resolução espacial entre a grade batimétrica do modelo (1 por 1 grau) e a grade do ETOPO (5 em 5 minutos).

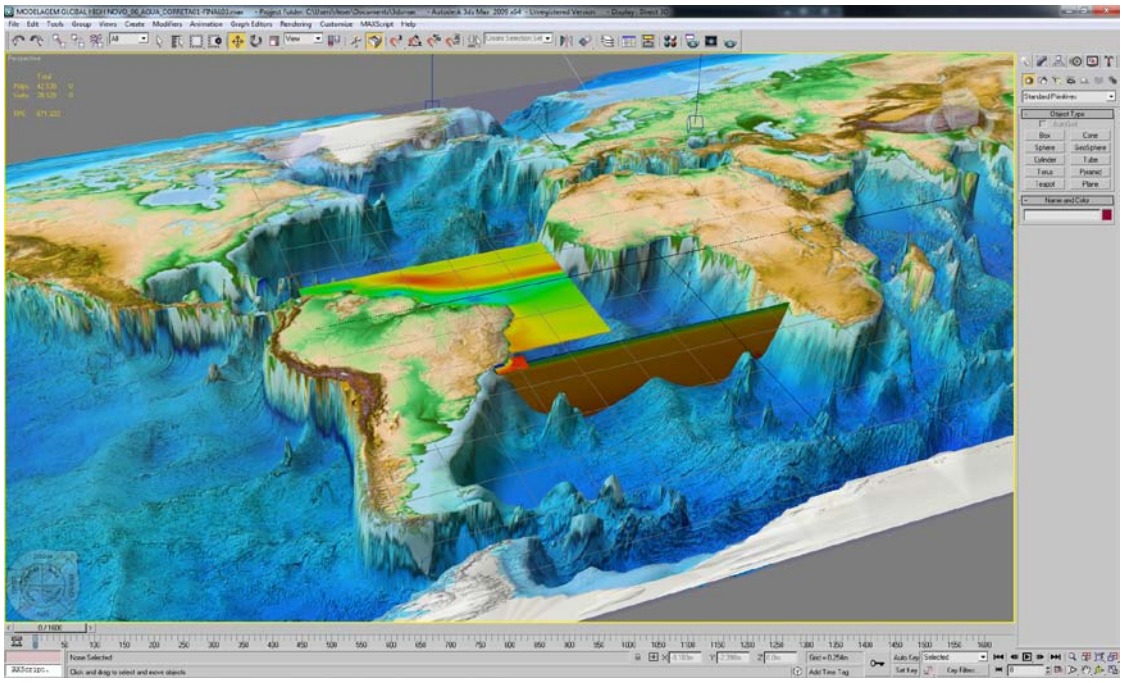

Figura 30 - Dados estáticos e transientes funcionais no software 3Ds Max.

Com todos os dados funcionando perfeitamente no software 3Ds Max, animações 3D foram realizadas e tornou-se possível seguir para a próxima fase: a preparação e exportação dos dados para ambientes em real-time.

5.8 – Redução de Polígonos e UVLayout no programa ZBrush

Ao se trabalhar com aplicativos em real-time, a palavra chave é **otimização.**

Para conseguir inserir uma vasta gama de objetos tridimensionais, texturas e efeitos, os modelos 3D devem ser tratados de forma a alcançar o menor tamanho, menor custo computacional (diga-se memória de vídeo ocupada) com a melhor qualidade gráfica possível. Para isto, a indústria do entretenimento, principalmente a de jogos, vem desenvolvendo técnicas para tornar a experiência virtual cada vez mais próxima da real e assim aumentar e fidelizar seus consumidores.

Como o objetivo deste projeto é criar um ambiente 3D interativo onde se possa interagir com dados científicos tridimensionais, as mesmas técnicas utilizadas na indústria de jogos foram aplicadas no desenvolvimento do visualizador.

A primeira trata da redução de polígonos. Quando se trabalha com aplicativos 3D em real-time, existe um limite para número de polígonos em cena que varia dependendo de

cada placa de vídeo. Como o objetivo é desenvolver um aplicativo que funcione nas placas mais comuns e de baixo desempenho, a regra principal é de que quanto menos polígonos, mais rápido o seu programa será executado.

Tendo isso em mente, diversas opções de redutores de polígonos foram estudadas para uso neste visualizador: o modificador *optmize*, do próprio 3Ds Max 2011, o *VizUp*, *Polygon Cruncher* e por último o *Decimation Master*, um plugin para o software ZBrush. Este último destacou-se por ser o produto mais recente no ramo de redutor de polígonos e demonstrou excelente capacidade técnica, ao remover até 99% do número de polígonos de um objeto, mantendo sua aparência original. Rapidamente o *Decimation Master* se tornou um *standard* na indústria de jogos, sendo utilizado mundialmente. Testes foram realizados com os modelos científicos e os resultados foram satisfatórios: no caso da batimetria do Campo de Santos, o modelo 3D original possuía 110.592 polígonos. Após a redução, o modelo ficou com 22.080 polígonos, uma otimização de 80%. Níveis maiores de redução podem ser alcançados, contudo, devido ao fato de se tratar de um modelo 3D orgânico, com muitas superfícies sinuosas, se os polígonos forem reduzidos demais, perde-se muito das características originais do modelo, deixando-o "facetado" (com superfícies poliédricas evidentes). A otimização de 80%, neste caso, foi escolhida por representar a melhor relação custo benefício entre quantidade de polígonos e semelhança com o modelo original.

O mesmo redutor de polígonos foi utilizado em todos os modelos presentes no aplicativo.

Figura 31 a - Imagem comparativa de um modelo antes e depois da redução de polígonos(95% de redução).

Figura 31 b - Imagem comparativa de um modelo antes e depois da redução de polígonos (95% de redução).

Geometrias	Original	Reduzido	% de Redução
Batimetria Solimões	183.762 polígonos	30.307 polígonos	83,5 %
Batimetria Campo de Santos	110.592 polígonos	22.086 polígonos	80%
Batimetria Global	142.822 polígonos	57.350 polígonos	60%
Topografia Solimões	64.090 polígonos	12.817 polígonos	80%

Tabela 2 - Quadro comparativo com as geometrias utilizadas, antes e depois da redução de polígonos com o *Decimation Master*.

Apesar do processo de redução de polígonos melhorar muito a exibição dos modelos em real-time, ele também gera um inconveniente: as texturas são perdidas. No caso dos modelos científicos, as texturas indicam os valores do dado científico, no caso do Rio Solimões, o gradiente de cor do azul para o vermelho representa o aumento gradativo da profundidade do rio. Este dado é extremamente importante, pois é através dele que os pesquisadores avaliam os dados e verificam sua autenticidade junto à natureza.

Logo, uma alternativa tinha que ser desenvolvida.

Como dito acima, quando o redutor de polígonos age sobre um modelo 3D, ele remove sua textura e seu UV Map, que contém as coordenadas de UV. O processo de UV Mapping "mapeia" uma textura em um objeto 3D. Ao contrário do "X,Y e Z" das coordenadas cartesianas, que são as coordenadas do objeto 3D no espaço, outro conjunto de coordenadas é necessário para descrever a superfície de um modelo, então as letras "U" e "V" são usadas [37] As coordenadas de UV simplesmente representam cada face de um objeto 3D, planificado em 2D, permitindo que uma textura plana seja aplicada ao mesmo, em todos os ângulos - frente, costas, em cima e em baixo. Quando o modelo em VRML é exportado do Paraview, o programa cria a sua própria coordenada de UV, "pintando" o dado científico em cima de cada polígono do modelo original. Assim, quando o redutor é aplicado no modelo, a quantidade e disposição dos novos polígonos é alterada, logo a coordenada de UV do modelo original se perde e a textura com os dados científicos não pode ser aplicada novamente.

Para resolver este problema, uma nova coordenada de UV deve ser criada para o modelo reduzido. Esse mapeamento do modelo 3D em 2D chama-se UVlayout. Novamente, diversos softwares podem realizar o UVlayout: o próprio 3DsMax, programas específicos de UV como o UVLayout, Unfold3d e o UVMaster, um plugin para o ZBrush.

O UVMaster também se trata de um lançamento recente da Pixologic, para o software ZBrush. Normalmente, o ato de se construir um layout de UV é extremamente

demorado e tedioso, tomando horas dos profissionais de modelagem 3D. Porém, com o UVMaster, esse processo é reduzido a minutos. Ele conta com um sistema muito intuitivo onde você define as áreas importantes do seu modelo 3D e as áreas onde pode haver corte e simplesmente apertando um botão, o layout UV é feito automaticamente.

Aprofundando um pouco mais, ao se realizar mapeamentos UV de modelos orgânicos, a técnica funciona exatamente como a criação dos famosos tapetes de urso. O usuário deve literalmente cortar o modelo 3D em áreas que não sejam visíveis com frequência e esticar as áreas cortadas de modo a aproveitar ao máximo o espaço quadrado definido pelo UVlayout.

Figura 32 - Comparação entre um tapete de urso e UVlayout de um modelo tridimensional de um dinossauro - o princípio é o mesmo.

Esta solução foi de tremenda ajuda para o desenvolvimento deste trabalho, pois o UVlayout do Rio Solimões seria extremamente complicado de ser feito manualmente, como foi em um primeiro momento, custando aproximadamente 10 horas de trabalho contínuo.

Usando o plugin UVMaster, esse trabalho foi reduzido para 3 minutos.

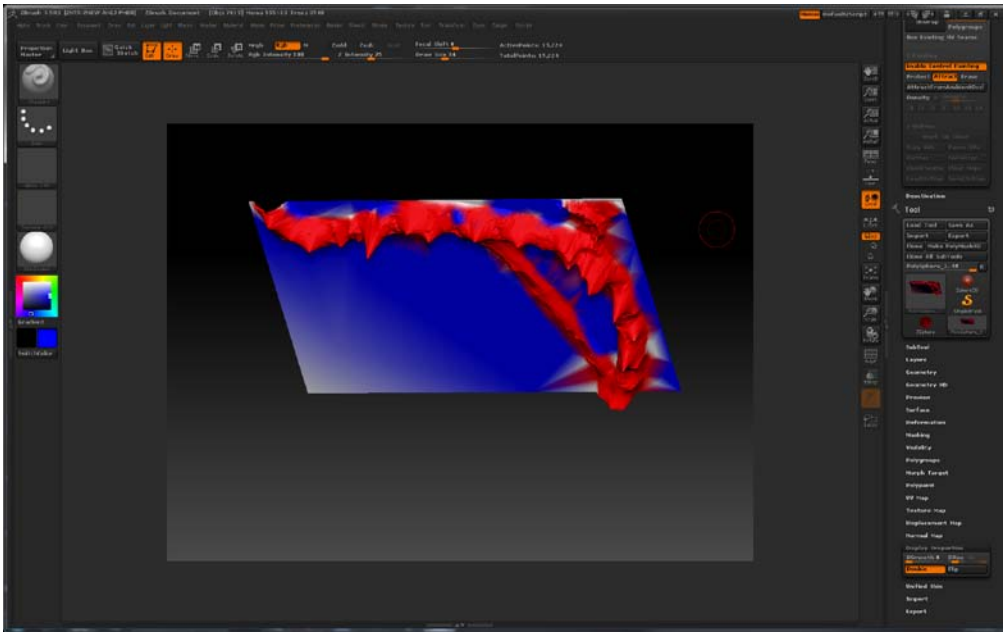

Figura 33 a - O plugin UVMaster em funcionamento: áreas vermelhas são protegidas e áreas azuis atraem possíveis cortes.

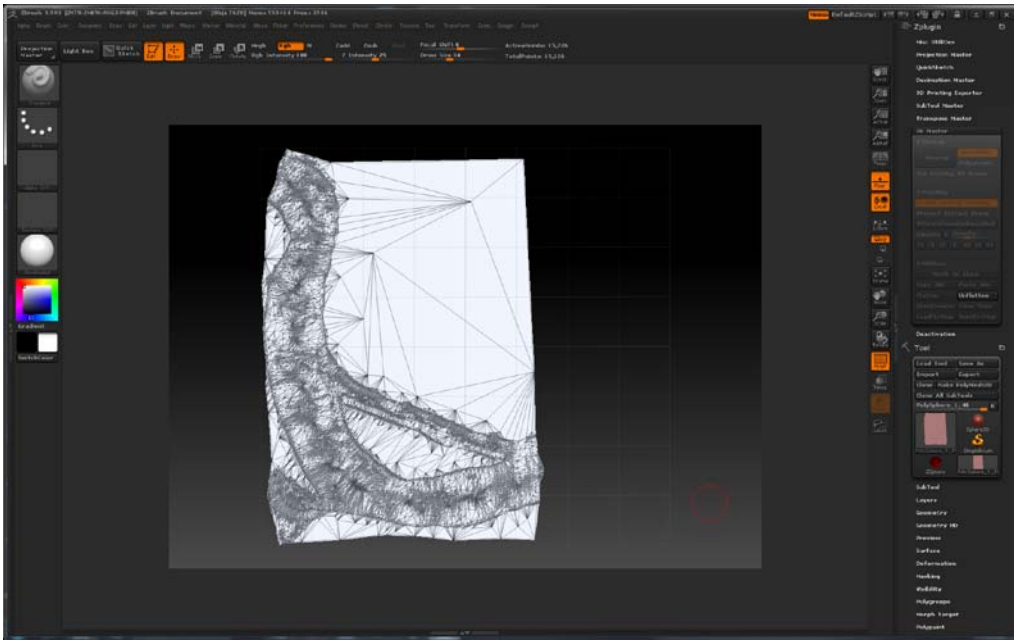

Figura 33 b - Após pressionar o botão "Unwrap", obtém-se o resultado acima: o modelo batimétrico tridimensional do Rio Solimões mapeado em 2D.

No centro da imagem 33 b, pode-se observar um grid quadradro, onde a figura do rio (em branco), se encaixa. A questão é: por que um grid quadrado?

As placas de vídeos dos computadores, por definição, só aceitam texturas quadradas, em potência de 2 (64x64, 128x128, 256x256, 512x512). Logo todas as texturas aplicadas em modelos 3D que são processados pela placa de vídeo necessitam

deste padrão, ou não são exibidas - alguns programas e placas de vídeo mais recentes fazem a transformação dos mapas retangulares, por exemplo, em texturas em potência de 2, automaticamente, mas mesmo assim, para fins de real-time, as texturas são sempre quadradas.

5.9 – Projeção de Mapas no 3Ds Max

Como foi dito anteriormente, a coloração ou textura do modelo em VRML extraído do Paraview é extramamente importante para interpretação do modelo científico.

Uma vez que o modelo com a contagem de polígonos reduzida tem o seu UVlayout realizado pelo ZBrush, este é exportado com a extensão .OBJ (*Wavefront Technologies*) e importado no 3Ds Max. Desta forma, temos o modelo original com muitos polígono ou *high poly* com a coloração original do dado científico e o modelo reduzido, *low poly* com o novo uvlayout.

O 3Ds Max conta com um painel chamado Render to Texture. Este painel é responsável pela criação de mapas real-time para os modelos do 3Ds Max, permitindo que estes sejam exibidos em tempo real dentro de aplicações 3D.

Uma de suas características é a capacidade de projetar texturas de um modelo para outro. Esta técnica é muito utilizada no ramo de jogos, ao se projetar as texturas de modelos com imenso grau de detalhamento e realismo em modelos que dispõem da mesma topologia, da mesma silhueta 3D, porém, com contagem de polígonos infinitamente inferior.

Modelos esculpidos digitalmente em softwares como o ZBrush e Mudbox, são remodelados com poucos polígonos, para que possam ser exibidos em real-time e têm o detalhamento da escultura digital projetados no modelo com poucos polígonos. Através do uso de mapas chamados Normal Maps, todos os detalhes da escultura simulam volume no modelo reduzido e em seguida pinta-se as cores do modelo.

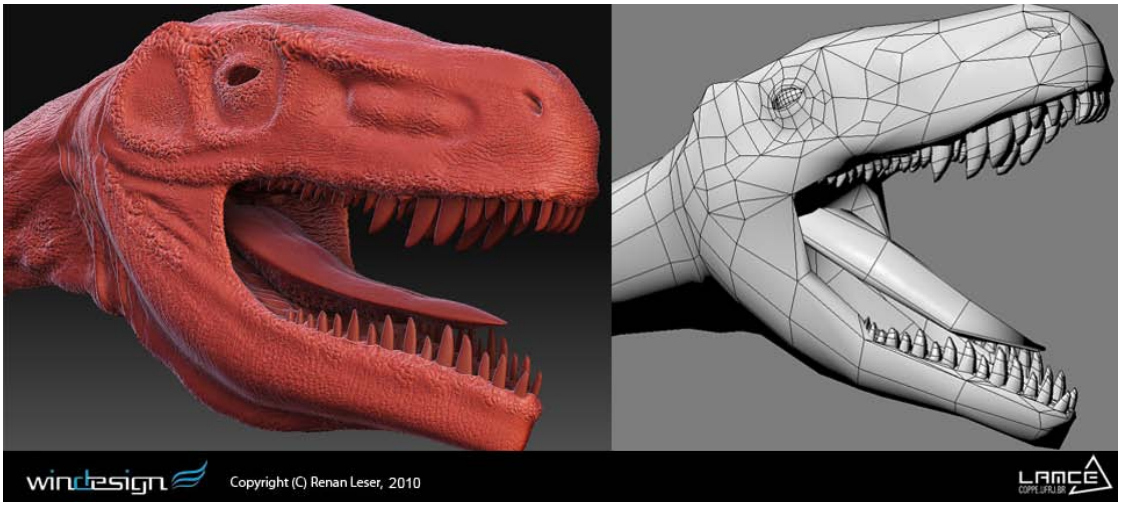

Figura 34 a - Escultura tridimensional à esquerda (high poly) e modelagem low poly à direita.

Figura 34 b - Modelo final em real-time, com baixa contagem de polígonos e texturas de alta qualidade.

Sendo assim, utilizando das mesmas técnicas, as cores (dados) do modelo *high poly* do Rio Solimões foram projetadas no modelo *low poly*, com baixa contagem de polígonos. Este método foi utilizado para todos os modelos presentes no visualizador, garantindo fidelidade dos dados visualizados a um menor custo computacional, permitindo a otimização dos dados e congregação de vários modelos em um único visualizador tridimensional interativo.

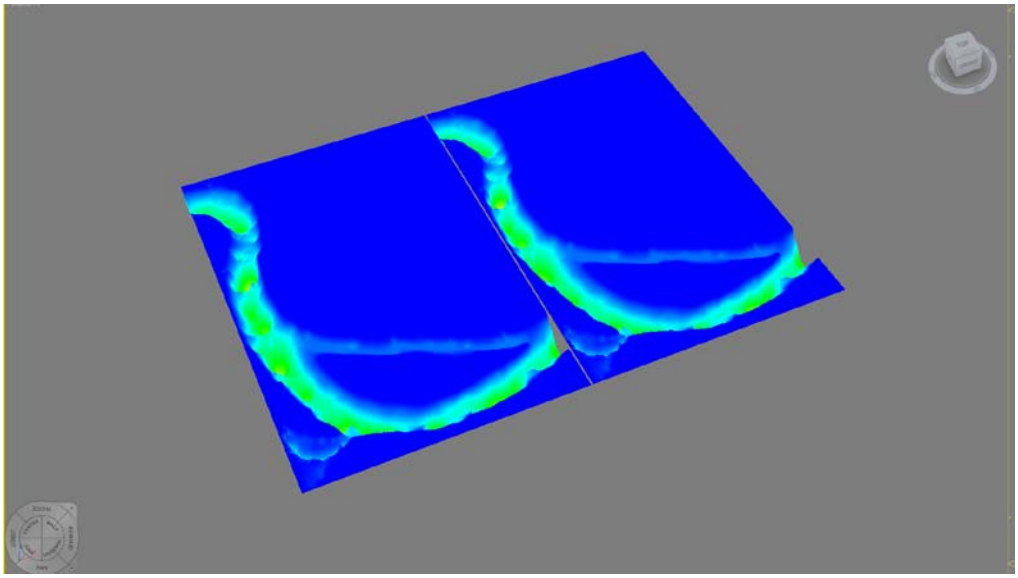

Figura 35 a - Modelo do Rio high poly à esquerda e low poly à direita - diferença imperceptível.

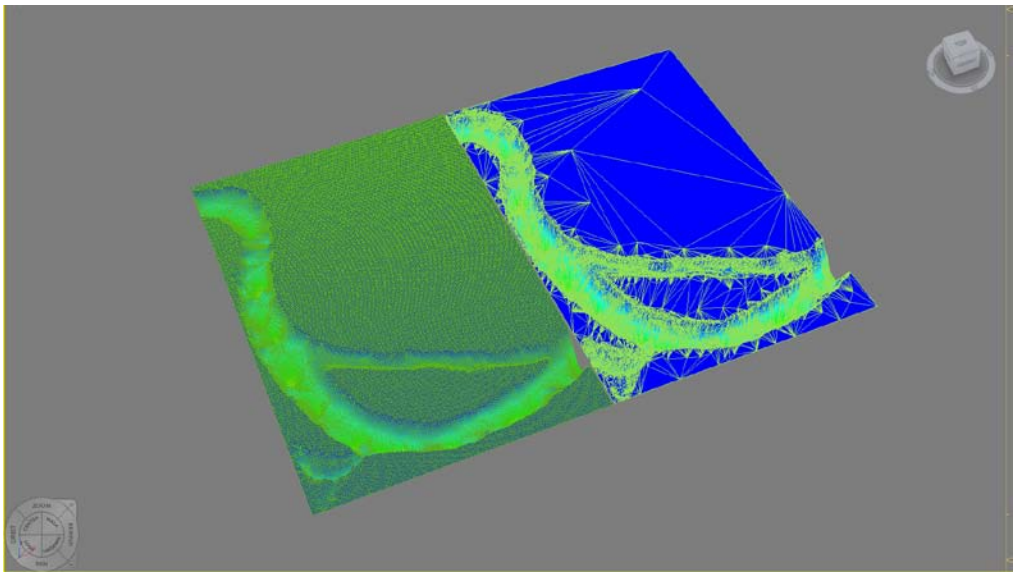

Figura 35 b - Semelhante a figura anterior, porém com visualização de cores e wireframe.

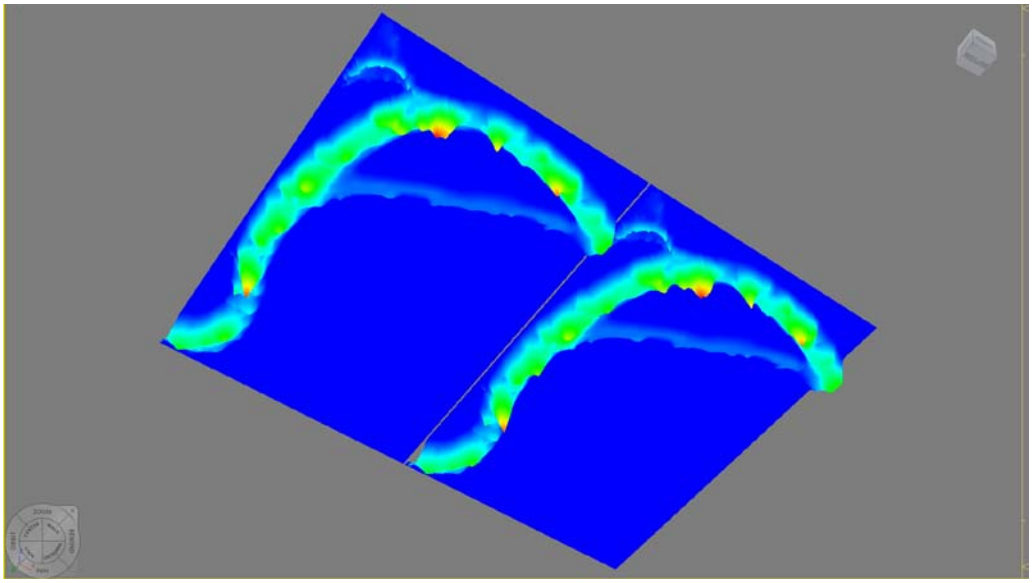

Figura 35 c - Vista inferior dos mesmos modelos - high poly e low poly.

5.10 – Entrada no programa Quest3D

Neste tópico serão discutidas as técnicas e metodologias para inserção de modelos e criação do executável em real-time do visualizador 3D interativo.

5.10.1 –Resumo básico do funcionamento do Quest3D

O Quest3D funciona através de um gráfico de canais, uma tela onde as estruturas dos canais são construídas. Os programas criados dentro do Quest3D são construídos com blocos que se assemelham à blocos de madeira, aqueles que as crianças usam para construir edifícios. No Quest3D, esse blocos são chamados "channels" ou canais. Abaixo segue a figura de um canal:

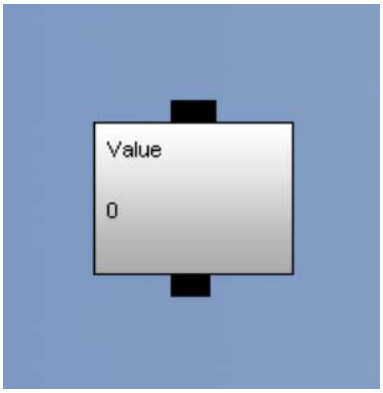

Figura 36 - Exemplo de um canal do Quest3D.

Esses blocos "canalizam" informação ou funcionalidade para outros blocos. Por isso o nome *channel* ou canal.

Cada canal é ligado ao outro através de setas, que dependendo de sua direção, levam ou trazem informação.

A imagem abaixo mostra o exemplo de um pequeno grupo de canais. Essa estrutura de canais é chamada *Channel Group* ou Grupo de Canais.

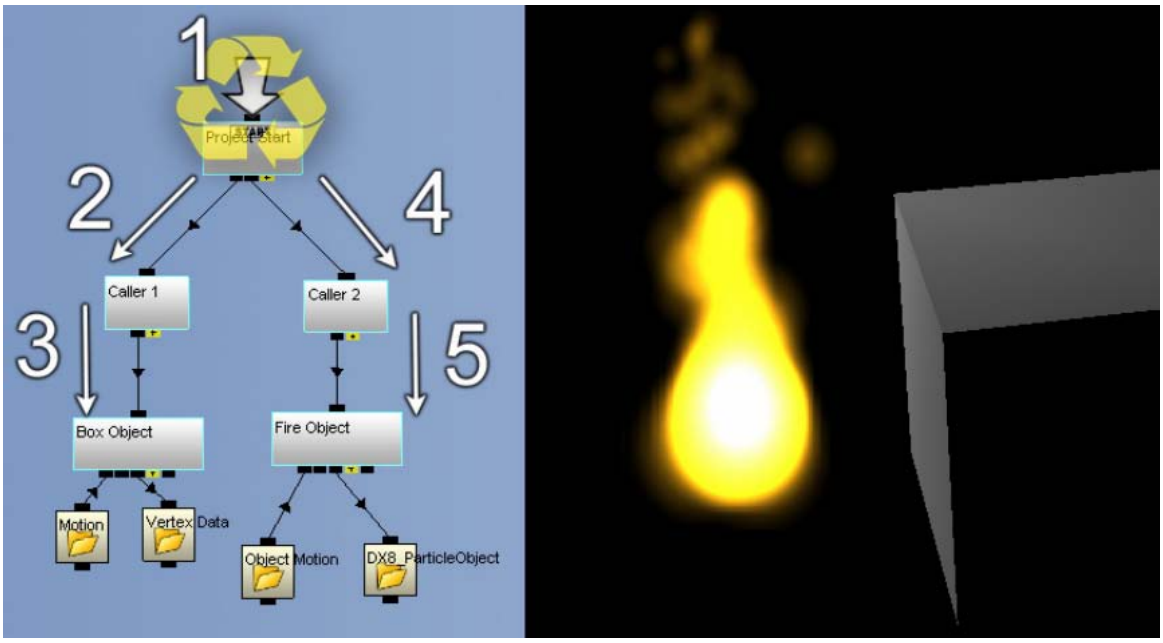

Figura 37 - *Flow* do Quest3D - programação visual à esquerda e resultado à direita.

1 - Início do Projeto, onde começa a contagem de frames do arquivo, com o canal *Start Channel*;

2 - *Channel Callers* funcionam da esquerda para direita, mas o *Start Channel* "chama" o canal da esquerda chamado "*Caller 1*" primeiro.

3 - O canal "*Caller 1*" chama a hierarquia abaixo dele. Esta hierarquia começa com o canal do tipo "*3D Object*", chamado "*Box Object*". Este canal provê a informação necessária para inserir a caixa cinza á direita, na tela.

4 - Depois de executar a hierarquia do lado esquerdo, o *Start Channel* continua com seu próximo *child*, o "*Caller 2*".

5 - O canal "*Caller 2*" chama seu *child* "*Fire Object*". Este canal e seus "filhos" criam o fogo visto na tela à direita.

Quando toda a estrutura é executada, o programa começa novamente a partir do *Start Channel.*

5.10.2 – Importando as geometrias e texturas

Com todos os modelos em low poly preparados e texturizados, estes foram exportados no formato OPENCOLLADA (.dae) para serem inseridos no Quest3D.

Fazendo um acréscimo, todas as texturas foram exportadas no formato .DDS (DirectDraw Surface), com a especificação DXT5 para os dados de batimetria (ARGB 8 bits por pixel) e as texturas da interface foram salvas na especificação 8.8.8.8 (ARGB 32 bits por pixel, para garantir uma transparência de qualidade). O formato .DDS é o mais indicado para aplicações em real-time, pois permite excelentes compressões de arquivo e usa o algoritmo S3TC (S3 Texture Compression), utilizado nas placas de vídeo e em consoles como Playstation 3 e Xbox 360. O que deve ser levado em conta é que nesse caso, o tamanho do arquivo em disco não conta muito, mas sim a quantidade de memória que ele vai ocupar da placa de vídeo. O formato .DDS é preparado para ocupar o menor espaço possível, porém texturas de 2048x2048 ainda assim ocupam 22Mb (vinte e dois mega bytes) de memória de vídeo. Além disso, O .DDS permite o pré-cálculo dos mip-maps. Mip-maps são cálculos gráficos da redução das texturas *tiled* (que se repetem, como a grama), quando vistas à distância. Ele permite que mesmo afastadas das câmeras, as texturas mantenham um bom grau de visibilidade e coerência visual. Esse pré-calculo agiliza o carregamento das texturas na placa de vídeo, uma vez que o software não precisa realizá-lo.

Quanto ao tamanho das texturas, os gráficos tridimensionais de maior extensão: batimetria Global e batimetria do Campo de Santos utilizaram texturas de 2048x2048 pixels. Já os outros gráficos utilizaram texturas de 1024x1024 pixels.

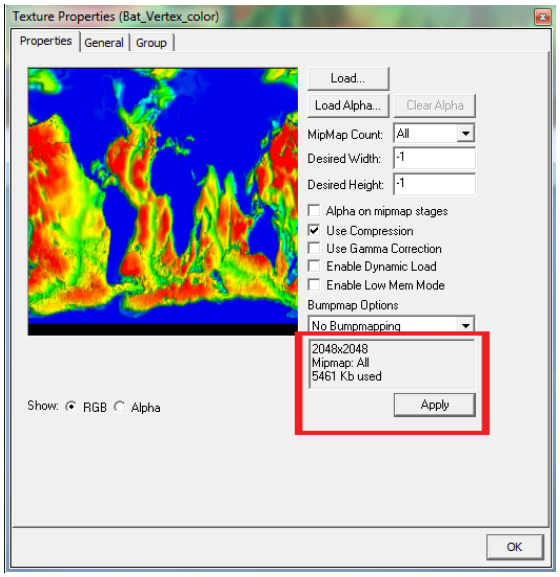

Figura 38 - Textura da batimetria global no Quest3D, com consumo de memória e tamanho em pixels.

As geometrias, em resumo, passaram pelo seguinte processo:

- Extraídas no formato .WRL do Paraview;
- Importadas no DeepExploration e exportadas em .WRL;
- Importadas no 3DsMax e exportadas em .OBJ (formato padrão do ZBrush);
- Redução de Polígonos e UVMap realizado no ZBrush - novamente exportadas em .OBJ;
- Importadas no 3DsMax, texturas projetadas via Render to Texture e exportadas em OpenCOLLADA;
- Importadas no Quest3D;

Assim que são importadas dentro do Quest3D, as geometrias assumem a aparência dos blocos, que compreendem a programação visual do software.

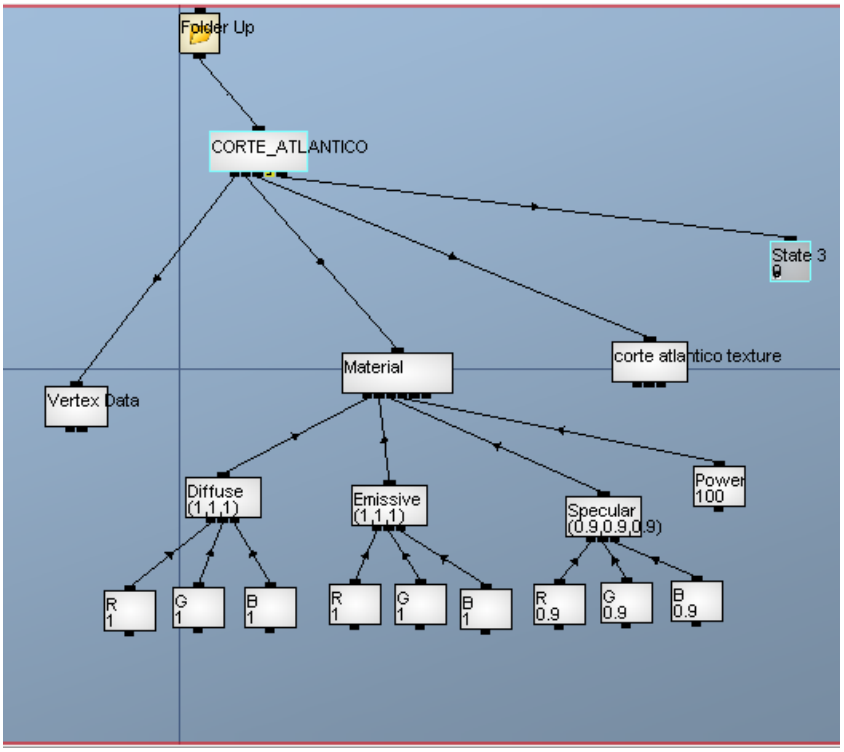

Figura 39 - Formatação das geometrias no Quest3D.

CORTE_ATLANTICO - esse canal, do tipo *Surface*, congrega todas as propriedades do modelo em questão e é ligado ao canal *3D Object*, que pode ser ligado diretamente ao canal de Render do Quest3D.

VERTEX DATA - compreende as informações de vértice, polígonos e edges.

MATERIAL - esse canal controla a visualização do modelo 3D, alterando suas cores (*diffuse*), emissão de luz (*emissive*), reflexos (*specular*) e nível de reflexão, se o material é muito ou pouco reflexivo (Power).

CORTE ATLANTICO TEXTURE - esse canal, chamado *Texture* é onde ficam as texturas do modelo, representado na figura 29. Mais de uma textura pode ser aplicada por modelo, mas nesse caso, apenas uma é utilizada.

STATE 3 - Esse canal, chamado *Value*, está com uma informação de programação, que informa quando o modelo será visível para a câmera ou não. Ele se liga ao *children "Draw Object"* do canal *Surface*, nomeado neste modelo como CORTE_ATLANTICO.

5.10.3 – Configuração de Câmeras

O Quest3D possui vários tipos de câmera para projetos em real-time, cada uma com funções e características diferentes.

Para este trabalho, foram utilizadas três câmeras:

Object Inspection Camera - Essa câmera tem o foco centralizado no objeto, girando e oferecendo *zoom in* e *zoom alt*. Ela é muito utilizada para demonstração de produtos ou modelos 3D. Neste caso, o modelo em foco são os dados científicos;

Fly Through Camera - Essa câmera é uma modificação da *3D Navigation Camera* que vem com o Quest3D, permitindo vôos que ignoram a gravidade.

As câmeras foram configuradas manualmente, para cada dado em específico. Isso ocorreu devido ao fato de todos os modelos estarem dentro da mesma escala de latitude e longitude. Logo, uma câmera configurada para visualizar a batimetria global, por questões de escala, não consegue visualizar a batimetria do Rio Solimões na Amazônia. Sendo assim, para cada dado de batimetria, uma câmera diferente teve de ser configurada.

Além disso, uma *Stereo Camera* foi configurada igualmente para cada dado. A *Stereo Camera* permite a visualização com óculos anaglifos (vermelho e azul) e polarizados.

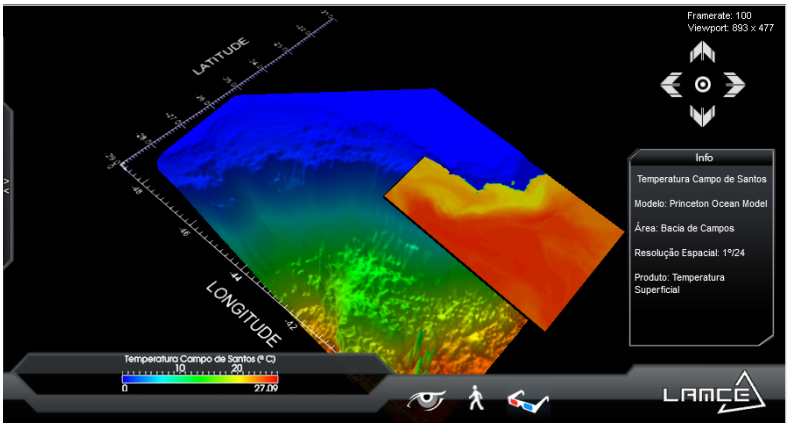

Figura 40 a - *Object Inspection Camera*.

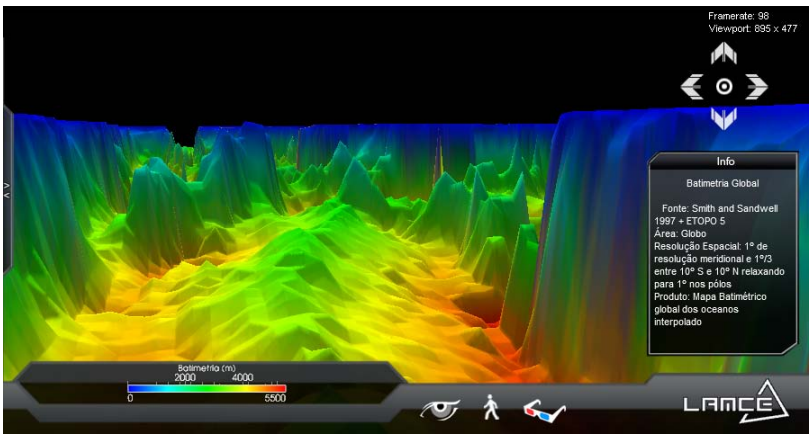

Figura 40 b - *Fly Through Camera*.

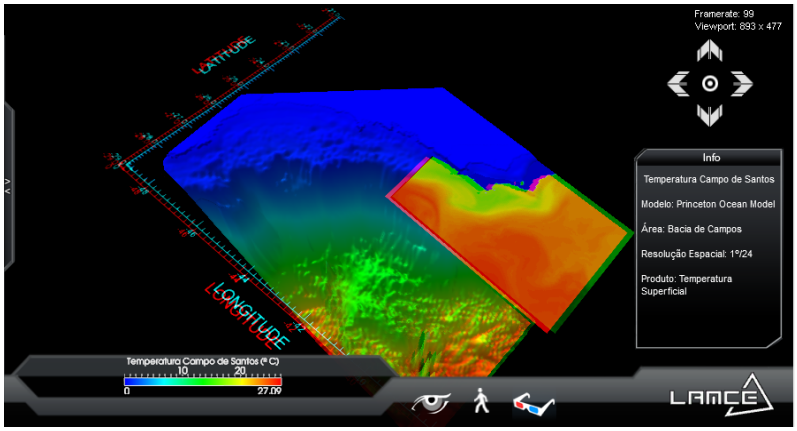

Figura 40 c - *Stereo Camera*.

Uma preocupação durante o setup de câmeras foi o *Aspect Ratio*. Ele define a proporção da câmera, que para monitores comuns é 4/3. Se uma programação específica não for embutida em cada câmera, ao usar diferentes resoluções, as câmeras ficariam

distorcidas: ou esticadas, ou apertadas, impossibilitando a correta visualização dos dados. Logo, uma pequena rotina foi introduzida em cada câmera para garantir que, independente da resolução de monitor em uso, a proporção fosse sempre 4/3.

5.10.4 – Aplicação de texturas e shaders

Conforme dito anteriormente, as texturas foram produzidas no formato .DDS, com resoluções que variam de 1024x1024 até 2048x2048.

Apesar do Quest3D suportar shaders de excelente qualidade, exibindo gráficos tridimensionais comparáveis aos consoles Playstation 3 e Xbox 360, para este projeto, nenhum shader avançado foi utilizado. Shaders são conjuntos de instruções de software, que são usadas primariamente para calcular efeitos de render em placas de vídeo, com grande flexibilidade, permitindo a criação de água, chuva, bolhas, efeitos de relevo e etc [38]. Neste caso, não foram usados shaders avançados devido ao fato de que o mais importante para o visualizador é a interpretação dos dados científicos. Logo, para que isso fosse possível, apenas as texturas com a informação do dado (vertex color) foram utilizadas. A única exceção é a textura da Batimetria Global, que dependendo do dado a ser visualizado, altera entre a textura com a informação batimétrica e a textura obtida a partir do ETOPO 1. E mesmo assim, essa exceção só veio à tona para facilitar a visualização dos demais dados, garantindo melhor contraste de figura e fundo.

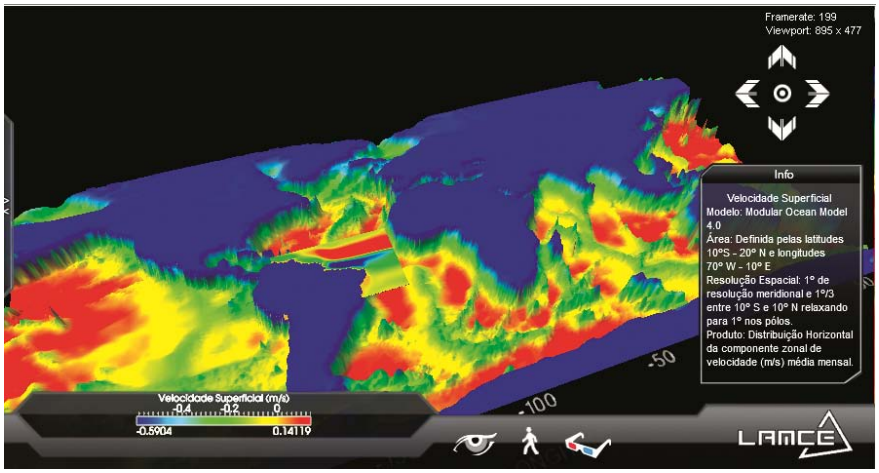

Figura 41 a - Visualização prejudicada pela ausência de contraste entre figura e fundo.

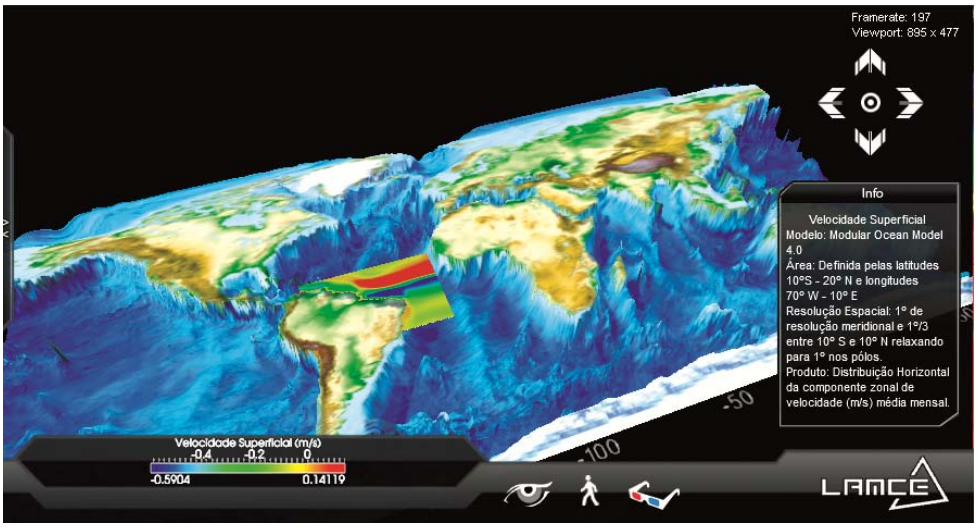

Figura 41 b - Visualização do dado garantida pelo bom contraste entre figura e fundo.

O único shader utilizado no visualizador foi um shader básico de transparência, para facilitar a implementação dos eixos de latitude e longitude para cada dado.

5.10.5 – Configuração de dispositivos de entrada e saída de dados

Quatro dispositivos de entrada e saída de dados foram utilizados neste trabalho, oferecendo três formas de interação:

- Mouse e teclado - utilizado na versão final do visualizador;
- TV Touch Screen - utilizada na primeira versão e teste de campo do visualizador;
- Controle sem fio para PC - utilizado na segunda versão e segundo teste de campo;

Para todos os dispositivos, a metodologia de programação no Quest3D é a mesma, utilizando o canal "*User Input*".

Este canal é responsável por controlar a entrada e saída de dados do teclado, mouse e controles (*joysticks*).

Ao fazer a verificação de *verdadeiro ou falso,* o canal insere o valor 0 ou 1 em outros canais, permitindo que ações sejam descritas de acordo com esse input de dado. Combinado com o canal *Expression Value,* diversas fórmulas matemáticas podem ser descritas, criando diferentes valores de saída, que podem ser aplicados das mais diversas formas.

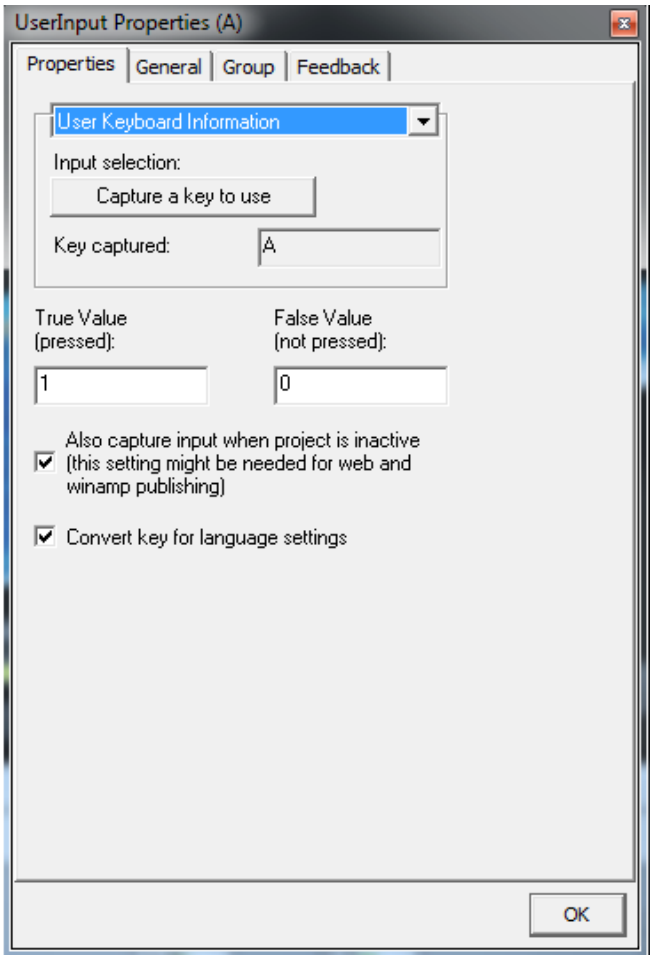

Figura 42 - Janela de propriedades do canal *User Input.*

5.10.6 – Criação de Executável para Plataforma PC e Publicação Online

Assim que todos os dados estão inseridos no Quest3D, toda a programação foi feita, as texturas aplicadas e a interface gráfica implementada (a ser discutida no próximo capítulo), o projeto está pronto para publicação.

Esta publicação pode ser feita de várias formas:

Como um arquivo Executável (.EXE);

Página da Web;

Quest3D Viewer;

Installer;

Screen Saver;

A publicação em arquivo executável é a mais indicada, pois cria um arquivo que pode ser visualizado em qualquer computador, mesmo sem ter o Quest3D instalado. Ao abrir este arquivo, ele mostra as diferentes resoluções de tela para o usuário escolher (caso o programador tenha deixado essa opção ativa) ou abre em *full screen*.

A publicação em Página da Web cria um arquivo .Q3D, e um arquivo .HTML. Este .HTML possui uma programação básica que ao ser aberto, pede pelo plugin de visualização online do Quest3D, o Quest3DWebViewer, que uma vez instalado executa o arquivo .Q3D.

A publicação em Quest3D Viewer cria um arquivo .Q3D, que também precisará do Quest3D instalado na CPU.

A publicação "Installer" cria um arquivo executável que ao ser aberto, instalará o programa no seu computador, da mesma forma que qualquer outro programa que exige instalação. Essa opção é mais indicada para projetos grandes, pois a instalação no disco rígido acelera a execução do programa.

A publicação screen saver se auto-explica. Ela cria um arquivo executável que pode ser usado como screen saver nos computadores.

Como a função deste trabalho é a difusão dos dados científicos, o projeto foi exportado como Executável e Página da Web. A preocupação com uso de memória e espaço em disco refletiu-se em um executável com 35 MB comprimido e 198 MB após descompressão.

Na página seguinte, pode-se verificar o fluxograma que resume todo o processo de construção do visualizador.

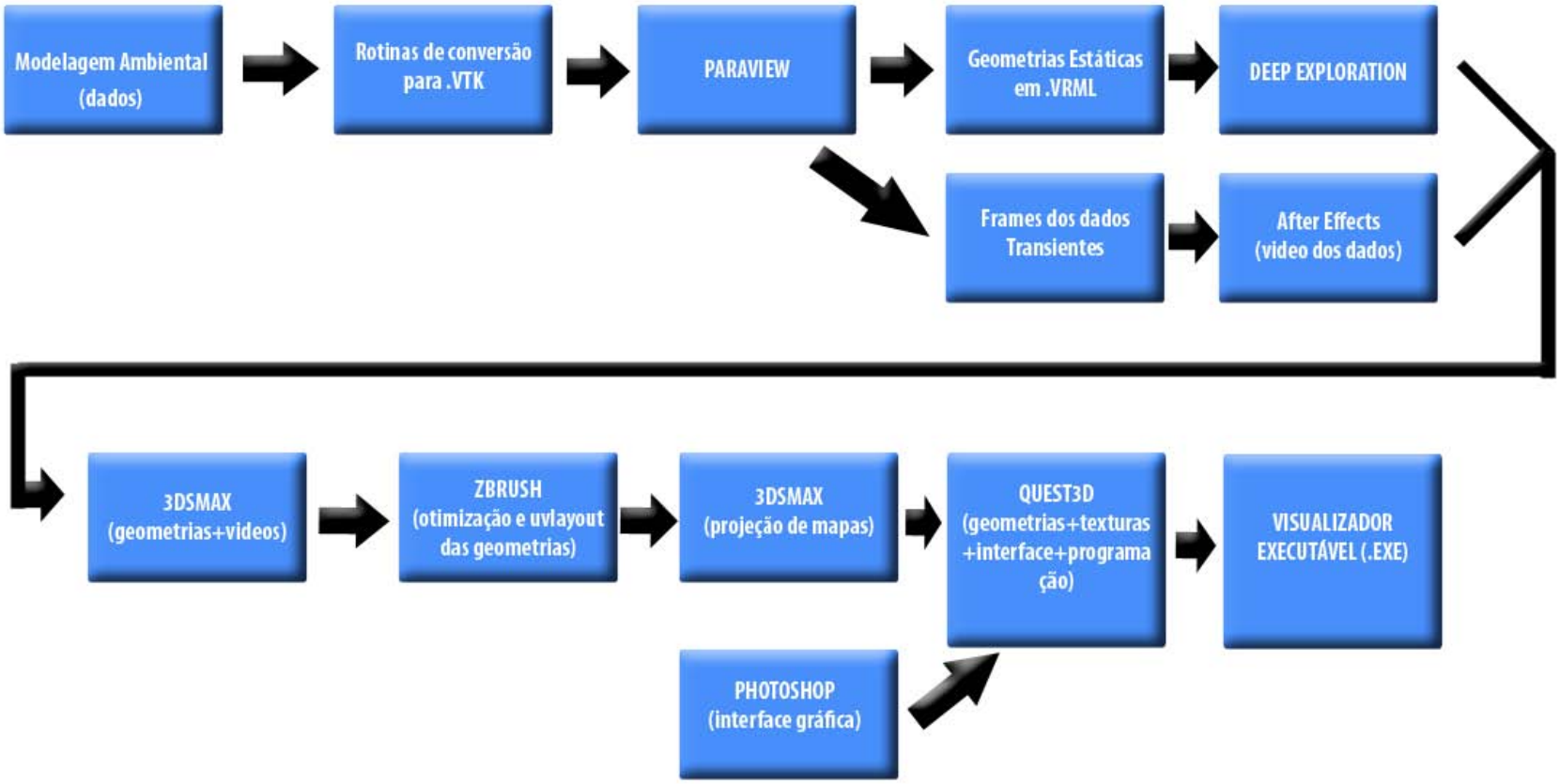

Figura 43 – Fluxograma do Processo de Construção do Visualizador.

5.11 – Design da Interface

5.11.1 - Introdução

O processo de construção da interface gráfica deste projeto ocorreu de forma dinâmica, a partir dos testes de campo realizados e necessidades que foram surgindo no decorrer da elaboração do visualizador.

Assim que delimitou-se toda a interação e dados a serem inseridos, a interface teve seu design final definido.

5.11.2 - Primeiro Layout

O primeiro layout do visualizador visava garantir a interatividade básica para visualização de somente três dados: a batimetria global, um dado de velocidade superficial da componente U na área do equador e a temperatura global.

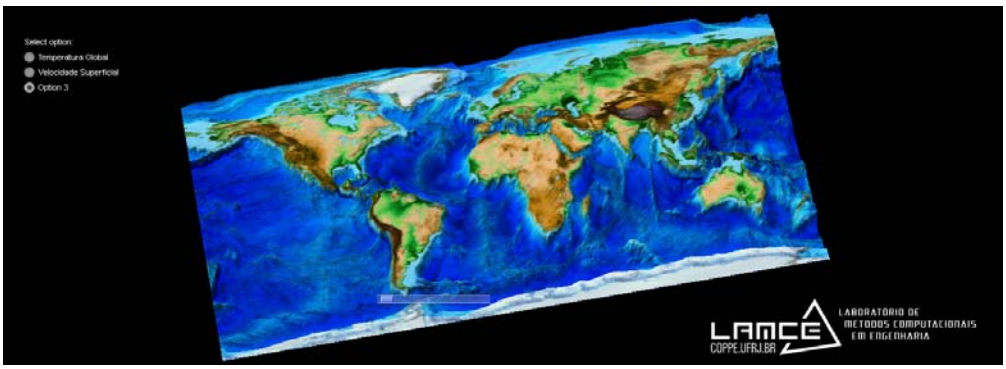

Figura 44 a - Visualização da Batimetria.

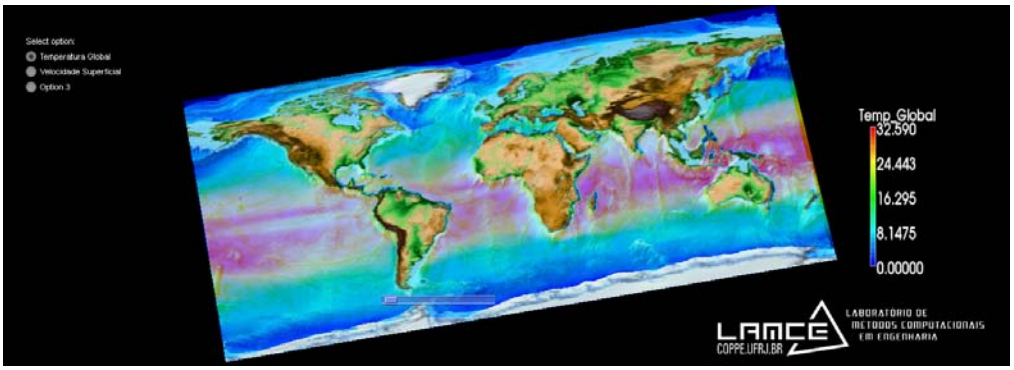

Figura 44 b - Visualização da Temperatura Global.

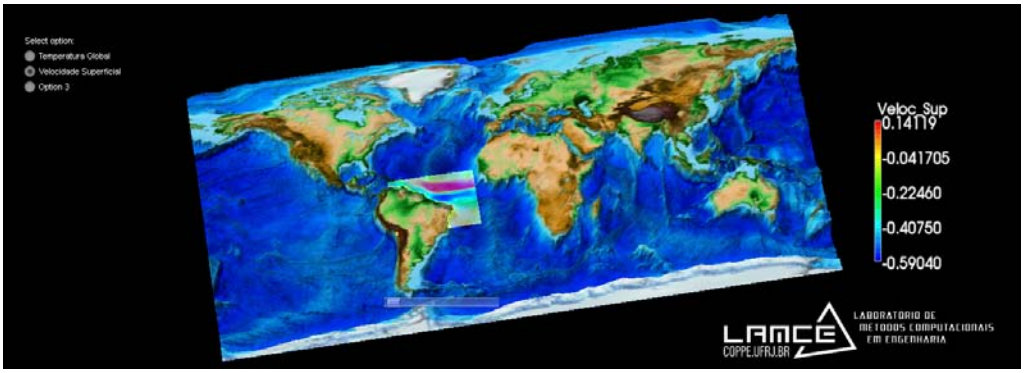

Figura 44 c - Visualização da Velocidade Superficial.

Este layout inicial contava com:

- Dados organizados por Radio Buttons, à esquerda, sempre presentes;
- Object Inspection Camera;
- Slider para controle de Zoom;
- Barras verticais com as legendas dos dados, sempre à direita;
- Logo do LAMCE no canto inferior direito;

Nos testes realizados, este layout funcionou de forma intuitiva, somente apresentando problemas de distorção da interface gráfica, dependendo da resolução de vídeo utilizada.

5.11.3 - Segundo Layout

Este layout foi empregado para apresentação no evento de comemoração de 10 anos do PRH-ANP, em setembro de 2009.

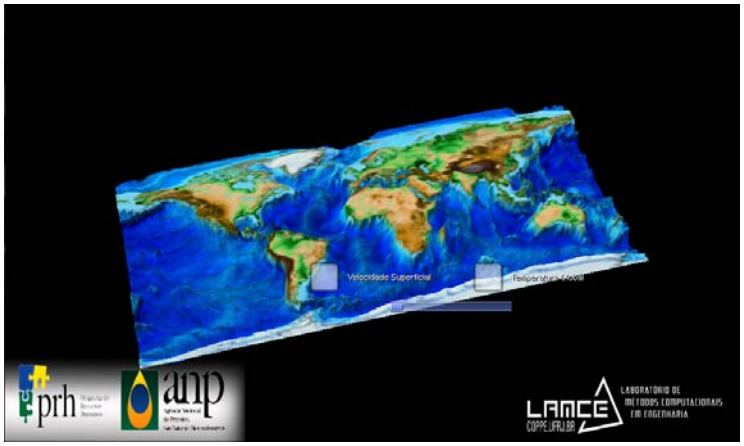

Figura 45 a - Visualização do segundo layout.

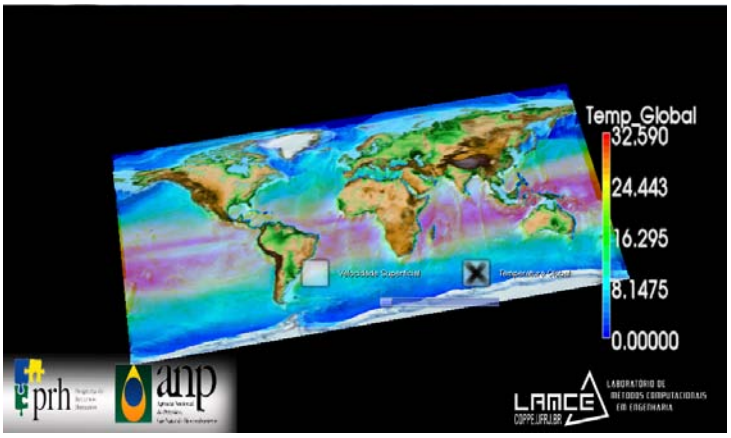

Figura 45 b - Visualização da temperatura global no segundo layout.

Nesta segunda versão da interface, o problema de posicionamento e escala foi ainda maior, fazendo com que os botões de controle de visualização dos dados fossem parar no meio do gráfico 3D, prejudicando muito a visualização do projeto como um todo.

Os logos aplicados também sofreram extrema distorção, algo que não podia permanecer para uma versão final da interface.

5.11.4 - Redesign da Interface

Depois dos dois primeiros layouts e dos testes de campo iniciais, foi definida a quantidade de dados a serem inseridas na versão final do visualizador. Com esses dados em mãos, a interação foi redefinida, assim como a interface gráfica.

Abaixo segue o layout final da interface, criado no software Adobe Photoshop:

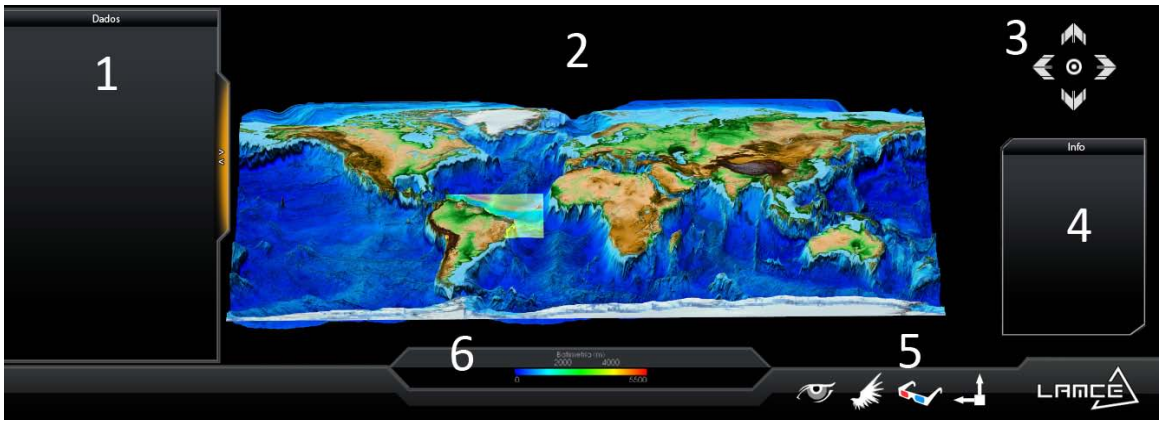

Figura 46 - Layout final da Interface gráfica.

A criação da interface gráfica levou em consideração vários conceitos, entre eles, estudos de gestalt, cor, funcionalidade, ergonomia e simbolismo (signo).

Segundo FILHO [39], pregnância é a Lei Básica da Percepção Visual da Gestalt e assim foi definida: “Qualquer padrão de estímulo tende a ser visto de tal modo que a estrutura resultante é tão simples quanto o permitam as condições dadas”.

Ou seja, “Quanto melhor for a organização visual da forma do objeto, em termos de facilidade de compreensão e rapidez de leitura ou interpretação, maior será o seu grau de pregnância”[39]. Nesse sentido, com o objetivo de facilitar a compreensão dos dados apresentados no visualizador, levou-se em conta a *pregnância da forma*. Sendo assim, desenvolveu-se uma estrutura de caixas e barras mais simples possível, disposto na ordem de leitura ocidental. E se escolheu a cor preta para o fundo do visualizador, garantindo a perfeita visualização de todas as informações. Além desse fator, a cor preta foi empregada com o objetivo de “expressar e reforçar a informação visual” [39], utilizando o conceito de *contraste*. Já que o preto contrasta com a cor azul, comum aos dados visualizados.

Outra questão é a funcionalidade. Com o fundo preto, reduziu-se o consumo de energia. Normalmente utilizam-se cores claras e mais brilhantes como fundo de tela de sites e programas. O consumo de energia por parte do monitor utilizando a cor preta é menor que o consumo de outras cores como o branco. Um exemplo deste tipo de iniciativa para redução de consumo é apresentado através do buscador Blackle, que se propõe a trabalhar como o buscador Google, mas que utiliza predominantemente o fundo de tela preto. Esta pesquisa, realizada por estudantes da UNICAMP [40], revela que podem ser obtidos grandes volumes de economia, tanto monetária quanto energética, através de alterações tecnicamente simples e em um curto período de tempo.

Levando em consideração que os dados apresentados pelo visualizador podem ser analisados durante horas por pesquisadores da área, preocupou-se em utilizar tonalidades que dessem conforto aos olhos. Segundo o estudo das cores, quanto mais clara é a cor, maior sua luminosidade, logo, mais brilho ela emite. Oftalmologistas do mundo inteiro recomendam que os usuários de computador evitem ficar em frente aos computadores por muitas horas, pois o brilho emitido pelo monitor (geralmente branco) facilita o aparecimento da “vista cansada”. Desse modo, os tons de cinza compõem a interface, emitindo pouco brilho e respeitando a visão dos usuários.

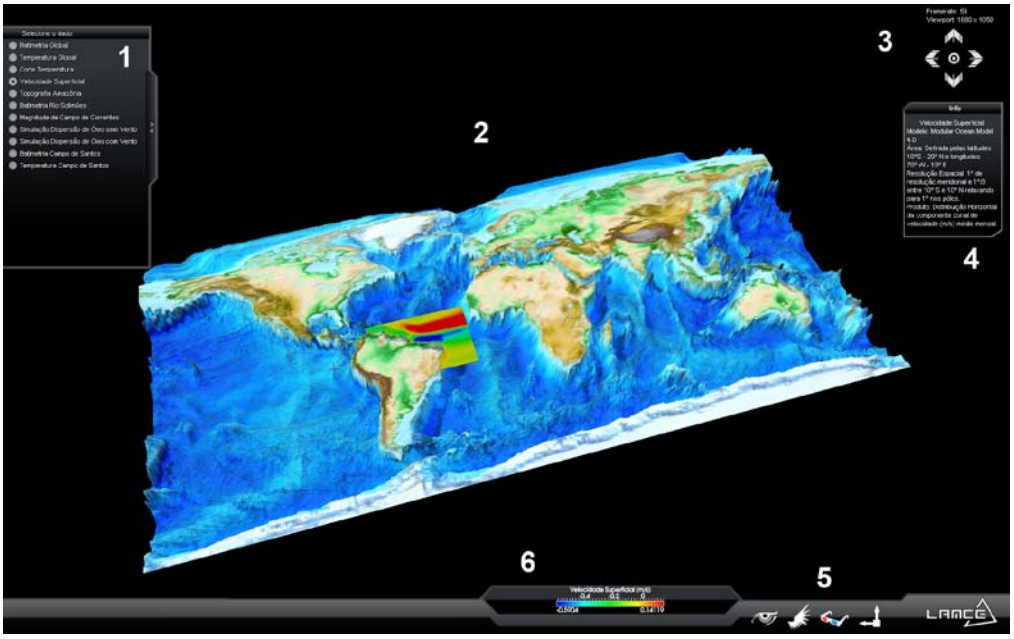

Figura 47 - Áreas da interface.

Definição das áreas da interface:

1 - Área de seleção dos dados: nesta barra ficam todos os dados presentes no visualizador, que podem ser visualizados através de seus respectivos botões;

2 - Área do *Stage* 3D: nesta área, ficam todos os gráficos tridimensionais e visualizações de câmera. Ela compreende toda a extensão da janela, porém está por trás da interface 2D;

3 - Setas para movimentação: essas setas movimentam o centro da *Object Inspection Camera*, a câmera que circunda ao redor de cada dado. O botão central reinicia a posição da câmera.

4 - Info : barra com as informações específicas de cada dado. Ao se alterar o dado na área 1, a informação nesta barra é atualizada dinamicamente.

5 - Botões barra inferior: esses botões se localizam imediatamente acima da barra inferior e apresentam as seguintes funções, da direita para a esquerda:

Botão “Olho” - visualização em modo de inspeção de objeto;

Botão “Asa” - visualização em modo de vôo livre;

Botão “Óculos” - visualização em stereo anaglifo;

Botão “Eixos” - exibição de eixos de latitude e longitude.

6 - Barra de cores: esta área contém a barra de cores com as informações pertinentes ao dado sendo visualizado no momento. Também é alterada dinamicamente de acordo com a mudança do dado na área 1.

A linguagem visual adotada para a interface seguiu o design da interface dos sistemas Windows Vista e Windows 7, com alguma influência dos sistemas operacionais da Apple (Macintosh).

Todas as barras possuem determinado grau de transparência, de modo a não eliminar totalmente o gráfico 3D que fica no fundo. Principalmente nas barras de info e dados, que ficam diretamente acima da área destinada aos modelos tridimensionais.

As barras de cores presentes na área 6 foram salvas do software Paraview, tiveram seu fundo recortado no Adobe Photoshop e foram salvas no formato .DDS com transparência. Para este item, o pesquisador tentou manter o padrão dos programas científicos aos quais os prováveis usuários do visualizador estão familiarizados: o Paraview e o Ensight [8]. Estes programas permitem que a barra de cor seja reposicionada na tela. Entretanto, ela é posicionada na parte inferior ou à direita das visualizações científicas com freqüência, sejam tridimensionais ou não.

O layout dos botões da barra inferior foi pensado de forma a oferecer o design gráfico mais intuitivo possível, para grandes audiências. Sendo assim, foram usados signos de significado amplamente difundido.

O botão “Olho” foi escolhido para a câmera que circunda ao redor dos dados, como se o espectador estivesse segurando o dado nas suas próprias mãos, observando-o de diferentes ângulos;

O botão “Asa” foi adotado para a câmera fly through (vôo livre) devido à sensação que o uso desta implica – o usuário tem a mesma liberdade que um pássaro – pode voar para qualquer lado em qualquer angulação, com bastante liberdade;

O botão “Óculos” foi produzido com base em uma foto de um óculos anaglifo (vermelho e azul) e as cores já deixam bem claro que se trata de um óculos para visualização em stereo, devido à disseminação desses óculos inclusive entre o público infantil;

O botão “Eixos” também foi desenhado de forma à deixá-lo o mais intuitivo possível, usando como referência o layout dos botões de eixos do próprio Paraview.

Na página a seguir, segue uma figura estendida da interface, de forma a mostrar todos os seus detalhes.

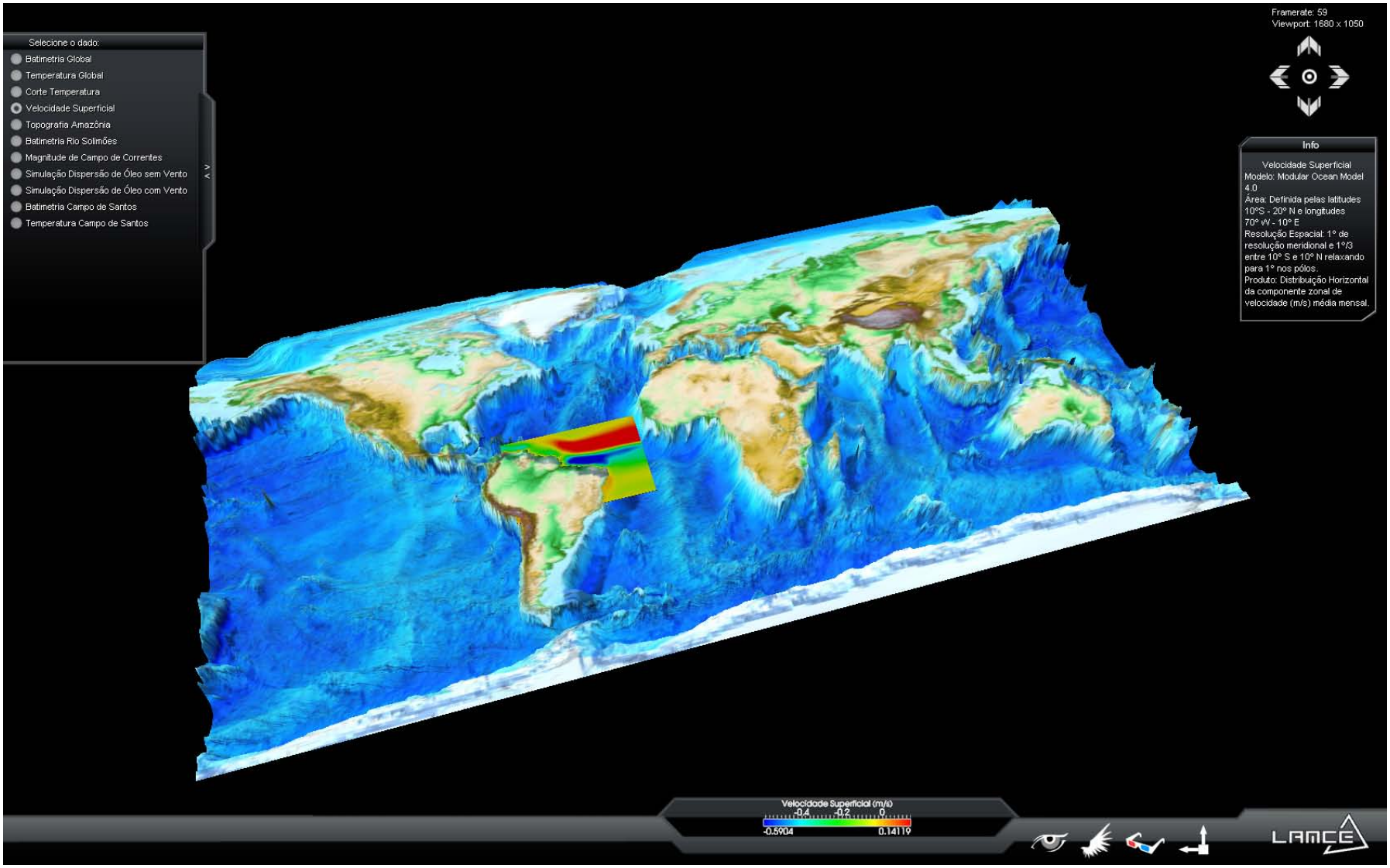

Figura 48 – Visão geral da interface final em funcionamento.

CAPÍTULO 6
RESULTADOS

6.1 – Introdução

Neste capítulo serão descritos os resultados dos diferentes testes realizados com a visualizador de dados tridimensionais, em diferentes fases de seu desenvolvimento.

6.2 – Teste 1

Durante o evento PRH-ANP 10 Anos (evento de comemoração de 10 anos da formação do Programa de Recursos Humanos da ANP), em Setembro de 2009, no Bloco A do Centro de Tecnologia da UFRJ, no Rio de Janeiro, o visualizador teve o seu segundo layout (descrito no capítulo anterior) utilizado para testes de campo no estande do PRH-02.

Para este teste utilizou-se a TV Touch Screen Panasonic de 42” descrita anteriormente neste trabalho. Ela possui apenas um toque e simula o uso de um mouse comum.

Figura 49 – Foto capturada durante o evento PRH-ANP 10 Anos com usuário interagindo com o visualizador tridimensional.

Para que o visualizador funcionasse apropriadamente com a TV Touch, alguns ajustes tiveram de ser feitos, como a velocidade de resposta do mouse.

Uma vez que a resposta quanto ao toque na tela da TV era muito alta, o aplicativo girava em uma velocidade incrível, prejudicando a visualização. Para minorar este efeito, a programação do visualizador teve de ser alterada, aplicando uma grande redução na velocidade de resposta do mouse. Sendo assim, quando o usuário utilizava a tela touch, obtinha-se uma resposta normal, com o aplicativo funcionando em uma velocidade aceitável.

Outro ajuste que teve de ser feito foi quanto o input para rotação dos dados. Inicialmente previsto para funcionar com o botão direito do mouse, este teve que ser alterado para o botão esquerdo, pois a tela touch reconhece apenas um toque. Isso prejudicou um pouco a visualização, pois sempre que o usuário tentava clicar no checkbox com os dados, o dado acabava sendo rotacionado ao mesmo tempo. Apesar deste contratempo, a possibilidade de interação com as mãos, sem nenhum outro dispositivo atraiu a atenção de muitas pessoas que passaram pelo evento, criando uma experiência favorável com o público.

6.3 – Teste 2

Este teste também foi realizado com o layout 2 do visualizador, porém utilizando o joystick sem fio como interface de entrada de dados. Este teste foi realizado durante a defesa de um artigo referente a esta dissertação, apresentado no 30º CILAMCE, ocorrido em Novembro de 2009, em Búzios, Rio de Janeiro [41].

O uso do joystick chamou muita atenção, principalmente pois os pesquisadores presentes na platéia durante a demonstração pertenciam à área de Modelagem Computacional Ambiental e ficaram surpresos com a qualidade da visualização e suas possibilidades de interação.

Porém, nehum membro da platéia se prontificou a utilizar o controle, quando oferecido.

6.4 – Teste 3

Este teste foi realizado durante o evento DEFENCIL – 5º Seminário Internacional de Defesa Civil, também ocorrido em Novembro de 2009, no bairro Anhembi, São Paulo. O tema deste evento foi o mapeamento de áreas de risco, assim a Petrobrás cedeu uma área do seu estande para o NUMA – LAMCE apresentar suas propostas e exibir o visualizador de dados tridimensionais.

Para este evento, foi utilizada uma versão do visualizador que operava tanto na tela touch como no joystick sem fio.

Figura 50 – foto do evento mostrando o stand da Petrobrás, com seta destacando a TV Touch utilizada pela equipe do NUMA-LAMCE.

Contudo, ao fornecer as duas opções de interação, o público optou sempre pela tela touch. Ainda assim, um aviso dizendo “Tela Touch – clique para interagir” teve de ser pendurado ao lado da TV para chamar a atenção do público.

CAPÍTULO 7
CONCLUSÃO E CONSIDERAÇÕES FINAIS

7.1 – Conclusão

O visualizador obteve resultados satisfatórios, alcançando todos os objetivos esperados. Ele possibilita a visualização dos dados tridimensionais referentes à modelagem computacional ambiental em real-time, provendo maior interatividade comparado a gráficos bidimensionais e facilitando a difusão dos dados científicos. Ele congrega dados transientes e não transientes, ocupa um espaço em disco reduzido - apenas trinta e cinco megabytes (35MB) - facilitando sua portabilidade (pode ser facilmente transportado em um pen drive) e disponibilização online (através do download do executável), além de servir como base para todos os resultados de modelos computacionais ambientais gerados pela equipe do NUMA-LAMCE.

Apesar dos computadores estarem cada vez mais potentes, suportando dados muito mais complexos, o escopo desta dissertação focou em criar um produto que funcionasse tanto em computadores de baixo desempenho quanto em máquinas com excelente hardware, permitindo a difusão do visualizador.

Com relação aos testes realizados, os resultados foram positivos em relação ao produto, necessitando apenas de correções no que diz respeito à interface gráfica, problema este que foi sanado na versão final do visualizador.

Cabe ressaltar que a metodologia empregada neste trabalho pode ser expandida para qualquer tipo de dado científico, não somente dados ambientais, mas também dados relacionados a medicina, dutos, engenharia naval, dentre outros. A capacidade das *engines* de visualização tridimensional em tempo real tem crescido vertiginosamente nas últimas décadas, permitindo gráficos cada vez mais detalhados e próximos da realidade. A função deste trabalho foi aproveitar a capacidade que essas *engines* oferecem para criar um visualizador de dados científicos tridimensionais portátil, interativo e em *stereo*.

7.2 – Implementações futuras

Pretende-se disponibilizar o visualizador na internet de forma interativa, dentro do próprio site do NUMA-LAMCE e divulgá-la no meio acadêmico, para que outros profissionais contribuam com dados para alimentá-la.

Caso esta estratégia funcione, a programação do visualizador será alterada para que funcione como cliente e servidor: os usuários baixariam uma versão cliente do

programa e os dados extras ficariam em um banco de dados online (servidor), que a versão cliente poderia acessar, funcionando da mesma forma que o Google Earth.

Serão realizadas interações com os usuário finais do visualizador, de forma a aferir a qualidade do visualizador.

Como objetivo chave, pretende-se diminuir o grau de comprometimento de ferramentas para pré-processar os dados - reduzir a quantidade de etapas do tratamento dos dados.

Também se estuda a possibilidade de visualização de geometrias transientes (água se chocando com uma coluna, por exemplo), algo que é perfeitamente possível na *engine* utilizada, porém não foi estudado neste trabalho, uma vez que os dados do NUMA-LAMCE não continham geometrias transientes não-estruturadas.

Outra proposta de implementação futura é a de projetar a grade batimétrica e todos os dados em projeção esférica, garantindo uma visualização do planeta Terra mais próxima da realidade, semelhante ao Google Earth.

APÊNDICE

Rotina em FORTRAN

```
      PROGRAM SIG2Z2VTK
      PARAMETER(IM=289,JM=193,KB=30,KZ=19)

          real NZ(KZ),ZZ(KB),H(IM,JM),FSM(IM,JM),TLEV(IM,JM)
          real xgrid(im,jm),ygrid(im,jm),dxgrid(im,jm),dygrid(im,jm)
          real dx(im,jm),dy(im,jm),latmed,exvert
          real T(IM,JM,KB),S(IM,JM,KB),V(IM,JM,KB),U(IM,JM,KB)
          real SZ(IM,JM,KZ),VZ(IM,JM,KZ),TZ(IM,JM,KZ),UZ(IM,JM,KZ)
          integer irec30,irec31,irec32,irec33,irec40,irec41,irec42,irec43
          integer ptcont,pto(im,jm,kz)
          character*4 param

c     Arquivos binários com os campos de interesse
      OPEN(UNIT=30,FILE='TTano_campsant.bin',FORM='UNFORMATTED',
     1    ACCESS='DIRECT',RECL=IM*1)
      OPEN(UNIT=31,FILE='SSano_campsant.bin',FORM='UNFORMATTED',
     1    ACCESS='DIRECT',RECL=IM*1)
      OPEN(UNIT=32,FILE='UUano_campsant.bin',FORM='UNFORMATTED',
     1    ACCESS='DIRECT',RECL=IM*1)
      OPEN(UNIT=33,FILE='VVano_campsant.bin',FORM='UNFORMATTED',
     1    ACCESS='DIRECT',RECL=IM*1)



c     Arquivo com a batimetria
      OPEN(55,FILE='batgau_prev_campsant.dat')

c     Arquivo de saída para o paraview
      OPEN(1,FILE='bat_campsant.vtk')

c     Arquivos de latitude e longitude
      open(2,file='xgrid_prev_campsant.dat')
      open(3,file='ygrid_prev_campsant.dat')
```

```
C**********************************************************************
      param='TEMP'

c     Niveis sigma
      data ZZ/-0.001,-0.004,-0.007,-0.015,-0.030,-0.060,-0.100,-0.140,
     $       -0.180,-0.220,-0.260,-0.300,-0.340,-0.380,-0.420,-0.460,
     $       -0.500,-0.540,-0.580,-0.620,-0.660,-0.700,-0.740,-0.780,
     $       -0.820,-0.860,-0.900,-0.940,-0.980,-1.020/


c     Niveis Z de interesse
      data nz/0.,-10.,-20.,-50.,-100.,-150.,-200.,-300.,-400.,-500.,
     $       -750.,-1000.,-1250.,-1500.,-2000.,-2500.,-3000.,-4000.,
     $       -5000./

C**********************************************************************
      pi=atan(1.e0)*4.e0    ! PI

      do j=1,jm
      READ(2,50) (dx(i,j),i=1,im)
      READ(3,50) (dy(i,j),i=1,im)
          ENDDO

   50 format(289(1x,e15.7))

      DO J=1,JM
        READ(55,*)(H(I,J),I=1,IM)
      END DO

          tbias=10.
          sbias=35.

      excess=1.

      DO 30 J=1,JM
      DO 30 I=1,IM
```

```
      FSM(I,J)=1.E0
      IF(H(I,J).GT.EXCESS) GO TO 30
      FSM(I,J)=0.E0
 30   CONTINUE

c     Leitura dos dados
      irec30=1
      irec31=1
      irec32=1
      irec33=1

      DO K=1,Kb
         DO J=1,JM
         READ(30,REC=IREC30) (T(I,J,K),I=1,IM)
         IREC30=IREC30+1
         END DO
      END DO
      DO K=1,Kb
         DO J=1,JM
          READ(31,REC=IREC31) (S(I,J,K),I=1,IM)
          IREC31=IREC31+1
         END DO
      END DO
      DO K=1,Kb
         DO J=1,JM
          READ(32,REC=IREC32) (U(I,J,K),I=1,IM)
          IREC32=IREC32+1
         END DO
      END DO
      DO K=1,Kb
         DO J=1,JM
          READ(33,REC=IREC33) (V(I,J,K),I=1,IM)
          IREC33=IREC33+1
         END DO
      END DO

      DO K=1,Kb
```

```
      DO J=1,JM
            DO I=1,IM
               T(I,J,K)=T(I,J,K)+TBIAS
               S(I,J,K)=S(I,J,K)+SBIAS
            ENDDO
         ENDDO
       ENDDO

   DO K=1,Kb
      DO J=1,JM
            T(1,j,k)=T(2,j,k)
            T(im,j,k)=T(im-1,j,k)
            S(1,j,k)=S(2,j,k)
            S(im,j,k)=S(im-1,j,k)
            U(1,j,k)=U(2,j,k)
            U(im,j,k)=U(im-1,j,k)
            V(1,j,k)=V(2,j,k)
            V(im,j,k)=V(im-1,j,k)
         ENDDO
       ENDDO

   DO K=1,Kb
      DO I=1,IM
            T(i,jm,k)=T(i,jm-1,k)
            S(i,jm,k)=S(i,jm-1,k)
            U(i,jm,k)=U(i,jm-1,k)
            V(i,jm,k)=V(i,jm-1,k)
         ENDDO
       ENDDO

C     Conversão de sigma para Z

c ZLEV eh o nivel z desejado

   DO K=1,Kz

      zlev=nz(k)
```

```
                CALL SIGTOZ(ZZ,H,T,TLEV,ZLEV,FSM,IM,JM,KB)
      do i=1,im
                do j=1,jm
                TZ(i,j,K)=TLEV(i,j)
    end do
  end do

                CALL SIGTOZ(ZZ,H,S,TLEV,ZLEV,FSM,IM,JM,KB)
      do i=1,im
                do j=1,jm
                SZ(i,j,K)=TLEV(i,j)
    end do
  end do

                CALL SIGTOZ(ZZ,H,V,TLEV,ZLEV,FSM,IM,JM,KB)
      do i=1,im
                do j=1,jm
                VZ(i,j,K)=TLEV(i,j)
    end do
  end do

                CALL SIGTOZ(ZZ,H,U,TLEV,ZLEV,FSM,IM,JM,KB)
      do i=1,im
                do j=1,jm
                UZ(i,j,K)=TLEV(i,j)
    end do
  end do

      END DO

C flags para a batimetria

  DO K=1,Kz
   DO J=1,JM
    DO I=1,IM
     IF(H(I,J).LE.1.01.or.h(i,j).le.abs(nz(k))) THEN
```

```
c            SZ(I,J,K)=-9999.
c                     VZ(I,J,K)=-9999.
c                     UZ(I,J,K)=-9999.
c                     TZ(I,J,K)=-9999.
           SZ(I,J,K)=0.
                     VZ(I,J,K)=0.
                     UZ(I,J,K)=0.
                     TZ(I,J,K)=0.
                     END IF
         END DO
        END DO
       END DO


c     Escrevendo o vtk
          write(1,1) '# vtk DataFile Version 2.0'
          write(1,2) 'grva/lamce/coppe/ufrj'
          write(1,3) 'ASCII'
          write(1,4) 'DATASET RECTILINEAR_GRID'
          write(1,5) 'DIMENSIONS',im,jm,1
c         write(1,5) 'DIMENSIONS',im,jm,kz

       exvert=0.0005

          write(1,6) 'X_COORDINATES',im,'float'
       write(1,7) (dx(i,1),i=1,im)
          write(1,6) 'Y_COORDINATES',jm,'float'
       write(1,8) (dy(1,j),j=1,jm)
c         write(1,6) 'Z_COORDINATES',kz,'float'
          write(1,6) 'Z_COORDINATES',1,'float'
c      write(1,9) (nz(k),k=1,kz)
       write(1,9) nz(1)
c      write(1,10) 'POINT_DATA',im*jm*kz
       write(1,10) 'POINT_DATA',im*jm
          write(1,11) 'SCALARS',param,'float',1
       write(1,12) 'LOOKUP_TABLE default'
```

```
c      do k=1,kz
c         do j=1,jm
c         write(1,14) (TZ(i,j,k), i=1,im)
c         enddo
c         enddo

          do j=1,jm
       write(1,20) (-h(i,j)*exvert,i=1,im)
          enddo


    1 format(A26)
    2 format(A21)
    3 format(A5)
    4 format(A24)
    5 format(A10,3(x,I3))
    6 format(A13,x,(I3,x),A5)
    7 format(289(e15.8,x))
    8 format(193(e15.8,x))
c   9 format(19(e15.8,x))
    9 format(e15.8,x)
   10 format(A10,I8)
   11 format(A7,x,A4,x,A5,x,I1)
   12 format(A20)
c  14 format(289(f6.2,x))
   20 format(289(f7.4,x))

      END


      SUBROUTINE SIGTOZ(ZZ,H,T,TLEV,ZLEV,FSM,IM,JM,KB)
C-----------------------------------------------------------------
C     THIS ROUTINE LINERLY INTERPOLATES TLEV AT THE LEVEL,
C     ZLEV, FROM T LOCATED ON SIGMA LEVELS, ZZ.
C     ZLEV AND ZZ ARE NEGATIVE QUANTITIES.
C     A SEARCH IS MADE TO FIND ZZ(K)*H AND ZZ(K+1)*H WHICH
C     BRACKETS ZLEV; THEN THE INTERPOLATION IS MADE.
```

```
C     IN THE REGION < 0 and > ZZ(1) AND,  IN THE REGION,
C     < ZZ(KB-1) and > -1, DATA IS EXTRAPOLATED.
C     NOTE THAT A NEW MASK ,FSM, IS CREATED.
C
C     THE VALUES OF H SUPPLIED TO THIS SUBROUTINE SHOULD BE
C     APPROPRIATE TO THE VARIABLE T. FOR EXAMPLE, IF T IS THE
C     X-COMPONENT OF VELOCITY, H SHOULD THE AVERAGE OF DEPTH
C     AT I AND I-1.
C-----------------------------------------------------------------
      DIMENSION ZZ(KB),H(IM,JM),T(IM,JM,KB),TLEV(IM,JM),FSM(IM,JM)

C
      DO 200 J=2,JM-1
      DO 200 I=2,IM-1
      IF(ZLEV.GT.ZZ(1)*H(I,J)) THEN
           K=1
           TLEV(I,J)=T(I,J,K)+(ZLEV-ZZ(K)*H(I,J))
     1        *(T(I,J,K+1)-T(I,J,K))/((ZZ(K+1)-ZZ(K))*H(I,J))
           GO TO 200
           FSM(I,J)=1.
      ENDIF
      K=1
  100 CONTINUE
C
C
      IF(ZLEV.LE.(ZZ(K)*H(I,J)).AND.ZLEV.GE.(ZZ(K+1)*H(I,J))) THEN
            TLEV(I,J)=T(I,J,K)+(ZLEV-ZZ(K)*H(I,J))
     1         *(T(I,J,K+1)-T(I,J,K))/((ZZ(K+1)-ZZ(K))*H(I,J))
            FSM(I,J)=1.
            GO TO 200
      ELSE
            K=K+1
            IF(K.LT.(KB-1)) GO TO 100
      ENDIF
      IF(ZLEV.LE.(ZZ(K)*H(I,J)).AND.ZLEV.GE.(-H(I,J))) THEN
             TLEV(I,J)=T(I,J,K-1)+(ZLEV-ZZ(K-1)*H(I,J))
     1          *(T(I,J,K)-T(I,J,K-1))/((ZZ(K)-ZZ(K-1))*H(I,J))
```

```
                FSM(I,J)=1.
      ENDIF
  200 CONTINUE
      RETURN
      END
C
```

REFERÊNCIAS BIBLIOGRÁFICAS

[1] JUNIOR, A.R.T., SILVA, M.P.R., SILVA, R.M., " PIATAM - Uma Parceria entre a Petrobrás e Instituições de Pesquisa para o Aumento do Entendimento das Questões Ambientais na Amazônia", ***Boletim da Sociedade Brasileira de Meteorologia***, v.31, n. 2-3, pp. 44-50, Ago - Dez. 2007

[2] PACANOWSKY, R.C. & S.M.GRIFFIES. 1999. "*The MOM3 Manual*". Geophysical Fluid Dynamics laboratory/NOAA, Princeton, USA, p. 680.

[3] MELLOR, G.L., *User guide for a three-dimensional, primitive equation, numerical ocean model.* 53 pp., Pro. in Atmos. and Ocan. Sci, Princeton University, Junho, 2003. Disponível em: <http://www.aos.princeton.edu/WWWPUBLIC/htdocs.pom/PubOnLine/POL.html#USERS_GUIDE> . Acesso em: 31/08/2010.

[4] NICOIL. Disponível em: < http://www.lamma.ufrj.br/lamma/pesquisa/oceanografia/pesquisa/transp_pol.htm>. Acesso em: 31/08/2010.

[5] NETTO, A. V.; MACHADO, L. S.; OLIVEIRA, M. C. F. *Realidade Virtual: fundamentos e aplicações*. Visual Books, 2002.

[6] INFOPAE
Disponível em : <http://www.k2sistemas.com.br/projetos/infopae.php>. Acesso em: 31/08/2010.

[7] NETTO, A. V., GOUVEIA, J. D., CATERIANO P. S. H. "Interface 3D para manipulação de dados em redes de distribuição de energia elétrica", ***INFOCOMP Journal of Computer Science***, Lavras, v. 4, n. 4, p. 73-81, 2005

[8] EnSight User Manual for Version 9.1. Disponível em: < http://www.ensight.com/9.1-Manuals/View-category.html >. Acesso em : 31/08/2010.

[9] PDF3D Overview. Disponível em: < http://www.pdf3d.co.uk/pdf3d_overview.php >. Acesso em: 31/08/2010.

[10] ANG, C.S., MARTIN, D.C., DOYLE, M.D., "Integrated control of distributed volume visualization through the World Wide Web", *Proc. IEEE Visualization' 94*. pp.13-20, Washingtom D.C., Outubro, 1994.

[11] LIU, P.W., CHEN, L.S., CHEN, S.C., CHEN, J.P., LIN, F.Y., HWANG, S.S., "Distributed computing: new power for scientific visualization", *IEEE Computer Graphics and Applications* Volume 16 Edição 3, pp.42-51, Março, 1996.

[12] ALVES, A.D., OLIVEIRA, M.C.F., MINGHIM, R., NONATO, L.G., "Interactive visualization over the WWW", *XIII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI' 00)*, pp.259, Outubro, 2000.

[13] BETHEL, E.W., CHEN, J., YOON, I., "Interactive, Internet Delivery of Visualization via Structured, Prerendered Imagery." *Two-page research highlight. LBNL-63674*. Lawrence Berkeley National Laboratory, 2007.

[14] CHEN, J., YOON, I., BETHEL, E.W., "Interactive, Internet Delivery of Visualization via Structured, Prerendered Multiresolution Imagery." *IEEE Transactions on Visualization and Computer Graphics*, Volume 14, Número 2, Março / Abril, 2008.

[15] BURIOL, T.M., Processamento e Visualização de Campos em Ambientes Virtuais e Sistemas CAD 3D Aplicados a Projetos de Iluminação em Subestações. Dissertação de M.Sc., Universidade Federal do Paraná, Curitiba, Brasil, 2006.

[16] Google Earth. Disponível em: <http://earth.google.com/>. Acesso em: 31/08/2010.

[17] MIRANDA, J. I. *Diretivas para disponibilizar mapas na Internet*. Campinas: Embrapa Informática Agropecuária,2002. páginas: 29 p. (Embrapa Informática Agropecuária. Documentos, 14). Disponível em: <http://www.cnptia.embrapa.br/modules/tinycontent3/content/2002/doc14>. Acesso em: 04 setembro 2009.

[18] Papervision3D Documentation. Disponível em: < http://papervision3d.googlecode.com/svn/trunk/as3/trunk/docs/index.html >. Acesso em: 31/08/2010.

[19] Unity3D User Manual. Disponível em: < http://unity3d.com/support/documentation/Manual/index.html>. Acesso em: 31/08/2010.

[20] Quest3D Reference Manual. Disponível em:
< http://support.quest3d.com/index.php?title=Reference_Manual>. Acesso em: 31/08/2010.

[21] Modelagem Computacional Ambiental. Disponível em:
< http://pt.wikipedia.org/wiki/Modelagem_computacional>. Acesso em: 31/08/2010.

[22] CUNHA, R.D., *Introdução à Linguagem de Programação Fortran 90* . Editora da UFRGS, Porto Alegre, 2005.

[23] *VTK User's Guide*. Kitware, Inc. 11ª Edição. Março, 2010.

[24] Paraview User's Guide (Version 1.6). Disponível em:
< http://www.paraview.org/files/v1.6/ParaViewUsersGuide.PDF>. Acesso em: 31/08/2010.

[25] CAREY., R., BELL, G. *The Annotaded VRML97 Reference Manual.* 1997. Disponível em: <
http://www.few.vu.nl/~eliens/documents/vrml/reference/BOOK.HTM>
Acesso em: 31/08/2010.

[26] Deep Exploration User Reference. Disponível em:
< http://www.righthemisphere.com/_base/static/files/manuals-tutorials/user_ref_de_6.0.pdf>. Acesso em: 31/08/2010.

[27] 3Ds Max 2011 Help. Disponível em:
< http://images.autodesk.com/adsk/files/3ds_max_2011_help.pdf>. Acesso em: 31/08/2010.

[28] ZBrush 4 Getting Started Guide. Disponível em:
< http://download.pixologic01.com/download.php?f=Plugins/zbrush4-PDF/ZBrush_Starting_Guide.pdf.zip>. Acesso em: 31/08/2010.

[29] Adobe After Affects. Disponível em:
< http://www.adobe.com/products/aftereffects/>. Acesso em: 31/08/2010

[30] Adobe Photoshop. Disponível em:
< http://www.adobe.com/br/products/photoshop/photoshop/>. Acesso em: 31/08/2010

[31] ARNAUD, R., BARNES, M., *COLLADA - Sailing the Gulf of 3D Digital Content Creation*. AK Peters. 2006.

[32] SMITH, W. H. F. and Sandwell, D. T. (1997). *Global seafloor topography from satellite altimetry and ship depth soundings*. *Science*, 277:195-196. Disponível em: http://topex.ucsd.edu/marine_topo/mar_topo.html. Acesso em 10 de Janeiro de 2010.

[33] ASSAD, L. P. F. *Influência do campo de vento anômalo tipo ENSO na Dinâmica do Atlântico Sul*. 2006. Tese (Doutorado em Engenharia Civil) – Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia, COPPE, Rio de Janeiro, RJ, Brasil, 2006.

[34] ETOPO 5. Disponível em:
< http://www.ngdc.noaa.gov/mgg/global/etopo5.HTML>. Acesso em: 31/08/2010

[35] MEHDI, N., Assad, L. P. F., Torres Junior, A. R., Landau, L. *Modelagem Hidrodinâmica no Rio Solimões*. In: *1°Congresso Internacional do PIATAM: Ambiente, Homem, Gás e Petróleo*, Manaus, Brasil, 11 a 15 de dezembro de 2005.

[36] ETOPO 2. Disponível em:
< http://www.ngdc.noaa.gov/mgg/image/2minrelief.html>. Acesso em: 31/08/2010

[37] MULLEN, T. *Mastering Blender*, Primeira edição, Indianapolis, Indiana: Wiley Publishing, Inc., 2009

[38] FERNANDO, R., KILGARD, M., *The Cg Tutorial: The Definitive Guide to Programmable Real-Time Graphics*, Addison-Wesley Professional, 2003.

[39] FILHO, G., *Gestalt do Objeto - Sistema de Leitura Visual da Forma,* Editora Escrituras, 2008.

[40] MAGALHÃES, D.F.C., CAZANGI, G.R., CAMARGO, E.A.A., CABRAL, C.M.S., "Estudo do Consumo Energético de Monitores com Fundo de Tela Preto". ***Revista Ciências do Ambiente On-Line***, Volume 5, Número 1, Julho, 2009.

[41] MEDEIROS, R.L.;LANDAU, L.; ASSAD, L.P.F., "Plataforma de Visualização Científica em Três Dimensões", *30º CONGRESSO IBERO-LATINO-AMERICANO DE MÉTODOS COMPUTACIONAIS EM ENGENHARIA*, Búzios, Brasil, Novembro de 2009.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo