FUNDAÇÃO DE ENSINO “EURÍPIDES SOARES DA ROCHA”
CENTRO UNIVERSITÁRIO EURÍPIDES DE MARÍLIA – UNIVEM
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

**MAYARA RIBEIRO FURLANETO**

# DESENVOLVIMENTO DE UM SOFTWARE EDUCACIONAL DO JOGO TANGRAM, USANDO OPENGL

MARÍLIA
2010

MAYARA RIBEIRO FURLANETO

# DESENVOLVIMENTO DE UM SOFTWARE EDUCACIONAL DO JOGO TANGRAM, USANDO OPENGL

Trabalho de Curso apresentado ao Curso de Bacharelado em Ciência da Computação da Fundação de Ensino "Eurípides Soares da Rocha", mantenedora do Centro Universitário Eurípides de Marília – UNIVEM, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador:
Profa. Dra. JULIANA DE OLIVEIRA

MARÍLIA
2010

Furlaneto, Mayara Ribeiro

Desenvolvimento de um Software Educacional do Jogo Tangram, usando OpenGL / Mayara Ribeiro Furlaneto; orientador: Juliana de Oliveira. Marília, SP: [s.n.], 2010.

58 f.

Trabalho de Curso (Graduação em Ciência da Computação) - Curso de Bacharelado em Ciência da Computação, Fundação de Ensino "Eurípides Soares da Rocha", mantenedora do Centro Universitário Eurípides de Marília –UNIVEM, Marília, 2010.

1. Software Educacional 2. Desenvolvimento de Jogos 3. Tangram

CDD: 005.12

CENTRO UNIVERSITÁRIO EURÍPIDES DE MARÍLIA
*BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO*

**TRABALHO DE CONCLUSÃO DE CURSO – AVALIAÇÃO FINAL**

Mayara Ribeiro Furlaneto

DESENVOLVIMENTO DE UM SOFTWARE EDUCACIONAL DO JOGO TANGRAM, USANDO OPENGL

Banca examinadora da monografia apresentada ao Curso de Bacharelado em Ciência da Computação do UNIVEM/F.E.E.S.R., para obtenção do Título de Bacharel em Ciência da Computação.

Nota: 10.0 ( Dez )

Orientador: Juliana de Oliveira

1°. Examinador: Renata Aparecida de Carvalho Paschoal

2°. Examinador: Leonardo Castro Botega

Marília, 02 de dezembro de 2010.

*À minha avó Darci,*
*que se foi bem*
*no meio desta trajetória,*
*dedico todas as minhas*
*vitórias à ela que foi meu*
*sustento, minha motivação,*
*meu espelho.*
*Devo tudo à ela,*
*minha eterna guerreira!*

# AGRADECIMENTOS

*Agradeço primeiramente à Deus pela vida, pela sabedoria e pela esperança. Sempre que precisei orei para Ele, e Ele me guiou para o certo.*

*Agradeço aos meus familiares e amigos que me motivaram e me deram o apoio necessário para concluir meu curso. Obrigada galera.*

*Agradeço ao meu noivo Yuri Fernandes pela paciência, dedicação e ajuda à mim prestadas. Te amo!*

*Agradeço aos professores do Curso de Ciência da Computação da Fundação de Ensino "Eurípides Soares da Rocha", os quais foram verdadeiros mestres e me deram a oportunidade, o apoio e o conhecimento necessários para a conclusão deste trabalho e deste curso.*

*Á profa. Dra. Juliana de Oliveira, pelo auxílio paciente na orientação, pela confiança, pelos elogios motivadores e pelas críticas construtivas que fizeram eu, com segurança e tranquilidade, desenvolver e finalizar este trabalho.*

FURLANETO, Mayara Ribeiro. **Desenvolvimento de um Software Educacional do Jogo Tangram, usando OpenGL**. 2010. 58 f. Trabalho de Curso (Bacharelado em Ciência da Computação) – Centro Universitário Eurípides de Marília, Fundação de Ensino "Eurípides Soares da Rocha", Marília, 2010.

## RESUMO

A computação aplicada à educação é uma área em grande crescimento, pois o computador é uma ferramenta eficaz e útil no processo ensino-aprendizagem. Para tanto, há a necessidade do desenvolvimento de softwares educacionais capazes de atingir esse objetivo e incentivar ainda mais o uso do computador para a educação. Neste trabalho são abordados os conceitos de softwares educacionais; sobre o jogo Tangram; os conceitos de desenvolvimento de jogos computacionais; e todo o processo necessário para a implementação de um protótipo do software educacional do jogo Tangram, usando uma poderosa ferramenta de desenvolvimento gráfico, a OpenGL, muito usada no desenvolvimento de jogos digitais 2D e 3D.

**Palavras-chave:** Software educacional. Jogo educacional. Tangram.

FURLANETO, Mayara Ribeiro. **Desenvolvimento de um Software Educacional do Jogo Tangram, usando OpenGL**. 2010. 58 f. Trabalho de Curso (Bacharelado em Ciência da Computação) – Centro Universitário Eurípides de Marília, Fundação de Ensino "Eurípides Soares da Rocha", Marília, 2010.

ABSTRACT

The computation applied to the education is an area in great growth, because the computer is a eficiente and useful tool in the process teach-learning. For this, it has the necessity of the development of educational softwares capable to reach this objective and to more stimulate the use of the computer for the education. This work has as purpose the study of educational softwares; about the game Tangram; concepts of game development; and computational tools for the development of an archetype of the educational software of the Tangram game, using a powerful tool of graphical development, the OpenGL, very used in the development of digital games 2D and 3D.

**Keywords:** Educacional Software. Educacional Game. Tangram.

# LISTA DE ILUSTRAÇÕES

# LISTA DE TABELAS

# LISTA DE ABREVIATURAS E SIGLAS

2D: Duas dimensões

3D: Três dimensões

AIX: *Advanced Interactive eXecutive*

API: *Application Programming Interface*

ARB: *Architecture Review Board*

BD: Banco/Base de Dados

BSD: *Berkeley Software Distributions*

CAD: *Computer Aidded Design*

GDI: *Graphics Device Interface*

GNU: *Gnu is Not Unix*

GUI: *Graphical User Interface*

HP-UX: *Hewlett Packard Unix*

IA: Inteligência Artificial

IDE: *Integrated Development Environment*

IRIS GL: *Integrated Raster Imaging System Graphics Library*

ISO: *International Standardization Organization*

JOGL: *Java bindings for OpenGL*

OpenGL: *Open Graphics Library*

OpenGL ES: *OpenGL for Embedded Systems*

PC: *Personal Computer*

RGBA: *Red, Green, Blue e Alpha*

RPG: *Role Playing Game*

SGI: *Silicon Graphics, Inc*

# SUMÁRIO

# INTRODUÇÃO

A utilização de computadores na Educação é um desafio para os pesquisadores preocupados com a disseminação dos computadores na nossa sociedade. A informática na educação é a inserção do computador no processo de ensino-aprendizagem de conteúdos curriculares de todos os níveis e modalidades de Educação.

Segundo Ralston & Meek (1976), em meados da década de 50, quando começaram a ser comercializados os primeiros computadores com capacidade de programação e armazenamento de informação, já apareceram as primeiras experiências do seu uso na Educação.

No Brasil, como em outros países, o uso do computador na Educação teve início com algumas experiências em universidades, no princípio da década de 70.

No entanto, a ênfase dada nessa época era praticamente a de armazenar informação em uma determinada seqüência e transmiti-la ao aprendiz. Hoje, a utilização de computadores na Educação é muito mais diversificada, interessante e desafiadora, do que simplesmente a de transmitir informação ao aprendiz.

Quando se investiga uma estratégia para a introdução dos computadores nas escolas, há a necessidade de se associar a discussão sobre o computador o conceito do chamado software educacional.

Um tipo muito usado de software educacional é o jogo educacional, que pode ser ferramenta instrucional eficiente, pois ele diverte enquanto motiva, facilitando o aprendizado e aumentando a capacidade de retenção do que foi ensinado, exercitando as funções mentais e intelectuais do jogador.

Assim sendo, o objetivo deste trabalho é desenvolver um software educacional do tipo jogo para servir de incentivo ao uso de computadores em atividades lúdicas pelos educadores.

Serão estudados conceitos sobre software educacional e sobre todo o processo de desenvolvimento de um jogo educacional. Ao final, usando a linguagem de programação C em conjunto com a biblioteca gráfica OpenGL, será implementado um protótipo do jogo educacional Tangram, muito usado como atividade lúdica nas aulas de matemática para o ensino da geometria.

# CAPÍTULO 1 – SOFTWARE EDUCACIONAL

Neste capítulo são abordados os conceitos de software educacional, apresentando a sua definição e as suas classificações. Também serão apresentados neste capítulo a definição de jogo educacional computadorizado, seus tipos e sua utilidade no ensino da Matemática.

## 1.1 Definição

A definição de software educacional passa por autores como Lucena (2002), Gomes (2003) e Gladcheff (2001) que manifestaram suas opiniões no início do segundo milênio.

Não basta apenas a escola ter computadores, para que haja um ambiente de ensino onde o professor possa usar o computador como ferramenta didática, se faz a necessidade de se associar o computador com o conceito do chamado software educacional (LUCENA, 2002).

O software educacional é um software desenvolvido com interfaces e artefatos educativos para servir de mediador entre atividades educativas nas diversas áreas do conhecimento (GOMES, 2003, p. 02).

Softwares educacionais podem ser fontes de informação, construtores de conhecimento e principalmente podem ser instrumentos lógicos que possibilitam o aprendiz a pensar, refletir, trabalhar com hipóteses e criar soluções (GLADCHEFF, 2001, p.03).

> Entretanto, para que um software seja utilizado com finalidade educacional ou em atividades curriculares, é necessário que sua qualidade, interface e pertinência pedagógica sejam previamente avaliadas de modo a atender às áreas de aplicação a que se destina e, principalmente, satisfazer às necessidades dos usuários (LUCENA, 2002, p. 04).

## 1.2 Classificação dos Softwares Educacionais

De acordo com Valente (1999), os softwares educacionais são classificados em 6 diferentes tipos: Tutoriais, Programação, Aplicativos, Multimídia e Internet, Modelagem e Simulação e Jogos. Sobre cada um dos tipos de softwares educacionais, serão descritos a seguir as suas características, objetivos pedagógicos e as formas como esse tipo de software atua no processo de ensino-aprendizagem.

### 1.2.1 Tutoriais

O software educacional do tipo tutorial se caracteriza por transmitir informações pedagógicas previamente definidas e organizadas, como se fossem um livro animado, um vídeo interativo ou um professor eletrônico.

Valente (1999, p. 72) mostra que no software do tipo tutorial:

> A interação entre o aprendiz e o computador consiste na leitura da tela ou escuta da informação fornecida, no avanço pelo material, apertando a tecla ENTER, na escolha da informação, usando o mouse e/ou resposta de perguntas que são digitadas no teclado.

Os tutoriais enfatizam a apresentação de informações e de lições ou exercícios, e a ação do aprendiz se restringe a ler e/ou escutar e realizar os exercícios. O professor, nesse caso, deve criar situações para o aluno manipular as informações recebidas de um tutorial, de modo que elas possam ser transformadas em conhecimento (VALENTE, 1999).

### 1.2.2 Programação

Os softwares educacionais de programação são caracterizados por terem uma linguagem fácil e intuitiva de programação de computadores, permitindo que pessoas, professores ou alunos, criem seus próprios protótipos de programas, sem que tenham que possuir conhecimentos avançados de programação.

O objetivo da programação é fazer com que o aprendiz pense estrategicamente, processando informações para resolver problemas, consequentemente transformando tudo isso em conhecimento.

Valente (1999, p. 75) descreve a vantagem da programação:

> O programa representa a idéia do aprendiz e existe uma correspondência direta entre cada comando e o comportamento da máquina. Essas características disponíveis no processo de programação facilitam a análise do programa, de modo que o aprendiz possa achar seus erros (*bugs*) e o professor possa entender o que ele está fazendo e pensando. Portanto, o processo de achar e corrigir o erro constitui uma oportunidade única para o aprendiz aprender sobre um determinado conceito envolvido na solução do problema ou sobre estratégias de resolução de problemas.

### 1.2.3 Aplicativos

Os aplicativos são programas voltados para aplicações específicas, tais como os processadores de texto, as planilhas eletrônicas, os gerenciadores de banco de dados, e entre outros aplicativos.

Apesar de não terem sido feitos com objetivo educacional, os aplicativos podem ser usados de forma interessante, pois facilitam a expressão do conhecimento através de interação mediada pelo idioma materno, comandos de formatação e cálculos.

Porém, os mesmos não favorecem muito no processo de construção do conhecimento, pois o resultado obtido só é uma comparação das idéias originais do formato com o resultado apresentado, o que torna limitado o espaço para a reflexão, depuração do conteúdo e compreensão das idéias pelo aprendiz. Sendo assim, os resultados obtidos no uso dos aplicativos devem ser sempre avaliados pelo professor (VALENTE, 1999).

### 1.2.4 Multimídia e Internet

Os softwares educacionais em forma de Multimídia apresentam o conteúdo educacional semelhante a um tutorial, porém de uma forma mais interessante através de recursos de textos, imagens, sons, animações e vídeos.

O software Multimídia disponibiliza várias opções ao aprendiz que escolhe dentre elas a de seu interesse. Uma vez escolhida uma opção, o computador apresenta a informação disponível e o aprendiz pode refletir sobre a mesma. Com isso, ele pode ir selecionando outras opções e assim navegar pelo software. A Internet e a Multimídia tem as mesmas características, os mesmos tipos de recursos e o objetivo de navegar pelas informações também é o mesmo, porém um software Multimídia tem um conteúdo limitado e específico, já a Internet tem um conteúdo amplo e diverso.

Ambos os softwares auxiliam o aprendiz a adquirir informação, mas não garante que o mesmo irá compreender ou construir conhecimento com a informação obtida. Nesse caso, cabe ao professor suprir essas situações para que a construção do conhecimento ocorra (VALENTE, 1999).

### 1.2.5 Modelagem e Simulação

Os softwares educacionais de Modelagem e Simulação permitem que sejam criadas e/ou apenas estudadas situações difíceis de serem reproduzidas em um ambiente comum, como por exemplo, experiências químicas, dissecação de cadáveres e o sistema solar.

No software de Simulação o fenômeno a ser estudado já está pronto e o mesmo é então fornecido ao aprendiz que poderá descrever ou implementar alguns aspectos do fenômeno e definir situações para ele no qual o software de simulação irá apresentar os resultados de forma semelhante a um tutorial (VALENTE, 1999).

No software de Modelagem, é o aprendiz quem escolhe o fenômeno, desenvolve o seu modelo e o implementa no computador de forma semelhante à atividade de programação.

Porém, por si só os softwares de simulação ou de modelagem não criam a melhor situação de aprendizado. Valente (1999, p. 80) conclui:

> Para que a aprendizagem ocorra, é necessário criar condições para que o aprendiz se envolva com o fenômeno e essa experiência seja complementada com elaboração de hipóteses, leituras, discussões e uso do computador para validar essa compreensão do fenômeno. Nesse caso, o professor tem o papel de auxiliar o aprendiz a não formar uma visão distorcida a respeito do mundo (que o mundo real pode ser sempre simplificado e controlado da mesma maneira que nos programas de simulação) e criar condições para o aprendiz fazer a transição entre a simulação e o fenômeno no mundo real.

### 1.2.6 Jogos

Os softwares educacionais do tipo jogo podem ser implementações computacionais de jogos educativos antes feitos de papel, madeira ou de algum outro material já usado pelos professores nas aulas, ou também, podem ser implementações de novos jogos de conteúdo educativo.

Os jogos têm a finalidade de desafiar e motivar o aprendiz em uma maneira prazerosa de se aprender e de reter o conteúdo que já foi ensinado nas aulas. Ao jogar espera-se que o aprendiz esteja elaborando hipóteses, usando estratégias e conhecimentos já existentes ou elaborando conhecimentos novos (VALENTE, 1999).

Mas é importante se atentar que os jogos têm a função de envolver o aprendiz em uma competição e essa competição pode atrapalhar o processo de ensino-aprendizagem. Portanto é importante que o professor avalie as situações apresentadas pelo aprendiz durante o jogo e depois as discuta com ele, recriando-as, apresentando conflitos e novos desafios, com o

objetivo de propiciar condições para o aprendiz adquirir conhecimentos além de só jogar (VALENTE, 1999).

## 1.3 Jogos Educacionais

De uma maneira geral os jogos na educação podem ser ferramentas instrucionais eficientes, pois eles divertem enquanto motivam, facilitam o aprendizado e aumentam a capacidade de retenção do que foi ensinado, exercitando as funções mentais e intelectuais do jogador (TAROUCO, 2004), e até mesmo motoras com os jogos de reabilitação.

Quando os jogos estão pedagogicamente embasados, ou são motivadores do processo de aprendizagem ou podem ser utilizados para algum objetivo educacional, então eles são definidos como jogos educacionais (TAROUCO, 2004).

A utilização de jogos na educação ajuda a desenvolver habilidades como a criatividade, a competição, a cooperação, a organização, a perseverança e a autonomia dos aprendizes (PORTO, 2008, p. 02).

Um jogo educacional não garante sucesso no processo ensino-aprendizagem, muitas vezes o professor precisa ser o mediador, segundo Tarouco (2004, p. 03):

> Os jogos educacionais se baseiam numa abordagem auto-dirigida, isto é, aquela em que o sujeito aprende por si só, através da descoberta de relações e da interação com o software. Neste cenário, o professor tem o papel de moderador, mediador do processo, dando orientações e selecionando softwares adequados e condizentes com sua prática pedagógica.

### 1.3.1 Tipos de Jogos Educacionais

Existem vários e diferentes tipos de jogos classificados de acordo com seus objetivos e características, tais como jogos de ação, aventura, cassino, lógicos, estratégicos, esportivos, roleplaying games (RPGs), entre outros. Abaixo seguem alguns tipos que podem ser utilizados com propósitos educacionais, de acordo com Tarouco (2004):

**Ação** – os jogos de ação tendem a auxiliar o desenvolvimento psicomotor da criança, desenvolver os reflexos, a coordenação e auxilia no processo de pensamento rápido frente a uma situação inesperada. O ideal para um jogo de ação, é que o mesmo alterne momentos de atividade cognitiva mais intensa com períodos de utilização de habilidades motoras.

**Aventura** – os jogos de aventura se caracterizam por ser um ambiente desconhecido a ser desvendado e explorado pelo jogador. Se modelado pedagogicamente, pode auxiliar na simulação de atividades impossíveis de serem vivenciadas em sala de aula, tais como um desastre ecológico ou um experimento químico.

**Lógico** – os jogos lógicos desafiam muito mais a mente do que os reflexos do jogador. Os jogos lógicos também podem oferecer um limite de tempo dentro do qual o usuário deve finalizar uma tarefa, exercitando ainda mais o uso do raciocínio rápido. Exemplos de jogos lógicos são: xadrez, damas, caça-palavras, palavras-cruzadas e jogos que exigem resoluções matemáticas.

**Estratégico** – os jogos estratégicos são focados em desafiar a sabedoria e as habilidades de negócios do jogador, principalmente quando o objetivo é a construção ou administração de algo. Esse tipo de jogo pode proporcionar uma simulação em que o jogador aplica conhecimentos adquiridos em sala de aula, criando uma forma prática de aplicá-los.

### 1.3.2 Jogos Educacionais de Computador aplicados ao Ensino da Matemática

Um jogo educacional, ao ser usado para o ensino da Matemática, auxilia no processo de construção do conhecimento, pois é um meio para se desenvolver o pensamento, a reflexão, habilidades de raciocínio e a criação de soluções a diversos problemas através de hipóteses e deduções (GLADCHEFF, 2001).

O jogo também pode tornar certos conceitos bem mais fáceis de compreender, é grande a variedade de temas do ensino fundamental e médio que podem ser explorados com tais recursos, com destaque aos temas relacionados à geometria (FANTI, 2004).

Muitos jogos educacionais, principalmente dos tipos lógico e estratégico, podem ser implementados no computador para servir de ferramenta instrucional no ensino da Matemática. Até mesmo jogos de tabuleiro e quebra-cabeças que são muito usados nas aulas de Matemática, como por exemplo o xadrez, damas, dominó e o quebra-cabeça Tangram, podem ser enriquecidos com animações, imagens, sons, textos e vídeos, tornando-os ainda mais envolventes e muito mais motivadores no processo ensino-aprendizagem da Matemática.

# CAPÍTULO 2 – O JOGO TANGRAM

Um jogo educacional muito utilizado nas aulas de Matemática é o quebra-cabeça de sete peças geométricas chamado Tangram.

Neste capítulo serão apresentados um breve histórico sobre o Tangram, suas características e como o mesmo pode ser usado como jogo educacional no ensino da Matemática.

## 2.1 Breve Histórico

O Tangram é um jogo que exige raciocínio e reflexão. Pouco se sabe sobre sua origem, alguns dizem que a palavra "Tangram" é inglesa e que significa *puzzle* (quebra-cabeça) ou quinquilharias. Outros afirmam que a palavra "Tangram" é originária de uma tribo da China, a Tanka, onde seus grandes comerciantes envolviam seus visitantes com este quebra-cabeça. E ainda uma outra história conta que o Tangram foi inventado por um homem que se chamava Tan, que tentava consertar os pedaços quebrados de um azulejo de porcelana (MOTTA, 2006).

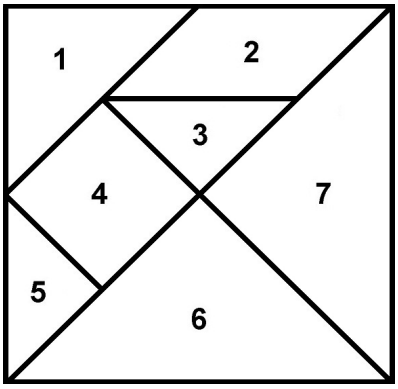
Foram criados Tangrams em todos os tipos de materiais, desde cartão até pedra, plástico ou metal. Uma Enciclopédia de Tangram foi escrita por uma mulher, na China, há 130 anos atrás e contêm mais de 1.700 problemas para resolver (MOTTA, 2006).

Ainda hoje o Tangram é muito utilizado, especialmente por professores no ensino da geometria. A sua simplicidade e capacidade de representar uma variedade enorme de objetos e, ao mesmo tempo a dificuldade em resolvê-los, exigem do jogador imaginação, paciência e criatividade (MOTTA, 2006).

## 2.2 Características

O Tangram tradicional é formado por sete peças geométricas cortadas a partir de um quadrado: cinco triângulos retângulos isósceles de tamanhos diferentes (1, 3, 5, 6, 7), um quadrado (4) e um paralelogramo (2), como mostra a Figura 1.

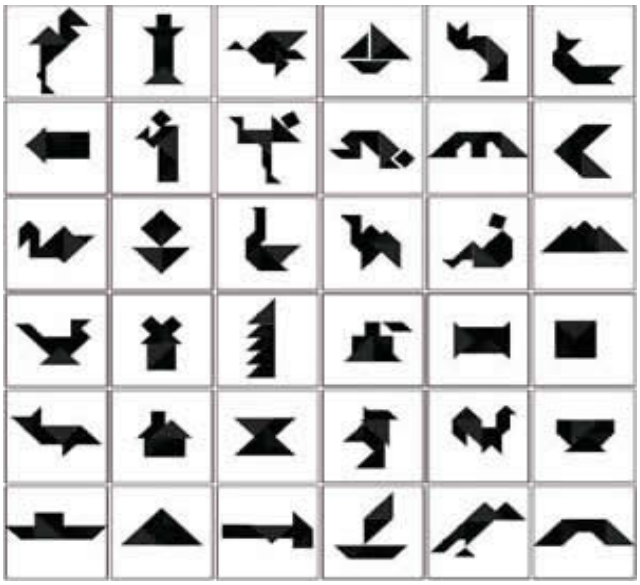
Figura 1 – Formação do Tangram Tradicional




Fonte: Autoria Própria

Através de combinações destas peças é possível criar e montar diversas figuras, como mostra a Figura 2.

Figura 2 – Exemplo de figuras formadas com as sete peças do Tangram



Fonte: BrasilEscola, 2010

O desafio do jogo é descobrir como uma sombra (imagem) foi criada usando todas as sete peças do Tangram. Devem-se combinar todas as peças, sem sobreposição, cobrindo totalmente a área exata da sombra, conforme mostra a figura 3, onde a figura à esquerda é uma sombra e à da direita é a formação desta sombra com as peças do Tangram.

Figura 3 – Exemplo de desafio do Tangram

Fonte: Autoria Própria

## 2.3 O Tangram como Jogo Educacional

O Tangram é muito usado nas escolas como uma forma lúdica de por em prática os conhecimentos teóricos adquiridos nas aulas de Geometria, pois o Tangram trabalha com as principais formas geométricas planas, além disso, através da composição e decomposição das figuras, o Tangram instiga o aluno a pensar geometricamente, no qual desenvolve sua criatividade, habilidade de análise e principalmente o seu raciocínio lógico.

> Estas são características necessárias e beneficiam o ensino e a aprendizagem da Matemática, pois quando o aluno estabelece relações, elabora e testa hipóteses e as reestrutura com o intuito de acertar, participa ativamente da construção do seu conhecimento. (ARRUDA; ALMEIDA, [s.d.], p. 06).

De acordo com Arruda ([s.d.], p. 06), o jogo Tangram favorece:

- O estudo de áreas, pois há diversas formas de composição e decomposição de figuras;

- A noção de espaço e combinação, porque os alunos têm de formar diferentes figuras sempre com a mesma quantidade de peças e;

- A autoavalição, porque o aluno precisa se corrigir a todo o momento para alcançar o objetivo.

Nas escolas o Tangram geralmente é feito em papel, onde os próprios alunos desenham e recortam o quebra-cabeça para brincar com a ajuda do professor. Implementar o Tangram como software educacional facilitaria e daria mais tempo ao professor para aplicar o jogo aos alunos, pois o software traz os desafios já prontos e em quantidade, avalia e ajuda o aprendiz nas resoluções e também motiva e atrai a atenção dos alunos para o desafio do jogo.

Além, da integração com o computador, ferramenta indispensável no cotidiano e na formação dos estudantes.

# CAPÍTULO 3 – DESENVOLVIMENTO DE JOGOS DE COMPUTADOR

Antes de iniciar o desenvolvimento de um software de um jogo educacional, é necessário conhecer os processos de desenvolvimento de um jogo de computador. Um processo de desenvolvimento completo envolve muitas questões e fases. Neste capítulo serão apresentados os conceitos mais básicos e objetivos de desenvolvimento de um jogo de computador, que servirão de base para o desenvolvimento do software educacional do jogo Tangram.

## 3.1 Definição de Jogo de Computador

De acordo com Battaiola (2003), um jogo de computador é um sistema composto em três partes principais: enredo, motor e interface interativa, conforme são explicados a seguir:

- **Enredo**: o enredo define o tema, a trama e os objetivos do jogo, e a sua definição pode envolver a participação de diferentes especialistas, tais como, pedagogos, psicólogos, historiadores, etc. Com criatividade e pesquisa sobre o assunto, deverá então ser definido o enredo.

- **Motor**: o motor é o sistema de controle do jogo, no qual controla a interação entre o usuário e a sua interface. A implementação do motor pode envolver conceitos presentes em diversas áreas da computação, tais como, computação gráfica, inteligência artificial, redes de computadores, engenharia de software, etc, conceitos estes que levam à escolha apropriada da linguagem de programação, ao desenvolvimento de algoritmos específicos, ao tipo de interface com o usuário, etc.

- **Interface Interativa**: a interface interativa apresenta graficamente o estado corrente do jogo e viabiliza a interação entre o jogo e o usuário. A sua implementação pode envolver aspectos artísticos, cognitivos e técnicos. O aspecto artístico de uma interface tem como objetivo valorizar a apresentação do jogo, atraindo usuários e aumentando a sua satisfação ao jogar. O aspecto cognitivo trata a interpretação da informação gráfica pelo usuário. O aspecto técnico envolve performance, portabilidade e a complexidade dos elementos gráficos. É importante

lembrar que no caso de jogos educacionais, a interface deverá obedecer a critérios pedagógicos.

O jogo tangram tem seu enredo baseado em um quebra-cabeça, onde o objetivo do jogo é completar uma sombra com todas as sete peças do Tangram. Em seu enredo também há uma base pedagógica em matemática para o ensino da geometria e para o desenvolvimento do raciocínio lógico.

O motor do jogo tangram é totalmente programado em C usando a biblioteca gráfica OpenGL, com algoritmos ligados à interações com o mouse.

A interface interativa do jogo tangram é bem simples, feita em 2D, colorida e de fácil entendimento. Foi usada técnica de transparência para ajudar o jogador a enxergar melhor os encaixes das peças. E também foi usada a técnica "arrastar e soltar", onde o jogador apenas precisa usar o mouse para arrastar e soltar as peças pela interface, facilitando ainda mais a jogabilidade.

## 3.2 Tipos de Jogos de Computador

Para se desenvolver um jogo, será necessário escolher o seu tipo. Os jogos são feitos em diferentes tipos, pois o jogo tem como objetivo entreter diversas e diferentes pessoas, com diversos gostos. O tipo de jogo poderá se adequar também ao objetivo que o jogo terá, como por exemplo, citado no Capítulo 1 - 1.3.1 Tipos de Jogos Educacionais, há certos tipos de jogos apropriados para serem usados como jogos educacionais. De acordo com Battaiola (2003) e Cavaco (2007), os principais tipos são os seguintes:

**Estratégia**: são jogos cujo propósito e objetivo é que o jogador gerencie recursos a fim de conquistar objetivo usando estratégias e táticas. O jogo depende das decisões do jogador, são as decisões que irão decidir as conseqüências para o jogador.

**Simuladores**: são jogos que tem como objetivo a imersão do jogador em um ambiente que tenta retratar a realidade. Simuladores são jogos geralmente baseados em tática e física.

**Aventura**: são jogos que combinam ações baseadas em raciocínio e reflexo, que impõem objetivos ao jogador, que terá como missão ultrapassar estágios que envolvam a solução de enigmas para completá-los.

**Passatempo**: são jogos simples, como quebra-cabeças, jogos de tabuleiro, jogo de cartas, cujo objetivo essencial é atingir uma pontuação alta.

**RPG (*Role Playing Game*)**: é uma versão computadorizada do RPG convencional, onde o jogador assume um personagem e enfrenta situações para ganhar experiência e ir se desenvolvendo ao longo da história do jogo.

**Esporte**: são basicamente jogos do tipo simuladores, porém são simulações baseadas nos esportes populares, como os jogos de futebol, vôlei, basquete, boxe, etc.

**Ação**: esse jogo gira em torno de um personagem principal, que geralmente pode andar, saltar, correr, atirar, chutar, etc. São os jogos de ação que proporcionam a visão ao jogador em primeira ou em terceira pessoa.

**Luta**: parecidos com os jogos de ação, porém os jogos de luta têm como característica uma disputa entre o jogador e o computador, ou entre outro jogador ou jogadores, onde cada um controla um personagem que possui uma combinação de movimentos e manobras para atacar ou se defender do oponente. As lutas são organizadas em *rounds*.

***Online/Massive Multiplayer***: este tipo de jogo pode englobar qualquer uma das categorias anteriores, com a diferença de ser jogado com muitos jogadores conectados em rede, via internet ou em *lan houses*.

O jogo tangram é um jogo totalmente do tipo passatempo voltado à lógica, pois é um quebra-cabeça que exige raciocínio geométrico.

## 3.3 Processo de Desenvolvimento de Jogos

Deve-se lembrar que um jogo é um software, e atualmente um software é desenvolvido em equipe, onde cada membro da equipe trabalha em sua especialidade até que o trabalho de todos seja integrado criando um trabalho único e coerente (ASTLE, 2004, p. 04).

Os jogos são desenvolvidos da mesma maneira, porém, ao contrário do software convencional, o jogo necessita de várias áreas de especialização. Artistas são necessários para gerar as imagens e um cenário de qualidade que é predominante em muitos dos jogos de hoje. Os designers trazem o mundo virtual à vida e usam a arte que lhes é fornecida pelos artistas para criar mundos além da imaginação. Técnicos de som e músicos criam o áudio necessário para fornecer ao jogador uma rica, multimídia, acreditável e virtual experiência. Finalmente os programadores juntam cada um dos elementos e certificam-se de que tudo funciona como um todo (ASTLE, 2004, p. 04-05).

Os jogos podem apresentar misturas complexas de tecnologia e projetos de software e, como tal, o desenvolvimento do jogo exige raciocínio abstrato e uma aplicação em nível mais elevado do que de um desenvolvimento de um software tradicional (ASTLE, 2004, p. 06).

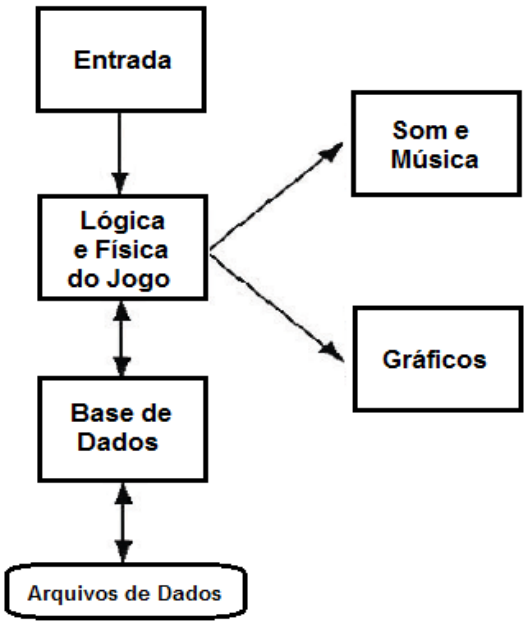
### 3.3.1 Arquitetura de um Jogo

O jogo é dividido em vários elementos para ser desenvolvido de acordo com cada diferente área de especialização, sendo reunidos todos os elementos no final do desenvolvimento. No geral, os jogos estão divididos nas seguintes áreas (ASTLE, 2004, p. 05):

- Gráficos
- Entrada
- Música e som
- Lógica do jogo e inteligência artificial
- *Networking*
- Sistema de interface e de interação de usuário

Cada uma destas áreas pode ser dividida em sistemas mais específicos. Por exemplo, a lógica do jogo consistiria em física e sistemas de partículas, enquanto que os gráficos podem ter um 2D e/ou 3D *renderer*. Na Figura 4 é mostrada uma arquitetura de jogo simples (ASTLE, 2004, p. 05).

Figura 4 – Simples composição dos subsistemas de um jogo

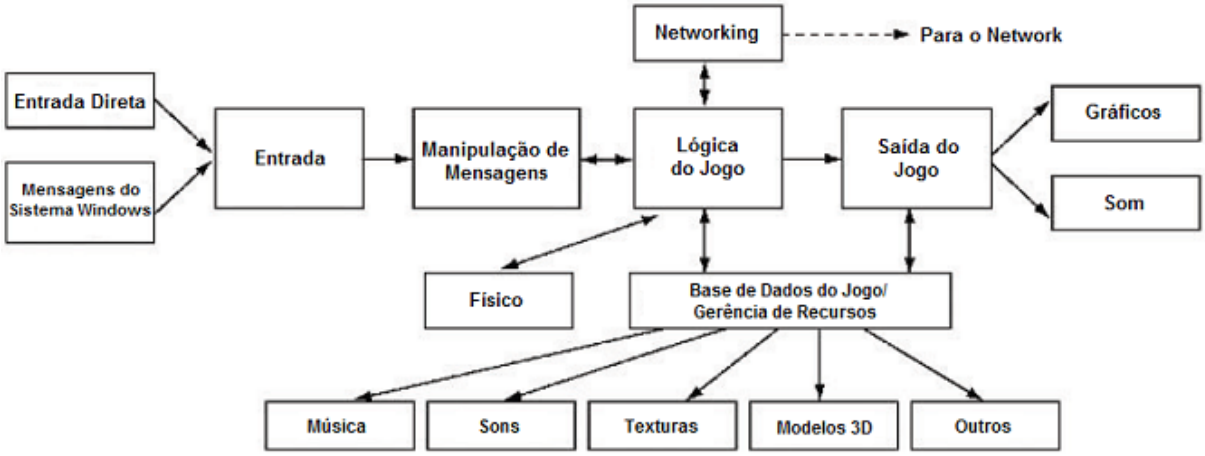



Fonte: Figura Adaptada de ASTLE; HAWKINS (2004, p.05).

Cada elemento de um jogo é dividido em seu próprio "pedaço" e se comunica com os outros elementos do jogo. O elemento Lógica e Física do Jogo tende a ser o centro do jogo, onde as decisões são feitas para o processamento de entrada e saída.

A arquitetura mostrada na Figura 4 é muito simples, no entanto, na Figura 5 é mostrada que a arquitetura de um jogo pode ser mais avançada (ASTLE, 2004, p. 06).

Figura 5 – Design de Arquitetura de jogo mais avançada

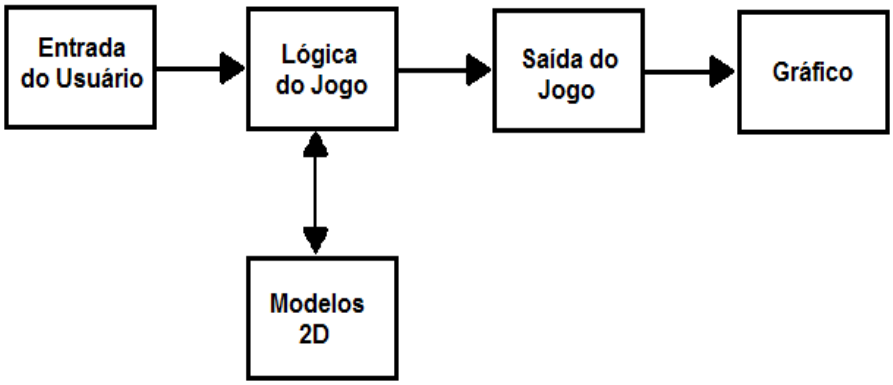



Fonte: Figura Adaptada de ASTLE; HAWKINS (2004, p. 06).

Como é mostrado na Figura 5, um jogo mais complexo exige um projeto arquitetônico mais complexo. Mais componentes são desenvolvidos e utilizados para implementar recursos específicos, ou uma funcionalidade, que o software do jogo precisa para funcionar sem problemas (ASTLE, 2004, p. 06).

Na Figura 6 é mostrada a arquitetura básica do jogo tangram:

Figura 6 – Design de Arquitetura do Jogo Tangram




Fonte: Autoria Própria

### 3.3.2 Design de um Jogo

O Design de um Jogo, ou Game Design, ou Projeto do Jogo, é o principal processo de desenvolvimento de um jogo, é o Design do Jogo que idealiza, imagina e define como será o jogo. No processo de Design do Jogo também são definidos os elementos que irão compor o jogo, as regras, os objetivos e a forma que o jogo funcionará. O processo do Game Design se resume em:

**Conceituação do Jogo:** é onde se cria e refina a idéia do jogo. As idéias não só surgem do próprio autor, os jogos podem ser baseados em livros, filmes, estórias, e até mesmo melhoria de jogos já existentes. Então a idéia do jogo pode ser documentada em uma proposta com os seguintes itens: **Introducão**; **Estória/Motivacão**; **Descrição**; **Características Chave**; **Gênero (tipo de jogo)**; **Plataforma(s)**; **Arte conceitual** (JUNIOR, 2002, p. 06-07).

**Definição dos Elementos do Jogo:** nesta etapa são definidos quais os elementos que farão parte desse jogo, de acordo com as características do jogo que será desenvolvido. Os principais elementos são: **Interface**; **Roteiro**; **Engine**; ***Gameplay***; **Sonorização**; **Artes Gráficas**; **entre outros subelementos** (CAVACO, 2007)**.**

Cavaco (2007, p. 49) finaliza citando:

> O Game Designer é a pessoa que determina cada detalhe do jogo, como o jogador pode controlar o jogo, quais informações serão transmitidas a ele, interface, cenário, enredo, jogabilidade, critério de vitória e derrota. Ela é a principal responsável pelo sucesso de um jogo.

### 3.3.3 Os Elementos de um Jogo

De acordo com Cavaco (2007), nas etapas de desenvolvimento do jogo, os seguintes elementos são desenvolvidos:

**Interface**: É a camada de apresentação do jogo, ela descreve como as informações serão apresentadas para o jogador, ela possui os seguintes elementos: **Interface do jogo**; **Menus**; **Tela de abertura**.

**Roteiro**: Ela descreve a história do jogo, descreve os cenários, as características psíquicas e psicológicas dos personagens, a seqüência de eventos que acontece no jogo, as animações do jogo e os vídeos. Possui os seguintes elementos: **Enredo**; **Eventos do jogo**; **Cenários**; **Animações**; **Vídeos**.

**Engine**: Ele é o núcleo do jogo, ele que controla as ações do jogador, ele é responsável pela entrada e saída de dados, exibição gráfica, protocolos de rede, regras do jogo, etc. Na engine são definidas: **Interface**; **Input**; **Editor gráfico**; G***ameplay***; **Network**; **IA**; **BD** (**Base de Dados**).

***Gameplay***: Descreve a jogabilidade do jogo, controles, sistema de batalha, sistema de pontuação, sistema de raças e magias, etc. São elementos do *gameplay*: **Controles**; **Sistemas de Batalhas**; **Inventário**; **Classes e Habilidades**; **Itens do jogo**; **Sistema de Pontos**; **Ranking** (**High Scores**).

**Sonorização**: O artista é responsável por todos os sons do jogo, dando ao jogador um aumento de realismo. Possui os seguintes elementos: **Trilha sonora**; **Efeitos Sonoros**; **Sonoplastia**; **Dublagens**.

**Artes Gráficas:** O artista é responsável pela arte gráfica do jogo, arte 2D, 3D e animações que acontece no jogo. Os elementos são: **Cenários**; **Personagens**; **Modelagem/Texturas**; **Animações**; **Vídeos**.

### 3.3.4 Programação do Jogo

É na fase de programação que tudo que foi planejado começa a ser implementado. Os desenvolvedores então escolhem a API (*Aplication Programming Interface*) e a linguagem para a programação do jogo, de acordo com suas necessidades.

Atualmente, pode-se utilizar praticamente todas as linguagens de programação para se programar um jogo, porém a linguagem mais utilizada no desenvolvimento de jogos é a linguagem C/C++ (LUZ, 2004, p. 50).

Para o desenvolvimento gráfico, utiliza-se uma API gráfica juntamente com a linguagem. As APIs gráficas mais utilizadas no momento são o OpenGL e o Direct3D.

OpenGL é uma API gráfica aberta e portável, disponibilizada pela Silicon Graphics. Ela é bastante utilizada no ensinamento de conceitos de Computação Gráfica, por ser mais simples e de implementação mais fácil (LUZ, 2004, p. 50).

O Direct3D faz parte do pacote DirectX, e também é uma API gráfica gratuita desenvolvida pela Microsoft. É a API mais utilizada para o desenvolvimento de jogos por ter atualizações mais constantes, bem como fazer parte do pacote DirectX, porém não é portável como a OpenGL (LUZ, 2004, p. 50).

> Mesclava-se também o uso do OpenGL com as demais bibliotecas do DirectX, para o desenvolvimento de jogos, mas atualmente essa é uma opção muito rara. Atualmente é utilizada o Direct3D juntamente com as demais bibliotecas do DirectX para o desenvolvimento de jogos. Entre os jogos feitos utilizando-se OpenGL com as bibliotecas do DirectX, podemos citar Worms 3D e Final Fantasy 7. Já utilizando-se DirectX, temos The Need For Speed Underground, Max Payne 1 e 2, Outlive etc (LUZ, 2004, p. 50).

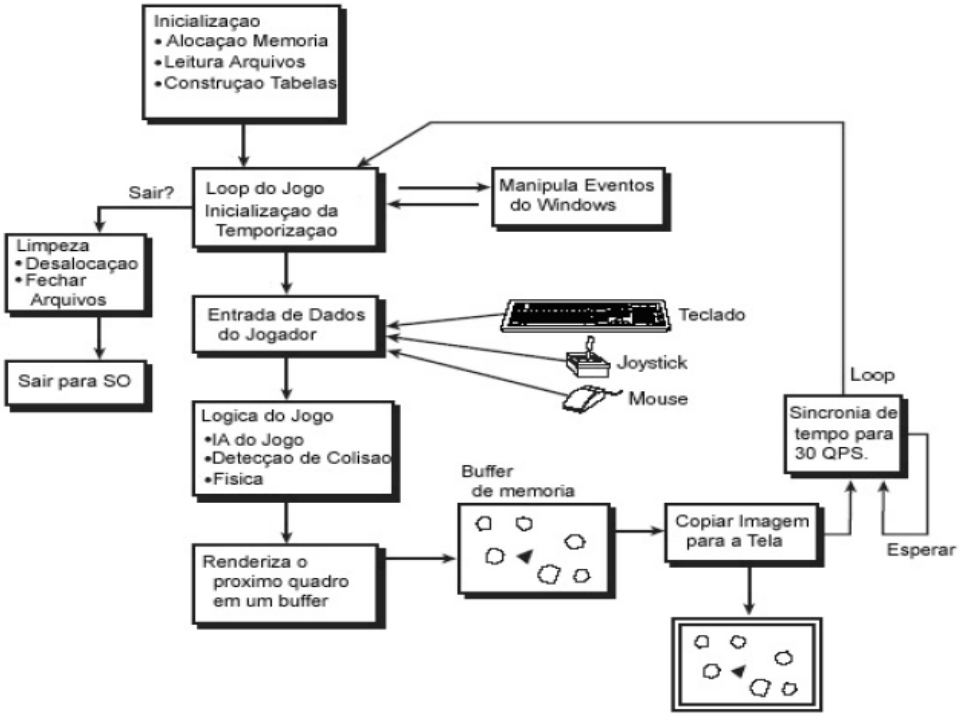
Em ambiente acadêmico, para pesquisas, utiliza-se bastante Java com a biblioteca JOGL, que nada mais é do que o uso da OpenGL para a linguagem Java. A biblioteca Java3D também é bastante utilizada para esse fim (LUZ, 2004, p. 50).

Porém a utilização da linguagem C/C++ para jogos comerciais é a mais aceita, pois é mais veloz e pelo fato das bibliotecas da linguagem Java ainda não explorarem todos os recursos das placas gráficas atuais (LUZ, 2004, p. 50).

Para o desenvolvimento do jogo Tangram foi escolhida a API OpenGL, bem como o uso da linguagem de programação C, por ser mais simples o uso da OpenGL com a linguagem C, e pelas demais vantagens da API OpenGL que será descrita no Capítulo 4.

De acordo com Luz (2004), o software do jogo segue uma determinada estrutura de funcionamento. Na Figura 7 de Luz (2004), é mostrado como geralmente um jogo é programado e como ele funciona seguindo a seguinte estrutura:

Figura 7 – Estrutura de um Código de Jogo




Fonte: LUZ **(**2004, p. 52).

**1º - Inicialização:** É a parte onde são efetuadas as operações padrão de qualquer programa, como alocação de memória, leitura de dados do disco, inicialização das variáveis, etc.

**2º - Loop do Jogo:** É nessa parte que o motor do jogo é executado, sempre em loop contínuo. É nessa parte que as ações do usuário são processadas até que o mesmo opte por finalizar o jogo.

**3º - Entrada de Dados do Jogador:** É nessa parte que as entradas de ação do jogador são processadas ou guardadas em um determinado *buffer* para depois serem utilizadas pela IA ou pela lógica do jogo.

**4º - Lógica do Jogo:** Nessa parte se encontra a maior parte do código do jogo. É nessa parte que estão o processamento da lógica do jogo, o da inteligência artificial e o da simulação da física, sendo os resultados utilizados para a renderização do próximo quadro que será exibido na tela.

**5º - Renderização do Próximo Quadro:** Nessa parte é criado o próximo quadro a ser exibido na tela pela coleta feita das entradas do jogador e dos resultados provenientes da execução da IA e da lógica do jogo. Geralmente, essa imagem é desenhada em um buffer que não é o mesmo da tela. Depois, ela é copiada rapidamente para a área de memória que representa o que está sendo mostrado na tela.

**6º - Sincronia da Exibição dos Quadros:** Essa é a parte onde se criam métodos para sincronização dos quadros, pois dependendo do computador e do quadro que deverá ser renderizado na tela, o tempo necessário para o processamento dos dados para a realização do mesmo pode variar muito e o número de quadros do jogo será muito variável. Neste caso, existem dois caminhos a serem seguidos: criar algum método de espera ou fazer com que as entradas do usuário sejam aplicadas de maneira proporcional ao tempo necessário para a renderização do quadro.

**7º – Limpeza:** Essa é a parte que finaliza o jogo. Isso significa que o usuário quer sair do jogo e voltar para o sistema operacional. Mas, antes disso, é necessária a desalocação dos dados e do fechamento dos arquivos que estavam sendo utilizados.

### 3.3.5 Testes

O processo de testes envolve a procura por falhas e também a jogabilidade e aceitação do jogo por parte dos usuários. Os testes geralmente são distribuídos entre todas as fases de desenvolvimento (JUNIOR, 2002, p. 12).

Os testes de funcionalidade e *bugfix* (acertar erros) em geral possuem as características das abordagens clássicas da Engenharia de Software. Mas também existe uma classe de testes que é própria para o desenvolvimento de jogos: o *playtest*. O *playtest* permite que os programadores analisem a aceitação do jogo e a reação dos jogadores (JUNIOR, 2002, p. 12). De acordo com Junior (2002), um *playtest* pode ser:

**Aberto**: uma versão do jogo é disponível para download para os usuários testarem ou marca-se um dia e local para a realização dos testes.

**Fechado**: os testadores são previamente selecionados, existem os chamados "testadores profissionais", eles são capazes de ir além de simplesmente testar e apontar o problema, eles são capazes também de dar informações mais detalhadas e sugestões que auxiliam na correção do software.

As abordagens de teste variam conforme a equipe e a fase de desenvolvimento, sendo que as mais conhecidas e utilizadas são:

**Usuários jogam e programador observa:** o programador não mantém nenhum contato com o jogador. O programador deverá apenas observar e fazer as anotações das reações e dificuldades dos jogadores. Essa metodologia se mostra eficiente na descoberta de dificuldades de jogabilidade e de partes tediosas de um game (JUNIOR, 2002, p. 12).

**Usuários jogam enquanto o programador faz perguntas:** o programador não apenas observa, ele também faz perguntas ao longo do jogo para o jogador. As perguntas são sobre os aspectos, a jogabilidade e a emoção do jogo. Funciona bem para jogos onde a ação é constante e dinâmica (JUNIOR, 2002, p. 12).

**Usuários jogam e fazem relatório:** é muito útil em *playtests* de grande porte (como os testes via internet), essa abordagem exige que o usuário jogue e faça um relatório sobre o que achou do jogo. É nesse modelo que se encaixam bem os "testadores profissionais". Porém, os desenvolvedores devem se atentar ao problema de vazamento de informações. É comum acontecer de versões piratas de jogos serem disponibilizados na Internet meses antes de seu lançamento, fruto de cópias de seções de testes (JUNIOR, 2002, p. 12).

# CAPÍTULO 4 – API DE DESENVOLVIMENTO GRÁFICO OPENGL

Neste capítulo será feita uma introdução à uma das principais APIs para desenvolvimento de jogos, a API de desenvolvimento gráfico OpenGL, na qual foi estudada para ser utilizada na implementação do jogo Tangram.

## 4.1 Histórico

Nos finais dos anos 80 a *Silicon Graphics, Inc.* (SGI) criou uma API de desenvolvimento 3D chamada IRIS GL, cuja finalidade era o desenvolvimento gráfico em 3D para seus sistemas operacionais IRIX. Porém a IRIS GL não era compatível com outros sistemas operacionais e isso tornou difícil o seu sucesso.

Então em 1992, com o intuito de criar um padrão público, a SGI criou uma nova API de desenvolvimento gráfico aberta, a OpenGL. Ao contrário do sistema original da IRIS GL, a OpenGL é totalmente portável.

Desde então o padrão OpenGL é mantido pelo órgão ARB (*Architecture Review Board*), que inclui companhias tais como a própria SGI, a nVIDIA, a ATI, a Matrox, a 3Dlabs, entre outras. Órgão esse responsável pela especificação e atualização do padrão OpenGL (SEDDON, p. 43).

A grande concorrente da OpenGL, a API 3D Direct3D foi criada em 1998 pela Microsoft e se tornou popular dentre os desenvolvedores de jogos pois havia muitas atualizações e recursos disponíveis à ela, porém há muitas razões ainda para se utilizar a OpenGL (SEDDON, 2005, p. 44-45):

- A OpenGL tem um órgão (ARB) de fabricantes e vendedores que planejam o futuro da API e têm opiniões de vários fornecedores relativas à como lidar com a API.

- A sintaxe OpenGL é muito simples e permite ao programador escrever programas gráficos complicados de forma simples e de maneira lógica.

– A API permite que os fabricantes de placas de vídeo criem recursos personalizados e aproveitem antes que se tornem parte da norma.

– Um código OpenGL também pode ser executado em uma série de plataformas, que outras APIs como o Direct3D não pode, como mostra a Tabela 1:

Tabela 1 – Comparação entre OpenGL e Direct3D em relação aos Sistemas Operacionais suportados

| **OpenGL** | **Direct3D** |
|---|---|
| Windows | Windows |
| Mac OS 9/Mac OS X | Xbox* |
| GNU/Linux | |
| UNIX | |
| AIX | |
| HP-UX | |
| FreeBSD | |
| NEXTstep | |
| OPENstep | |
| OS/2 | |
| BeOS | |
| Playstation 2<br>(via PS2Linux add-on) | |

*Versão modificada do Direct3D

Fonte: Tabela adaptada de SEDDON (2005, p. 45).

A OpenGL foi e é considerada uma API fácil de usar, pois há uma riqueza de documentação: livros, programas de exemplo, e assim por diante. Foi por isso que anos atrás a OpenGL começou a florescer (WRIGHT, 2007, p. 37).

Os desenvolvedores se preocupam com o tempo de comercialização, portabilidade e reutilização de código. A utilização da OpenGL, permitiu que muitos desenvolvedores conseguissem atender melhor à demanda de clientes. A popularidade da OpenGL continuou a crescer como alternativa à tecnologia de processamento específico do Windows e é agora amplamente apoiada em todos os principais sistemas operacionais e dispositivos de hardware (WRIGHT, 2007, p. 37).

Até mesmo os celulares com tecnologia gráfica 3D suportam um subconjunto da OpenGL chamada de OpenGL ES. Hoje, todas as novas placas de vídeo 3D para PC trazem suporte aos dois drivers: OpenGL e o Direct3D. Isso é devido, em grande parte, ao fato de que muitos desenvolvedores continuam a preferir a OpenGL para novos desenvolvimentos.

Atualmente, a OpenGL também é reconhecida e aceita como um padrão API para desenvolvimento de aplicações gráficas 3D e 2D em tempo real (WRIGHT, 2007, p. 37).

Os jogos não foram a única aplicação na qual a OpenGL foi bem adequada, imagem acelerada por hardware e aplicações de processamento de vídeo também foram. Isso faz com que em um futuro próximo a OpenGL seja a API escolhida para uma ampla gama de aplicações e plataformas de hardware (WRIGHT, 2007, p. 37).

Tudo isso também faz com que a OpenGL seja bem posicionada para tirar proveito das futuras inovações em gráficos 3D. Devido à extensão do mecanismo da OpenGL, os vendedores podem expor novas funcionalidades de hardware sem esperar a Microsoft ou alguma comissão da indústria, e desenvolvedores podem explorá-las logo que atualizações dos *drivers* forem disponíveis (WRIGHT, 2007, p. 37).

Com a inclusão da OpenGL *Shading Language*, a OpenGL tem demonstrado sua adaptabilidade permanente para atender os desafios de um desenvolvimento em *pipeline* de programação gráfica 3D. Finalmente, a OpenGL é uma especificação que mostrou que ela pode ser aplicada a uma grande variedade de paradigmas de programação. De C/C++ à Java e Visual Basic e ainda linguagens mais recentes como o C#, também já estão sendo usadas para criar jogos de computador e aplicações usando OpenGL (WRIGHT, 2007, p. 37).

## 4.2 Definição

OpenGL é definido estritamente como “um *software* de interface para *hardware* gráfico”. Essencialmente, é uma biblioteca de gráficos e modelagem 3D altamente portável e rápida. Usando a OpenGL, é possível criar gráficos 3D elegantes e bonitos com qualidade visual excepcional (WRIGHT, 2007, p. 37).

OpenGL não é uma linguagem de programação, como o C ou C++, ele é uma API (*Application Programming Interface*). Quando se diz que um programa está baseado em OpenGL ou é uma aplicação OpenGL, significa que foi implementado em alguma linguagem de programação (tal como C ou C++) que faz chamadas a uma ou várias das bibliotecas da OpenGL (WRIGHT, 2007, p. 34). Ferramentas CAD e programas de modelagem usados para criar personagens animados para o cinema são alguns exemplos de aplicações OpenGL (COHEN, 2006, p. 18).

OpenGL permite desde do desenho de primitivas gráficas, tais como, linhas e polígonos, até o suporte de iluminação, colorização, mapeamento de textura, transparência, animação, entre outros efeitos especiais (COHEN, 2006, p. 18).

## 4.3 Funcionamento e Utilização da API OpenGL

Para utilizar a OpenGL, ao invés de descrever a cena e como ela deve aparecer, o programador deverá prescrever as etapas necessárias para se conseguir algum efeito ou aparência. Essas "etapas" são construídas através de várias chamadas aos vários comandos da OpenGL (WRIGHT, 2007, p. 38).

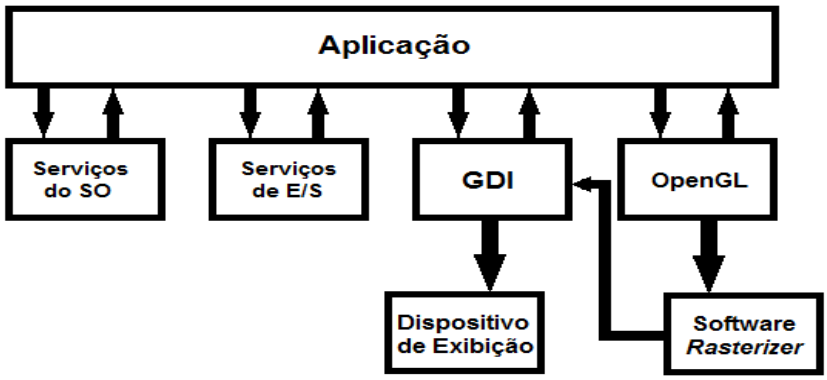
A OpenGL não inclui funções para gerência de janelas, nem para interação com o usuário e nem para entrada e saída de arquivos, então cada plataforma (tal como o Mac OS X ou o Microsoft Windows) deverá ter suas próprias funções para essa finalidade (WRIGHT, 2007, p. 38).

> Como uma API, OpenGL segue a convenção de chamada da linguagem C, isso significa que programas escritos em C podem facilmente chamar funções desta API, tanto porque estas foram escritas em C, como porque é fornecido um conjunto de funções C intermediárias que chamam funções escritas em *assembler* ou outra linguagem. Apesar da OpenGL ser uma biblioteca de programação padrão, existem muitas implementações desta biblioteca, como por exemplo para *Windows* e para *Linux*. A implementação utilizada no ambiente *Linux* é a biblioteca Mesa, que está disponível em http://www.mesa3d.org/. Também existem implementações para os compiladores *Borland C++*, *Dev-C++*, *Delphi* e *Visual Basic* (MANSSOUR, [S.d.], p. 01).

### 4.3.1 Implementação Genérica

Genericamente, uma execução do OpenGL pode tecnicamente funcionar em qualquer sistema, desde que o mesmo possa mostrar as imagens gráficas geradas. Na Figura 8 é mostrado o local típico onde a OpenGL e uma implementação genérica atuam quando uma aplicação do Windows está sendo executada (WRIGHT, 2007, p. 38).

Figura 8 – Local típico da OpenGL em uma aplicação




Fonte: Figura adaptada de WRIGHT (2007, p. 38).

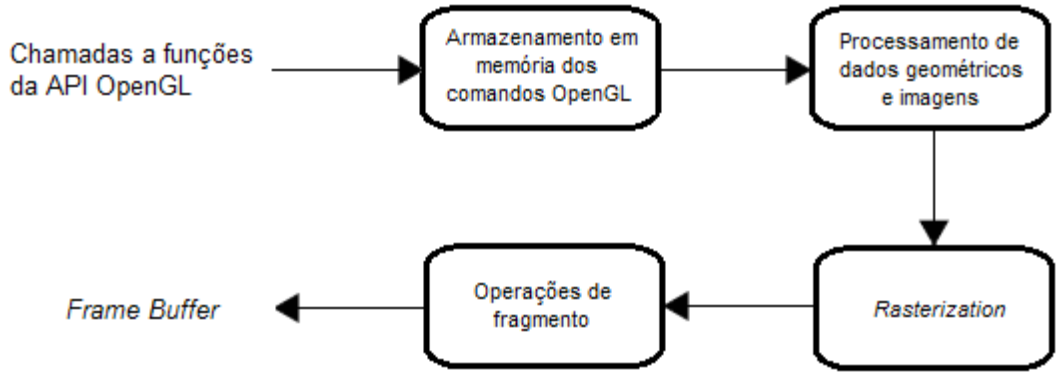
Um programa típico chama muitas funções, algumas que o próprio programador cria e algumas que são fornecidas pelo sistema operacional ou pela biblioteca da linguagem de programação em execução. As aplicações do Windows que querem criar uma saída de tela geralmente chamam uma API do Windows chamada de interface de dispositivos gráficos ou GDI (*graphics device interface*). O GDI contém os métodos que permitem que você escreva um texto em uma janela, simples linhas em 2D, e assim por diante (WRIGHT, 2007, p. 38).

### 4.3.2 O *pipeline* da OpenGL

A OpenGL trabalha como um *pipeline*, isso significa que ele executa um processo composto de duas ou mais etapas para a geração de uma imagem.

Quando a aplicação faz chamadas às funções da API OpenGL, os comandos são armazenados em uma memória específica chamada *buffer* de comandos. Então quando o *buffer* é preenchido com comandos de desenho de primitivas, iluminação e dados de textura, entre outros, esses comandos e dados são passados para o próximo estágio do *pipeline* (COHEN, 2006, p .19). Na Figura 9 é mostrada uma versão simplificada do *pipeline* da OpenGL.

Figura 9 – Versão simplificada do *pipeline* da OpenGL




Fonte: COHEN; MANSSOUR (2006, p. 19)

Na etapa de processamento de dados geométricos e imagens, os dados geométricos, tais como os vértices, são processados de forma diferente de dados de imagens, tais como *pixels*. Porém, após algumas etapas específicas, ambos passam pela etapa de *rasterization* do *pipeline*, que os converte em fragmentos. Um fragmento em OpenGL é uma posição na tela que contém cor e outras informações associadas, tais como profundidade e coordenadas de textura. Cada fragmento contribui para a atualização dos *pixels* do *frame buffer*, que corresponde à memória do dispositivo gráfico. Depois disso, várias operações podem ser executadas, tais como oclusão e transparência, antes que finalmente a imagem final seja exibida no monitor (COHEN, 2006, p .19).

### 4.3.3 Máquina de Estados

OpenGL é uma máquina de estados onde é possível colocá-la em vários estados (ou modos) que não são alterados, até que uma função seja chamada para isto. Por exemplo, a cor corrente é uma variável de estado, então todos os objetos são desenhados com essa cor, até que outra cor corrente seja definida (MANSSOUR, [S.d.], p. 03).

OpenGL tem várias variáveis de estado, tais como estilo de uma linha, posições e características das luzes, e propriedades do material dos objetos que estão sendo desenhados. Muitas delas podem ser habilitadas ou desabilitadas com os comandos *glEnable()* e *glDisable()*. Cada variável de estado possui um valor inicial que pode ser alterado. O trecho de programa mostrado na Figura 10 é um exemplo da utilização destas funções (MANSSOUR, [S.d.], p. 04).

Figura 10 – Trecho de um código exemplo

```
int luz;
:
glEnable(GL_LIGHTING); // Habilita luz - GL_LIGHTING é a variável de estado
:
luz = glIsEnabled(GL_LIGHTING); // retorna 1 (verdadeiro)
:
glDisable(GL_LIGHTING); // Desabilita luz
:
luz = glIsEnabled(GL_LIGHTING); // retorna 0 (falso)
:
```

Fonte: MANSSOUR ([S.d.], p. 04).

### 4.3.4 Tipo de dados OpenGL

Para que o código OpenGL fosse portável, foi necessário definir tipos de dados próprios para OpenGL. Estes tipos de dados são mapeados como os tipos de dados comuns da linguagem C, que também podem ser utilizados (MANSSOUR, [S.d.], p. 02).

Os compiladores e os ambientes possuem regras diferentes para determinar o tamanho das variáveis C, portanto usando os tipos OpenGL é possível separar o código das aplicações destas alterações (MANSSOUR, [S.d.], p. 02).

Na Tabela 2 são apresentados os tipos de dados OpenGL, os tipos de dados C

correspondentes e o sufixo apropriado, esses sufixos são usados para especificar os tipos de dados para as implementações ISO C de OpenGL. Pode-se observar que todos os tipos começam com "GL", e a maioria é seguido pelo tipo de dado C correspondente (MANSSOUR, [s.d.], p. 02).

Tabela 2 – Tipos de dados OpenGL

| Tipo de dado OpenGL | Representação interna | Tipo de dado C equivalente | Sufixo |
|---|---|---|---|
| GLbyte | 8-bit integer | signed char | b |
| GLshort | 16-bit integer | short | s |
| GLint, GLsizei | 32-bit integer | int ou long | i |
| GLfloat, GLclampf | 32-bit floating-point | float | f |
| GLdouble, GLclampd | 64-bit floating-point | double | d |
| GLubyte, GLboolean | 8-bit unsigned integer | unsigned char | ub |
| GLushort | 16-bit unsigned integer | unsigned short | us |
| GLuint, GLenum, GLbitfield | 32-bit unsigned integer | unsigned long ou unsigned int | ui |

Fonte: MANSSOUR ([S.d.], p. 02).

### 4.3.5 Principais Bibliotecas OpenGL

A OpenGL fornece um poderoso conjunto de comandos, porém primitivos. Então as rotinas de desenho de alto nível devem ser elaboradas em função destes comandos. E para simplificar a programação foram desenvolvidas algumas bibliotecas, das quais as principais são apresentadas a seguir:

**- GLU (*OpenGL Utility Library)***: essa biblioteca contém rotinas que utilizam os comandos OpenGL de baixo nível para executar tarefas como, por exemplo, definir as matrizes para projeção e orientação da visualização. Suas funções usam o prefixo **glu** (MANSSOUR, [S.d.], p. 03).

**- GLUT (*OpenGL Utility Toolkit)***: essa biblioteca é um *toolkit* independente de plataforma, que inclui alguns elementos GUI (*Graphical User Interface*), tais como menus *pop-up* e suporte para *joystick*. A GLUT esconde a complexidade das APIs dos diferentes sistemas de janelas. Suas funções usam o prefixo **glut** (MANSSOUR, [s.d.], p. 03).

# CAPÍTULO 5 – DESENVOLVIMENTO E IMPLEMENTAÇÃO DO SOFTWARE EDUCACIONAL DO JOGO TANGRAM

Neste capítulo são apresentadas as etapas de desenvolvimento do software, desde do seu objetivo até a sua codificação.

## 5.1 Especificação do Software

O Software Educacional do Jogo Tangram é um jogo em 2D para computador que simula o quebra-cabeça feito com o Tangram. Seu objetivo é facilitar o uso do Tangram como atividade lúdica pelos professores nas aulas de matemática para o ensino da geometria, pois o Tangram já está pronto e em formato de jogo de computador, tornando a atividade mais fácil e cativante para os alunos.

O Software Educacional do Jogo Tangram tem como público alvo alunos do Ensino Fundamental, crianças com idade entre 9 e 14 anos, por isso o mesmo foi desenvolvido com uma interface colorida e muito simples, e com jogabilidade fácil.

Foram usadas figuras tradicionais do Tangram para os desafios, em formatos conhecidos para as crianças.

### 5.1.1 Conteúdo

O jogo contém o desafio, que é composto pelas peças do tangram e a figura em sombra (preta), na parte de desafio ainda há a solução do desafio, onde o jogador poderá escolher se deixa vísivel ou não enquanto joga.

O jogo ainda traz um texto explicativo sobre os comandos necessários para jogar:

"Como Jogar:"

"1 - Arrastar a peça: clique na peça desejada e arraste com o botão esquerdo do mouse pressionado."

"2 - Girar a peça: clique na peça desejada com o botão direito do mouse para cada giro."

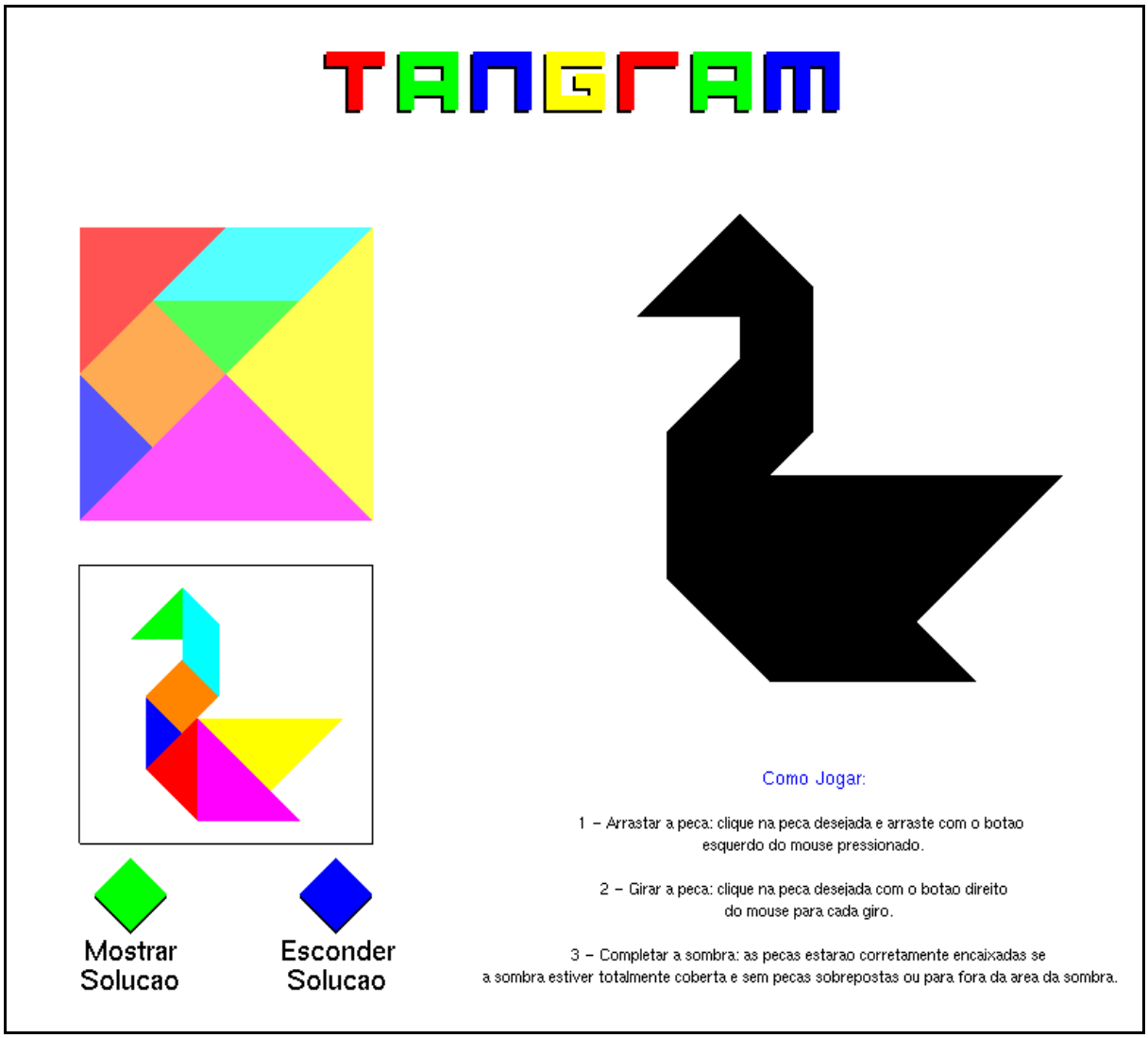
"3 - Completar a sombra: as peças estarão corretamente encaixadas se a sombra estiver totalmente coberta e sem peças sobrepostas ou para fora da área da sombra."

### 5.1.2 Interface do Jogo

A interface do jogo é simples, colorida e de fácil entendimento.

O protótipo do jogo mostra, em uma única tela, todas as funcionalidades do jogo, como é mostrado na Figura 11.

Figura 11 – Tela Principal do Protótipo do Jogo Tangram




A interface contém o título do jogo, as peças do tangram que serão utilizadas para montar o quebra-cabeça, a figura em sombra (preta) onde deverão ser encaixadas as peças, a solução do desafio, dois botões com as opções “Mostrar Solução” e “Esconder Solução” que podem ser ativados a qualquer momento durante o jogo e, o texto explicativo sobre os comandos para jogar.

### 5.1.3 Jogabilidade

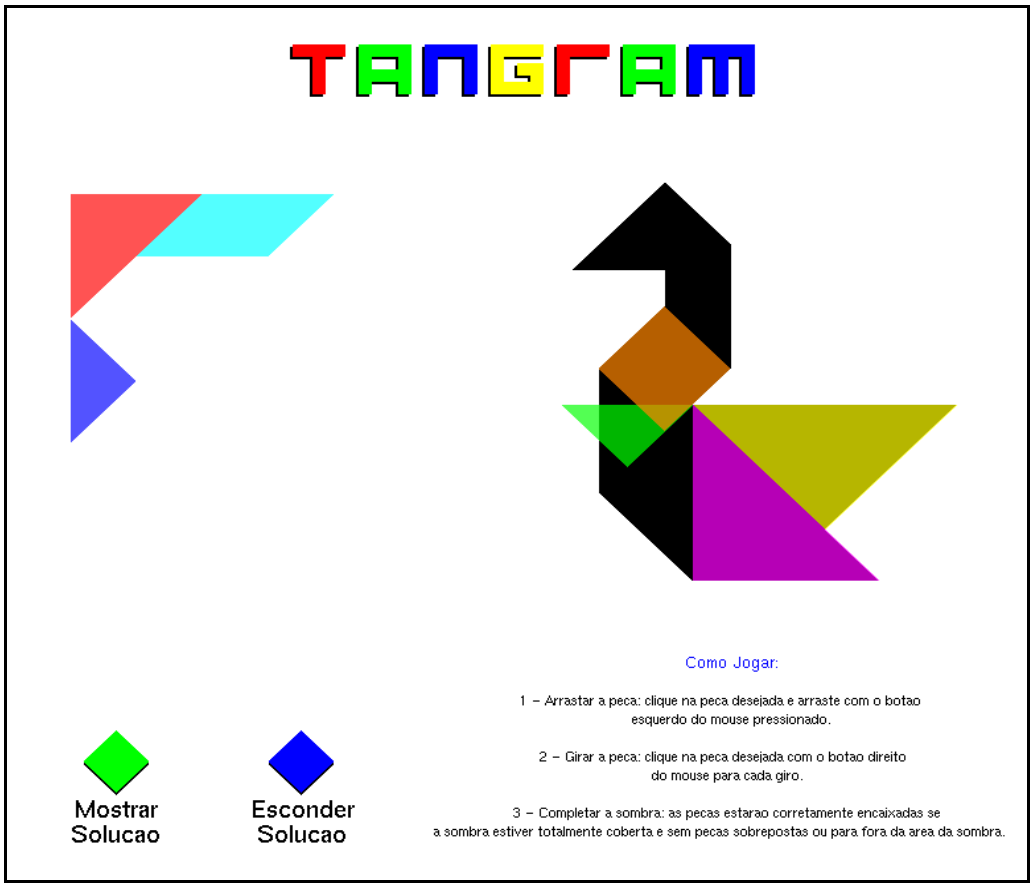
A jogabilidade é do modo “arrastar e soltar”, o jogador efetuará as ações sobre o jogo pelo mouse. Ao clicar na peça a mesma é selecionada, com o botão esquerdo pressionado a peça poderá ser movimentada, com o clique do botão direito sobre a peça a mesma é girada em 45º a cada clique.

O software também apresenta no título da janela o nome da peça que foi selecionada naquele momento pelo jogador.

### 5.1.4 Transparência das peças

As peças do Tangram são transparentes para que o próprio jogador consiga ver se completou a figura da maneira correta. Se há encaixes errados das peças, a transparência irá deixar visível ao jogador para que ele mesmo possa perceber o erro e se auto corrigir. Na Figura 12 é mostrado um exemplo do uso da transparência.

Figura 12 – Tela de exemplo da Transparência



## 5.2 Implementação do Jogo

O jogo Tangram foi desenvolvido em 2D utilizando a linguagem C em conjunto com a API OpenGL (Capítulo 4 - API de Desenvolvimento Gráfico OpenGL). Foi utilizada a IDE Dev-C++ e não foi necessário o uso de nenhuma *engine* de desenvolvimento de jogos, pois o motor do jogo tangram é facilmente implementado apenas com o uso da API OpenGL.

Abaixo são descritas cada etapa do desenvolvimento e trechos de códigos importantes para tal desenvolvimento.

### 5.2.1 Configuração da Janela

A janela do jogo foi configurada com a dimensão 775 x 700 *pixels* definida pela função *glutInitWindowsSize(int width, int hight)*. A projeção foi configurada usando a função *gluOrtho2D(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top)*, usada para determinar que a projeção ortográfica (2D) será utilizada para exibir na tela a imagem 2D que está na janela de seleção definida através dos parâmetros passados para esta função. No sistema de coordenadas cartesianas, os valores *left* e *right* especificam os limites mínimo e máximo no eixo X; analogamente, *bottom* e *top* especificam os limites mínimo e máximo no eixo Y (MANSSOUR, 2005), conforme é exibido na Figura 13.

Figura 13 – Código da Configuração da Janela

```
int main(void)
{
    glutInitDisplayMode(GLUT_DEPTH | GLUT_DOUBLE | GLUT_RGBA);
    glutInitWindowSize(viewW, viewH);
    glutInitWindowPosition(200,40);
    glutCreateWindow("Software Educacional - Tangram");
    Inicializa();
    glutMouseFunc(pickingPecas);
    glutDisplayFunc(Desenha);
    glutIdleFunc(Desenha);
    glutMainLoop();
}
```

```
int viewW = 775, viewH = 700;

void Inicializa(void)
{
    glViewport(0, 0, viewW, viewH);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    gluOrtho2D(0, viewW, viewH, 0);
    glMatrixMode(GL_MODELVIEW);
    glShadeModel(GL_SMOOTH);
    glClearColor(1.0, 1.0, 1.0, 1.0);
    glEnable(GL_COLOR_MATERIAL);
}
```

Outras configurações importantes foram feitas utilizando as funções: *glViewport(GLint x, GLint y, GLsizei width, GLsizei height)*, função que define a área dentro janela, em coordenadas de tela, que a OpenGL pode usar para fazer o desenho. A função *glMatrixMode(GLenum mode)* informa a OpenGL que todas as futuras alterações, tais como transformações geométricas, irão afetar os modelos da cena, ou em outras palavras, o que é desenhado. A função *glLoadIdentity(),* chamada em seguida, faz com que a matriz corrente seja inicializada com a matriz identidade (nenhuma transformação é acumulada). A função *glShadeModel(GLenum mode)* é usada para especificar a técnica de colorização (MANSSOUR, 2005).

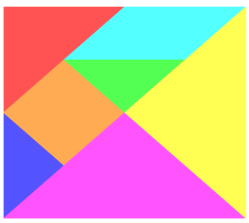
### 5.2.2 As Sete Peças do Tangram

Para desenhar as sete peças foram utilizadas as funções de desenho de primitivas da OpenGL: *glBegin(GL_TRIANGLES)* para os triângulos, *glBegin(GL_QUADS)* para os quadriláteros e *glBegin(GL_LINE_LOOP)* para a borda usada na solução do desafio. Na Figura 14 é mostrado o trecho de código da implementação de uma das peças.

Figura 14 – Trecho do Código de uma Peça

```
void Peca1(){
    //Desenho da Peça nº 1
    glPushMatrix();
    glTranslatef(mouse_x1+50, mouse_y1+150, 0.0);//transladar às coordenadas do
    glRotatef(ang1, 0.0, 0.0, 1.0);
    glLoadName(1);
    glBegin(GL_TRIANGLES);
        glColor4f(1.0, 0.0, 0.0, alfa_transp);
        glVertex2i(0, 100);
        glVertex2i(0, 0);
        glVertex2i(100, 0);
    glEnd();
    glPopMatrix();
}
```

Figura 15 – Resultado da Implementação das Sete Peças do Tangram



Na Figura 15 é mostrado o resultado da implementação das sete peças do Tangram, para cada peça foi aplicada uma cor própria, que não se repete, para que fosse fácil distingui-las umas das outras, usando a função *glColor4f(GLfloat red, GLfloat green, GLfloat blue, Glfloat alpha)*, que define a cor e o grau de transparência RGBA do desenho.

### 5.2.3 A Sombra

Para desenhar a figura sombra que deverá ser usada como base do quebra-cabeça, foram utilizadas cópias das peças codificadas acima, porém na cor preta e com ações de transformações geométricas para que ficassem no formato da figura desejada. Na Figura 16 é mostrado o trecho do código e na Figura 17 é mostrado o resultado da implementação de uma sombra.

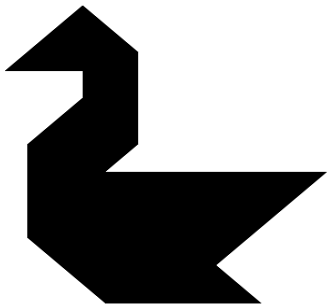
Figura 16 – Trecho do Código da Figura Sombra

```
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////        GANSO        ////////////////////////////////
////////////////////////////////////////////////////////////////////////////////

void DesenhaSombraGanso()
{
    glColor3f(0, 0, 0);

    //Desenho da Peça nº 1
    glPushMatrix();
    glTranslatef(326, 301, 0);
    glRotatef(-45.0, 0.0, 0.0, 1.0);
    glBegin(GL_TRIANGLES);
        glVertex2i(25, 250);
        glVertex2i(25, 150);
        glVertex2i(125, 150);
    glEnd();
    glPopMatrix();
```

Figura 17 – Resultado da Implementação da Figura Sombra



### 5.2.4 *Picking*

Após desenhadas as peças e a figura sombra, foi necessário utilizar uma técnica chamada *Picking* para que fosse possível selecionar os objetos. O *Picking* é a programação das funções necessárias para que seja possível selecionar objetos na cena e realizar operações com elas.

Primeiramente foi usada a função *glutMouseFunc(void (*func)(int button, int state, int x, int y)*, essa função captura os parametros do mouse e os envia para uma função que fará as operações do *Picking* utilizando esses parametros.

Depois, ao desenhar os objetos, deve-se dar nomes à eles, usando a função *glLoadName(GLuint name)*, uma pilha de nomes será preenchida com esses nomes e o *Picking* utilizará ela para reconhecer os objetos selecionados.

Após isso, é feita a função onde o *Picking* será configurado e utilizado. Conforme é mostrado na Figura 18.

Figura 18 – Trecho do Código da Função do *Picking*

```
//Função chamada pela ação do mouse para selecionar as peças
#define BUFFER_LENGTH 512
void pickingPecas(int botao, int estado, int x, int y){

    GLint i, j, index, nitems, item;
    GLint hits, viewport[4];
    char cMessage[64];
    strcpy(cMessage,"Erro, nenhuma seleção detectada!");
    static GLuint selectBuff[BUFFER_LENGTH];

    ////////////////////////////////////////////////////////////////////////////////
    ////////////////      FUNCIONALIDADE "CLICAR E ARRASTAR"      ////////////////////
    ////////////////////////////////////////////////////////////////////////////////

    if (botao == GLUT_LEFT_BUTTON || estado != GLUT_DOWN){

            glGetIntegerv(GL_VIEWPORT, viewport);
            glSelectBuffer(BUFFER_LENGTH, selectBuff);
            glRenderMode(GL_SELECT);
            glInitNames();
            glPushName(0);
            glPushMatrix();
            glLoadIdentity();
            gluPickMatrix((GLdouble) x, (GLdouble) (viewport[3] - y), 0.5, 0.5, viewport);
            gluOrtho2D (0.0, viewW, viewH, 0.0);
            DesenhaTangram();
            glPopMatrix();
            hits = glRenderMode(GL_RENDER);
            if(hits == 0){
                glRenderMode(GL_SELECT);
                Desenha();
                hits = glRenderMode(GL_RENDER);
            }else{
                item = selectBuff[3];

                    switch(item){
                            case 1:
                                strcpy(cMessage,"Você selecionou o TRIÂNGULO MÉDIO");
                                glutMotionFunc(MovePeca1);
                                break;
```

Na função do *Picking*, é criado um buffer de seleção, *glSelectBuffer(GLsizei size, GLuint *buffer)*, onde ficarão armazenados os dados dos objetos selecionados, depois é ativado o modo de renderização, *glRenderMode(GLenum mode)*, no modo GL_SELECT, para reconhecer seleções. A pilha de nomes deverá ser inicializada com as funções *glInitNames()* e *glPushName(0)*. A área de seleção do *Picking* é definida usando a função *gluPickMatrix(GLdouble x, GLdouble y, GLdouble delX, GLdouble delY, GLint *viewport)*. A função na qual estão desenhados os objetos que sofrerão ação do *Picking* deve ser chamada logo em seguida. Quando o *hits* é diferente de 0 significa que pelo menos um objeto foi selecionado e dados dessa seleção foram armazenados no buffer de seleção, somente assim poderemos acessar o buffer, capturar o nome do objeto e utilizá-lo para algum fim, como por exemplo, no código acima, caso o objeto de nome 1 seja selecionado, o mesmo sofrerá uma ação pela função *glutMotionFunc(Movepeça1)*.

### 5.2.5 Movimento e Rotação das Peças

Após a implementação do *Picking*, foram implementadas as funções de movimento e rotação da peça. Para arrastar a peça é necessário clicar na peça e arrastá-la com o botão esquerdo do mouse pressionado, para tanto, primeiramente o programa identifica se a peça foi selecionada com o botão esquerdo, conforme mostra a linha de código: *if (botão == GLUT_LEFT_BUTTON || estado != GLUT_DOWN)*, após selecionada o *Picking* identifica a peça e então é chamada a função *glutMotionFunc(void (*func)(int x, int y))*, essa função ativa quando um botão do mouse está pressionado; então passa-se como parâmetro a função *Movepeça1(int x, int y)*, que substitui os valores de X e Y da translação da peça pelos valores de X e Y do mouse, por fim fazendo com que ela seja transladada conforme as coordenadas do mouse: *glTranslatef(mouse_x1+50, mouse_y1+150, 0.0)*.

Foram implementadas cada uma dessas funções para cada peça. Na Figura 19 são mostrados trechos de código usados na implementação do movimento.

Figura 19 – Trechos do Código da Função de Movimento da Peça

```
////////////////////////////////////////////////////////////////////////////////
////////////////      FUNCIONALIDADE "CLICAR E ARRASTAR"      //////////////////
////////////////////////////////////////////////////////////////////////////////

if (botao == GLUT_LEFT_BUTTON || estado != GLUT_DOWN){
```

```
                item = selectBuff[3];

                    switch(item){
                            case 1:
                                strcpy(cMessage,"Você selecionou o TRIÂNGULO MÉDIO");
                                glutMotionFunc(MovePeca1);
                                break;
```

```
////////////////////////////////////////////////////////////////////////////////
////////////////////      FUNÇÕES PARA MOVER AS PEÇAS      /////////////////////
////////////////////////////////////////////////////////////////////////////////

void MovePeca1(int x, int y){

     mouse_x1 = (x-50);
     mouse_y1 = (y-150);
}
```

```
////////////////////////////////////////////////////////////////////////////////
/////////        FUNÇÕES PARA DESENHAR AS PEÇAS DO TANGRAM        ///////////////
////////////////////////////////////////////////////////////////////////////////
void Peca1(){
    //Desenho da Peça nº 1
    glPushMatrix();
    glTranslatef(mouse_x1+50, mouse_y1+150, 0.0);//transladar às coordenadas do ponteiro do mouse
    glRotatef(ang1, 0.0, 0.0, 1.0);
    glLoadName(1);
    glBegin(GL_TRIANGLES);
        glColor4f(1.0, 0.0, 0.0, alfa_transp);
        glVertex2i(0, 100);
        glVertex2i(0, 0);
        glVertex2i(100, 0);
    glEnd();
    glPopMatrix();
}
```

Para rotacionar a peça é necessário clicar na peça com o botão direito do mouse, sendo um clique para cada giro de 45º, para tanto, primeiramente para dar certo o cálculo de rotação, foi necessário desenhar as peças com pelo menos um vértice na coordenada da origem (0,0). Após isso, o programa identifica se a peça foi selecionada com o botão direito, conforme mostra a linha de código: *if (botão == GLUT_RIGHT_BUTTON || estado != GLUT_DOWN)*, após selecionada o *Picking* identifica a peça e então o valor do ângulo *ang1* é alterado para que a peça seja rotacionada em 45º através da função *glRotatef(ang1, 0.0, 0.0 1.0)*, nota-se que há uma *flag clique1* que controla a quantidade de cliques dados na peça, que ao chegar no oitavo clique, onde *ang1* = 315°, a contagem de *ang1* seja zerada para recomeçar o ciclo de rotações.

Foram implementadas cada uma dessas funções para cada peça. Na Figura 20 são mostrados os trechos de código usados na implementação da rotação.

Figura 20 – Trechos do Código da Função de Rotação da Peça

```
////////////////////////////////////////////////////////////////////////////////
//////////////////////      FUNCIONALIDADE "GIRAR"      ////////////////////////
////////////////////////////////////////////////////////////////////////////////

}else if (botao == GLUT_RIGHT_BUTTON || estado != GLUT_DOWN){
```

```
                item = selectBuff[3];

                    switch(item){
                            case 1:
                                strcpy(cMessage,"Você selecionou o TRIÂNGULO MÉDIO.");
                                if(clique1 == 8){
                                   ang1 = 0.0;
                                   clique1 = 1.0;
                                }else{
                                   ang1 = ang1-45;
                                   clique1++;
                                }
                                break;
```

```
////////////////////////////////////////////////////////////////////////////////
/////////        FUNÇÕES PARA DESENHAR AS PEÇAS DO TANGRAM        ///////////////
////////////////////////////////////////////////////////////////////////////////
void Peca1(){
    //Desenho da Peça nº 1
    glPushMatrix();
    glTranslatef(mouse_x1+50, mouse_y1+150, 0.0);//transladar às coordenadas do ponteiro do mouse
    glRotatef(ang1, 0.0, 0.0, 1.0);
    glLoadName(1);
    glBegin(GL_TRIANGLES);
        glColor4f(1.0, 0.0, 0.0, alfa_transp);
        glVertex2i(0, 100);
        glVertex2i(0, 0);
        glVertex2i(100, 0);
    glEnd();
    glPopMatrix();
}
```

### 5.2.6 Criando o Protótipo

Para finalizar, foi montado o protótipo com um desafio. Para tanto, foi necessário primeiramente desenhar o título do jogo no topo da tela, depois desabilitar o efeito de transparência: *glDisable(GL_BLEND)*, para chamar a função que desenha a figura sombra (preta): *DesenhaSombraGanso()*, a função que desenha a solução do desafio: *DesenhaSombraGansoSolucao()* e a função que desenha o texto explicativo de ajuda: *Ajuda()*. Ambas as funções não necessitam do efeito de transparência. Após isso, foi habilitado o efeito de transparência para as peças do Tangram: *glEnable(GL_BLEND)* e *glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA)*, para chamar a função que desenhará cada peça do Tangram e os botões: *DesenhaTangram()*.

Na Figura 21 são mostrados os trechos de código usados na implementação da tela principal do protótipo, cujo o resultado é mostrado na Figura 16.

Figura 21 – Trechos dos Códigos que Implementam a Tela Principal do Protótipo

```
////////////////////////////////////////////////////////////////////////////////
//////////////////////       FUNÇÃO QUE DESENHA A TELA     //////////////////////
////////////////////////////////////////////////////////////////////////////////

void Desenha(){
     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

     //desenho do Título - titulo.h
     glPushMatrix();
        glScalef(0.5,0.5,0.5);
        glTranslatef(393, 13, 0);
        glColor3f(0,0,0);
        DesenhaTituloSombra();
     glPopMatrix();
     glPushMatrix();
        glScalef(0.5,0.5,0.5);
        glTranslatef(396, 10, 0);
        DesenhaTitulo();
     glPopMatrix();

     //desenho opaco - Ganso Preto
     glDisable(GL_BLEND);
     DesenhaSombraGanso();
     DesenhaSombraGansoSolucao();
     Ajuda();

     //desenho transparente - Peças do Tangram
     glEnable(GL_BLEND);
     glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
     DesenhaTangram();
     glutSwapBuffers();
}
```

```
void DesenhaTangram(){
    /*As Peças de 1 a 7 estão especificadas de acordo com a Figura 1
    no Cap. 2 - O Jogo Tangram*************************************/
    Peca1();
    Peca2();
    Peca3();
    Peca4();
    Peca5();
    Peca6();
    Peca7();
    BotaoMostrarSolucao();
    BotaoEsconderSolucao();
}
```

# CONCLUSÕES

Com os estudos realizados foi possível notar a vantagem do uso de softwares educacionais na educação, o computador facilita o uso de atividades de ensino-aprendizagem aplicados pelo professor e motiva o aluno ao mesmo tempo em que ensina. Porém um software educacional deve ser projetado e desenvolvido com qualidade, pois um software educacional sem um boa base pedagógica pode acabar servindo apenas de distração para o aluno e um software educacional sem interface de qualidade pode fazer com que o aluno não se motive à utilizá-lo. Dependendo do tipo, do enredo e da qualidade do software educacional, é possível sim ele ser uma poderosa ferramenta para o ensino.

Para este trabalho foi muito importante o estudo dos conceitos necessários para planejar como seria o desenvolvimento do jogo Tangram. Foram abordados desde conceitos de softwares educacionais até aos conceitos de desenvolvimentos de jogos. A partir desse estudo foi possível criar o jogo Tangram com qualidade, mediante às características de um bom software educacional.

A fase de implementação foi a mais importante, nessa fase foi intensamente estudada a API de desenvolvimento gráfico OpenGL e teorias de Computação Gráfica. O jogo foi dividido em partes e implementadas uma a uma, para cada parte foi necessário o estudo de uma determinada técnica, tal como o *picking*, transformação geométrica, transparência e interação com o mouse. Os estudos destes conceitos computacionais para desenvolvimento de jogos foi bastante compensador.

Ao final foi desenvolvido com sucesso um protótipo com a maioria das funcionalidades pretendidas para o software do jogo Tangram. Para trabalhos futuros, seria interessante dar continuidade no desenvolvimento do jogo Tangram incluindo implementações de menus, novos desafios, banco de imagens, sons, animações e realizar testes com alunos para avaliar a sua qualidade.

# REFERÊNCIAS

ARRUDA, Francislene Aparecida Oliveira; ALMEIDA, Vera Lia Marcondes Criscuolo de. **Os jogos tangram e dominó geométrico como estratégia para o ensino da geometria.** São Paulo: UNESP, [S.d.].

ASTLE, Dave; HAWKINS, Kevin. **Beginning OpenGL® Game Programming.** United States of America**:** Thomson Course Tecnology, 2004.

AZEVEDO, Eduardo; CONCI, Aura. **Computação Gráfica:** Teoria e Prática. Rio de Janeiro: Campus, 2003.

BATTAIOLA, André Luiz. **Jogos por Computador:** Histórico, Relevância Tecnológica e Mercadológica, Tendências e Técnicas de Implementação. São Carlos: Departamento de Computação da Universidade Federal de São Carlos, 2003.

**BRASILESCOLA**. Disponível em: <http://www.brasilescola.com/upload/e/tangram10.jpg>. Acesso em: julho 2010.

CAVACO, Marcelo. **Desenvolvimento de Jogos:** Programação de Jogos. [S.l], 2007.

COHEN, Marcelo; MANSSOUR, Isabel Harb. **OpenGL:** Uma Abordagem Prática e Objetiva. São Paulo: Novatec, 2006.

FANTI, Ermínia de Lourdes Campello; SILVA, Aparecida Francisco da. **Informática e Jogos no Ensino da Matemática.** Bahia: II Bienal da Sociedade Brasileira de Matemática, 2004.

GLADCHEFF, Ana Paula; ZUFFI, Edna Maura; SILVA, Dilma Menezes da. **Um Instrumento para Avaliação da Qualidade de Softwares Educacionais de Matemática para o Ensino Fundamental.**[S.l.], 2001.

GOMES, Alex Sandro; WANDERLEY, Eduardo Garcia. **Elicitando requisitos em projetos de software educativo.** Campinas: WIE'2003, 2003.

LUCENA, Marisa. **Diretrizes para a capacitação do professor na área de tecnologia educacional:** critérios para a avaliação de software educacional. Rio de Janeiro: PUC-Rio, 2002.

LUZ, Mairlo Hideyoshi Guibo Carneiro da. **Desenvolvimento de Jogos de Computador.** Itajubá: Universidade Federal de Itajubá – UNIFEI, 2004.

MANSSOUR, Isabel Harb. **Introdução à OpenGL.** Porto Alegre**:** Faculdade de Informática da PUCRS, [S.n. : S.d.].

MANSSOUR, Isabel Harb. **Tutorial Online - Introdução à OpenGL.** Porto Alegre**:** Faculdade de Informática da PUCRS, 2005. Disponível em: <http://www.inf.pucrs.br/~manssour/OpenGL/ Tutorial.html>. Acesso em: 10 novembro 2010.

MOTTA, Ivany Aparecida Rodrigues da. **Tangram.** Guaratinguetá: Escola Estadual Professor Luiz Menezes, 2006.

PORTO, Irlaine da P. Gomes; CARVALHO, Carlos V. de A.; OLIVEIRA, Rosana de. **O Jogo Computacional Tangram:** Um objeto de aprendizagem sobre geometria. Rio de Janeiro: Universidade Severino Sombra, 2008.

SEDDON, Chris. **OpenGL® Game Development.** United States of America**:** Wordware Publishing, Inc. 2005.

TAROUCO, Liane Margarida Rockenbach et al. **Jogos educacionais**.[S.l.], CINTED-UFRGS – Novas Tecnologias na Educação, 2004.

VALENTE, José Armando et al. **O Computador na Sociedade do Conhecimento**. Campinas: UNICAMP, 1999.

WRIGHT, Richard S. Jr.; LIPCHAK, Benjamin; HAEMEL, Nicholas. **OpenGL® SuperBible**. 4th. ed. United States of America**:** Pearson Education, Inc. 2007.