

# 1. Introdução

*Este capítulo apresenta as áreas, as aplicações, os profissionais e as ferramentas de programação utilizadas na Computação Gráfica. Sugerimos uma leitura rápida para se familiarizar com a nomenclatura da área. É muito importante o exercício de implementação de um programa fornecido que tem como objetivo exemplificar o uso do OpenGL e da GLUT.*

A Computação Gráfica acompanha a evolução dos computadores na sociedade e também teve um começo restrito e pouco natural para hoje se tornar presente em praticamente todas as nossas atividades. Se observarmos a difusão dos computadores vemos que a Computação Gráfica tem tido um papel decisivo nela. As interfaces gráficas na forma de janelas, botões, menus e dispositivos de seleção<sup>1</sup>, por exemplo, foram muito importantes para a popularização dos computadores pessoais da década de 1980. Difícilmente esta difusão dos computadores teria ocorrido não fossem estas interfaces gráficas. Atualmente as câmeras digitais acopladas a computadores portáteis de mão e a telefones móveis ainda estão impulsionando novos usos da informática no nosso dia a dia.

## Áreas da CG

A área de entretenimento com os jogos eletrônicos e os efeitos especiais do cinema digital é, provavelmente, a face mais conhecida e sedutora da Computação Gráfica para o público em geral. A Computação Gráfica, entretanto, não se restringe ela. Ela está presente nas atividades de projeto e construção de carros, casas e plantas industriais. A Fig. 1.1 ilustra um modelo de uma plataforma marítima de petróleo, acrescida de efeitos artísticos para divulgação.



Fig. 1.1- Modelo de uma plataforma marítima de petróleo.

<sup>1</sup> Estas interfaces, denominadas WIMP para *windows, icons, menus and pointer devices*, apareceram na Xerox e se popularizaram com as diversas versões do sistema operacional Windows da MicroSoft.

Outras aplicações importantes podem ser encontradas nas áreas de imagens médicas, cirurgia com realidade aumentada, supervisão e controle, e simulação na Engenharia, Geologia, Sísmica, entre muitas. Uma boa maneira de estudarmos uma área tão ampla como a Computação Gráfica consiste em dividi-la de forma a compreender suas partes e a estrutura do todo.

Inicialmente a Computação Gráfica foi dividida em três grandes áreas: síntese de imagens (*redenring*), processamento de imagens e visão computacional. Estas áreas foram categorizadas em função da entrada e saída dos seus algoritmos como ilustra a Fig. 1.2.

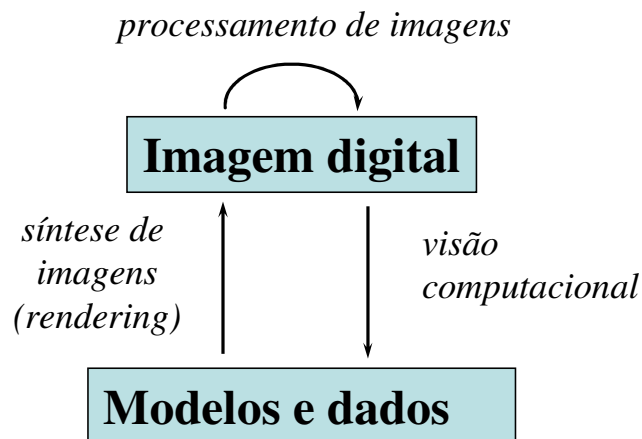


Fig. 1.2 – Divisão clássica das áreas da Computação Gráfica.

### *Síntese de imagens (rendering)*

Os algoritmos de síntese de imagens produzem imagens digitais a partir de dados e modelos computacionais de objetos, luzes e câmera. De certa forma são as máquinas fotográficas do mundo virtual. Tiram fotos dos objetos que estão lá. A Fig. 1.3 mostra, por exemplo, uma imagem de uma mesa de sinuca tirada de uma implementação do trabalho proposto no Capítulo de Rastreamento de Raios.

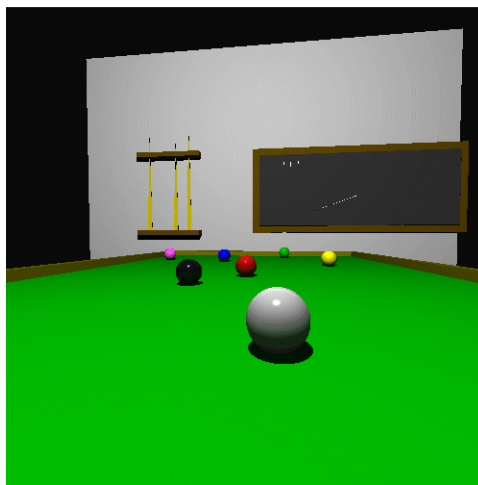


Fig. 1.3 – Imagem pelo algoritmo de Rastreamento de Raios

Os algoritmos de síntese de imagens são fundamentais para sistemas de Realidade Virtual. A Fig. 1.4 ilustra um destes sistemas onde o modelo de uma plataforma marítima está sendo examinado com o auxílio de óculos que permitem que cada um dos olhos receba uma imagem correspondente a sua posição, reproduzindo o efeito de estereoscopia e criando uma sensação de imersão no mundo virtual.

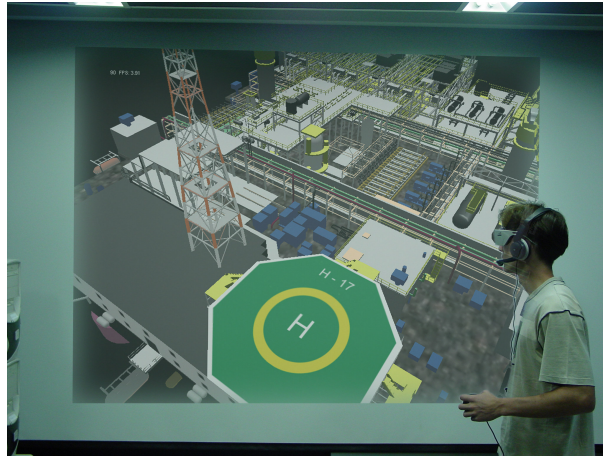
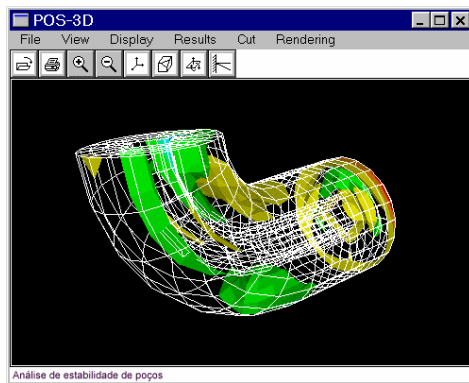
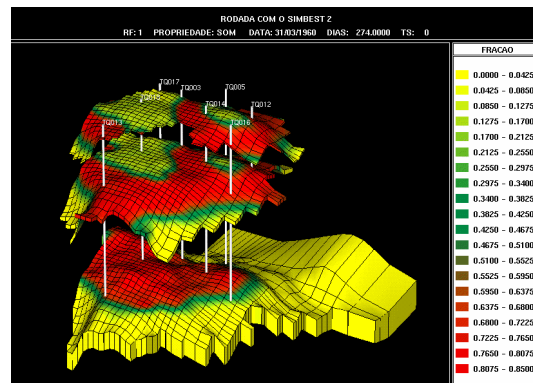


Fig. 1.4- Síntese de imagens.

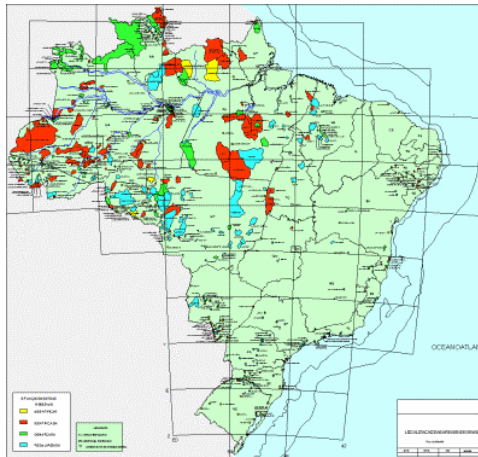
O objetivo dos algoritmos de síntese de imagens pode ser o realismo visual, como ilustra a Fig. 1.4, ou a transmissão de informações técnicas como mostram as visualizações científicas da Fig. 1.5. Nesta figura os programas de: (a) geotecnia, (b) engenharia de reservatórios, (c) cartografia, (d) estruturas navais, (e) sísmica, e (f) medicina, tem fortes requisitos de precisão geométrica mas tratam de objetos abstratos que fogem a experiência visual de pessoas leigas.



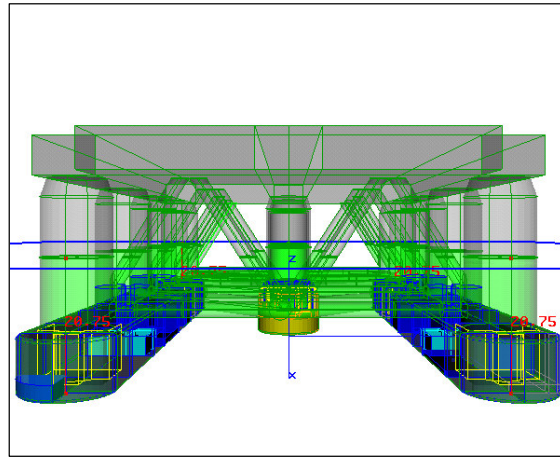
(a) poço horizontal



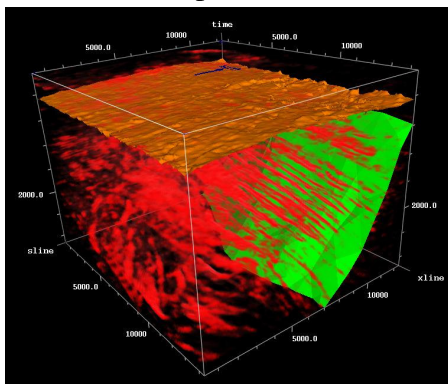
(b) reservatório de petróleo



(c) mapa temático



(d) estrutura marítima



(e) sísmica



(f) exames médicos

Fig. 1.5- Síntese de imagens para visualização científica.

### *Processamento de imagens*

Os algoritmos de processamento de imagens modificam uma imagem para melhorar sua qualidade, eliminando ruídos e realçando características que facilitem o seu entendimento. Estes algoritmos não mudam a natureza dos dados, ou seja, eles processam imagens de entrada e geram imagens de saída. A Fig. 1.6 ilustra a este processamento sobre uma imagem médica de um cérebro humano. O objetivo do processamento ilustrado consiste em realçar as regiões do cérebro para torná-las mais visíveis.

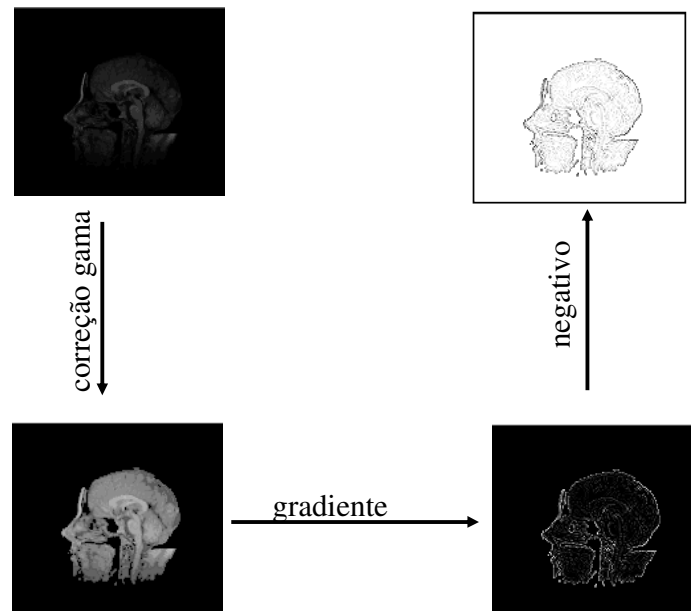


Fig. 1.6 - Processamento de imagens.

### *Visão computacional*

A visão computacional processa imagens visando obter informações sobre objetos presentes nela. Ou seja, interpretam o que está na imagem extraindo dela informações não puramente visuais. A Fig. 1.7 ilustra, por exemplo, um algoritmo capaz de “ler” o número de uma placa de automóvel a partir de uma imagem dela.

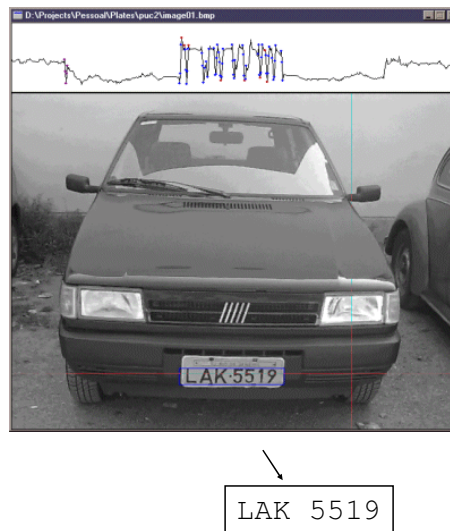


Fig. 1.7 – Visão Computacional.

### *Realidade aumentada*

As aplicações mais recentes da Computação Gráfica não se encaixam em apenas uma das classes de algoritmos apresentada acima: síntese, processamento e visão. Os sistemas de realidade aumentada que misturam elementos virtuais em imagens reais são os exemplos



mais marcantes desta nova classe de aplicações da Computação Gráfica. A Fig. 1.8 mostra, no lado esquerdo um jogo da velha onde o computador “vê” onde o usuário riscou o jogo e as suas jogadas e responde colocando na tela a imagem da câmera acrescida de objetos virtuais.

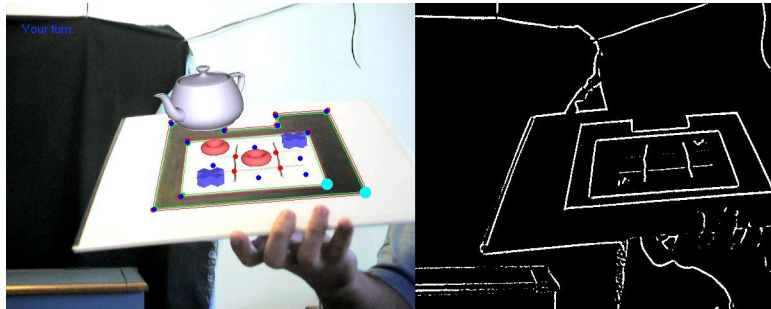


Fig. 1.8 – Jogo da velha com realidade aumentada

O programa da Fig. 1.8 tem três fases. Na primeira o programa captura cada quadro da câmera de vídeo, processa as imagens para reduzir os ruídos e realça as linhas da figura obtendo a imagem mostrada do lado direito da Fig. 1.8. Ou seja, é uma fase de Processamento de Imagens.

Na segunda fase, o programa reconhece onde está a câmera e quais são seus parâmetros pela deformação da moldura preta que envolve a área onde o jogo ocorre. Reconhece também onde estão as linhas do jogo e as jogadas do usuário. É uma fase típica de Visão Computacional.

Finalmente o computador sintetiza novos elementos na cena: as peças do jogo, as marcas dos cantos que fazem parte do algoritmo de visão e o bule de chá que não tem nada a ver com o contexto do jogo. É uma fase de sínteses de imagens.

A Fig. 1.9 ilustra um outro programa de Realidade Aumentada que serve para acompanhar jogos de esportivos de campo ou quadra. Na fase de processamento de imagens o sistema descobre onde estão as linhas do campo. Na fase de visão o sistema acha as equações destas linhas e a partir delas encontra o modelo de uma câmera que as “vê” daquela forma distorcida. Na ultima fase o sistema acrescenta elementos na imagem para torná-la mais clara. Na imagem do lado direito da Fig. 1.9(a) o sistema realça as linhas de campo e a trave para ilustrar a precisão com que ele está capturando estes objetos. Na Fig. 1.9(b) vemos uma verificação de impedimento com marcação das linhas de profundidade dos jogadores mais importantes na jogada.



(a) calibração de câmeras



(b) medições no campo.

Fig. 1.9 – Acompanhamento de jogos esportivos

Concluindo esta seção devemos dizer que a divisão da Computação Gráfica ilustrada na Fig. 1.2 ainda persiste, mas é cada vez mais comum encontrarmos programas que não se encaixam em apenas uma destas áreas. Uma divisão mais atual seria a ilustrada na Fig. 1.10 que introduz a área de Realidade Aumentada nas áreas clássicas da Computação Gráfica.

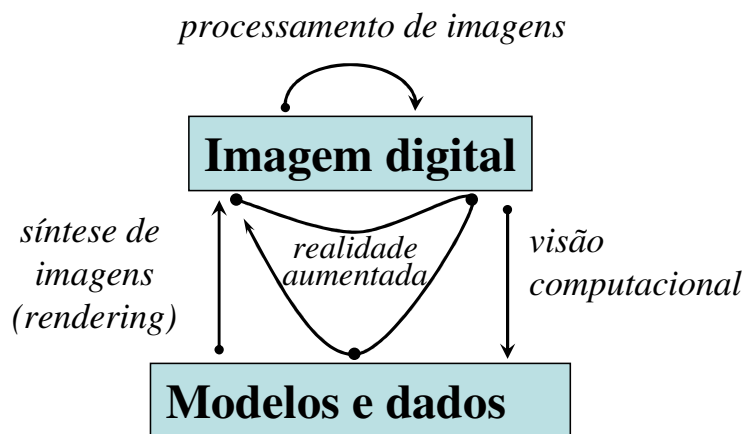


Fig. 1.10 – Áreas da CG incluindo Realidade Aumentada

## ***Requisitos de realismo e de eficiência***

Uma outra divisão importante da Computação Gráfica diz respeito aos requisitos de realismo e eficiência que o programa deve ter. Na produção de filmes, por exemplo, é comum utilizarmos grades de computadores trabalhando por longos períodos para gerar cada um dos quadros de uma animação. O importante nesta tarefa é a qualidade da imagem gerada. O tempo um é fator menos crítico. Programas que produzem vídeos e filmes são geralmente baseados no algoritmo de Rastreamento de Raios, que apesar de não ter baixa eficiência produz imagens bastante realistas.

Em contrapartida a produção de filmes, nós temos os jogos de computador e os programas de Realidade Virtual, onde o usuário controla interativamente o que ele está vendo. A Fig. 1.4 ilustra o uso de um mouse 3D para controlar o “vôo” do usuário no modelo da plataforma. Programas como estes precisam gerar as imagens num tempo muito pequeno para que não se quebre a sensação de imersão do usuário no mundo virtual. Estes programas normalmente utilizam Sistemas Gráficos, como o OpenGL™, que são baseados no algoritmo de Mapa de Profundidade (*ZBuffer*). Este algoritmo é bem mais eficiente que o algoritmo de Rastreamento de Raios mas produz resultados com menor realismo visual.

Estas duas linhas de algoritmos são importantes na Computação Gráfica e são tratadas neste livro. É interessante notarmos que cada um destes algoritmos tem evoluído de forma superar suas deficiências. O algoritmo de Rastreamento de Raios já tem implementações eficientes e os jogos de computador conseguem imagens bastante realistas. A Fig. 1.11 mostra uma imagem de um modelo de uma plataforma gerada por um programa interativo feito com o OpenGL™ que utiliza o algoritmo de mapa de profundidade. Se compare a qualidade desta imagem com a mostrada na Fig. 1.1, obtida por um programa que utiliza o Rastreamento de Raios, vemos que não são tão diferentes.

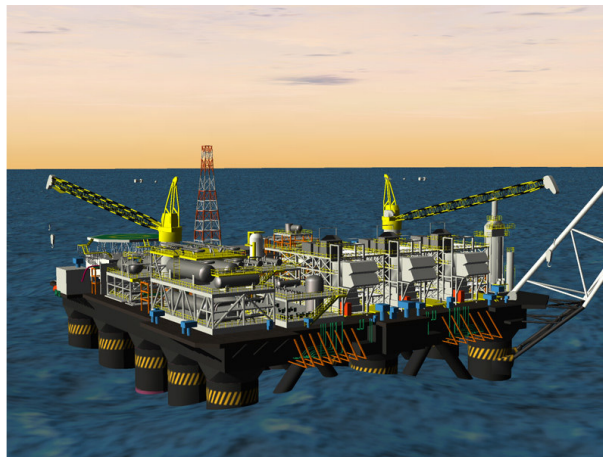


Fig. 1.10 – Imagem gerada com o algoritmo de mapa de profundidade.

Os jogos de computador também produzem imagens muito boas, mas quando o programa não está respondendo a eventos do usuário e passa uma animação em vídeo podemos perceber claramente a melhora de qualidade.



## ***Profissionais da Computação Gráfica***

A Computação Gráfica envolve diversos tipos de profissionais. Alguns precisam de mais do outros dos conhecimentos dos fundamentos da Computação Gráfica. Os fundamentos são entendidos aqui não só como os modelos físicos e matemáticos que dão suporte a disciplina mas principalmente as classes de algoritmos que implementam as tarefas de síntese, processamento e visão computacional.

Sob o ponto de vista de desenvolvimento de programas os profissionais envolvidos na Computação Gráfica são de quatro tipos: usuários, customizadores, programadores de aplicações e desenvolvedores de ferramentas.

Os **usuários** constituem um grupo grande e variado. Suas características dependem da área do conhecimento onde a Computação Gráfica está sendo utilizada. Se o sistema for de interpretação Sísmica os usuários são geralmente Geofísicos, se for de exames médicos os usuários são médicos, e assim por diante. O conhecimento dos fundamentos da Computação Gráfica não é um requisito importante para estes profissionais.

Um destaque especial deve ser dado aos usuários que produzem material áudio visual utilizando programas como o 3ds max da Autodesk™. A qualidade dos vídeos gerados por estes usuários depende, em grande parte, do conhecimento deles na área da Computação Gráfica propriamente dita. Uma boa noção dos modelos e algoritmos ajuda bastante a criação artística destes profissionais.

Os **customizadores** adaptam os programas existentes, através das linguagens de extensão, escrevendo procedimentos que visam criar efeitos especiais ou automatizar tarefas repetitivas. Eles precisam de um conhecimento básico dos fundamentos da Computação Gráfica.

Os **programadores de aplicação** desenvolvem programas gráficos utilizando sistemas como o OpenGL™. Nesta categoria se enquadram os profissionais que escreveram os programas ilustrados nas figuras deste capítulo. Eles certamente precisam de sólidos conhecimentos dos fundamentos da Computação Gráfica.

Os **desenvolvedores de ferramentas** escrevem os sistemas do tipo OpenGL™ ou DirectX da Microsoft™. Eles necessitam não só do conhecimento dos fundamentos da Computação Gráfica, como também de uma sólida formação na área da Ciência da Computação em geral.

O material deste livro enfoca estes fundamentos da Computação Gráfica e procura explicá-los a partir de requisitos mínimos de Matemática e Programação. Como a maioria dos profissionais envolvidos na Computação Gráfica tem estes conhecimentos, poderíamos dizer que o livro pode ser interessante para todos os profissionais citados acima. Este material, entretanto, foi preparado para servir de base para um curso de Computação Gráfica num programa de graduação em Engenharia da Computação ou em de Informática. Esperamos que os leitores de áreas técnico-científicas, como as Engenharias, que estejam interessados em desenvolver programas de aplicação também achem o enfoque apresentado interessante.

## Ferramentas de programação

O desenvolvimento de um programa que tenha uma interface gráfica interativa requer duas ferramentas básicas: um Sistema Gráfico e um Sistema de Interface com o Usuário. O primeiro para gerar imagens em janelas gerenciadas pelo segundo. Ou seja, o sistema de interface gerencia os elementos do tipo de janelas, menus e botões enquanto que o sistema gráfico desenha primitivas geométricas do tipo pontos, linhas e polígonos e/ou imagens digitais nos elementos de interface denominados de *canvas* ou *forms*. A Tab. 1.1 ilustra a evolução destas duas ferramentas.

Sistemas Gráficos	Sistemas de Interface
1978-core	1984-Macintosh
1985-GKS	1987-XWindow v.11
1988-PHIGS	1988-Motif
1990-PEX	1990-Windows 3.0
1991-GL	1993-Visual Basic v.3
1993-OpenGL	1995-Delphi
1996-Direct 3D	1996-Java

Tab. 1.1 – Evolução dos Sistemas Gráficos e de Interface com o Usuário

A primeira proposta de Sistema Gráfico, independente de fabricantes, foi o Core da ACM<sup>2</sup> em 1978. Esta proposta visava um sistema gráfico 3D capaz de fazer a síntese de imagens (*rendering*) mostrada na Figura 1.10. O sistema Core possuía um modelo de câmera bem semelhante ao que utilizamos até hoje. A sua principal deficiência foi a falta de detalhes de como a proposta deveria ser implementada. Por exemplo, o nome e os parâmetros das funções da proposta nas diversas linguagens de programação não era especificado. Assim, dadas duas implementações do Sistema core, um programa escrito para uma, geralmente, teria que ser modificado para rodar na outra. A proposta também era bem deficiente no tratamento de imagens. Os monitores gráficos da época ainda eram predominantes vetoriais e o suporte para imagens digitais era praticamente inexistente.

A partir da experiência do core, a DIN<sup>3</sup> alemã propôs o GKS, Graphical Kernel System. O GKS é uma especificação de um Sistema Gráfico que é basicamente um subconjunto bi-dimensional do core acrescido de algumas funções para a síntese de imagens digitais. A grande vantagem desta proposta foi o seu alto grau de detalhamento. Programas escritos para uma implementação do GKS podem rodar em outra sem grandes modificações do código fonte.

O PHIGS (Programmer's Hierarchical Interactive Graphics System), retomou a idéia de cenas 3D do Core com modelos hierárquicos de objetos e transformações. O PHIGS foi

<sup>2</sup> Association for Computing Machinery – associação internacional educacional e científica dedicada ao avanço das artes, ciências e aplicações da tecnologia da informação. Apesar de internacional é uma organização fortemente americana.

<sup>3</sup> Deutsches Institut für Normung – Instituto alemão de normas.

posteriormente estendido para suportar imagens digitais. Este novo sistema passou a ser denominado PHIGS+.

Ao mesmo tempo em que a discussão acadêmica sobre Sistemas Gráficos evoluía os Sistemas de Interface gráficos foram sendo desenvolvidos. A coluna da direita da Tab. 1.1 ilustra as datas destes sistemas. É interessante notarmos que os sistemas gráfico e de interface competem pela gestão da área da tela onde o desenho vai ser exibido. Sistemas como o XWindow, por exemplo, são baseados em arquiteturas cliente-servidor em que o acesso a uma janela é encapsulado pela XLib que é intrinsecamente um sistema gráfico 2D. Implantar um sistema gráfico 3D sob a XLib restringe fortemente a Ou seja, a programação 3D teria que ser feita por *software*. Para contornar este problema o PHIGS evoluiu para o PEX, que foi extensão do PHIGS para o X-Window.

O PEX não vingou como padrão porque na mesma época a Silicon Graphics™ criou o GL (*graphics library*) que foi um sucesso absoluto. Este sistema foi posteriormente aberto e se tornou o OpenGL™ que é atualmente o padrão mais consagrado de Sistemas Gráficos.

Compete com o OpenGL™ o Direct3D que é um sub-sistema do DirectX que a MicroSoft™ criou para dar suporte a programação de jogos. Com o avanço dos computadores e as placas de vídeo programáveis o OpenGL tem ocupado este espaço também.

O foco central deste livro é conceitual para que ele seja útil para programadores de qualquer linguagem de computador utilizando qualquer Sistema Gráfico e de Interface. Os códigos apresentados no texto e no CD em anexo são todos escritos em ANSI C. Os códigos utilizam a Sistema Gráfico OpenGL™ e os Sistemas de Interface com o Usuário GLUT e IUP.

Todos os sistemas utilizados neste livro são livres. O OpenGL™ e a GLUT podem ser obtidos em <http://www.opengl.org> e a IUP em <http://www.tecgraf.puc-rio.br/iup>.

### **Projeto de programação**

Faça o *download* das bibliotecas OpenGL, GLUT e/ou IUP e instale na versão seu compilador C/C++ favorito em sua máquina. Compile, monte e rode o programa glutOpenGL.c fornecido.

### **Exercícios**

- 1) Qual a diferença entre Processamento de Imagens, Visão Computacional e Síntese de Imagens? Procure descrever um programa de cada uma destas áreas que você conheça.
- 2) Porque um programa gráfico interativo utiliza o algoritmo de mapa de profundidade e não o de Rastreamento de Raios?
- 3) O que é uma programa de aplicação?
- 4) Faça um breve histórico dos principais Sistemas Gráficos.