

# Utilização de Ferramentas Multimídia para a Construção de Simuladores de Fenômenos Físicos

(Use of Multimedia Tools to Construction of Physics Phenomena Simulators)

Antonio Luiz Almeida<sup>1</sup> e Murilo Freire Oliveira Araújo\*

Universidade do Estado da Bahia, Salvador, BA, Brasil

\*Aluno do curso de Análise de Sistemas (UNEB)

O uso de computadores no processo de ensino-aprendizagem vem se tornando cada vez mais comum nas salas de aula. Entre jogos, simuladores e ambientes de ensino, conta-se hoje com vários *software* que trazem a promessa de tornar as aulas dinâmicas e interativas. Existem ferramentas muito completas, que englobam várias áreas do saber, como o *Educandus*, que oferece um conteúdo rico para o ensino de matemática e ciências. Em Física, pode-se citar o *Easy Java Simulations*, um *Applet* desenvolvido para fornecer suporte à construção de simuladores físicos. Entretanto, os recursos de *software* multimídia ainda são pouco explorados no desenvolvimento destas ferramentas de ensino, principalmente os recursos gráficos. Este trabalho objetiva demonstrar que diversas linguagens de programação e ferramentas utilizadas hoje para a mídia de entretenimento podem ser direcionadas para a construção de ambientes de ensino-aprendizagem o que constrói, ao final do processo, um produto muito mais robusto e eficiente.

The use of computers in process of teach-learning has becoming each time more common in classrooms. Among games, simulators a teach environments, we count today with a lot of software that bring the promise of turn the classes dynamic and interactive. There is tools much complete, that circle a lot of known areas, like the *Educandus*, that offers a rich content for math and science teaching. In Physics, we can quote the *Easy Java Simulations*, an *Applet* developed to provide support to physics simulators construction. Meanwhile, the resources of multimedia software are still few explored on development of these teaching tools, mostly the graphic resources. This work objects to show that a lot of programming languages and tools used today by the entertainment media can be redirected to the building of teach-learning environments what makes, at the end of process, a product more robust and efficient.

Palavras-Chave: Ensino de Física, Simulador, Software, Java, Applet, Multimídia.

*Key-Words: Physics teaching, Simulator, Software, Java, Applet, Multimedia*

---

<sup>1</sup> Email: sos.sepetiba@gmail.com

# 1. INTRODUÇÃO

## 1.1. Uso de Simuladores nas Salas de Aula

A utilização de simuladores em salas de aula torna-se uma prática cada vez mais comum atualmente. Por simuladores, entende-se ferramentas para o estudo do comportamento de sistemas reais através do exercício de modelos. Em relação ao software, esta representação muitas vezes é mostrada através de gráficos, desenhos e esquemas.

Entre as vantagens do uso de simuladores pode-se citar a representação de experimentos que, se realizados sobre o sistema real, poderiam consumir anos; a redução de custos que, mesmo exigindo recursos humanos e computacionais, ainda se mantém muito abaixo dos gastos com experimentos sobre o sistema real; representação de situações onde não é possível desenvolver as experiências sobre o sistema real em função de questões de segurança, tempo ou acesso; formas diversificadas de demonstração do estado do sistema e do resultado dos experimentos em prol dos recursos gráficos – tabelas, gráficos, esquemas, entre outros – oferecidos pelos computadores; e, de suma importância, possibilidade de observar as variações do sistema em função da alteração dos dados de entrada.

Observa-se então que o uso destas ferramentas pelo educador contribui para a clareza, dinâmica e interatividade do conteúdo. Este fato também se apóia na aceitação da maior parte dos alunos pela utilização dos computadores em sala de aula como ferramentas de auxílio à aprendizagem. Através destas informações pode-se afirmar que os simuladores devem representar com eficiência muitas das características dos sistemas reais estudados e evitar ao máximo elevadas discrepâncias entre os resultados dos experimentos simulados e os resultados dos experimentos reais, porém, algumas vezes, a excessiva abstração do sistema real

durante a elaboração do projeto do simulador pode prejudicar o desempenho deste como ferramenta de demonstração de um sistema real. É claro que, sendo o simulador construído a partir de um modelo extraído do sistema real, não se deve esperar que o primeiro represente todas as características deste, principalmente em função da complexidade que um sistema real possa apresentar.

Além disso, como fator de importância, as ferramentas de simulação devem prover representações gráficas claras e objetivas, possuindo também uma interface intuitiva para que a utilização da ferramenta, por parte dos alunos, não se torne um empecilho para a classe.

No item dois serão abordados problemas e limitações observados nos simuladores, enfatizando pontos pertinentes aos recursos gráficos, desempenho e detalhamento do modelo. A seguir, no item três, serão tratados aspectos referentes à construção de simuladores tendo em vista o *Java*, que é a linguagem mais utilizada hoje para a construção destes, além do enfoque em ferramentas e linguagens de programação multimídia que podem ser direcionadas para esta área. As aplicações destas linguagens em simuladores de sistemas físicos serão apresentadas através de quatro estudos de caso, com a finalidade de mostrar as facilidades obtidas através da escolha de ferramentas apropriadas para projetos distintos.

## 2. LIMITAÇÕES DOS SIMULADORES

As limitações dos simuladores podem ser observadas conforme as características de cada projeto. Quando existe a necessidade de processamento de complexas equações matemáticas ou análise de uma massiva quantidade de dados, observam-se problemas referentes à capacidade de processamento dos computadores (memória *RAM* ou

capacidade do processador insuficientes). Quando executa-se um simulador que exige alta capacidade de processamento em microcomputadores inadequados, observam-se problemas comuns ao baixo desempenho como lentidão no tempo de resposta e travamento do sistema. Isso pode comprometer a exposição do conteúdo em sala de aula e gerar desinteresse nos alunos.

Em relação aos quesitos gráficos, a forma de apresentação do conteúdo, assim como o design da interface, podem influenciar no aprendizado tanto oferecendo estímulo ao uso da ferramenta como obtendo o efeito inverso ao esperado. Alguns pontos observados foram sugestões e críticas dos alunos em relação à melhoria na interface das aplicações [1]. Apesar dos avanços na área multimídia – tratamento de som, imagem e vídeo - ainda pode-se encontrar simuladores com capacidade gráfica inexplorada além de interfaces pouco intuitivas.

Uma outra questão importante a ser abordada é a abstração do conteúdo para a construção do simulador. Tendo em vista as informações que podem ser obtidas através de experimentos reais e que os simuladores propõem a reprodução destes experimentos em ambiente virtual faz-se necessário, na maioria dos casos, limitações no conteúdo para o projeto. Isso limita o estudo do ambiente proposto através do simulador, uma vez que os resultados apresentados podem não ser compatíveis com os resultados reais. Tais abstrações em excesso ou definidas sem o devido cuidado, podem contribuir para a construção de um conteúdo deformado, não só dificultando o aprendizado dos alunos como construindo perspectivas errôneas do conteúdo [2]. Em muitos casos, as abstrações estão ligadas às duas questões citadas acima.

### 3. CONSTRUÇÃO DE SIMULADORES

A qualidade gráfica, a capacidade de representação dos fenômenos e o desempenho possuem um papel fundamental na decisão de uso de um simulador pelos usuários. Logo, através da escolha de uma ferramenta de construção – linguagens de programação, aplicativos gráficos ou ferramentas de modelagem – adequada, pode-se atender estas três necessidades de um modo competente possibilitando o desenvolvimento de um projeto melhor contextualizado, mais lúdico e com maior possibilidade elucidativas de apreciação e exame.

#### 3.1. Linguagem Java

O desenvolvimento de simuladores apresenta hoje abundantes exemplos de *software* desenvolvidos utilizando a linguagem *Java*, mais precisamente os *Applets* [3]. Entre esses aplicativos observamos o Laboratório Virtual de Computação Natural (<http://lsin.unisantos.br/lvcon/>), que utiliza *Applets* para a simulação de fenômenos da Computação Natural<sup>2</sup>, e o *Easy Java Simulations* [4], que provê um mecanismo de criação de simuladores de Física em *Java*.

Um *Applet* é um programa escrito na linguagem *Java* que é executado dentro de uma página *HTML*<sup>3</sup>, da mesma forma que uma imagem é incluída nesta mesma página. A tecnologia *Java* [5] surgiu em 1991, criada pela *Sun Microsystems* [6], e sua promessa era prover uma plataforma eficiente de desenvolvimento de *software* para dispositivos portáteis. De fato, sua

---

<sup>2</sup>A Computação Natural pode ser vista como uma versão computacional do processo de extração de idéias da natureza para o desenvolvimento de sistemas "artificiais", ou então a utilização de materiais e mecanismos naturais para realizar computação

<sup>3</sup>Hiper Text Markup Language – Linguagem de formatação de hiper-texto padrão na Internet usada para a construção de páginas.

portabilidade<sup>4</sup> contribuiu para o sucesso desta linguagem, mas foi o conjunto de todas as suas características - entre estas os *Applets* - que proporcionaram a feroz ascendência do *Java* no mercado de desenvolvimento de *software*. Assim, pode-se encontrar uma documentação abrangente para esta linguagem em forma de livros, artigos e tutorias, além do manual no sítio do seu desenvolvedor ([www.sun.com](http://www.sun.com)) e muitos tópicos em fóruns de discussão.

A linguagem *Java*, por apresentar um código mais simples em relação a outras linguagens utilizadas até então, como o *C++*, tornou-se preferida entre muitos programadores. Desta maneira, a construção de simuladores seguiu este mesmo rumo. Todavia, a compreensão da metodologia de desenvolvimento de *software* em *Java* exige qualificação e, quando o projetista não é um profissional de tecnologia da informação, a concepção do projeto pode ficar restrita devido a limitações da linguagem ou desconhecimento das possibilidades desta. Tais limitações de projeto mostram-se graficamente ou estruturalmente nos simuladores. É possível visualizar isso, dentre outras situações, através de um pesquisador que limita seu projeto a um ambiente bidimensional, imaginando dificuldades em relação ao desenvolvimento em três dimensões. Este também sentiria-se desestimulado a explorar recursos multimídia mais elaborados como criação de vídeos e animações. Apesar do *Java* oferecer uma plataforma para desenvolvimento tridimensional, o *Java3D* [7], é exigido que o programador tenha conhecimentos muito sólidos em *Java* e em computação gráfica. Assume-se então que um pesquisador, conhecendo a existência de outras ferramentas e linguagens de programação, possa ter uma visão mais

clara das possibilidades do seu projeto de simulador, menos abstrações neste e possa obter um desenvolvimento de *software* mais eficiente. Hoje, muitas destas ferramentas e linguagens são bastante utilizadas no desenvolvimento multimídia em vários setores, como jogos, onde as características assemelham-se às de construção de simuladores.

### **3.2. Ferramentas e Linguagens Multimídia**

Através das características dos simuladores mostradas nos itens anteriores, nota-se que o desenvolvimento destes pode estar incluso diretamente na área multimídia, neste caso apresentada com foco na computação gráfica. Esta área, devido à sua popularidade e diversidade, apresenta suas próprias ferramentas de desenvolvimento de *software*, cada qual com suas peculiaridades. Apesar do *Java* prover recursos que podem ser usados na computação gráfica, tal linguagem tem um uso muito mais geral, tornando assim o uso de outras linguagens, mais específicas, bastante eficiente em alguns casos. Tais ferramentas e linguagens multimídia são utilizadas em várias outras áreas do conhecimento como Apresentações Gráficas, Entretenimento, Educação, Treinamento e Visualização Científica e poderia-se ter tais recursos facilmente direcionados para a construção de simuladores. Propõe-se então o estudo de ferramentas para a escolha de uma ferramenta adequada a um projeto específico.

Através do estudo de ferramentas podem-se agilizar processos e reduzir custos no projeto de construção de simuladores. Estes ganhos podem ser direcionados para outras etapas, como detalhamento do modelo. Afirma-se então que, se for possível empregar menos esforço no desenho, modelagem gráfica e programação, torna-se viável a construção de modelos mais elaborados, que

---

<sup>4</sup>Uma aplicação feita em *Java* pode ser executada na maioria dos sistemas operacionais existentes e, em alguns casos, em dispositivos móveis como telefones celulares e Palms.

abrangem mais eficientemente as características do projeto.

Basicamente um estudo de ferramenta consiste em determinar qual será a mais eficiente para o desenvolvimento do projeto. Algumas delas apresentam características específicas que podem trazer inúmeros benefícios ao desenvolvimento de *software* gráficos/educativos. Através da escolha de uma ferramenta condizente com o projeto proposto, pode-se agregar mais valor ao produto final, além da possibilidade de exploração de diversos outros recursos computacionais.

Estas definições podem ficar mais claras se observadas através de estudos de caso. Para o estudo destas propostas serão observados alguns dos pontos fundamentais para a análise de requisitos de projetos multimídia, como qualidade gráfica, dimensões dos gráficos (duas ou três dimensões), alguns requisitos da interface (forma como o conteúdo será organizado e exibido) e modo de distribuição do sistema (rede, Internet ou local, como em *CD-ROM*). É através do conjunto destas características que a ferramenta de construção será sugerida.

### 3.3. Estudo de Casos

#### 3.3.1. Proposta I

Construção de um simulador planetário em três dimensões para mostrar os movimentos dos planetas do Sistema Solar em relação ao Sol. O sistema deve prover um ambiente navegável que demonstre a posição dos astros em um determinado período de tempo e será disponibilizado através da Internet.

Tendo em vista as características deste sistema, em relação aos requisitos de *software*, observa-se a necessidade de utilização de uma linguagem que permita esta distribuição na Internet e forneça mecanismos de suporte à construção 3D. Apesar dos *Applets* serem facilmente distribuídos em páginas *HTML*, para a

construção da interface 3D o *Java3D* por ser mais complexo, pode aumentar o prazo e o custo do projeto. Neste caso, a tecnologia *X3D* [8] pode atender esta necessidade devido ao seu bom desempenho em aplicações para Internet, além da simplicidade de sua linguagem de programação. *X3D* é um formato de arquivo padrão, livre e aberto e uma estrutura de tempo de execução para representar e comunicar cenas e objetos 3D em *XML*<sup>5</sup>. Através do *X3D*, podemos construir desde modelos de edifícios e avenidas, até pequenos jogos, com um sistema de navegação que permite o usuário percorrer o ambiente virtual. Uma aplicação desenvolvida com uso do padrão *X3D* pode ser executada nos principais navegadores e sistemas operacionais utilizados atualmente. Pode-se otimizar ainda mais este processo utilizando junto ao *X3D* uma ferramenta de modelagem 3D.

Apesar do *X3D* oferecer por si só os recursos necessários para o desenvolvimento desta aplicação, a utilização de uma ferramenta visual de modelagem, como o *Flux Studio* [9], torna este processo ainda mais simples. O *Flux Studio* é uma aplicação de modelagem e orientação 3D fácil de usar, de propósito geral, orientado visualmente, que cria e exporta conteúdo *web 3D* de tempo-real em *X3D*, *VRML97*<sup>6</sup> e vários outros formatos de arquivos abertos e especiais. Neste caso seria feita a modelagem dos objetos tridimensionais utilizando o *Flux Studio* e estes seriam exportados para o formato *X3D* para uso no ambiente criado com esta tecnologia.

---

<sup>5</sup>Extensible Markup Language (linguagem extensível de formatação) - A linguagem XML é definida como o formato universal para dados estruturados na Internet.

<sup>6</sup>Virtual Reality Modeling Language – Tecnologia que deu origem ao X3D.

### 3.3.2. Proposta II

Construção de um ambiente tridimensional que simule os principais elementos da dinâmica de Newton. Este ambiente deve ser navegável, conter exemplos pré-definidos, mas possibilitar a construção de situações diversas e permitir que os usuários alterem os parâmetros de entrada dos exemplos para acompanhar suas implicações. Um sistema para ser distribuído em *CD-ROM*.

Em uma proposta com tais características averigua-se a necessidade de uso de ferramentas de desenvolvimento e modelagem 3D, como o *Blender* [10] ou *3D Studio Max*[11]. Neste caso, seriam aproveitadas a confiabilidade das aplicações desenvolvidas no *Blender*<sup>7</sup> e as funções de suporte oferecidas por esta ferramenta tanto para recursos gráficos quanto para simulação de Física.

O *Blender* é gratuito, *opensource*<sup>8</sup> e oferece uma suíte completa de ferramentas de modelagem, animação e edição de objetos e cenários 3D, além de um motor de física, o *Ketsji* [12], provendo assim muitas possibilidades para o desenvolvimento de ambientes tridimensionais. Ele oferece suporte a diversas linguagens de programação, entre estas o *C++* - muito utilizado no desenvolvimento de jogos - e o *Python* [13] que permitem a implementação de funcionalidades nos simuladores, como rotinas de cálculo, menus interativos e relatórios sobre estado do sistema.

O motor de física é uma das partes que integram um motor de jogo (*Game Engine*). Um Motor de Jogo é um conjunto de bibliotecas e rotinas que tem a função de facilitar o desenvolvimento de jogos ou

<sup>7</sup>O 3D Studio Max, assim como o Blender, é uma ferramenta de desenvolvimento e modelagem de objetos em três dimensões. Este trabalho teve o enfoque no Blender porque este apresenta licença gratuita, ao contrário do 3D Studio Max.

<sup>8</sup>Uma aplicação *opensource* tem seu código-fonte aberto, normalmente distribuído através de uma licença de software livre, para que qualquer pessoa possa estudar, corrigir ou melhorar este código.

simulações em tempo real. Entre as partes deste, está o motor de renderização, responsável pela aplicação de texturas a gráficos 2D ou 3D e o motor de física, responsável pelos cálculos físicos do ambiente como detecção de colisão e inércia. Ele também inclui recursos adicionais como suporte a animação, sons, inteligência artificial, *networking* (rede) e um *scene graph* (ambiente para integração e organização dos vários elementos da aplicação). Os elementos controlados pelo motor – também chamados de *Game Objects* – comportam-se autonomamente através de um conjunto de ferramentas chamadas *Logicbricks* (conjuntos de funções pré-definidas, como movimentação e som) e propriedades.

### 3.3.3. Proposta III

Construção de um simulador acessado através da Internet que realize operações de cálculo vetorial. Será exibido um plano cartesiano com linhas de grade e o usuário poderá fazer marcações referentes aos vetores a serem calculados e o vetor resultante será desenhado na tela.

Nota-se que o sistema em questão requer poucos recursos gráficos, porém exige um tratamento matemático complexo. Fazendo-se uso das bibliotecas matemáticas *Java*, este sistema molda-se na construção de um *Applet*. Estas bibliotecas matemáticas [14] (*Class Math*) são um conjunto de funcionalidades que possibilitam a execução de operações numéricas diversas como a raiz quadrada, funções trigonométricas, logarítmicas, exponenciais, entre outras. Nesta proposta os recursos gráficos não excedem a exibição do resultado das operações matemáticas. Sendo um *Applet* um aplicativo em Java que pode ser incluso em uma página *HTML*, esta ferramenta atende às necessidades deste projeto.

### 3.3.4. Proposta IV

Construção de um ambiente dinâmico com textos, figuras e animações que se complementam para o aprendizado do conteúdo de magnetismo. O sistema deve conter um menu para navegação que agrupará os sub-temas desta disciplina. Este sistema poderá ser distribuído tanto em *CD-ROM* quanto pela Internet.

Neste exemplo, a ferramenta multimídia sugerida é o *Flash* [15]. Nota-se que este projeto tem baixa complexidade tanto em programação quanto em processamento e requer mais recursos gráficos do que lógica em si, ou seja, através do uso do *Flash*, obtém-se um desenvolvimento eficiente. Esta ferramenta tem sido muito utilizada na Internet para a construção de conteúdo interativo e tem se mostrado bastante competente neste sentido. É um *software* comercial, pertencente hoje à empresa *Adobe* [16] sendo um ambiente para criação de animações que dispõe de uma linguagem de programação própria, o *ActionScript*. Através do *Flash* pode-se criar *websites* interativos, anúncios ricos em mídia, mídias instrutivas, apresentações, jogos, entre outros. Nesta ferramenta são oferecidos recursos de design e animação e, através do *ActionScript*, podem ser implementadas várias outras funcionalidades, atendendo assim as necessidades deste projeto.

### 3.3.5. Análise Comparativa das Propostas

Será analisada agora a utilização das ferramentas mostradas nas propostas anteriores e serão avaliados seus critérios de uso nestes projetos.

Na Proposta I, na qual foi sugerido o uso do *X3D*, caso fosse optado pela utilização do *Blender*, não seria necessária a utilização de uma ferramenta de modelagem a parte – no caso, o *Flux Studio* – uma vez que este oferece recursos próprios de modelagem 3D. Entretanto, obteria-se problemas quanto à forma de distribuição, tendo em vista que o *Blender*, diferente do *X3D*, oferece poucos recursos

para a execução de suas aplicações em páginas da Internet. Nesta proposta, assim como na proposta II, não seria sugerido o uso do *Flash* devido às suas limitações no desenvolvimento de aplicações em três dimensões.

Na proposta seguinte, onde o uso do *Blender* foi proposto, seria possível utilizar tanto o *X3D* quanto o *Java3D* para a construção do gráfico desta, entretanto, o *Blender* traz um recurso nativo essencial neste projeto: o motor de física.

A construção de um motor para as duas outras linguagens poderia inviabilizar o projeto, devido à complexidade de desenvolvimento deste. Além disso, no *Blender*, seria mais fácil promover a integração entre os objetos modelados em três dimensões e o motor em si, uma vez que estes componentes já estão ligados à mesma ferramenta através do *Scene Graph*.

Em relação à terceira proposta, temos necessidade de gráficos simples e funções matemáticas mais elaboradas. Se utilizadas as outras três ferramentas mostradas – *X3D*, *Blender* e *Flash* – seria necessário desenvolver as rotinas de cálculo matemático necessárias, o que aumenta a complexidade do projeto. Além disso, o desenvolvimento de um *Applets* torna muito simples a distribuição posterior deste simulador na Internet.

Na quarta proposta, apenas o *Flash* e o *Java*, entre as ferramentas mostradas, trazem recursos que simplificam a construção de menus e telas de navegação. O *Flash* foi escolhido porque apresenta, como ponto forte deste *software*, exatamente o desenvolvimento deste tipo de aplicativo, diferente do *Java* que tem uso mais geral. Neste caso o ganho de produtividade foi o motivo para a sugestão de uso do *Flash*.

## 4. CONCLUSÕES

Neste trabalho, procurou-se apontar algumas, entre muitas, ferramentas multimídia que podem ser direcionadas

para a construção de simuladores, mostrando-se pontos importantes a serem observados para a escolha de uma ferramenta de desenvolvimento adequada a cada tipo de projeto. Desta maneira, possibilita-se otimizar o processo de construção além de obter um produto final com desempenho mais eficiente, explorar novos recursos gráficos e desenvolver interfaces mais intuitivas. Possibilita-se também elaborar simuladores mais completos, fazendo com que a abstração do sistema real seja feita durante uso do simulador e não em sua construção, contribuindo para a versatilidade de utilização destas ferramentas. Explicitou-se também que a escolha da ferramenta pode estar relacionada a aspectos vitais do projeto, como a viabilidade.

Como fator de relevância agrega-se à área de simuladores novas possibilidades de desenvolvimento, fomentando-se assim a criação de outras ferramentas de simulação, assim como o incentivo para a criação de grupos de pesquisa nesta área, tendo em vista a diversidade dos *software* multimídia para a concepção e uso destas ferramentas.

Como sugestão de trabalhos futuros, poderia-se construir os simuladores propostos neste trabalho além de utilizar os conhecimentos sobre ferramentas multimídia aqui descritos para a elaboração de outros projetos.

## 5. REFERÊNCIAS

- [1] I. Yamamoto e V. B. Barbeta Revista Brasileira de Ensino de Física **23**,222 (2001).
- [2] A. Medeiros, C. F. de Medeiros Revista Brasileira de Ensino de Física **24**, 80 (2002).
- [3] Applets disponível em <http://java.sun.com/applets/> Acesso em 14 abr. 2008.
- [4] J. S. Figueira, Revista Brasileira de Ensino de Física **27**, 613 - 618 (2005).
- [5] Learn about Java Technology disponível em <http://www.java.com/en/about/> Acesso em 16 abr. 2008.
- [6] Sun Microsystems disponível em <http://www.sun.com> Acesso em 15 mai. 2008.
- [7] Java 3D disponível em <https://java3d.dev.java.net/> Acesso em 27 jul. 2008.
- [8] Open Standards for Real-Time 3D Communication disponível em <http://www.web3d.org> Acesso em 15 abr. 2008.
- [9] Media Machines – Developer Resources disponível em <http://www.mediamachines.com/developer.php> Acesso em 30 jun. 2008.
- [10] Blender disponível em <http://www.blender.org> Acesso em 09 abr. 2008.
- [11] 3D Studio Max disponível em <http://www.autodesk.com.br/3dsmax> acesso em 05 jun. 2008.
- [12] Ketsji disponível em [http://www.blender.org/documentation/intranet/docs/develop/gameengine\\_high\\_level/index.html](http://www.blender.org/documentation/intranet/docs/develop/gameengine_high_level/index.html) Acesso em 28 jun. 2008.
- [13] About Python disponível em <http://www.python.org/about/> Acesso em 20 jun. 2008.
- [14] Java Class Math disponível em <http://java.sun.com/javase/6/docs/api/java/lang/Math.html> Acesso em 29 mai. 2008.
- [15] Adobe Flash CS3 Professional disponível em <http://www.adobe.com/products/flash/> Acesso em 30 jun. 2008
- [16] Adobe disponível em <http://www.adobe.com> Acesso em 26 jun. 2008