# Introdução à Computação Gráfica

Notas do Curso de Computação Gráfica I,
2000

Paulo Roma Cavalcanti

Universidade Federal do Rio de Janeiro - UFRJ

# Sumário

# Capítulo 1

# Introdução

O objetivo deste texto é servir como base a um curso introdutório de Computação Gráfica. Considerando a Computação Gráfica, em última análise, como uma forma de matemática aplicada, isto significa que irão ser utilizados, na maioria das vezes, modelos matemáticos simplificados, ficando para os cursos mais avançados o estudo dos modelos mais complexos, que procuram retratar a realidade mais fielmente.

## 1.1 Áreas Correlatas

O primeiro ponto a ser discutido é: qual o objetivo da Computação Gráfica? Colocando de maneira extremamente espartana, pode-se afirmar que o objetivo primordial da Computação Gráfica é transformar dados em imagens. Assim, existe o problema da modelagem dos dados (criação, estruturação e análise dos dados) e o problema da visualização destes dados.

O objetivo inverso, ou seja, a recuperação dos dados a partir de uma imagem (análise de imagem), corresponde a área de Visão Computacional, que é muito importante, por exemplo, em robótica. Por fim, ainda existe a necessidade de manipulação de imagens com o objetivo de processar, de alguma forma, uma imagem, para produzir uma nova imagem, a partir de operações de filtragem e de deformação, ou simplesmente com o objetivo de compactação. Estes problemas são tratados na área de Processamento de Imagens. Tanto a área de Visão Computacional como a área de Processamento de Imagens não serão abordadas neste texto. A figura 1.1 busca esquematizar os relacionamentos entre as áreas citadas.

## 1.2 Paradigma dos Universos

Um paradigma de abstração útil consiste em estabelecer quatro universos (conjuntos) distintos:

- o universo físico $F$, que contém os objetos do mundo real que pretende-se estudar;
- o universo matemático $M$, que contém uma descrição matemática abstrata dos objetos do universo físico, em geral, idealizados (simplificados), para permitir a sua descrição através de um modelo matemático simples;

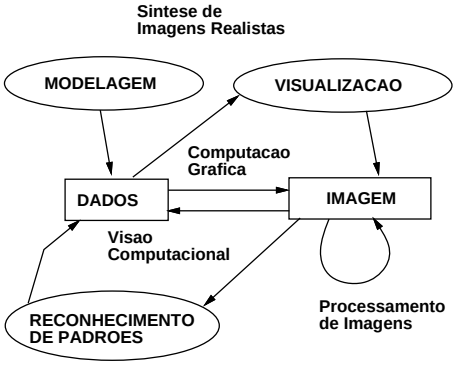



Figura 1.1: Relacionamentos entre as Áreas de CG, VC e PI.

- o universo de representação $R$, que é constituído por descrições simbólicas e finitas associadas aos objetos do universo matemático;
- e o universo $C$ de codificação, que é constituído pelas estruturas de dados utilizadas na codificação da representação em uma dada linguagem de programação.

O paradigma dos quatro universos parte do suposto de que para estudar, com o auxílio do computador, um determinado fenômeno ou objeto do mundo real, associa-se ao mesmo um modelo matemático para, em seguida, procurar-se uma representação finita do modelo associado. O espaço dos modelos consiste, em geral, em uma quantidade infinita de pontos e os esquemas de representação buscam uma descrição finita destes pontos que seja manipulável (passível de implementação) em um computador digital. Podem-se traçar, então, em linhas gerais, os diversos problemas a serem estudados:

- definição dos elementos de $M$;
- estabelecimento de relações entre os universos;
- definição das relações de representação de $M$ em $R$ (esquemas de representação);
- estudo das propriedades das diversas representações de $M$ em $R$;
- conversão entre diferentes representações.

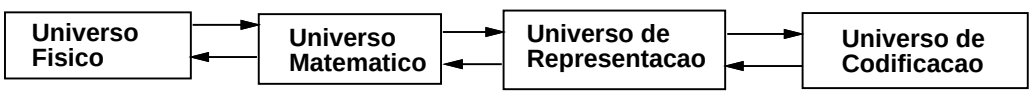



Figura 1.2: Paradigma dos Universos.

Este modelo conceitual vai ser utilizado em quase todos os capítulos, de forma a que o leitor tenha sempre em mente qual o nível de abstração correto a cada momento. Para isto, no início de cada capítulo, existe uma ilustração com o modelo conceitual apropriado e um texto explicativo, que vai se tornando mais resumido à medida que os capítulos vão avançando, pois espera-se que o leitor já esteja devidamente familiarizado com ele.

## 1.3 Escopo

Este texto se atém à correta conceitualização do problema de *Síntese de Imagens de Cenas Tridimensionais* e ao estudo dos pricincipais métodos e algoritmos envolvidos. As áreas de aplicação são diversas, podendo ser citadas as seguintes:

- Entretenimento
  - Televisão
  - Filmes
  - Jogos
- CAD / CAM
  - Engenharia
  - Arquitetura
  - Design
- Visualização Científica
  - Medicina
  - Biologia
  - Matemática

Não é o objetivo deste texto tratar os seguintes assuntos: Gráficos bidimensionais, Processamento de imagens e Interação Homem-Máquina. Como áreas de aplicação citam-se:

- Editoração Eletrônica
  - Illustração
  - Sistemas de Pintura
  - Layout de Página
- Processamento de Imagens
  - Visão Computacional
  - Efeitos para Vídeo
  - Reconhecimento de Padrões
- Projeto de Interfaces
  - Sistemas de Gerenciamento de Interface com o Usuário
  - Sistemas de Janela
  - Toolkits

## 1.4 Estrutura do Livro

Como já foi dito, o problema básico da Computação Gráfica consiste em transformar dados em imagens (gerar uma fotografia virtual). Desse modo, de um ponto de vista didático, esse processo de transformação pode ser dividido em duas etapas: Modelagem e Visualização. A Modelagem se ocupa da criação e representação dos objetos, chamados modelos, no computador. Na Modelagem, tenta-se representar no computador o mundo físico real. Já a visualização estuda os métodos e técnicas utilizados para obter-se, a partir do modelo, uma imagem, que é o produto final da Computação Gráfica.

Uma proposta deste texto é que as noções básicas sobre cor e imagem sejam apresentadas antes que se discutam a visualização e iluminação, e não, como faz a maioria dos livros de Computação Gráfica, após. Isto evita que o aluno fique com uma visão distorcida ou simplificada do problema e que possa entender (ou implementar) algoritmos capazes de lidar adequadamente com cores (algoritmos policromáticos). Desta forma, este texto se divide do seguinte modo:

- Noções Preliminares
  - Fundamentos de Cor
  - Dispositivos Gráficos
  - Imagem Digital
- Modelagem Geométrica
  - Geometria e Transformações
  - Esquemas de Representação
  - Sistemas de Modelagem
- Visualização
  - Câmera e Recorte
  - Rasterização e Amostragem
  - Visibilidade
- Iluminação
  - Propriedades dos Materiais e Modelos de Iluminação
  - Colorização e Mapeamentos
- Decisões de Sistema

## 1.5 Objetivos e Pré-Requisitos

Os objetivos que pretendem-se atingir são:

- Oferecer uma visão global da área de Computação Gráfica
- Discutir os conceitos pré-estabelecidos

- Apresentar os principais problemas da área
- Fornecer uma base sólida para um desenvolvimento futuro

O público alvo deste texto é composto por:

- Futuros Pesquisadores
- Projetistas de Sistemas Gráficos
- Programadores de Computação Gráfica

e os pré-requisitos necessários são:

- Programação de Computadores
- Álgebra Linear
- Cálculo

## 1.6 Referências Bibliográficas

Existem vários livros de Computação Gráfica, geralmente em língua inglesa, no mercado. Eles podem ser consultados dependendo da disponibilidade e interesse de cada leitor. Indicam-se aqui os mais adequados, na opinião dos autores:

- De caráter Geral

  Foley [1]

  Newman [2]

  Watt [3]

  Slater

- Modelagem

  Farin [4]

  Mäntylä [5]

  Hoffman [6]

  K B's [7]

- Visualização

  Glassner [8]

  Hall [9]

- Processamento de Imagem

  Gonzalez [10]

  Wolberg [11]

- Implementação

  Harrington [12]

  Gems [13] [14] [15] [16] [17]

- Outros

  Samet [18]

  Pavlidis [19]

  Rogers [20]

## 1.7 Exercícios

**1.1** *Faça uma comparação entre as áreas de Computação Gráfica, Processamento de Imagens e Visão Computacional. Dê pelo menos dois exemplos de aplicação em cada uma dessas áreas.*

**1.2** *Explique o Paradigma dos Universos. Discuta a natureza de cada nível de abstração.*

**1.3** *Considere o seguinte problema: Implementar no computador um sistema para desenhos de discos metálicos. Discuta detalhadamente esse problema do ponto de vista do paradigma dos quatro universos (incluindo implementação).*

# Capítulo 2

# Cor

Uma imagem é definida, em última análise, pela cor dos seus pontos. Surge aí o primeiro problema: o que é cor? Embora seja um termo utilizado amplamente no cotidiano, é de difícil formalização. O paradigma dos quatro universos fornece a conceitualização necessária. No universo físico o fenômeno a ser estudado é a luz que corresponde, no universo matemático, a um modelo que descreve o fenômeno luz adequadamente. Já no universo de representação existem esquemas que fornecem uma representação finita para o modelo adotado, conforme pode ser visto na figura 2.1.


Figura 2.1: Modelo Conceitual.

## 2.1 Modelo Espectral de Cor

Voltando ao paradigma dos quatro universos, do ponto de vista físico, a sensação de cor é produzida por uma radiação eletro-magnética que chega até aos nossos olhos. Esta radiação pode ser modelada matematicamente por uma função, chamada de função de distribuição espectral, que associa a cada comprimento de onda presente na radiação[1], o valor de alguma grandeza radiométrica (em geral energia radiante). Assim, do ponto de vista matemático, cor é uma função. Neste caso, o universo matemático é constituído pelo conjunto $D$ de todas as funções de distribuição espectral:

$$D = \{f : U \subset \Re^+ \rightarrow \Re^+\}.$$

[1] Newton foi o primeiro pesquisador a mostrar qua a luz branca possui energia em todos os comprimentos de onda do espectro visível.

## 2.2 Sistemas Físicos de Cor

O olho humano é um sistema físico de processamento de cor. Em particular, trata-se de um sistema refletivo, pois faz uma amostragem em três faixas do espectro. Existem, em contrapartida, sistemas emissivos que fazem o inverso, ou seja, reconstroem a cor. Generalizando um pouco mais, em geral, um sistema refletivo possui um número finito de sensores, $s_1, s_2, ..., s_n$, que fazem a amostragem em $n$ faixas do espectro. Matematicamente, este fato é expresso da seguinte forma:

$$C(\lambda) \rightarrow (c_1, c_2, ..., c_n), \;\; c_i = \int_0^{+\infty} C(\lambda) s_i(\lambda) d\lambda,$$

onde $s_i(\lambda)$ é a função de resposta espectral do i-ésimo sensor (fig. 2.2).

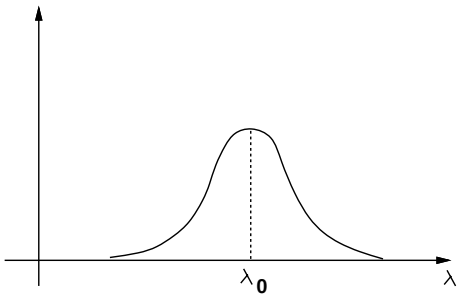



Figura 2.2: Função de Resposta Espectral Típica de um Sensor.

A amostragem ideal é aquela na qual a resposta dos sensores é um impulso:

$$s_i(\lambda) = \infty \;\; \text{se} \;\; \lambda = \lambda_i \;\; \text{ou} \;\; 0 \;\; \text{se} \;\; \lambda \neq \lambda_i.$$

Neste caso, $c_i = \int_{-\infty}^{\infty} C(\lambda)\ \delta(\lambda - \lambda_i)\ d\lambda = C(\lambda_i)$.

Um sistema emissivo é responsável pela reconstrução da cor, a partir de emissores que emitem luz com uma certa distribuição espectral. Assim, se os emissores são caracterizados por $n$ funções de distribuição espectral $P_1(\lambda), P_2(\lambda), ..., P_n(\lambda)$, chamadas de base de primárias, tem-se que a cor reconstruída é dada por:

$$C_r(\lambda) = \sum_{k=1}^{n} c_k P_k(\lambda).$$

Para compreender melhor o processo de amostragem e reconstrução de cor, considere-se o que ocorre quando se utiliza uma câmara de televisão para captar uma imagem da natureza, que é então transmitida por tele-difusão. A luz que atinge a lente é focada, gerando uma imagem no tubo da câmara, e os circuitos eletrônicos da câmara fazem uma varredura da imagem, gerando um determinado número de linhas (que depende do padrão de vídeo adotado). A imagem é transmitida desta forma. As ondas são então captadas por

uma antena e o sinal original é reconstruído pelo monitor da televisão. A cor reconstruída é idêntica (perceptualmente) à cor original devido ao fenômeno do *metamerismo*, definido na próxima seção. A figura 2.3 ilustra o processo.


Figura 2.3: Amostragem e Reconstrução.

## 2.3 Representação Discreta de Cor

O espaço de todas as distribuições espectrais possui dimensão infinita. Para obter-se uma representação finita, ou seja, aproximar este espaço de dimensão infinita por um espaço de dimensão finita, é necessário fazer algum tipo de amostragem. Isto faz com que se utilize um vetor de dimensão finita na representação de uma função:

$$R : f \in D \rightarrow (f(x_1), f(x_2), ..., f(x_n)) \in \Re^n.$$

Esta representação define uma transformação linear, pois: $R(af_1 + bf_2) = aR(f_1) + bR(f_2)$. É claro que este processo de amostragem acarreta em perda de informação (fig. 2.4) (a representação é ambígua, ou seja, um mesmo vetor pode representar mais de uma função).

Este tipo de representação é válida, no entanto, porque o problema de cor deve ser abordado do ponto de vista perceptual e não do ponto de vista físico (na realidade é um problema psico-físico). Em 1807, Young concluiu, a partir de experimentos, que o olho humano possui três tipos de receptores luminosos (células) que são mais sensíveis ao intervalo (da radiação) que corresponde aos comprimentos de onda na faixa do vermelho, verde e azul, respectivamente. Estes três tipos de receptores fazem uma amostragem da radiação nestes três comprimentos de onda. Desta forma, o espaço perceptual de cor é um espaço de dimensão finita (dimensão três). Isto significa que uma mesma sensação de cor pode ser obtida a partir de distribuições espectrais distintas, um fenômeno conhecido por metamerismo. Graças a isto a televisão produz imagens aceitáveis ao ser humano, uma vez que o conjunto de distribuições espectrais existentes na natureza é muito mais rico do que aquele que pode ser produzido artificialmente no monitor de uma televisão.

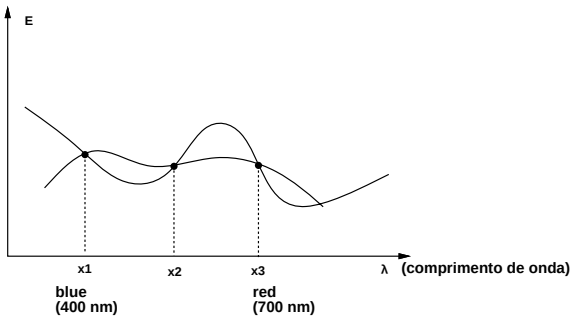



Figura 2.4: Representação Finita de duas Funções.

## 2.4 Espaços de Cor

Dada uma função de distribuição espectral $C(\lambda)$ (que corresponde a uma sensação de cor), um sistema emissivo com uma certa base de primárias $P_k(\lambda)$ e um sistema refletivo, como calcular as componentes na base de primárias de forma a que a cor reconstruída $C_r(\lambda)$ seja perceptualmente equivalente à cor original em relação ao sistema refletivo?

É possível mostrar que conhecendo-se a resposta espectral do sistema em cada comprimento de onda, obtêm-se as funções de reconstrução de cor $r_k(\lambda)$ (fig 2.5), que indicam as proporções nas quais as cores primárias devem ser combinadas para igualar a cor desejada.

Por definição, a resposta espectral do sistema é dada para um certo comprimento de onda pelas componentes, na base de primárias, da distribuição espectral conhecida como cor espectral (fig. 2.6), que é diferente de zero apenas neste comprimento de onda. Desta forma, tem-se que

$$C_r(\lambda) = \sum_{k=1}^{n} c_k P_k(\lambda), \qquad c_k = \int_0^{+\infty} C(\lambda) r_k(\lambda) d\lambda.$$

A resposta espectral do sistema pode ser obtida experimentalmente. Para isto usam-se quatro emissores de luz. Os três primeiros correspondem às cores primárias e podem ter as intensidades controladas. O quarto emite a luz monocromática que se deseja igualar. Direcionando os três fachos de luz provenientes dos três emissores primários para um único ponto, e ajustando as suas intensidades até que, para um observador padrão, a cor resultante seja idêntica à cor do quarto emissor, obtêm-se as componentes da luz monocromática de teste em relação à base de primárias (fig. 2.7). Algumas cores não são igualadas, a menos que se adicione uma primária junto com a cor de teste. Matematicamente, isto corresponde a uma intensidade negativa.

Neste ponto o leitor já deve ter percebido que o objetivo é escrever uma dada cor como

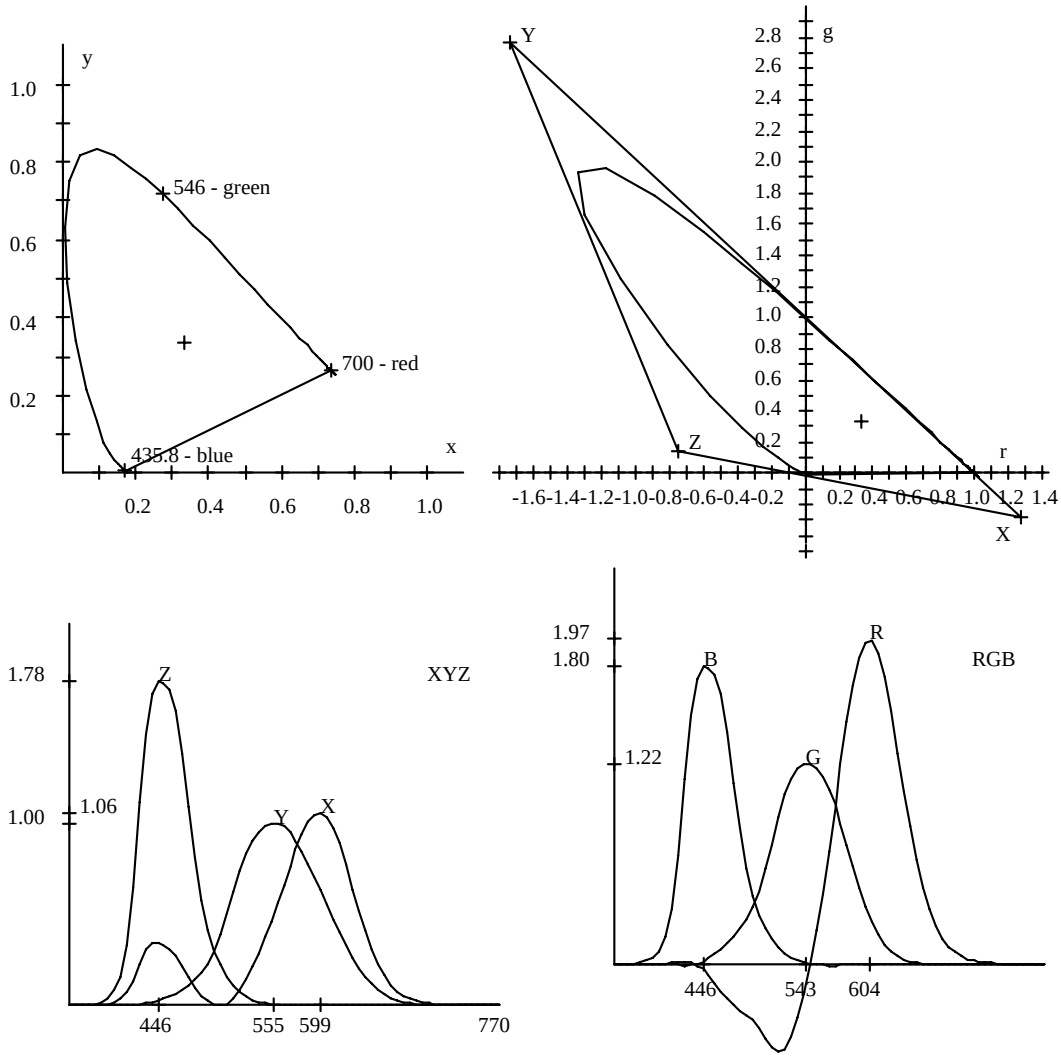



Figura 2.5: Funções de Reconstrução de Cor.

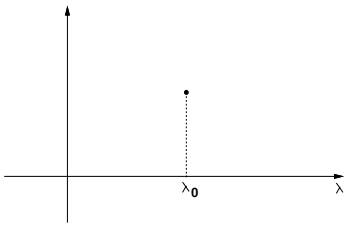


Figura 2.6: Cor Espectral.

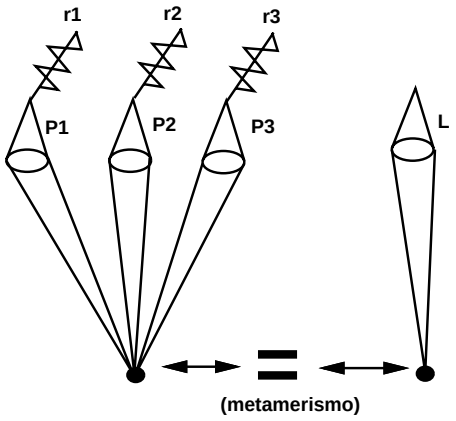



Figura 2.7: Obtenção Experimental das Funções de Reconstrução de Cor.

combinação linear das cores primárias[2]. Voltando ao universo matemático, é fácil concluir que a multiplicação de uma função de distribuição espectral por um escalar positivo não altera a sensação de cor, mas apenas o que se costuma chamar de intensidade. Diz-se que a informação de croma foi preservada. Uma combinação convexa de duas distribuições espectrais ainda é uma distribuição espectral. Como a cada distribuicão espectral corresponde um único ponto no espaço de cor, e a aplicação de representação (amostragem) nada mais é do que uma transformação linear, pode-se concluir que o espaço de cor é o espaço formado pelas retas que passam pela origem. Conclui-se então que o conjunto de todas as cores visíveis, chamado de sólido de cor, é um cone convexo.

- $C(\lambda)$ é cor visível $\Rightarrow t\ C(\lambda)$ é cor visível $\Rightarrow R(t\ C(\lambda)) = t\ R(C(\lambda))$.
- $C_1(\lambda)$ e $C_2(\lambda)$ são cores visíveis $\Rightarrow (1-t)\ C_1(\lambda) + t\ C_2(\lambda),\ t \in [0,1]$, é cor visível $\Rightarrow R((1-t)\ C_1(\lambda) + t\ C_2(\lambda)) = (1-t)\ R(C_1(\lambda)) + t\ R(C_2(\lambda))$.

Um sistema físico não suporta intensidades arbitrariamente altas, pois os seus sensores são destruídos. Na prática, o que se faz é trabalhar com uma representação normalizada de cor. Escolhe-se uma cor de referência, chamada de branco padrão $(b_r, b_g, b_b)$, e atribuem-se a ela as componentes $(1,1,1)$. Qualquer outra cor $(c_r, c_g, c_b)$ têm as suas componentes normalizadas:

$$\left(\frac{c_r}{b_r},\ \frac{c_g}{b_g},\ \frac{c_b}{b_b}\right).$$

## 2.5 Diagrama de Cromaticidade

Uma forma de considerar apenas um representante em cada reta de cor é utilizar a projeção radial de um determinado ponto da reta sobre um plano. Maxwell utilizou o plano

$$x + y + z = 1$$

[2]Se as cores primárias pertencerem ao espectro visível, haverá, eventualmente, componentes negativas.

que, por este motivo, é chamado plano de Maxwell. Sabe-se que:

- As cores visíveis definem um sólido convexo de cor (um cone).
- A interseção do sólido de cor com o plano de Maxwell gera uma curva convexa (diagrama de cromaticidade).
- As cores espectrais correspondem a pontos na fronteira do diagrama de cromaticidade (por que?).

A projeção de uma cor sobre o plano de Maxwell produz as coordenadas de cromaticidade da cor (fig 2.8). O seu cálculo é imediato. Sejam $(c_r, c_g, c_b)$ as coordenadas de uma cor $c$ no espaço de cor. A reta que passa pela origem e por $c$ é dada pelos pontos da forma:

$$\{p;\ p = tc,\ t \in \Re\}.$$

A projeção $c' = (c'_r, c'_g, c'_b)$ da cor $c$ sobre o plano de Maxwell impõe que $c'_r + c'_g + c'_b = 1$. Em particular, tem-se que $t(c_r, c_g, c_b) = (c'_r, c'_g, c'_b)$ para algum $t$. Logo,

$$t(c_r + c_g + c_b) = c'_r + c'_g + c'_b = 1,\ \text{ou}\ t = \frac{1}{(c_r + c_g + c_b)}.$$

Assim,

$$c'_i = \frac{c_i}{(c_r + c_g + c_b)}.$$

O diagrama de cromaticidade é obtido, então, projetando-se ortogonalmente, sobre o plano $rg$, a interseção do sólido de cor com o plano de Maxwell. O diagrama de cromaticidade é útil por permitir uma visualização padrão de um corte do sólido de cor. Baseado neste diagrama, pode-se definir um comprimento de onda dominante para qualquer cor $c$, traçando-se uma semi-reta com origem no ponto acromático $(1/3, 1/3, 1/3)$ e contendo $c$. A interseção da semi-reta com o bordo do diagrama de cromaticidade fornece, diretamente, o comprimento de onda dominante de $c$. O comprimento de onda dominante pode ser interpretado como correspondendo a uma cor pura (espectral) que se combinada com branco nas proporções apropriadas produz a cor $c$.

O conceito de cor complementar de uma cor $c$ também pode ser definido de forma análoga. É uma cor que se misturada com a cor $c$ nas proporções apropriadas produz branco ($\alpha c_1 + \beta c_2 =$ cor acromática(branco)). O ponto acromático deve estar contido no segmento de reta que une $c$ à sua cor complementar (por que?).

### 2.5.1 Decomposição Luminância-Crominância

As coordenadas de cromaticidade captam a noção de cor propriamente dita. Juntamente com a informação de intensidade ou luminância, determinam unicamente qualquer cor. A luminância é, por definição, um funcional linear: $L : \zeta = \Re^3 \to \Re$. Dada uma cor $c = (c_r, c_g, c_b)$, existem constantes $l_r, l_g, l_b$ tais que $L(c) = l_r c_r + l_g c_g + l_b c_b$. Da linearidade vem que:

$$L(c + c') = L(c) + L(c') \text{ e } L(tc) = tL(c).$$

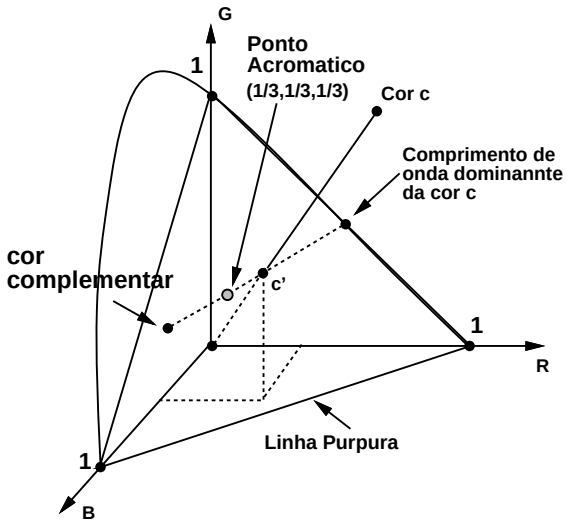



Figura 2.8: Diagrama de Cromaticidade.

Todo vetor de cor $c$ pode ser escrito, de modo único, como soma direta de um vetor pertencente a $ker(L) = \{c \in \Re^3 : L(c) = 0\}$ (núcleo de $L$) e de um vetor pertencente a um subespaço complementar de $ker(L)$ — na forma $c = ker(L) \oplus \ell,\ c = c_c + c_l$.

Sabe-se da álgebra linear que a dimensão do núcleo mais a dimensão da imagem de uma transformação linear $L : \Re^n \to \Re^m$ é igual a dimensão do domínio da transformação:

$$dim(Ker(L)) + dim(Im(L)) = n.$$

No caso presente, isto significa que a dimensão do núcleo do funcional de luminância é 2. Se duas cores têm a mesma luminância, $L(c_1) = L(c_2)$, conclui-se que $c_1$ e $c_2$ estão em um hiperplano afim $c_v = c_0 + v$, paralelo ao núcleo do operador de luminância (fig. 2.9):

$$L(c_1 - c_2) = 0 \Rightarrow c_1 - c_2 \in ker(L).$$

Cada hiperplano afim paralelo ao núcleo do operador de luminância é chamado de um hiperplano de crominância (luminância constante).

A decomposição em crominância-luminância é de extrema importância na definição de diversos sistemas de coordenadas no espaço de cor.

Para finalizar, define-se um sólido de cor como sendo o conjunto de cores realizáveis em um espaço de cor e um mapa de cor como sendo uma curva no sólido de cor

$$\varphi : I \subset \Re \to S \subset \Re^3.$$

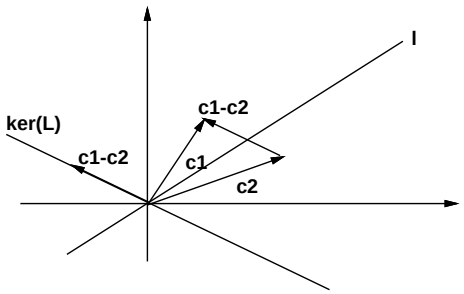



Figura 2.9: Decomposição Crominância-Luminância.

## 2.6 Luminância

Após a discusão sobre decomposição crominância-luminância, o leitor deve estar se perguntando o que é, afinal de contas, a luminância. Para compreender este conceito, suponha-se uma luz monocromática com potência constante de 1 watt. Será que a resposta do olho a este estímulo é linear, ou seja, perceptualmente, se variarmos o comprimento de onda, será que um observador padrão concluirá que as luzes tem brilho constante? A resposta é não. A resposta é máxima para comprimento de onda igual a 555 $nm$ (faixa do verde). A figura 2.10 mostra a sensibilidade relativa do olho em função do comprimento de onda.

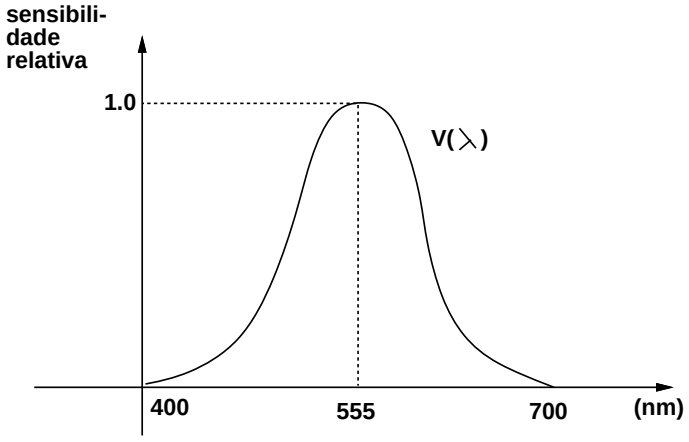



Figura 2.10: Curva de Sensibilidade Relativa.

Convencionou-se que uma luz monocromática com comprimento de onda igual a 555 $nm$ e com 1 watt de potência produz 680 $lumens$. A constante $K(\lambda) = 680\ V(\lambda)\ lm/w$ permite converter de watts para lumens. Note-se então que a luminância[3] é uma grandeza

[3] Luminância = $lumens/(m^2 Sr)$.

colorimétrica que corresponde aos termos perceptuais de brilho (para emissores) ou luminosidade (para refletores).

Se a luz não for monocromática, mas sim caracterizada por uma distribuição espectral $C(\lambda)$, tem-se que:

$$L(C(\lambda)) = K(\lambda) \int_0^\infty C(\lambda)\ V(\lambda)\ d\lambda.$$

Isto pode ser expresso em função da representação de $C$ no sistema CIE[4] -RGB por

$$L(C(\lambda)) =< L, c >=< (0.177, 0.812, 0.0106), (c_r, c_g, c_b) > .$$

## 2.7 Padrão CIE-XYZ

Para evitarem-se coordenadas de cromaticidade negativas (por que isto é importante?), foi criado um outro padrão de cor chamado de CIE-XYZ. As primárias deste sistema não estão contidas no sólido de cor justamente para que qualquer cor possa ser expressa, apenas com coordenadas positivas, como combinação linear das cores primárias. A conversão do sistema CIE-RGB para o CIE-XYZ é uma mera mudança de sistema de coordenadas.

Como foi criado o sistema CIE-XYZ? Definiu-se que duas cores primárias têm luminância zero (pertencem ao núcleo do operador de luminância). Traça-se então uma reta coincidente com o segmento (quase) retilíneo do diagrama de cromaticidade. A interseção desta reta com a reta de luminância zero (que passa pela origem e é perpendicular ao vetor $(0.176, 0.81)$) define a primária $X$. As duas outras primárias ficam definidas pelo traçado de uma segunda reta que é tangente ao diagrama de cromaticidade pela esquerda e que minimiza a área do triângulo formado pela reta de luminância zero, a reta anterior e esta reta. A primária $Z$ está na reta de luminância zero e a $Y$ no terceiro vértice do triângulo (fig. 2.5).

| rgb | x | y | z | xyz | r | g | b |
|---|---|---|---|---|---|---|---|
| r | 1.2750 | -1.7395 | -0.7431 | x | 0.73467 | 0.27376 | 0.16658 |
| g | -0.2779 | 2.7675 | 0.1409 | y | 0.26533 | 0.71741 | 0.00886 |
| b | 0.0029 | -0.0280 | 1.6022 | z | 0.00000 | 0.00883 | 0.82456 |

Tabela 2.1: Bases do CIE (vetores coluna).

Aqui talvez valha a pena reavivar a memória daqueles que não estão muito familiarizados com álgebra linear. Considere-se o seguinte problema: obter as coordenadas do objeto da figura 2.11 no sistema de coordenadas b).

### 2.7.1 Mudança de Sistema de Coordenadas

Uma mudança de sistema de coordenadas corresponde a uma transformação linear. Desta forma, basta que seja determinado como a transformação age nos vetores de uma base para que se tenha a transformação correspondente. Assim se $v = \sum a_i e_i,\ a_i \in \Re,\ \mathrm{v}, e_i \in \Re^n$ ($\{e_i\}$

[4]Comission Internationale de Eclairage.

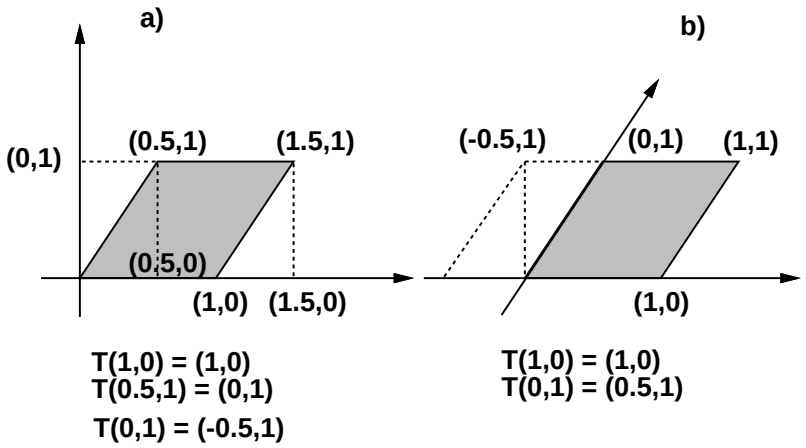



Figura 2.11: Mudança de Sistema de Coordenadas.

formam uma base), tem-se que $T(v) = T(\sum a_i e_i) = \sum a_i T(e_i)$. Este é o motivo pelo qual uma transformação linear pode ser representada por uma matriz (a matriz onde os vetores coluna representam os vetores da base já transformados).

No exemplo proposto $T(1,0) = (1,0)$ e $T(0.5,1) = (0,1)$, o que, na forma matricial, produz o seguinte sistema de equações:

$$\begin{pmatrix} T(e_1) & T(e_2) \\ a & b \\ c & d \end{pmatrix} \begin{pmatrix} a_1 & a_2 \\ 1 & 0.5 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} T(v_1) & T(v_2) \\ 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Resolvendo o sistema acima, obtêm-se: $a = 1,\ 0.5a + b = 0,\ c = 0,\ 0.5c + d = 1$, ou $T = \begin{pmatrix} 1 & -0.5 \\ 0 & 1 \end{pmatrix}$. A transformação $T$ leva do sistema de coordenadas a) para o sistema de coordenadas b), sendo denotada por $T_{ab}$.

A transformação $T_{ba} = {T_{ab}}^{-1}$ pode ser obtida pelo mesmo procedimento ou pela inversão da matriz correspondendo a $T_{ab}$. Deve-se notar que fixada uma base de vetores existe um sistema de coordenadas associado. A representação gráfica do sistema de coordenadas é feita, normalmente, utilizando-se um par de eixos perpendiculares, e não como foi apresentado no sistema b) da figura 2.11. Se os vetores representam alguma grandeza física, no caso presente cor, o sistema de coordenadas especifica em que proporções as grandezas correspondendo aos vetores da base devem ser combinadas para gerar qualquer outra grandeza representada por um elemento do espaço vetorial.

### 2.7.2 Sistema $xyY$

O diagrama de cromaticidade retira a luminância. Logo sensações de cores relacionadas com luminância não aparecem (por exemplo, marrom = vermelho-alaranjado com luminância muito baixa). Assim, ve-se que o diagrama de cromaticidade não é uma palete de cores.

Há uma infinidade de planos no espaço $XYZ$ que se projetam sobre o diagrama, perdendo a luminância nesse processo. Projetando uma cor no plano de Maxwell tem-se:

$$x = \frac{X}{X+Y+Z}, \quad y = \frac{Y}{X+Y+Z}, \quad z = \frac{Z}{X+Y+Z},$$

$$X = \frac{x}{y}Y, \quad y = Y, \quad Z = \frac{z}{y}Y \Rightarrow (X,Y,Z) = Y\left(\frac{x}{y}, 1, \frac{1-x-y}{y}\right).$$

As coordenadas $xyY$ permitem que se faça uso do diagrama de cromaticidades na especificação de cores.

### 2.7.3 Mudança do Sistema $RGB \leftrightarrow XYZ$

A mudança do sistema RGB para o sistema XYZ não pode ser feita diretamente a partir da tabela 2.1 porque as coordenadas lá presentes são coordenadas de cromaticidade. Então procura-se uma matriz que represente a transformação apropriada:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} x_r C_r & x_g C_g & x_b C_b \\ y_r C_r & y_g C_g & y_b C_b \\ (1-x_r-y_r)C_r & (1-x_g-y_g)C_g & (1-x_b-y_b)C_b \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} = M \begin{pmatrix} C_r R \\ C_g G \\ C_b B \end{pmatrix},$$

onde $C_r = X_r + Y_r + Z_r$; $\quad C_g = X_g + Y_g + Z_g$; $\quad C_b = X_b + Y_b + Z_b$.

Na realidade só dispoem-se das coordenadas $xyz$ e precisa-se determinar o valor dos três escalares $C_r, C_g, C_b$, que escalam apropriadamente os vetores da base. É necessário então que se conheça o valor das coordenadas tricromáticas de um ponto. Normalmente, tem-se as coordenadas tricromáticas $(X_w, Y_w, Z_w)$ do branco padrão de referência: $(R_w, G_w, B_w) = (1,1,1)$. Assim:

$$\begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} = M \begin{pmatrix} C_r \\ C_g \\ C_b \end{pmatrix} \Rightarrow \begin{pmatrix} C_r \\ C_g \\ C_b \end{pmatrix} = M^{-1} \begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix}.$$

Fazendo-se estas contas utilizando os dados da tabela 2.1 e considerando $(X_w, Y_w, Z_w) = (R_w, G_w, B_w) = (1,1,1)$, obtem-se:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.489989 & 0.310008 & 0.200003 \\ 0.176962 & 0.812400 & 0.010638 \\ 0.000000 & 0.009999 & 0.990001 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix},$$

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 2.364666 & -0.896583 & -0.468083 \\ -0.515155 & 1.426409 & 0.088746 \\ 0.005203 & -0.014407 & 1.009204 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}.$$

Note-se que o resultado é o esperado, por exemplo:

$$r_y = \frac{-0.896583}{-0.896583 + 1.426409 - 0.014407} = -1.7395226.$$

Este mesmo processo pode ser usado para fazer a mudança da base mRGB de um dispositivo para a base XYZ e vice-versa.

## 2.8 Sistemas de Cor

Um sistema de cor é o espaço de cor mais um sistema de coordenadas nele definido. Existem vários sistemas de cor. Uma classificação possível é a seguinte:

- Sistemas padrão;
- Sistemas dos dispositivos;
- Sistemas computacionais;
- Sistemas de interface.

### 2.8.1 Sistemas Padrão

Os sistemas padrão são aqueles homologados por alguma instituição normativa. Pode-se citar o sistema de cor CIE-RGB, criado em 1931, que fixa uma base de primárias composta pelas cores monocromáticas de comprimento de onda 700 $m\mu$(Red); 546 $m\mu$(Green) e 435.8 $m\mu$(Blue); o sistema CIE-CMY, que utiliza como primárias as cores complementares ciano (azul-piscina), magenta (violeta) e amarelo para simular um sistema subtrativo de cor (fig. 2.12); e o sistema CIE-XYZ, cuja base de primárias, chamadas X, Y e Z, está fora do espectro visível.

A principal finalidade de um sistema padrão é que, por ser independente de qualquer dispositivo físico, possibilita a mudança de coordenadas entre sistemas distintos e fornece uma maneira de armazenamento de imagens que independe de um dispositivo particular.

### 2.8.2 Sistemas dos Dispositivos

Os sistemas dos dispositivos são definidos pelas bases de primárias dos dispositivos. O gamute de um dispositivo (o conjunto de cores realizáveis pelo dispositivo) é um triângulo contido no diagrama de cromaticidade (por que?). Os sistemas dos dispositivos têm uma importância intrínseca porque, em última análise, são com eles que as imagens são reconstruídas. O espaço de cor de um dispositivo é um subconjunto do sólido de cor que, em geral, tem a forma de um paralelepípedo cujas faces são paralelogramos. Quando se efetua a mudança de coordenadas tem-se então um cubo (fig. 2.12).

### 2.8.3 Sistemas Computacionais

Os sistemas computacionais são sistemas utilizados tanto para síntese de imagens como no processamento de imagens. Podem ser, por exemplo, sistemas padrão, ou sistemas com alguma característica própria, tal como utilizar uma base com mais de três primárias.

### 2.8.4 Sistemas de Interface

Sistemas de cor baseados em espaços vetoriais são práticos do ponto de vista computacional, mas são muito ruins para serem usados na interface com o usuário. Os sistemas de interface objetivam oferecer uma interface adequada a especificação de cores por um usuário comum,

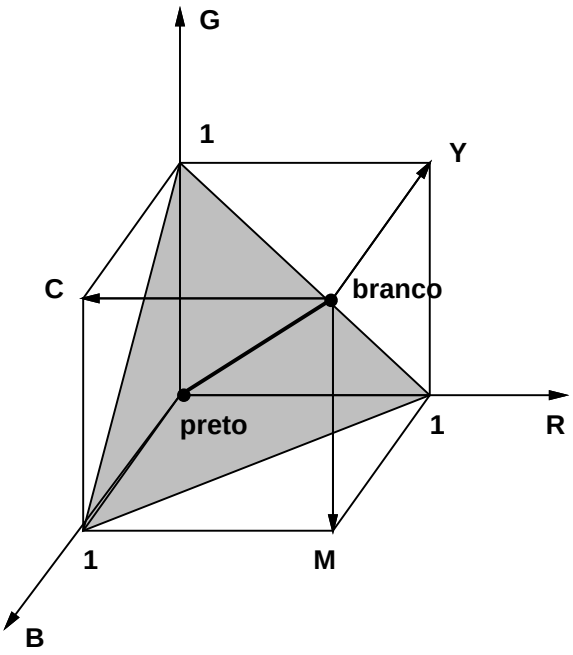



Figura 2.12: Espaço de Cor de um Dispositivo (cubo RGB).

sem qualquer conhecimento específico a respeito de colorimetria. Alguns sistemas permitem escolher uma cor a partir de três parâmetros: tonalidade, saturação e luminância.

- tonalidade ou matiz (comprimento de onda dominante).
- saturação (pureza da cor).
- brilho ou valor (luminância).

A tonalidade varia quando se caminha angularmente no diagrama de cromaticidade e a saturação quando se caminha radialmente. A saturação tem a ver com a pureza da cor (o quanto ela contém de branco). As cores espectrais são as cores consideradas puras. Consulte-se a figura 2.13.

Existem dois métodos básicos para se criar um sistema de interface de cor:

- por coordenadas: utilizam-se coordenadas (ex, HSV e HSL).
- por amostras: discretiza-se o sólido de cor criando-se um atlas de cor (ex, Pantone e Munsell).

O sistema HSV (*Hue, Saturation, Value*), criado por Alvy Ray Smith, projeta o cubo RGB sobre o plano perpendicular a linha de luminância (a diagonal do cubo) $x+y+z=3$, conforme pode ser visto na figura 2.14.

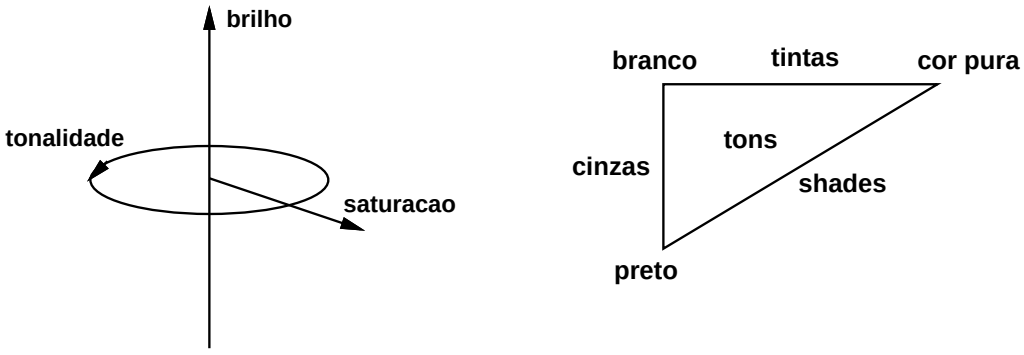



Figura 2.13: Paradigmas de Cor.

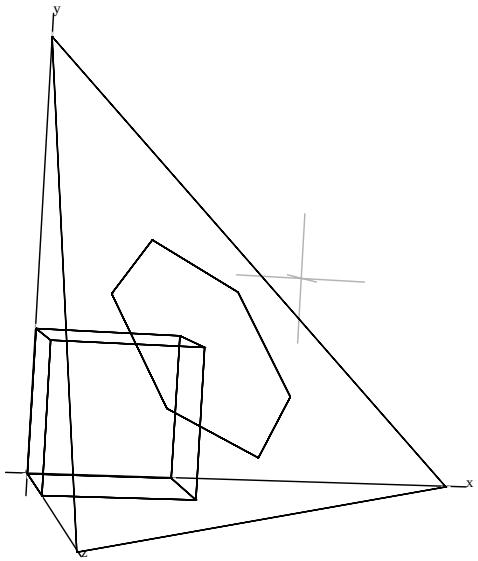



Figura 2.14: Projeção do Cubo RGB no plano x+y+z=3.

Este sistema de coordenadas não se baseia numa base de um espaço vetorial. Logo, a conversão do sistema HSV para o sistema RGB não é dada por uma transformação projetiva e conseqüentemente não pode ser representada por uma matriz. A coordenada valor de uma cor $c = (c_r, c_g, c_b)$ é definida como $max(c_r, c_g, c_b)$ (fig. 2.15).

O sistema HSL (*Hue, Lightness, Saturation*) foi criado pela Tektronix e é muito parecido com o sistema HSV. Neste sistema o brilho é definido por $1/2(max(c_r, c_g, c_b) - min(c_r, c_g, c_b))$ (preto tem brilho 0 e branco brilho 1). O modelo geométrico do sistema HSL é um hexacone duplo.

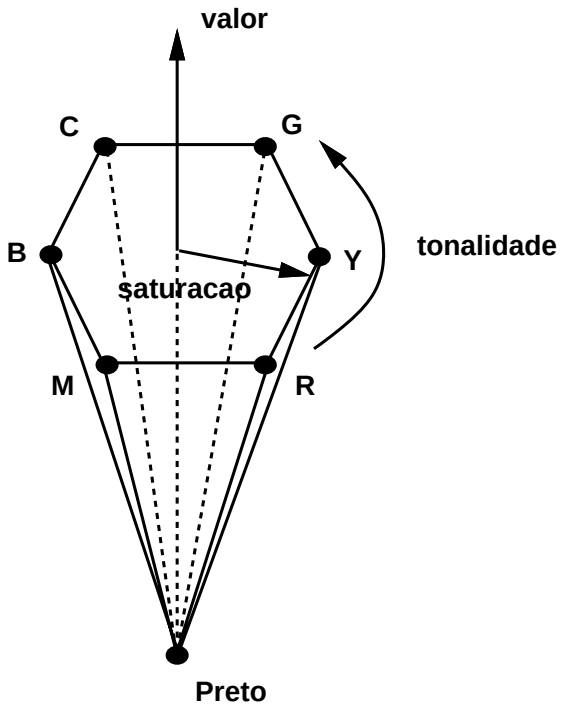



Figura 2.15: Sistema HSV.

Um atlas de cor é constituído por um número finito de amostras retiradas do espaço de cor. O sistema de Munsell (1915) obedece o critério de uniformidade perceptual e o sistema Pantone (1960) foi criado para ser usado no processo de impressão em papel pela indústria gráfica. Estes sistemas amostram as tonalidades; para cada tonalidade, amostram a saturação; e para cada saturação, amostram a luminância (fig. 2.16).

## 2.9 Exercícios

**2.1** *O que é metamerismo? Dê exemplos de metamerismo em nosso cotidiano.*

**2.2** *Defina o espaço espectral de cor.*

**2.3** *Dê uma justificativa para mostrar que o espaço espectral de cor tem dimensão infinita.*

**2.4** *Mostre que o sólido das cores visíveis é um conjunto convexo e apresente uma visualização 3D do sistema CIE-RGB.*

**2.5** *Justifique por que as cores espectrais estão situadas no bordo do diagrama de cromaticidade.*

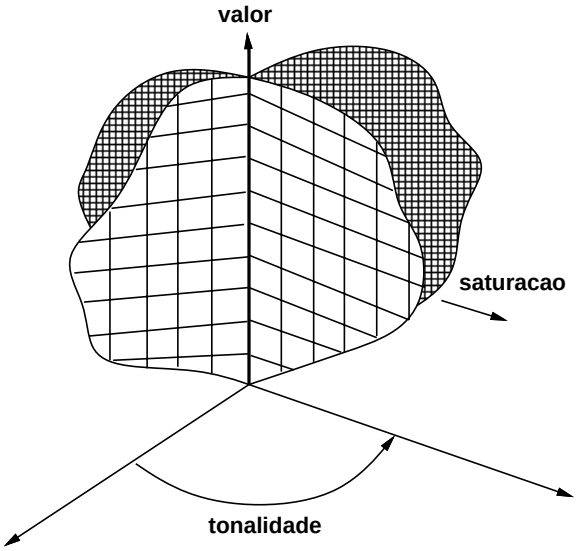



Figura 2.16: Sistema de Munsell.

**2.6** *Explique por que a chamada linha púrpura é a única parte retilínea do bordo do diagrama de cromaticidade.*

**2.7** *Defina mapa de cor (computacionalmente e matematicamente) e crie um mapa de cor que contenha todas as cores espectrais.*

**2.8** *Crie uma imagem para visualizar o mapa de cor do exercício anterior e discuta-a em relação à palavra todas.*

**2.9** *Discuta a representação do diagrama de cromaticidade no computador e crie uma imagem do diagrama de cromaticidade do sistema CIE-XYZ.*

**2.10** *Dê uma classificação dos sistemas de cor utilizados em Computação Gráfica, dando um exemplo de cada tipo de sistema.*

**2.11** *Defina e dê uma descrição dos parâmetros perceptuais utilizados na escolha de cor no computador.*

**2.12** *O sistema de interface HSV é definido a partir do sistema padrão RGB.*

*a) Exiba um algoritmo para a conversão de uma cor do sistema mRGB (RGB de um monitor) para o sistema HSV correspondente e outro para converter do sistema HSV para o mRGB.*

*b) Explique o princípio de funcionamento dos dois algoritmos.*

*c) Discuta as diferenças entre os dois sistemas.*

*d) Diga como pode ser feita uma interpolação entre duas cores no sistema HSV.*

**2.13** *Dado um monitor com base de primárias com coordenadas de cromaticidade R(0.628, 0.346), G(0.268, 0.588) e B(0.150, 0.070).*

*a) Mostre que a cor com coordenadas de cromaticidade (0.274, 0.717) não pode ser exibida de modo preciso neste monitor.*

*b) Qual a denominação da cor do item anterior?*

**2.14** *Considere o sólido de cor de um dispositivo gráfico e uma cor que não pode ser representada acuradamente neste dispositivo.*

*a) Quais são os tipos de situação que podem causar este problema.*

*b) Proponha métodos de aproximar esta cor no dispositivo.*

*c) Discuta os aspectos positivos e negativos de cada método proposto.*

**2.15** *Calcule as coordenadas de cromaticidade de uma cor com coordenadas* $C = (r, g, b)$.

**2.16** *Dado que a mistura de duas cores, segundo as leis de Grassman, pode ser calculada pela adição de suas coordenadas tricromáticas (X,Y,Z) e que os valores de cromaticidade* $(x, y, z)$ *de uma cor são*

$$x = \frac{X}{(X+Y+Z)}; \qquad y = \frac{Y}{(X+Y+Z)}; \qquad z = \frac{Z}{(X+Y+Z)} \quad ou$$

$$X = \frac{xY}{y}; \qquad Y = Y; \qquad Z = \frac{(1-x-y)Y}{y},$$

*determine as coordenadas de cromaticidade da mistura das três cores* $c_1, c_2, c_3$ *dadas no sistema CIE-xyY, respectivamente, por (0.1,0.3,10.0), (0.35,0.2,10.0), (0.2,0.5,10.0).*

$$\begin{aligned} T_1 &= \frac{Y_1}{y_1}; \quad T_2 = \frac{Y_2}{y_2}, \\ X_1 &= (x_1T_1, Y_1, (1-x_1-y_1)T_1), \\ X_2 &= (x_2T_2, Y_2, (1-x_2-y_2)T_2). \end{aligned}$$

$$\begin{aligned} X_{12} = X_1 + X_2 &= (x_1T_1 + x_2T_2, Y_1 + Y2, (1-x_1-y1)T_1 + (1-x_2-y_2)T_2) \Rightarrow \\ X_{12_x} + X_{12_y} + X_{12_z} &= x_1T_1 + x_2T_2 + Y_1 + Y_2 + T_1 - x_1T_1 - y_1T_1 + T_2 - x_2T_2 - y_2T_2 \\ &= T_1 + T_2. \end{aligned} \tag{2.1}$$

$$\Rightarrow x_{12} = \frac{x_1T_1 + x_2T_2}{T_1 + T_2}; \quad y_{12} = \frac{y_1T_1 + y_2T_2}{T_1 + T_2}; \quad Y_{12} = Y_1 + Y_2. \tag{2.2}$$

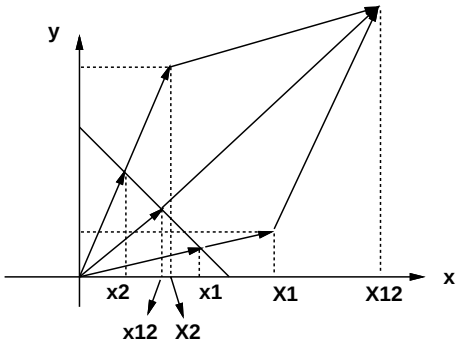



Figura 2.17: Combinação de Cores.

**2.17** *Dada uma cor $c_1 = (x_1, y_1, Y_1)$ e o ponto acromático $a = (a_x, a_y, 1)$ derive a fórmula para calcular a cor $c_2 = (x_2, y_2, Y_2)$ complementar de $c_1$.*

$$c_1 + c_2 = a \ (2.2) \Rightarrow \begin{cases} x_1T_1 + x_2T_2 = a_x(T_1 + T_2); \\ y_1T_1 + y_2T_2 = a_y(T_1 + T_2); \\ Y_1 + Y_2 = y_1T_1 + y_2T_2 = 1. \end{cases} \tag{2.3}$$

$$a_y(T_1 + T_2) = 1 \Rightarrow (T_1 + T_2) = \frac{1}{a_y} \Rightarrow T_2 = \frac{1}{a_y} - T_1.$$

$$\begin{aligned} x_2 &= \frac{a_x(T_1 + T_2) - x_1T_1}{T_2} = \frac{\frac{a_x}{a_y} - x_1T_1}{\frac{1}{a_y} - T_1}, \\ y_2 &= \frac{a_y(T_1 + T_2) - y_1T_1}{T_2} = \frac{1 - y_1T_1}{\frac{1}{a_y} - T_1}, \\ Y_2 &= 1 - Y_1. \end{aligned} \tag{2.4}$$

**2.18** *O comprimento de onda dominante de uma cor c é obtido, calculando-se a interseção da reta definida pelas coordenadas de cromaticidade de c e do ponto acromático com o bordo do diagrama de cromaticidade. Pela definição acima, alguns pontos contidos no diagrama de cromaticidade não têm comprimento de onda dominante no espectro visível.*

*a) Por que?*

*b) Proponha um esquema que permita representar o comprimento de onda dominante de um desses pontos, baseado no conceito de complementaridade.*

**2.19** *Dados dois espaços tricromáticos de cor A e B com bases $(A_1, A_2, A_3)$ e $(B_1, B_2, B_3)$, respectivamente, e a matriz $[a_{ij}]$ de mudança de base entre A e B.*

*a) Mostre como calcular as coordenadas $(c_1, c_2, c_3)$ de uma cor c em A a partir de suas coordenadas em B.*

*b) Sendo $rB_i(\lambda)$ as funções de reconstrução de cor associadas às primárias $(B_1, B_2, B_3)$, diga como podem-se obter as funções de reconstrução de cor $rA_i(\lambda)$ associadas às primárias $(A_1, A_2, A_3)$.*

*c) Justifique os itens anteriores.*

**2.20** *Considere o sistema de cor RGB do monitor (mRGB).*

*a) Qual a geometria do sólido de cor desse sistema?*

*b) Faça um esboço do sólido de cor.*

*c) Considere as cores do sistema mRGB normalizadas para o interval0 $[0, 1]$. Defina o sistema de cores complementares $CMY$, correspondentes às cores primárias do sistema $mRGB$. Mostre que*

$C = 1 - R$;

$M = 1 - G$;

$Y = 1 - B$.

# Capítulo 3

# Dispositivos Gráficos

Os equipamentos desempenham um papel bastante importante na computação gráfica, não só pela relevância da imagem como um dos produtos da atividade, mas também pela influência que as características dos equipamentos exercem nos processos computacionais da área.

Neste capítulo advoga-se que a representação dos dados no computador e a apresentação da imagem nos equipamentos gráficos formam uma via de mão dupla, com muitas interrelações e interdependências. Serão estudados os equipamentos com relação a certos critérios de classificação, em particular do ponto de vista da classificação funcional e, em cada caso, serão dados exemplos de equipamentos que utilizam os diversos formatos de dados gráficos.

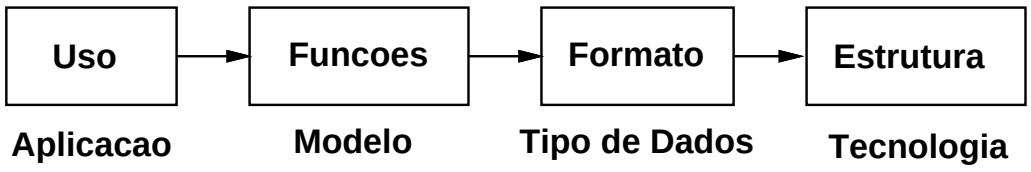



Figura 3.1: Modelo Conceitual.

## 3.1 Formato de Dados Gráfico

Existem dois formatos básicos para a representação e o armazenamento de dados gráficos no computador: o *formato vetorial* e o *formato matricial*. O formato vetorial é utilizado, em geral, para descrever a estrutura geométrica dos objetos gráficos, enquanto o formato matricial está freqüentemente associado à imagem digital.

### 3.1.1 Formato Vetorial

No formato vetorial, os dados são representados por unidades básicas de informação, descritas por coordenadas em um espaço vetorial. Estes elementos são associados a posições ou a vetores deste espaço. No primeiro caso, eles podem ser usados na especificação dos pontos iniciais e finais de segmentos de reta, vértices de polígonos e malhas de controle de curvas ou superfícies paramétricas. Já no segundo caso, eles podem especificar forças, direções ou orientações. A dimensão do espaço vetorial determina o número de coordenadas de seus elementos básicos. Além disso, o espaço pode ser contínuo ou discreto, tendo suas coordenadas representadas por números reais ou inteiros, respectivamente.

As informações geométricas, em muitos casos, precisam ser complementadas por informações topológicas para especificar completamente o modelo de um objeto gráfico. Este assunto será discutido em profundidade nos capítulos dedicados à Modelagem.

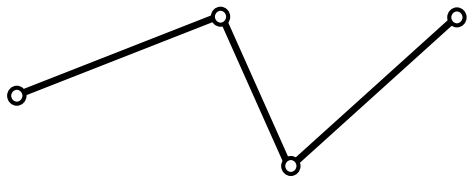


Figura 3.2: Formato Vetorial.

### 3.1.2 Formato Matricial

As imagens digitais são tratadas no capítulo 4. Entretanto, no que se segue, define-se este conceito de uma forma um tanto limitada, porém suficiente, ao propósito de estudo dos equipamentos gráficos.

Uma *imagem digital* é uma matrix $M \times N$ onde cada elemento da matriz é um elemento de um espaço vetorial $V$. O caso mais comum é quando $V$ é um espaço de cor. Chama-se de *resolução* da imagem à ordem $M \times N$ da matriz. De modo análogo, define-se *imagem volumétrica* tomando matrizes de ordem $M \times N \times P$, onde cada entrada também é um elemento do espaço de cor. É comum utilizarem-se os termos *imagens bidimensionais* e *imagens tridimensionais* para cada um dos dois casos, respectivamente.

O formato matricial de dados permite a representação de imagens bidimensionais e volumétricas. Note-se que a estrutura da matriz determina implicitamente a conectividade de seus elementos. Cada elemento é chamado de *pixel*.

### 3.1.3 Conversão entre Formatos

A conversão entre formatos é desejável e muitas vezes necessária.

Dá-se o nome de *rasterização* à transformação de dados do formato vetorial para o

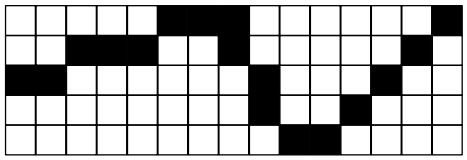
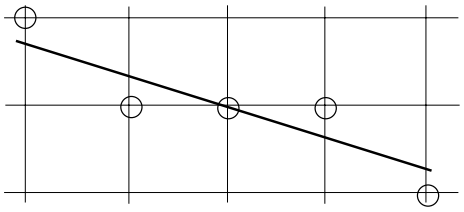


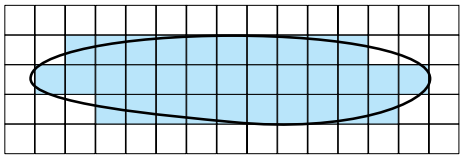
Figura 3.3: Formato Matricial.

formato matricial (fig. 3.4 (a)). Essa técnica, que constitui uma parte fundamental dos algoritmos de síntese de imagens, é estudada em detalhe no Capítulo 10.

A transformação oposta, ou seja, de dados no formato matricial para o formato vetorial, é chamada de *segmentação* e faz parte da área de Visão Computacional. Essa conversão, em alguns casos, não é bem definida, sendo impossível de ser realizada (fig. 3.4 (b)).



(a)



(b)

Figura 3.4: Conversão entre Formatos.

## 3.2 Classificação dos Dispositivos

Os dispositivos gráficos são projetados usualmente de forma a privilegiar um dos formatos de dados descritos acima. Isso não significa que um tipo de equipamento somente possa operar com um determinado formato de dados. Os dispositivos do tipo matricial podem, por exemplo, reproduzir segmentos de reta utilizando processos de rasterização. Alguns equipamentos deste tipo dispõem até mesmo de suporte em hardware para tal operação. Embora não seja tão comum, os dispositivos do tipo vetorial podem também reproduzir imagens utilizando padrões de linhas. Por exemplo, em mapas e desenhos técnicos, vários tipos de hachuras são empregados para diferenciar áreas, simulando tonalidades de cinza.

A evolução dos equipamentos gráficos reflete, de uma certa forma, o desenvolvimento da computação gráfica como um todo. Inicialmente, quando havia uma grande preocupação com a modelagem geométrica, os dispositivos vetoriais eram mais populares. Depois, com a ênfase na síntese de imagens sofisticadas, os equipamentos matriciais passaram a ser mais utilizados. Atualmente, a tendência é a busca de soluções integradas, combinando dispositivos do tipo vetorial e matricial nas diversas fases do processo da computação gráfica para atender classes específicas de aplicações. De um modo geral, os dispositivos vetoriais estão vinculados à especificação e manipulação dos modelos geométricos, enquanto que os dispositivos matriciais estão relacionados com a exibição e o processamento de imagens.

Além disso, vários fatores de natureza técnica, industrial e econômica, determinaram a evolução dos equipamentos gráficos. Os dispositivos do tipo matricial necessitam do uso de muita memória para armazenar a imagem. Por outro lado, os dispositivos do tipo vetorial se beneficiaram da tecnologia de radar, numa época em que o custo da mémoria inviabilizava o uso de dispositivos de formato matricial (década de 60 e 70). Já os dispositivos do tipo matricial foram impulsionados na década de 80 por dois fatores: a queda do preço de memória, e a revolução nas comunicações que a televisão provocou (neste capítulo será visto que os dispositivos de saída gráfica mais comuns utilizam o formato matricial e se baseiam na tecnologia de monitores de televisão). Os avanços recentes nas áreas da supercomputação e da computação paralela têm tido um impacto significativo nos dispositivos de processamento gráfico.

### 3.2.1 Critérios de Classificação

No estudo dos dispositivos gráficos é necessário criar abstrações das suas características operacionais, de modo que o vínculo entre programas e equipamentos não se transforme num fator de dependência. Vários aspectos contribuem para o estabelecimento de critérios para uma classificação dos equipamentos gráficos. Nessa análise há categorias de equipamentos estruturadas, hierarquicamente, segundo dois pontos de vista: o funcional, e o do formato dos dados gráficos.

Em relação ao critério funcional, dividem-se os dispositivos gráficos em:

- equipamentos de entrada;
- equipamentos de processamento;
- equipamentos de saída.

Quanto ao formato de dados os dispositivos gráficos se dividem em equipamentos do *tipo vetorial* e do *tipo matricial*. O diagrama da figura 3.5 mostra como essas classificações se relacionam.

## 3.3 Equipamentos de Entrada Gráfica.

Os equipamentos de entrada de dados gráficos são equipamentos de captação de informações gráficas. Do ponto de vista do formato da imagem, os dispositivos podem ser classificados como vetoriais e matriciais.

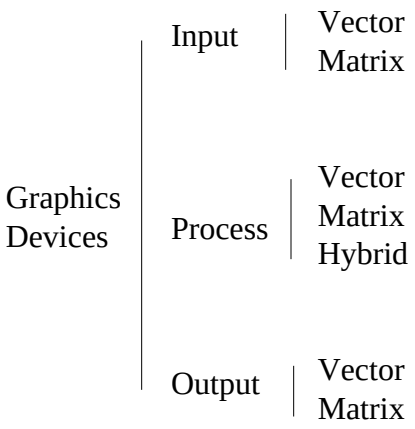



Figura 3.5: Classificação dos Equipamentos.

Os dispositivos de entrada vetorial são em sua maioria utilizados como componentes de estações gráficas interativas. Um exemplo típico é o "mouse", componente indispensável em uma estação de trabalho interativa que utilize ambiente de janelas.

Os dispositivos de entrada matricial são tradicionalmente utilizados de modo não-interativo, devido, principalmente, ao grande volume de dados que devem ser manipulados. Esta situação tende a se modificar com a evolução dos equipamentos de aquisição, exibição e processamento de imagens, que poderão tornar possível aplicações em tempo real envolvendo dados matriciais.

### 3.3.1 Dispositivos de Entrada Vetorial

Os dispositivos cujo sistema de coordenadas é absoluto são: a "light pen", a "tablet", o "touch pannel", e o "3D-digitizer". As características técnicas relevantes dos dispositivos de entrada vetorial absolutos são a sua resolução, linearidade, repetibilidade, e área de ação.

A *light pen* é um dispositivo bidimensional que funciona necessariamente acoplado a um terminal de vídeo. Este equipamento é composto por uma caneta com uma foto-célula na ponta ligada ao circuito de vídeo do terminal. Dessa maneira, é possível detectar pontos apresentados na tela e conseqüentemente sua localização. Este dispositivo surgiu com os primeiros equipamentos gráficos interativos. Atualmente ele caiu em desuso devido a alguns problemas técnicos apresentados.

O *touch pannel* também é um dispositivo bidimensional de entrada que deve ser integrado a um terminal de vídeo. Ele consiste em uma tela transparente, sensível ao toque, que é sobreposta à tela do terminal. Este dispositivo apresenta severas limitações em termos de resolução. Por este motivo, ele é indicado apenas para a seleção de objetos gráficos apresentados na tela. Um exemplo desse tipo de utilização pode ser visto em alguns terminais eletrônicos de banco.

A *tablet* ou *mesa digitalizadora* consiste em uma base plana e um instrumento indicador em forma de caneta ou bloco. No indicador existem um ou mais botões. O equipamento

fornece a posição do indicador em relação ao sistema de referência da base, juntamente com o estado dos botões ("on" ou "off"). Além desses parâmetros, com relação aos dados de entrada, a tablet em geral é um dispositivo bidimensional. Porém em alguns dispositivos é possível especificar a pressão exercida na ponta da caneta e também a sua orientação. Nesse caso, o espaço vetorial de entrada tem dimensão 6 (i.e. posição $(X, Y)$, pressão $(\psi, \rho)$ e orientação $(\alpha, \beta, \gamma)$). Estes dados podem ser interpretados de maneira bastante efetiva em um programa de pintura eletrônica, para permitir a simulação de instrumentos tradicionais como o pincel, o crayon, etc. O instrumento indicador pode ou não estar ligado à mesa digitalizadora por um fio. A caneta sem fio possibilita uma interação mais natural, especialmente para aplicações de desenho livre.

O *3D digitizer* permite captar posição $(X, Y, Z)$ e orientação $(\alpha, \beta, \gamma)$ no espaço tridimensional. Este dispositivo é constituído por um emissor magnético e um sensor que, em geral, tem a forma de uma caneta. Ele é bastante conveniente para digitalizar diretamente pontos na superfície de um objeto tridimensional.

Os dispositivos vetoriais que operam com referencial relativo são: o "mouse", a "trackball", o "joystick" e os "dials". Esses dispositivos registram deslocamentos que são transformados em informações de movimento relativo. Isso implica que o programa de controle do dispositivo deve manter a posição corrente que é atualizada a cada movimento relativo.

O *mouse*, como foi dito anteriormente, é um dos dispositivos de entrada gráfica mais comuns atualmente, por estar associado à estações de trabalho que utilizam sistemas de janelas. Esse equipamento consiste em um pequeno bloco com botões de pressão, que se comunica com o computador.

A *trackball* é constituída por uma esfera que gira livremente numa base. Os movimentos de rotação em relação a dois eixos ortogonais são transformados em informações de posição de maneira semelhante à do mouse.

O *joystick* é formado por uma haste conectada a uma base. Em geral, o movimento da haste é transformado em um vetor de velocidades que controla a variação dos dados posicionais. Ou seja, na medida em que a haste se afasta do eixo central, a velocidade aumenta proporcionalmente naquela direção. Alguns tipos de joystick possuem um terceiro grau de liberdade, associado à rotação da haste. Esse dispositivo é utilizado com freqüência como interface de entrada dos vídeo-games.

Os *dials* são potenciômetros tipicamente montados em grupos de seis ou oito. Os potenciômetros fornecem valores escalares e podem ter uma faixa de rotação fixa menor do que 360 graus, ou podem permitir a rotação livre. No primeiro caso, o intervalo escalar dos dados é mapeado na faixa de operação do dispositivo, enquanto no segundo caso os dados são especificados de forma relativa.

### 3.3.2 Dispositivos de Entrada Matricial

A estrutura dos dispositivos de entrada do tipo matricial consiste em um sensor que capta sinais no espaço ambiente e um circuito digitalizador que converte esses sinais analógicos para o formato matricial.

O processo de conversão de uma imagem para uma imagem digital é conhecido como *digitalização*. Os dispositivos de entrada matricial são, em sua maioria, destinados à digitalização de imagens. Dependendo do meio no qual se encontra a imagem a ser digitalizada

têm-se o "frame grabber", o "scanner", o "film scanner", e o "depth scanner".

O *frame grabber* faz a digitalização a partir de um sinal analógico de vídeo. O sinal de vídeo pode ser gerado diretamente por uma câmera ou por um equipamento de reprodução de vídeo. A resolução geométrica da imagem digitalizada é a resolução de vídeo, que é aproximadamente de $512 \times 512$ pixels. Os dispositivos mais sofisticados digitalizam com uma resolução de cor de 24 bits.

O *scanner* digitaliza a partir de imagens em papel. A imagem é colocada sobre uma superfície transparente, em geral plana ou cilíndrica, que se move numa direção ortogonal a um elemento de digitalização de linha. Este elemento se compõe de uma fonte de luz e de um sensor que mede a luz refletida linha por linha, em sincronismo com o deslocamento da imagem. A resolução deste dispositivo está situada entre 200 a 1500 pontos por polegada.

O *film scanner* digitaliza a partir de imagens em transparências utilizando o laser para maior resolução. Um feixe de luz precisamente colimado é dirigido ao filme e a quantidade de luz transmitida é medida por uma célula fotoelétrica. Este dispositivo pode atingir uma resolução superior a 2000 pontos por polegada.

O *depth scanner*, ao invés de digitalizar uma imagem, captura informações de uma cena tridimensional, produzindo uma matriz de coordenadas com a profundidade de cada ponto da cena. A estrutura dessa matriz depende do processo de varredura utilizado. Os tipos mais comuns possuem varredura plana ou cilíndrica.

## 3.4 Equipamentos de Processamento Gráfico

Os equipamentos de processamento gráfico são computadores com uma arquitetura especial, orientada para a manipulação e processamento de dados gráficos.

Dois problemas recorrentes do equacionamento da arquitetura dos equipamentos gráficos de processamento estão relacionados com aspectos de funcionalidade e acoplamento. O primeiro aspecto diz respeito ao grau de especialização das funções do processador gráfico. Processadores especializados são mais eficientes e caros, enquanto processadores de propósito geral são mais flexíveis e baratos. O segundo aspecto diz respeito ao canal de comunicação do processador gráfico com o sistema de computação. Um alto acoplamento possibilita um acesso rápido aos dados do sistema, mas implica em grande interdependência. Em contrapartida, um baixo acoplamento permite maior independência entre o processador gráfico e o sistema, mas implica em uma comunicação restrita entre eles.

Estes aspectos estão claramente interrelacionados. Um processador gráfico especializado necessita de um canal de comunicação de alta capacidade porque muitas das funções gráficas serão realizadas pelo processador principal. Por outro lado, um processador gráfico mais geral dispõe de recursos para executar localmente grande parte das funções de visualização, podendo ter um canal de comunicação limitado com o computador principal.

Existe uma tendência na indústria de equipamentos gráficos que consiste em gradativamente adicionar funcionalidade aos processadores gráficos especializados, até que eles se tornam equivalentes a um computador de uso geral. A única opção, que existe então, é completar o ciclo voltando à especialização. Este fenômeno ficou conhecido como a "roda da reencarnação" (weel of reincarnation).

### 3.4.1 Dispositivos de Processamento Vetorial

Os dispositivos do tipo vetorial se destinam principalmente ao processamento de modelos geométricos. Eles atuam portanto sobre as coordenadas das diversas componentes dos modelos, tais como segmentos de reta, polígonos, e etc. Em função do número de processadores, podem-se ter dispositivos do tipo SISD (single-instruction, single data stream), ou MISD (multiple-instruction, single data stream).

Os dispositivos do tipo SISD são uniprocessadores que possuem instrucões especiais para processamento de dados geométricos, do tipo multiplicação de matrizes por vetores.

Os dispositivos do tipo MISD são pipelines compostas de vários processadores organizados seqüencialmente. O processamento gráfico é dividido em etapas, onde cada processador é especializado numa classe de operações gráficas, como projeção, recorte, etc.

### 3.4.2 Dispositivos de Processamento Matricial

Os dispositivos do tipo matricial são equipamentos multiprocessadores utilizados para o processamento de imagens, para a rasterização (Capítulo 10) e outros algoritmos gráficos paralelizáveis. Há os dispositivos do tipo SIMD (single-instruction, multiple data stream), ou MIMD (multiple-instruction, multiple data stream) com diferentes configurações dos processadores.

Os dispositivos do tipo SIMD são utilizados para realizar a mesma operação em vários elementos simultaneamente. Um exemplo desse tipo de equipamento é o computador de imagem Pixar.

Os dispositivos do tipo SIMD são processadores paralelos que se comunicam entre si. A maneira como eles estão interligados define uma topologia de rede e, conseqüentemente, o fluxo de dados. Um equipamento desse tipo é a estação gráfica Pixel Machine.

## 3.5 Equipamentos de Saída Gráfica

Os equipamentos de saída gráfica são equipamentos que permitem a visualização de dados gráficos. Podem ser subdivididos em dispositivos vetoriais ou matriciais, de acordo com o tipo de dado gráfico por eles manipulado. Dentre todos os equipamentos gráficos de saída, os *dispositivos de exibição de vídeo* são, sem dúvida alguma, os mais importantes e mais comuns. A tecnologia de vídeo implica em uma série de características comuns aos equipamentos vetoriais e matriciais. Por esse motivo, inicia-se esta seção analisando a estrutura básica dos dispositivos de exibição de vídeo. Em seguida discutem-se os detalhes específicos dos diversos tipos de equipamentos de vídeo no contexto dos dispositivos vetoriais e matriciais.

### 3.5.1 Dispositivos de Exibição de Vídeo

Um dispositivo de exibição de vídeo é constituído por quatro elementos: um monitor, um controlador de vídeo, uma memória de exibição ("frame buffer") e um *conversor digital analógico* (fig. 3.6).

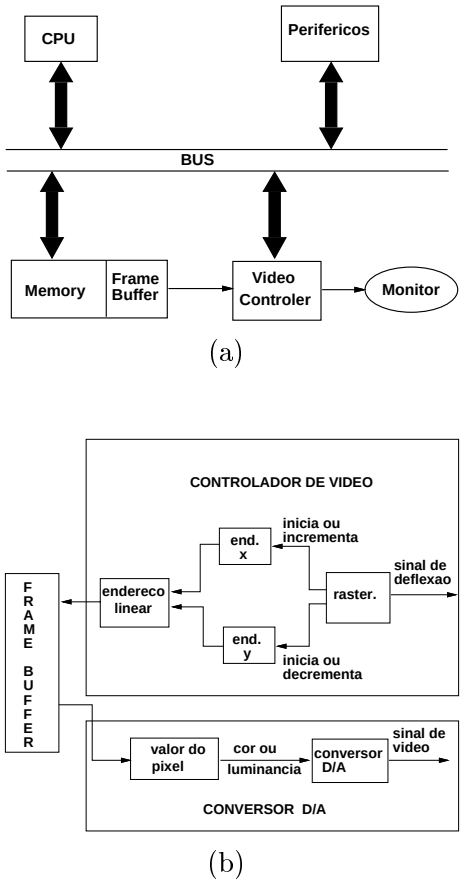



Figura 3.6: Dispositivo de Vídeo.

O *monitor de vídeo* (fig. 3.7) consiste em um tubo de raios catódicos com uma tela e um canhão que produz um ou mais feixes de eletrons controlado por um sistema de focalização e exploração. Em cada ponto da tela se colocam uma ou mais camadas de fósforo, de modo que, ao atingir um desses pontos, o feixe de eletrons provoca a emissão de radiação eletromagnética na faixa visível do espectro. O funcionamento básico de qualquer monitor de vídeo é bastante similar ao funcionamento de um monitor de televisão. No sistema NTSC[1] e PAL-M[2] existem 525 linhas (483 visíveis) e 644 pixels por linha (razão de aspecto 4:3). Monitores para aplicações gráficas, no entanto, costumam ter uma resolução muito maior, algo em torno de $1024 \times 1024$ pontos endereçáveis.

O espaço de cor do monitor de vídeo depende do número de camadas de fósforo em cada ponto. Os monitores monocromáticos, ou de dois níveis (bitmapped), utilizam uma única camada de um fósforo que é sensibilizado com voltagem mínima ou máxima; os monitores que permitem a exibição de tons de cinza (gray scale) utilizam uma única camada de fósforo cuja sensibilidade produz uma radiação com a luminância proporcional à voltagem aplicada

[1] National Television System Cometee.
[2] Phase Alternating Lines.

ao feixe de eletrons do canhão.

O espaço tricromático de cor é implementado por meio de três tipos de fósforo diferentes em cada ponto (e três feixes de eletrons), de modo que cada fósforo emite uma das cores primárias do sistema. A distância entre os centros dos pontos R, G e B define o *pitch* do tubo, que varia de 0.60 mm num monitor de televisão até 0.21 mm em monitores de alta resolução. Uma máscara metálica perfurada colocada à frente da camada de fósforo garante que cada feixe de eletrons atinge e excita apenas o ponto correspondente ao tipo correto de fósforo.

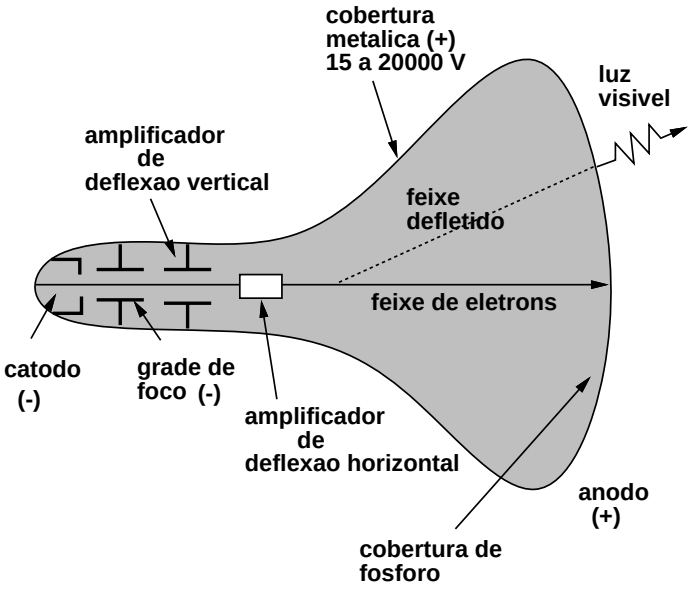



Figura 3.7: Monitor de Vídeo.

Como a resposta luminosa do fósforo utilizado na tela decai exponencialmente com o tempo, a imagem precisa ser redesenhada periodicamente. O número de vezes por segundo que a imagem deve ser exibida na tela para que seja percebida como um fenômeno contínuo no tempo, é chamado de *freqüencia crítica de fusão*. Este número é determinado por vários fatores, desde a persistência do fósforo, até a aspectos psicofisiológicos. Em média ele se situa próximo dos 50 Hz.

O *controlador de vídeo*, também chamado de DPU (Display Processing Unit), tem a finalidade de controlar o movimento de exploração na tela do feixe de eletrons, para que a imagem desejada seja produzida. Esse processo, denominado de *varredura*, pode ser aleatório ou regular. Na varredura aleatória, o feixe se desloca numa trajetória que segue o desenho das curvas da imagem. Na varredura regular, o feixe se movimenta de acordo com um padrão fixo que percorre toda a tela da esquerda para a direita e de cima para baixo, cobrindo-a por linhas horizontais. Este padrão regular é chamado de *raster*. A figura 3.8 ilustra os dois padrões utilizados. Os dispositivos de exibição vetorial empregam a varredura

aleatória enquanto os dispositivos de exibição matricial empregam a varredura regular.

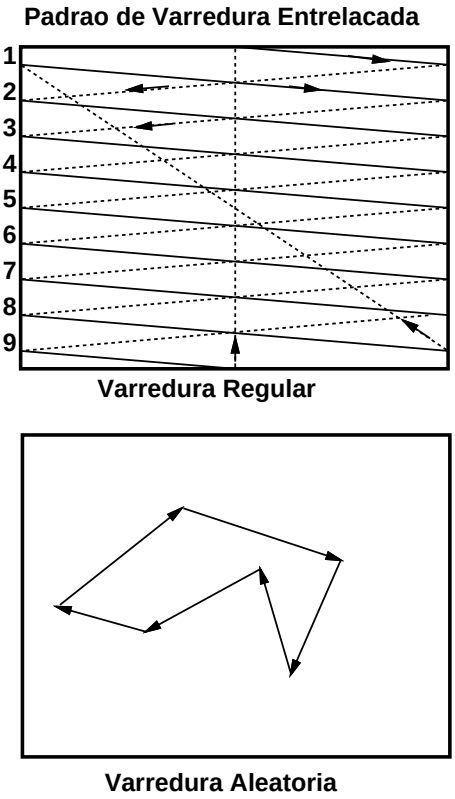



Figura 3.8: Padrões de Varredura.

A *memória de exibição* (frame buffer) armazena os dados que vão ser utilizados para gerar a imagem. Através do conversor digital analógico os valores armazenados nessa memória são convertidos para uma voltagem que é utilizada pelo canhão para gerar o feixe de eletrons. Nos dispositivos matriciais, a memória de exibição é organizada em uma estrutura matricial de modo a armazenar os valores de cada cor dos pixels da imagem. O tamanho da memória de exibição determina a resolução de cor e a resolução geométrica da imagem. A resolução de cor está ligada diretamente ao tipo do espaço de cor do dispositivo. Podem-se identificar três tipos mais comuns de memória de exibição: os do tipo *bitmap* que utilizam 1 bit por pixel, sendo portanto monocromáticos com apenas dois níveis (preto e branco) (fig. 3.9 (a)); os do tipo *cor falsa* ("pseudo-color") que utilizam de 2 a 12 bits para cada pixel, podendo reproduzir imagens monocromáticas e também imagens coloridas, em geral com auxílio de uma look-up table (fig. 3.9 (b)); os do tipo *cor real* ("true color"), que utilizam 24 bits por pixel, 8 bits para cada uma das componentes das cores primárias R (red), G (green) e B (blue). Vale ressaltar que mesmo monitores de exibição de vídeo que utilizam cor real podem possuir uma look-up table cuja finalidade é permitir alterações de cor da imagem

de uma forma mais rápida e flexível, além de permitir que a cooreção gama seja feita por intermédio da look-up table. Nesse caso o esquema correto para os dispositivos de cor real é como mostrado na figura 3.9 (c).

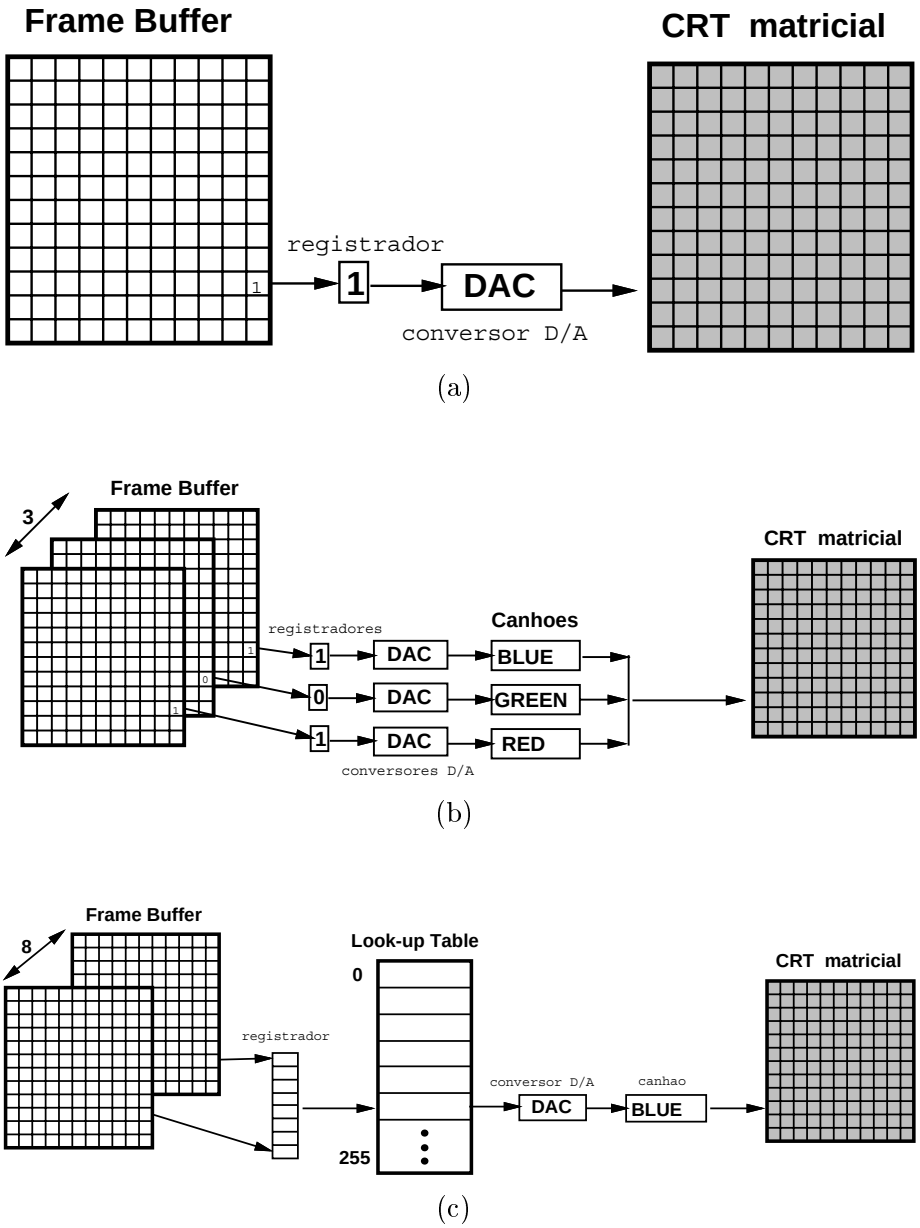



Figura 3.9: Memória de Exibição.

Nos dispositivos vetoriais, a memória de exibição contém instruções de desenho com as coordenadas de tela dos objetos gráficos. Este conjunto de instruções, denominado de *lista de exibição* ("display list"), é executado ininterruptamente pela controladora de vídeo para

manter a imagem visível na tela.

### 3.5.2 Dispositivos de Saída Vetorial

Os dispositivos de saída vetorial produzem imagens traçando segmentos de reta descritos pelas coordenadas de seus pontos iniciais e finais. Nesta categoria de equipamentos estão: o *display caligráfico*, o *display de armazenamento*, e as *traçadoras*.

O *display caligráfico* é um dispositivo de exibição de vídeo interativo. O sistema de varredura é aleatório e o fósforo do monitor é de baixa persistência e distribuído de forma contínua sobre a tela, ou seja, o feixe de eletrons se movimenta livremente sobre a tela e a imagem precisa ser regenerada constantemente. Essas características permitem a manipulação dos dados em tempo real (fig. 3.10).

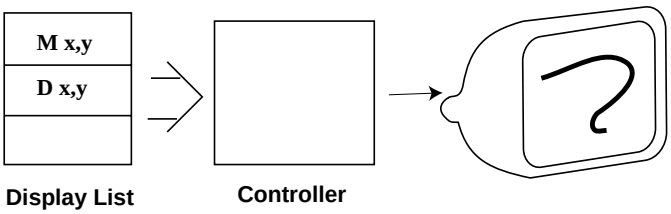



Figura 3.10: Display Caligráfico.

O *display de armazenamento* ou *DVST* (Direct View Storage Tube) dispõe de um monitor de vídeo com fósforo de alta persistência (cerca de uma hora no máximo). Nesse equipamento a imagem traçada é mantida na tela através de um circuito especial, sem necessidade de regeneração. Uma desvantagem desse tipo de monitor é que partes da imagem não podem ser modificadas sem que a imagem inteira seja apagada (o que leva cerca de um segundo) e redesenhada. Esses monitores têm uma importância histórica em Computação Gráfica: devido ao baixo custo, pelo fato de não necessitarem de uma memória de exibição, eles deram início a grande expansão das aplicações da Computação Gráfica nas diversas áreas [3].

As *traçadoras* são equipamentos eletromecânicos para o desenho de linhas sobre papel ou filme. Esse equipamento é constituído por um suporte para a superfície de desenho, e um mecanismo de controle do instrumento de traçado. Quanto ao suporte, as traçadoras podem ser de *mesa* ou de *tambor*. Quanto ao mecanismo de desenho, podem ser do tipo *contínuo* ou *incremental*.

[3] A importância desse tipo de display pode ser avaliada pelo seguinte fato. Esse foi o critério escolhido pela SIGGRAPH para definir quem pode pertencer ao grupo dos Pioneiros da Computação Gráfica. A pessoa deve ter trabalhado na área antes do aparecimento do primeiro display DVST da Tektronix

### 3.5.3 Dispositivos de Saída Matricial

Os dispositivos de saída matricial produzem imagens a partir de uma matriz de intensidades. Nesta categoria de equipamentos estão: o display raster, o painel de plasma, o display de cristal liquído, as impressoras de impacto, as impressoras gráficas, e o film recorder.

O *display raster* é um dispositivo de exibição de vídeo matricial. Ele consiste em um monitor, um controlador de vídeo e uma memória de display que armazena a matriz de imagem. Esse equipamento normalmente dispõe de uma "look up table" (fig. 3.11). Projetores de vídeo também podem ser utilizados nos displays raster em substituição aos monitores.

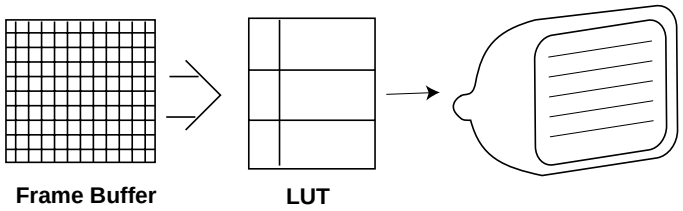



Figura 3.11: Display Raster.

O *painel de plasma* é constituído por uma matriz de células microscópicas de neon. Ele é um display monocromático, bitmap de armazenamento, não precisando portanto de memória adicional nem de regeneração da imagem.

O *display de cristal liquído* é semelhante ao painel de plasma, mas utiliza células de cristal líquido na matriz de imagem. Essa tecnologia tem um grande potencial, por permitir displays coloridos e monitores com tela plana.

As *impressoras de impacto* são destinadas principalmente para saída alfa-numérica. Algumas delas, do tipo "dot matrix", têm capacidade gráfica. Possuem cabeças com um conjunto que contém de sete a vinte e quatro pinos (7, 9, 18 ou 24) que são impulsionados contra uma fita impregnada de tinta sobre o papel.

As *impressoras gráficas* podem utilizar uma técnica de reprodução a laser, eletrostática, térmica, ou por jato de tinta. As *impressoras laser* utilizam um raio de laser para sensibilizar os pontos que não aparecem na imagem (permanecem brancos), retirando a carga de um ponto (tornando-o negativo) sobre um cilindro rotativo de selênio — que é um material foto-sensível — carregado positivamente. Partículas de toner (carregado negativamente) são atraídas para a superfície do cilindro e, em seguida, transferidas para o papel (fig. 3.12).

As *impressoras eletrostáticas* carregam negativamente os pontos do papel — por meio de um pente com contatos elétricos — que devem receber tinta. É utilizado um toner carregado positivamente. As *impressoras térmicas* transferem seletivamente pigmentos de tinta para o papel por calor. As *impressoras da jato de tinta* possuem uma cabeça de impressão que lança gotículas de tinta no papel por meio de pequenos jatos. As impressoras gráficas podem ser monocromáticas ou coloridas.

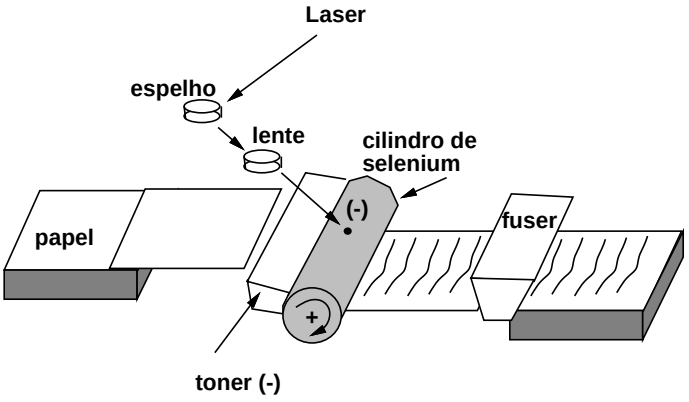



Figura 3.12: Impressora a Laser.

O "*film recorder*" registra em película fotográfica imagens geradas por computador. Para reproduzir a cor, o filme é exposto três vezes, através de filtros, para as componentes de cor azul, verde e vermelha.

## 3.6 Estações Gráficas Interativas

As estações gráficas combinam dispositivos gráficos de entrada, saída e processamento em um sistema destinado ao uso interativo. Nesta seção descrevem-se os tipos mais comuns de estações gráficas, que podem ser classificadas em *estações caligráficas* e *estações matriciais*. Estas, por sua vez, se divididem em estações de baixa e alta performance. Além destas, existem estações com uma arquitetura dedicada para aplicações específicas, tais como o processamento de imagens em tempo real e a visualização de dados volumétricos.

As estações caligráficas são constituídas por um monitor de exibição de vídeo do tipo caligráfico, um processador de display e vários dispositivos de entrada, tais como teclado, mesa digitalizadora e dials. Um exemplo clássico desse tipo de equipamento são as estações da série PS-300 produzidas pela Evans & Sutherland (fig. 3.13).

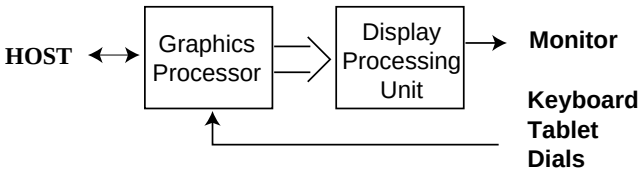



Figura 3.13: Estação Caligráfica.

As estações gráficas de baixa performance integram um computador de uso geral com um

display bitmap, teclado e mouse. Opcionalmente, o display pode incorporar um processador gráfico para operações com imagens do tipo BitBlt (Bit Block Transfer). Um exemplo desse tipo de equipamento são as estações mais simples da linha "SPARCstation" da Sun Microsystems e da linha RS-6000 da IBM (fig. 3.14).

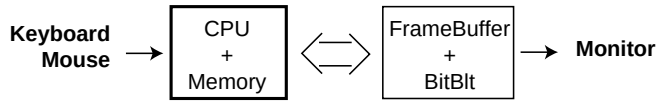



Figura 3.14: Estação Bitmap.

As estações gráficas coloridas de alta performance conhecidas como "super-workstations" são os equipamentos interativos de tecnologia mais moderna. Elas são compostas por um display raster colorido, processadores gráficos organizados de forma seqüencial ou em paralelo, além de diversos dispositivos de entrada como teclado, mouse e dials (fig. 3.15). Exemplos desse tipo de equipamento são as estações gráficas IRIS da Silicon Graphics e a DN-10000 da HP/Apolo. A estação IRIS se utiliza de uma CPU com processamento paralelo, e de um processador gráfico com processamento vetorial e paralelo; A DN10000 se utiliza de processamento paralelo na CPU que é utilizada, também, para fazer operações gráficas.

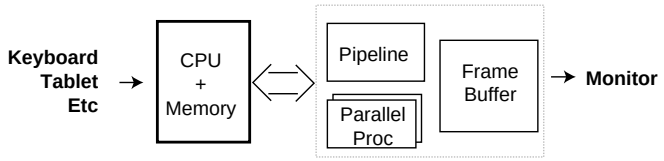



Figura 3.15: Estação Gráfica de Alta Performance.

Entre as estações gráficas do tipo bitmap com baixa performance, e as super-workstations, existe toda uma faixa intermediária de estações gráficas variando a capacidade de processamento da CPU e do processador gráfico.

## 3.7 Exercícios

**3.1** *Explique a diferença entre os formatos gráficos vetorial e matricial.*

**3.2** *Discuta o problema da conversão de dados gráficos. Dê exemplos práticos de aplicações.*

**3.3** *Explique o funcionamento de um CRT. Em seguida, particularize a sua explicação para:*

*a) Dispositivo caligráfico;*

*b) Dispositivo raster;*

*c) Dispositivo de armazenamento (storage tube);*

*d) Destes, quais são os dispositivo chamados de randômicos? Por que?*

*e) Qual o formato dos dados de cada um?*

*f) Classifique-os a partir do critério funcional e quanto ao modo de utilização.*

*g) Explique por que os primeiros monitores de vídeo matriciais utilizavam varredura entrelaçada e por que este tipo de varredura não produz flicker.*

**3.4** *Descubra a resolução espacial e de cor das placas CGA, EGA, VGA, SuperVGA e de uma impressora matricial padrão Epson. Elas possuem look-up table? Com quantas entradas? Qual o universo (tamanho da palete) de cor?*

**3.5** *Liste a funcionalidade básica de um subsistema gráfico 2D e 3D.*

# Capítulo 4

# Imagem

A imagem digital é a materialização de grande parte dos processos de Computação Gráfica, servindo como elo de ligação entre o usuário e esses processos, evidenciando, desta forma, os seus resultados. É válido afirmar-se que a imagem está presente em todas as áreas da Computação Gráfica, seja como produto final, como no caso da visualização, ou como parte essencial do processo de interação, no caso da Modelagem. Por este motivo, é de fundamental importância a perfeita compreensão do significado da imagem nos diversos contextos. Neste capítulo vai-se desenvolver uma conceitualização da imagem digital, apresentando-se modelos abstratos para uma imagem e diversas formas de representação. Aqui, mais uma vez, o paradigma dos quatro universos é bastante apropriado para obter-se um perfeito entendimento dos diversos modelos de imagem que vão-se estudar (fig. 4.1).

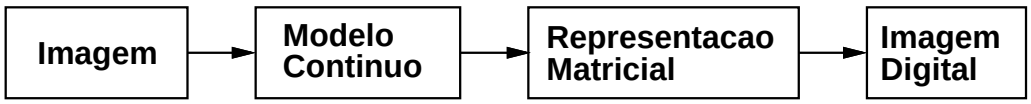



Figura 4.1: Modelo Conceitual.

## 4.1 Modelo de Imagem

Uma imagem é o resultado de estímulos luminosos associados a um suporte bidimensional que corresponde a superfície (curva) da retina do olho humano. Essa é a forma de percepção do universo físico no qual habitamos através da visão, um sentido que processa imagens, sejam elas o resultado de um processo intermediado, como por exemplo uma fotografia, ou o resultado de um processo de projeção do mundo tridimensional diretamente na retina.

Para representar e manipular imagens no computador, devem-se definir modelos matemáticos adequados a esses objetivos. Uma imagem pode ser definida como uma aplicação

$$\iota : U \subset \Re^2 \rightarrow \zeta \subset \Re^3$$

de um subconjunto do plano (um retângulo, na prática) em um espaço de cor. Com um espaço de cor de dimensão 1, o gráfico da função imagem é o conjunto dos pontos do $\mathbb{R}^3$ dado abaixo:

$$G(\iota) = \{(x, y, z);\ (x, y) \in U,\ z = \iota(x, y)\}$$

Esta definição, embora pareça um tanto abstrata, capta com precisão a noção de que uma imagem é definida pelas cores dos seus pontos. O espaço de cor é identificado com o $\Re^3$ se for usado um espaço perceptual de cor de dimensão 3. Assim, cada ponto da imagem tem a sua cor definida por uma tripla de números reais.

Uma fotografia é um exemplo de uma imagem contínua, no sentido de que assume valores em todos os pontos do domínio[1]. Um problema no armazenamento da imagem no computador é que ela possui um número infinito de pontos. Novamente, deve-se obter uma representação finita por um processo de discretização.

## 4.2 Discretização

O processo de discretização produz uma imagem discreta, a partir de um número finito de amostras colhidas da imagem contínua. Isto significa que a função $\iota$ é amostrada num subconjunto discreto $U' \subset U$:

$$\begin{aligned} U &= [a, b] \times [c, d] = \{(x, y) \in \Re^2;\ x \in [a, b],\ y \in [c, d]\}; \\ U' &= \{(x_i, y_j) \in U;\ x_i = i\Delta x,\ y_j = j\Delta y;\ \ i, j \in Z,\ \ \Delta x, \Delta y \in \Re.\} \end{aligned}$$

Para codificar uma imagem, em geral, discretiza-se, também, o espaço de cor, para que a informação de cor possa ser armazenada com um número finito de bits. A discretização do espaço de cor é chamada de quantização. A rigor, quando se trabalha com números em ponto flutuante, significa que já foi feita uma quantização, no entanto, ignora-se esse fato. Uma imagem digital é uma imagem discretizada no domínio espacial e no espaço de cor (fig. 4.2).

## 4.3 Representação Matricial

Uma imagem digital pode utilizar a representação matricial. Considerando que o domínio espacial de uma imagem é um retângulo, pode-se fixar um par de eixos ortogonais, paralelos aos lados do retângulo. Cada um desses eixos é particionado uniformemente em células de comprimento $\Delta x$ e $\Delta y$, respectivamente. Ao se fazer o produto cartesiano das células correspondentes, obtém-se um reticulado. Cada célula $(i, j)$ deste reticulado é chamada de pixel (fig. 4.3). A representação matricial utiliza uma matriz $A_{m \times n}$, onde $m$ é a resolução vertical e $n$ a resolução horizontal da imagem.

O produto $m \times n$ do número de pixels em cada dimensão da imagem é chamado de resolução espacial da imagem. A densidade de resolução é o número de pixels por unidade de comprimento e a resolução de cor é o número de bits usados para armazenar uma componente de cor do pixel. Com $m$ bits é possível codificar $2^{3m}$ cores.

[1] Isto não quer dizer que a função $\iota$ é contínua. Em geral, ela não é.

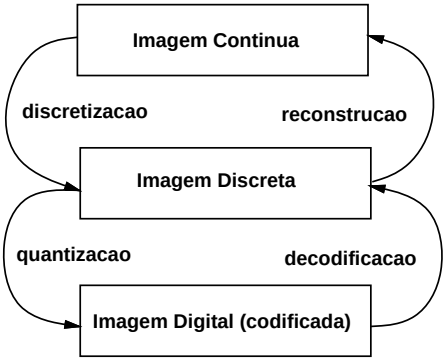



Figura 4.2: Imagem Digital.

## 4.4 Classificação para Imagem Digital

Uma imagem digital pode ser classificada de acordo com o seu espaço de cor. Chama-se gamute ao conjunto das cores do espaço quantizado de cor da imagem. Quanto à dimensão do espaço de cor, a imagem pode ser monocromática ou colorida.

Uma imagem monocromática possui um espaço de cor de dimensão 1 e pode ser classificada como:

- bitmap ou dois tons (gamute de duas cores): luminância máxima e luminância mínima;
- gray scale: mais de dois tons (intensidade variável).

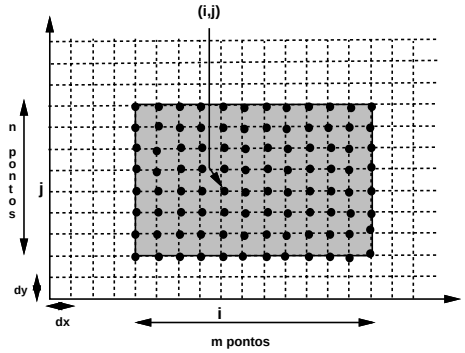



Figura 4.3: Modelo Matricial de Imagem.

Uma imagem colorida possui um espaço de cor com dimensão maior do que 1 e pode ser classificada como:

- pseudo-cor: o gamute do dispositivo é menor do que o número de pixels;
- cor real (*true color*): número de pixels é igual a $2^{3m}$ (resolução de cor de pelo menos 8 bits, para efeitos práticos).

## 4.5 Quantização

A quantização tem dois propósitos principais: permitir que a imagem digital seja exibida em um dispositivo com uma resolução de cor menor ou então servir como uma forma de compactação da imagem.

O processo de quantização costuma fazer surgir contornos de quantização, que marcam a transição de um nível para outro. Em imagens monocromáticas, o número mínimo de tons, para obter-se um gradiente de cor razoável, é 64 (6 bits). O ideal, para evitar o surgimento de contornos de quantização, são 256 (8 bits) tons. No entanto, existem aplicações importantes, por exemplo impressão em papel, onde é necessário fazer a quantização para 1 bit, ou duas cores – preto e branco.

Uma fotografia pode ser digitalizada por um scanner gerando uma representação matricial, na qual a cor de cada ponto pode ser armazenada em 8 bits, por exemplo. No processo de impressão desta imagem digital, uma impressora laser faz a quantização para 1 bit. É claro que a intenção não é obter uma imagem chapada em dois tons. Para que a imagem seja visualmente aceitável empregam-se os algoritmos de dithering.

- Fotografia→Scanner→Impressora Laser.
- Imagem→quantização→dithering→exibição.

Formalizando, a quantização para $m$ bits

$$q : S^1 \to S^2;\ S^2 \subset S^1,$$

$$\iota : U \to \zeta \Rightarrow \iota' : U \to \zeta';\ \iota'(x, y) = q(\iota(x, y))$$

é uma transformação sobrejetiva que para toda cor $c \in S^1$, representada com $n$ bits, associa $q(c) \in S^2$ representada por $m$ bits, $m < n$.

Logicamente, a quantização é um processo de partição do espaço de cor. Cada célula da partição é chamada de célula de quantização. Todas as cores contidas em uma célula de quantização são mapeadas em uma certa cor chamada de nível (valor) de quantização. Dada uma célula de quantização $\varrho_i$, com nível de quantização $c_i$, então para toda cor $c \in \varrho_i$, $d(c, c_i)$ é o seu erro de quantização (fig. 4.4).

Um algoritmo de quantização deve selecionar as células de quantização e, para cada célula, achar o nível de quantização correspondente. Matematicamente, é uma questão de minimização do erro de quantização. O problema é que deve ser utilizada uma métrica perceptual. Os métodos de quantização podem ser classificados em:

- métodos uniformes (as células de quantização são congruentes);

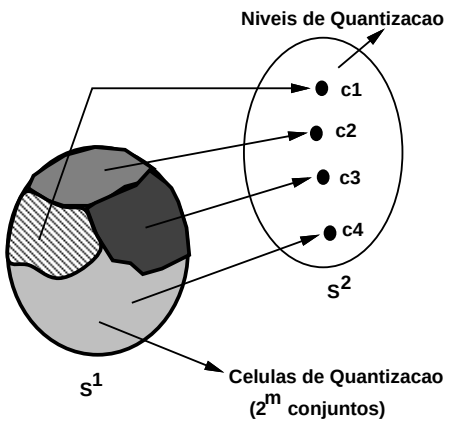



Figura 4.4: Células de Quantização.

- métodos adaptativos (levam em conta a distribuição das cores na imagem).

Neste texto, vão-se considerar dois algoritmos de quantização: o algoritmo por populosidade e o algoritmo do corte mediano.

### 4.5.1 O Algoritmo de Populosidade

O algoritmo por populosidade baseia-se no histograma de freqüência das cores na imagem. No eixo das abcissas estão as cores e no eixo das ordenadas é indicado quantos pixels possuem aquela cor (fig. 4.5). Dado o número de níveis de quantização, são escolhidas as cores mais freqüentes. As células de quantização são determinadas, então, a partir de alguma métrica. Se for utilizada a métrica Euclideana, as células podem ser determinadas com a construção do diagrama de Voronoi.

Uma observação importante é que o algoritmo de populosiade ignora completamente as cores com baixa populosidade.

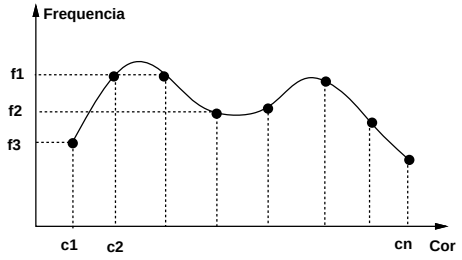



Figura 4.5: Histograma de Freqüência.

### 4.5.2 O Algoritmo do Corte Mediano

O algoritmo do corte mediano, criado por Paul Heckbert em 1982, determina, em primeiro lugar, as células de quantização. O objetivo do algoritmo é que cada célula de quantização possua mais ou menos o mesmo número de cores da imagem.

O primeiro passo é determinar o menor retângulo no espaço de cor (*bounding box*) que engloba todas as cores da imagem. Em seguida ele ordena as cores usando como chave a componente ($r, g$ ou $b$) que corresponde ao eixo coordenado paralelo à maior dimensão do retângulo. Por fim ele calcula a mediana deste conjunto ordenado levando em conta o histograma de freqüência das cores. Por definição a mediana de um conjunto ordenado $\varepsilon = \{\epsilon_1 \leq \epsilon_2 \leq ... \leq \epsilon_{n-1} \leq \epsilon_n\}$ é:

- $n$ impar: $\epsilon_{(\frac{n+1}{2})}$;
- $n$ par: $(\epsilon_{(\frac{n}{2})} + \epsilon_{(\frac{n}{2}+1)})/2$.

Feito isto, corta-se o retângulo por um plano que contém a mediana do conjunto de cores e é perpendicular ao eixo coordenado escolhido. Este processo é repetido então, recursivamente, para cada sub-retângulo resultante, até que se atinja o número de células de quantização desejado.

Os níveis de quantização podem ser escolhidos, por exemplo, tomando-se a média das cores de cada célula de quantização. O algoritmo do corte mediano é o melhor algoritmo de quantização para 8 bits conhecido. A implementação eficiente do algoritmo necessita de uma estrutura de dados espaciais adequada ao processo de subdivisão recursiva do espaço.

## 4.6 Dithering

Dithering é um processo de filtragem para minimizar a percepção dos contornos de quantização. Em certos casos, mesmo utilizando-se bons algoritmos de quantização fica difícil não perceber os contornos de quantização, por exemplo, na quantização para dois níveis.

Em geral, quando se faz uma quantização comete-se um erro. Os contornos de quantização surgem devido a forte correlação entre a cor de um pixel e a cor dos seus vizinhos. Geometricamente trata-se de uma curva conexa, o que acarreta que a passagem entre os diferentes níveis de quantização seja perceptível.

O objetivo de um algoritmo de dithering é descorrelacionar o erro de quantização. O erro torna-se crítico quando a quantização é para 1 bit apenas. Neste caso, existem somente duas células de quantização, que correspondem a dois níveis de quantização, conforme pode ser visto na figura 4.6.

### 4.6.1 Por que Funciona?

O sistema visual humano integra (soma) os estímulos luminosos recebidos dentro de um certo ângulo sólido. Assim é possível perceberem-se intensidades que não existem individualmente. Na realidade, o que é importante é o meio tom em uma região e não os tons dos pixels individualmente.

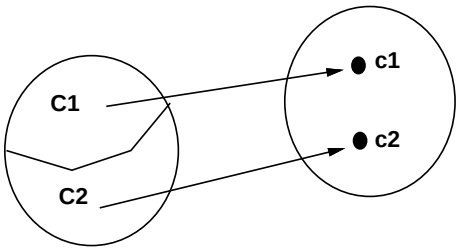



Figura 4.6: Quantização de 1 bit.

O ângulo de visão humano é de 150° na horizontal e 120° na vertical e a acuidade mínima é de $1/60° = 1'$. A percepção de detalhes depende então de três fatores:

- distância da imagem ao olho;
- densidade de resolução da imagem;
- abertura do olho.

### 4.6.2 Classificação dos Algoritmos

Os algoritmos de meio tom para indústria gráfica datam do início do século. O método utiliza um processo fotográfico tradicional. A imagem é refotografada com uma retícula sobreposta. Regiões de alta luminância (claras) geram pontos pequenos enquanto regiões de baixa luminância (escuras) geram pontos grandes que se superpõem.

Os algoritmos de dithering devem preservar as altas freqüências, que correspondem aos contornos das áreas de interesse, e substituir as baixas freqüências, que correspondem as texturas presentes no interior das áreas, por outras perceptualmente equivalentes.

A classificação dos algoritmos é feita utilizando padrões produzidos em áreas de intensidade constante. De acordo com a regularidade dos padrões, estes podem ser periódicos ou aperiódicos e de acordo com a estrutura podem estar aglomerados ou dispersos.

- periódicos: processos determinísticos que usam grades de amostras regulares.
- aperiódicos: minimizam o erro distribuindo-o globalmente.
- dispersos: criam os tons de cinza com pontos individuais uniformemente distribuídos (indicado para dispositivos com controle preciso, ex. monitores de vídeo).
- aglomerados: concentram os pontos em pequenos grupos de mesmo valor (indicado para dispositivos sem controle preciso, ex. impressoras).

### 4.6.3 Dithering com Modulação Aleatória

O algoritmo de dithering com modulação aleatória aplica uma pequena perturbação aleatória, uniformemente distribuída no intervalo das intensidades, ao valor de uma cor $c$ antes de quantizá-la:

$$q(c) = (c + random()) \in C_1 ?\ c_1 : c_2.$$

Este processo simples descorrelaciona a intensidade do pixel da intensidade dos pixels vizinhos. Este foi o primeiro algoritmo de dithering criado e é o mais simples. No entanto, ele introduz um ruído aleatório (pontos brancos) na imagem. Porém, este algoritmo pode ser efetivo se for utilizado junto com algoritmos de quantização para mais de 1 bit.

### 4.6.4 Dithering Periódico Ordenado

O algoritmo de dithering ordenado utiliza uma pertubação aleatória de média nula, dada sob a forma de uma matriz quadrada $D_{n\times n}$ de números pseudo-aleatórios:

$$Q(k,l) = (f(k,l) > D(i,j)); \quad i = k\%n, \quad j = l\%n.$$

Esta matriz funciona como um filtro[2] de passa baixa.

### 4.6.5 Dithering Ordenado com Aglomeração de Pontos

O dithering ordenado com aglomeração de pontos simula o método fotográfico tradicional. O primeiro passo é normalizar as intensidades da imagem para o intervalo dos níveis de quantização. Considere-se o padrão da figura 4.7. Neste exemplo a primeira coluna e linha estão repetidas e a matriz é $6 \times 6$, o que dá um total de 37 níveis (0 a 36). Assim $int_n = 36 * int/int_{max}$ e a intensidade média é 18.5.

Numa imagem com intensidade constante igual a 18.5, a entrada com o valor 18, por exemplo, gera branco na imagem filtrada porque $18.5 > 18$.

| | | | | | | |
|---|---|---|---|---|---|---|
| 35 | 30 | 18 | 22 | 31 | 36 | 35 |
| 29 | 15 | 10 | 17 | 21 | 32 | 29 |
| 14 | 9 | 5 | 6 | 16 | 20 | 14 |
| 13 | 4 | 1 | 2 | 11 | 19 | 13 |
| 28 | 8 | 3 | 7 | 24 | 25 | 28 |
| 34 | 27 | 12 | 23 | 26 | 33 | 34 |
| 35 | 30 | 18 | 22 | 31 | 36 | 35 |

Figura 4.7: Matriz de Dithering Ordenado por Aglomeração.

[2]Um operador unário que recebe um sinal na entrada e produz um outro sinal na saída.

Após processar a imagem pelo filtro acima, a imagem resultante apresenta a repetição do padrão da figura 4.7:

- intensidade constante acima da média (+clara) : região clara (branco) é aumentada.
- intensidade constante abaixo da média (+escura): região clara (branco) é diminuída.
- intensidade variável: área do aglomerado varia.

### 4.6.6 Dithering Ordenado com Dispersão Pontual

O algoritmo de dithering ordenado com dispersão reduz a resolução espacial da imagem para aumentar artificialmente a sua resolução de cor. A imagem é particionada em células quadradas congruentes. As células são percorridas ordenadamente e é tirada uma média das intensidades de cada célula de pixels. Esta média endereça uma tabela de matrizes (cada matriz possui a mesma dimensão da célula). Se o i-ésimo elemento da matriz vale 1, o pixel correspondente da imagem reduzida é aceso, caso contrário ele é apagado. Cada uma dessas matrizes possui um arranjo de 0's e 1's que faz com que o sistema visual humano integre as intensidades da célula, de forma que é percebida uma única intensidade. Bayer criou uma família de matrizes que podem ser geradas iterativamente dada a sua dimensão.

A figura 4.8 mostra a matriz de Bayer de ordem 2. Neste caso há 5 níveis de intensidade. Após a normalização das intensidades da imagem para o intervalo $0 - 5$, uma intensidade de 2.4, por exemplo, é mapeada para a matriz de índice 2.

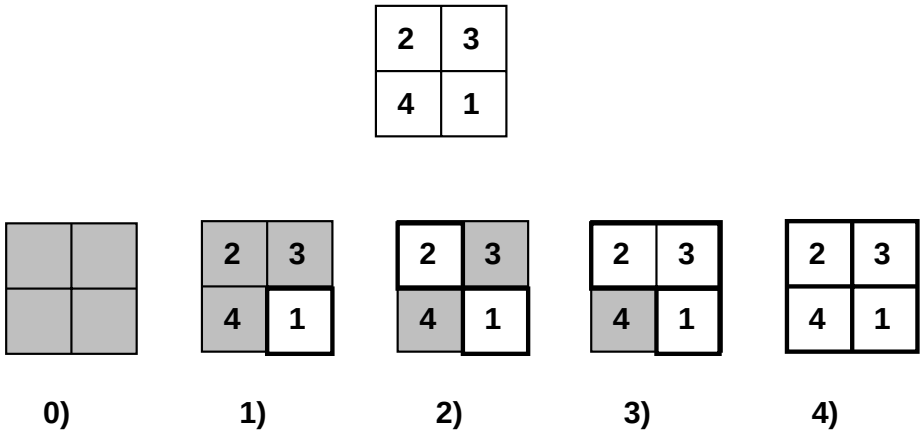



Figura 4.8: Distribuição das Intensidades com Dithering de Bayer de Ordem 2.

Para a reprodução de imagens com alta qualidade, a dimensão da matriz de dithering (com aglomeração ou dispersão, tanto faz) deve estar entre 8 e 10, ou seja, gerando entre 65 e 101 níveis respectivamente. A densidade de células de dithering é chamada de freqüência de tela. Valores de freqüência de tela entre 120 e 150 linhas (células) por polegada produzem

bons resultados. Isto significa que o dispositivo de exibição deve ter uma resolução entre $150 \times 8 = 1200$ dpi[3] e $150 \times 10 = 1500$ dpi.

### 4.6.7 Algoritmo de Dithering Aperiódico

O algoritmo de Floyd-Steinberg procura dispersar o erro de quantização, ocorrido em um pixel, para os pixels vizinhos (fig. 4.9).

```
Q(i,j)      = (f(i,j)>0.5);
erro        = Q(i,j) - f(i,j);
f(i+1,j)   += erro*3/8;
f(i,j+1)   += erro*3/8;
f(i+1,j+1) += erro*1/4;
```

Como conseqüência do modo com que o erro é propagado, surge um contorno de quantização que se propaga na direção da diagonal da imagem.

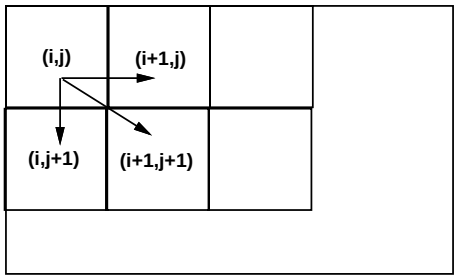



Figura 4.9: Algoritmo de Floyd-Steinberg.

Em geral, os algoritmos de dithering eliminam as altas freqüências presentes na imagem, como conseqüência do descorrelacionamento do erro. Este problema pode ser atenuado, usando-se um filtro de passa alta, para realçar as altas freqüências, antes de empregar-se o algoritmo de dithering[4].

## 4.7 Codificação de Imagem

A codificação é o terceiro nível de abstração de uma imagem, conforme o modelo conceitual apresentado no início deste capítulo (fig. 4.1). Na codificação a representação discreta da imagem é quantizada e a imagem digital resultante é transformada em um conjunto de símbolos organizados segundo uma estrutura de dados. A codificação pode ser feita utilizando um número de bits constante para codificar cada parte da imagem, sendo chamada neste caso de codificação uniforme, ou pode se valer da probabiliade de ocorrência dos diferentes

[3] Dots per inch.

[4] A tradução de dithering seria exitação.

níveis de quantização e utilizar um número de bits variável de acordo com a distribuição de probalidade de ocorrência destes níveis. Neste caso, a codificação é dita adaptativa.

Para se conseguir uma redução no espaço necessário ao armazenamento de uma imagem, costumam-se utilizar métodos de compressão de imagens. Isto é possível porque as imagens tendem a apresentar um alto grau de coerência, que se traduz em uma redundância de informação quando codificada. Existem métodos de compressão reversíveis e irreversíveis, dependendo da possibilidade de recuperação exata da imagem ou não (com ou sem perda de informação). Quando utilizar um ou outro método irá depender da natureza da aplicação, que pode ser objetiva ou subjetiva.

Os métodos de compressão podem ser classificados em métodos espaciais, por transformada ou por modelo. Os métodos espaciais utilizam diretamente a representação espacial da imagem para fazer a compressão, por exemplo, o método conhecido como *Run-length Encoding*.

Nos métodos por transformada, a compressão é feita com base em uma representação não espacial da imagem (por exemplo, representação espectral). Como exemplos citam-se os métodos baseados em transformadas de Fouriér, Cosseno ou de Hadamard.

Nos métodos de compressão por modelo utiliza-se um modelo da imagem para obter-se a compressão. Como exemplo, cita-se a compressão por fractais.

Existem, na prática, diversos padrões (a maioria padrões de fato) para armazenamento de imagens. Cada padrão costuma utilizar algum método de compressão, alguns suportam true-color, outros são mais adequados ao armazenamento de seqüências animadas, etc. Os mais famosos são os padrões GIF (Graphics Interface Format), TIFF (Tagged Image File Format), JPEG (Joint Photograph Expert Group), MPEG (Motion Photograph Expert Group), SUN-raster file, Encapsulated PostScript.

## 4.8 Exercícios

**4.1** *Defina e explique a correção gama.*

**4.2** *Defina quantização de uma imagem.*

**4.3** *Discuta o problema de um sistema de cor para armazenamento de imagens.*

**4.4** *Considere o algoritmo de populosidade para quantização de cor.*

*a) Qual o procedimento de ordenação mais adequado ao algoritmo?*

*b) Em que condições o algoritmo apresenta resultados insatisfatórios?*

*c) Escreva um pseudo-código para o algoritmo.*

*d) Faça uma análise da complexidade do algoritmo.*

**4.5** *O método de quantização por subdivisão do espaço obtém as células de quantização pelo particionamento recursivo do volume de cor. Em cada iteração, este é dividido em dois sub-volumes por um hiperplano ortogonal a uma das direções principais do espaço.*

*a) Qual o critério utilizado pelo algoritmo do corte mediano para escolher a direção e o ponto de subdivisão?*

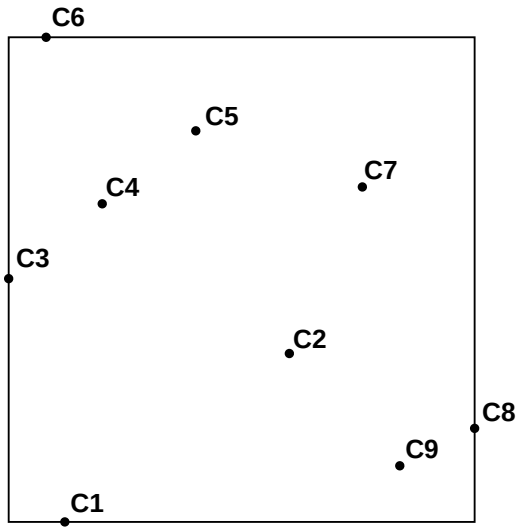
*b) Qual a razão desta escolha?*

*c) Sugira um outro critério de subdivisão.*

**4.6** *Considere o conjunto de cores, e a tabela de freqüência dessas cores numa imagem digital, mostrado na figura 4.10. Obtenha a quantização da imagem para quatro níveis.*

*a) Determine a função de quantização para o algoritmo de populosidade.*

*b) Esboce as células de quantização pelo algoritmo do corte mediano.*




| Cor | Freq |
|---|---|
| C1 | 2 |
| C2 | 3 |
| C3 | 2 |
| C4 | 1 |
| C5 | 2 |
| C6 | 1 |
| C7 | 1 |
| C8 | 1 |
| C9 | 2 |

Figura 4.10: Conjunto de Cores e Tabela de Freqüência.

**4.7** *Considere-se a imagem de uma esfera metálica com a reflexão especular de uma fonte de luz pontual ("highlight").*

*a) O que pode ocorrer com esse highlight se for feita uma quantização da imagem por populosidade.*

*b) Que modificação pode ser feita no algoritmo de populosidade para minimizar o problema levantado no item anterior?*

**4.8** *Como transformar uma imagem digital colorida em uma imagem em preto e branco (ou seja, "descolorizar" a imagem)?*

**4.9** *Como pode ser definido o mapa de cor de uma look-up table de modo a mostrar o negativo de uma imagem em gray-scale?*

**4.10** *Considere-se um dispositivo raster com 4 bits por pixel e uma LUT(look-up table) com 12 bits por entrada (4 bits para R, G e B). Suponha-se que os quatro planos de memória são particionados para armazenar duas imagens: imagem A nos dois planos de mais alta ordem e imagem B nos dois planos de mais baixa ordem. Sabendo-se que as atribuições de cor para os 2 bits dos valores dos pixels de cada imagem são:* $00 = R,\ 01 = G,\ 10 = B$ *e* $11 = W$.

*a) Mostre como carregar a LUT para que somente a imagem A seja exibida.*

*b) Mostre como caregar a LUT para que somente a imagem B seja exibida.*

**4.11** *Dadas uma LUT com* 16 *entradas* $\times$ 12 *bits e imagens com 2 bits por pixel, mostre como a tabela deve ser carregada para produzir uma fusão entre duas imagens A e B, correspondendo a expressão,* $A * t + B * (1 - t)$*, para valores de* $t = 0.0;\ 0.25;\ 0.5;\ 1.0$. *As atribuições de cor são as mesmas do item anterior.*

**4.12** *Um dos problemas de uma imagem digital é o erro de quantização.*

*a) Defina "dithering" e explique o seu objetivo frente ao problema apontado.*

*b) Escreva o pseudo-código do algoritmo de dithering de floyd-steinberg.*

*c) Escreva o pseudo-código de um algoritmo de dithering ordenado.*

*d) Que tipos de dispositivo são adequados ao uso de um algoritmo de dithering com dispersão pontual?*

*e) Que tipos de dispositivo são adequados ao uso de um algoritmo de dithering com aglomeração de pontos?*

**4.13** *Considere-se o algoritmo de run-length encoding que codifica imagens usando o seguinte formato:*

$$m, n, L_1 I_1, L_2 I_2, ..., L_k I_k,$$

*onde m e n são números inteiros de 32 bits e especificam a resolução da imagem; os pares* $(L_j I_j), j = 1, ..., k$ *são números de 8 bits e indicam respectivamente o comprimento e a intensidade do* $j-$*ésimo bloco de elementos da imagem com valor de intensidade constante igual a I.*

*a) Que tipo de imagem produz a menor codificação?*

*b) Que tipo de imagem produz a maior codificação?*

*c) Calcule (em bytes) o espaço necessário para armazenar imagens de* $640 \times 480$ *pixels, correspondendo aos items anteriores, comprimidas pelo algoritmo.*

# Capítulo 5

# Geometria

As transformações Geométricas ocupam uma posição de destaque em Computação Gráfica. Elas permitem o reposicionamento de objetos no espaço, favorecendo o agrupamento desses objetos e a geração de famílias de modelos a partir de um modelo básico. Os objetos geométricos representados por estes modelos podem ser transformados aplicando-se uma transformação geométrica a um número finito de pontos, por exemplo, nos vértices de curvas poligonais ou de polígonos de controle de curvas algébricas por partes.

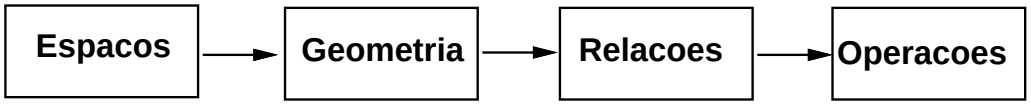



Figura 5.1: Modelo Conceitual.

As transformações geométricas que podem ser aplicadas a um objeto são induzidas pelo tipo de geometria adotado e a pergunta a ser respondida é: qual o tipo de geometria mais adequado às necessidades da Computação Gráfica?

## 5.1 Geometria Euclideana

Uma geometria pode ser definida de forma sintética, a partir de axiomas e teoremas, ou por coordenadas, como é feito em Álgebra Linear.

No caso da Geometria Euclideana, por exemplo, pode-se construí-la a partir de um espaço vetorial ($\Re^3$) munido de produto interno: $< x, y >= \sum_{i=1}^{3} x_i y_i$; os comprimentos são dados pela norma Euclideana $\|x\| = \sqrt{< x, x >}$.

Será que a Geometria Euclideana — com o seu postulado básico que diz que dado um ponto $p$ e uma reta $r$ existe uma única reta paralela a $r$ que passa pelo ponto $p$ — é a mais adequada? Se não, como negá-la? Basicamente existem duas formas: dizer que existe uma infinidade de retas paralelas (Geometria Hiperbólica) ou dizer que não existe paralelismo (Geometria Projetiva).

Neste capítulo vai ser visto que, embora seja natural definir os objetos no espaço Euclideano, a geometria projetiva é a mais adequada para a computação gráfica, pois estende a geometria Euclideana com uma série de vantagens.

## 5.2 Transformações Lineares

Neste texto, entenda-se por transformação $n$-dimensional uma aplicação invertível $T : U \subset \Re^n \rightarrow \Re^n$. Os modelos poligonais e poliedrais são comuns em Computação Gráfica, pois podem ser exibidos em qualquer dispositivo gráfico e, assim, uma classe importante de transformações é formada por aquelas que preservam estruturas lineares. Estas transformações são conhecidas como transformações lineares em $\Re^n$, ou operadores lineares, e se caracterizam por transformarem retas em retas e a origem na própria origem.
**Definição:** Um operador linear é uma transformação $T$ que $\forall x, y \in \Re^n, \lambda \in \Re$:

- $T(x + y) = T(x) + T(y)$,
- $T(\lambda x) = \lambda T(x)$.

Da álgebra linear, sabe-se que o conjunto de todos os operadores lineares em $\Re^n$ forma um espaço vetorial de dimensão $n^2$. Um resultado importante é que existe um isomorfismo entre a álgebra dos operadores lineares em $\Re^n$, determinado por uma base, sobre a álgebra das matrizes quadradas $n \times n$. Note-se que qualquer matriz quadrada $n \times n$, $A$, sobre $\Re$ define um operador linear em $\Re^n$, pela transformação $v \rightarrow Av$ (onde $v$ é escrito como um vetor coluna). É possível mostrar que a representação matricial deste operador é, justamente, a matriz $A$, se for usada a base canônica do $\Re^n$.

A representação matricial de um operador linear é bastante adequada do ponto de vista computacional, pois permite que a composição de diversas transformações seja alcançado através do produto de matrizes.

### 5.2.1 Transformações Lineares Bi-dimensionais

Vai-se iniciar o estudo das transformações lineares bi-dimensionais considerando o seu efeito sobre os pontos do plano. Como foi visto anteriormente, um operador linear bi-dimensional pode ser representado por uma matriz $2 \times 2$. O resultado da multiplicação de um vetor coluna $(x, y)^T$, que contém as coordenadas de um ponto, por uma matriz $2 \times 2$ genérica pode ser escrito como:

$$T(X) = \begin{pmatrix} a & c \\ b & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} ax + cy \\ bx + dy \end{pmatrix} = \begin{pmatrix} x^* \\ y^* \end{pmatrix}.$$

Atribuindo valores aos elementos $a, b, c, d$, pode-se obter uma relação entre o ponto transformado e o ponto original, conforme mostrado abaixo.

$a, d > 0, b = c = 0 \quad \rightarrow x^* = ax, y^* = dy \quad$ – escalamento das componentes $x$ e $y$.
$a = -1, d = 1, b = c = 0 \rightarrow x^* = -x, y^* = y \quad$ – reflexão em relação ao eixo $y$.
$a = 1, d = -1, b = c = 0 \rightarrow x^* = x, y^* = -y \quad$ – reflexão em relação ao eixo $x$.
$a = d = -1, b = c = 0 \quad \rightarrow x^* = -x, y^* = -y \quad$ – reflexão em relação à origem.

$a = d = 0, b = c = -1 \quad \rightarrow x^* = -y, y^* = -x$ – reflexão em relação à reta $y = -x$.
$a = d = 0, b = c = 1 \quad \rightarrow x^* = y, y^* = x$ – reflexão em relação à reta $y = x$.
$a = d = 1, b = 0 \quad \rightarrow x^* = x + cy, y^* = y$ – cisalhamento na direção $x$.
$a = d = 1, c = 0 \quad \rightarrow x^* = x, y^* = bx + y$ – cisalhamento na direção $y$.

Um objeto geométrico qualquer pode ser transformado por um operador linear, aplicando-se a matriz de transformação a cada um de seus pontos. A área do objeto transformado é dada pelo produto do determinante da matriz de transformação pela área do objeto original.

Uma mudança de variável no $\Re^n$ é dada por $T : \Re^n \rightarrow \Re^n$, onde $T$ é um operador continuamente diferenciável. Sendo $R \subset \Re^n$ com fronteira constituída por um número finito de conjuntos diferenciáveis e supondo-se que $R$ e sua fronteira estão contidos no domínio de $T$, e que:

- $T$ é injetiva em $R$;
- det $T'$, o determinante jacobiano de $T$, é diferente de zero em $R$;

então, se a função $f$ é limitada e contínua em $T(R)$ (a imagem de $R$ por $T$), tem-se:

$$\int_{T(R)} f dv = \int_R (f.T) \left|\det T'\right| dv, \qquad T' = \begin{pmatrix} \frac{\partial x}{\partial u}(u,v) & \frac{\partial x}{\partial v}(u,v) \\ \frac{\partial y}{\partial u}(u,v) & \frac{\partial y}{\partial v}(u,v) \end{pmatrix}.$$

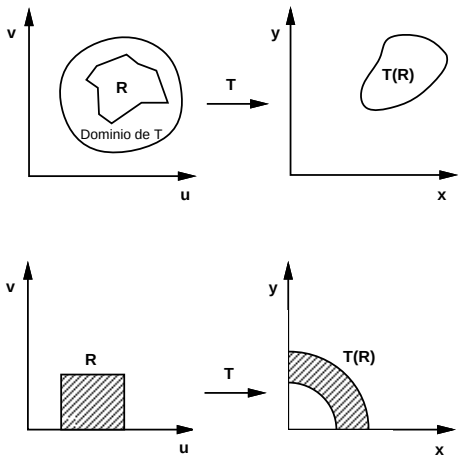



Figura 5.2: Mudança de Variável.

Como exemplo, tem-se a conhecida mudança para coordenadas polares, dada pela transformação abaixo (fig. 5.2):

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x(u,v) \\ y(u,v) \end{pmatrix} = \begin{pmatrix} ucos(v) \\ usin(v) \end{pmatrix}, \; J = \begin{pmatrix} cos(v) & -usin(v) \\ sin(v) & ucos(v) \end{pmatrix}, \; \det J = u.$$

Se $T$ é uma transformação linear de $\Re^n$ em $\Re^n$ com matriz $A$, então $T$ multiplica volumes pelo fator $|\det A|$ :

$$V(T(R)) = \int_{T(R)} dv = \int_R |J|\, dv = |J|\, V(R).$$

Afortunadamente, a transformação de segmentos de reta pode ser obtida apenas pela transformação das suas extremidades[1], existindo uma correspondência um a um entre os pontos do segmento transformado e os pontos do segmento original:

$$T[\overline{p_1 p_2}] = T[(1-t)\overline{p_1} + t\overline{p_2}] = (1-t)T[\overline{p_1}] + tT[\overline{p_2}].$$

Um operador linear aplicado a segmentos de reta paralelos sempre produz segmentos paralelos. Pode ser mostrado que se dois segmentos $\overline{AB}$ e $\overline{EF}$ são paralelos com inclinação $m$ então os segmentos transformados possuem inclinação $m^* = (b + dm)/(a + cm)$. Isto significa que paralelogramos são transformados em paralelogramos.

Quando um operador linear é aplicado a um par de segmentos que se interceptam, o resultado é um par de segmentos que também se interceptam. Além disso, o ponto de interseção dos segmentos originais é transformado no ponto de interseção dos segmentos transformados. Entretanto, dois segmentos perpendiculares que se interceptam podem ser transformados em dois segmentos que se interceptam e não são perpendiculares ou vice-versa, ou seja, os ângulos, em geral, não são preservados. Como vai ser visto adiante, os ângulos são preservados se a matriz de transformação for ortonormal.

**Definição:** Uma matriz $A$, $n \times n$, é ortonormal quando os seus vetores linha (ou coluna) formam uma base ortonormal, ou seja:

$$a_i.a_j = 0 \text{ se } i \neq j \text{ e } a_i.a_i = 1,\ \forall i, j \in [1..n].$$

Neste caso, $AA^T = I$, uma vez que o elemento $(i, j)$ é obtido pelo produto da $i$-ésima linha de $A$ com a $j$-ésima coluna de $A^T$, justamente $a_i$ e $a_j$. Segue que $A^T = A^{-1}$ também é uma matriz ortonormal.

Além dos efeitos de escalamento, reflexão e cisalhamento, o efeito de rotação de um objeto, por um ângulo $\theta$ ao redor da origem, também pode ser obtido por um operador linear representado por uma matriz da forma:

$$R = \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix}.$$

Claramente, $R$ é uma matriz ortonormal.

## 5.3 Transformações Rígidas

Do estudo dos operadores lineares foi visto que, em geral, um objeto pode ser deformado pela aplicação de uma transformação genérica. Para evitar deformações é necessário que seja aplicado apenas um tipo particular de transformação, que preserve os ângulos entre os segmentos de reta que se interceptam e o comprimento destes segmentos. Utilizando a noção de produto escalar e vetorial, pode-se mostrar que esta condição é satisfeita quando os elementos da matriz de transformação satisfazem às seguintes restrições:

[1] Combinações afim de pontos são invariantes sob transformações afim.

- $a^2 + b^2 = 1, \quad c^2 + d^2 = 1,$
- $ac + bd = 0, \quad ad - bc = 1.$

Isto significa que a matriz de transformação deve ser ortonormal. Os comprimentos e os ângulos entre segmentos que se interceptam são preservados (ou seja, o produto escalar ou interno) por transformações de rotação pura. As matrizes de reflexão pura possuem determinante = -1 e, embora o comprimento dos segmentos seja preservado, tecnicamente, o ângulo entre os segmentos transformados é $2\pi - \theta$. Rotações e reflexões pura são chamadas de transformações rígidas, que são as isometrias de um espaço Euclideano[2]. Deve-se notar que os escalamentos uniformes também preservam os ângulos, mas não a magnitude dos vetores transformados.

Os movimentos rígidos do espaço, juntamente com as transformações lineares de semelhança[3], constituem o grupo de transformações da Geometria Euclideana. Os conceitos estudados nessa geometria são os de congruência e semelhança. O conjunto das transformações lineares de um espaço vetorial, juntamente com a translação, constituem o grupo das transformações da Geometria Afim do espaço, que estuda as razões e proporções entre objetos geométricos.

## 5.4 Geometria Projetiva

Como foi dito anteriormente, a Geometria Projetiva[4] é a que se adequa melhor à Computação Gráfica. Intuitivamente, isto pode ser constatado imaginando-se uma fotografia de uma estrada sem curvas. Claramente, as linhas que delimitam a estrada se encontram num ponto distante, na linha do horizonte, conforme pode ser visto na figura 5.3. A transformação fotografia não pode ser obtida a partir da Geometria Afim (translações mais transformações lineares), uma vez que, neste tipo de geometria, retas paralelas são transformadas em retas paralelas. Como esta aplicação é importante no contexto da Computação Gráfica, é-se levado na direção da Geometria Projetiva, que oferece a solução por meio da transformação projetiva chamada de perspectiva.

O modelo sintético da Geometria Projetiva foi introduzido por Pascal, que criou a noção de ponto no infinito ou ponto ideal. O conjunto de todos os pontos no infinito forma uma reta chamada de reta ideal. Aqui, vai-se propor um modelo para a Geometria Projetiva, para depois adicionarem-se as coordenadas (tanto reais como ideais).

## 5.5 Transformações Projetivas

Um número muito grande de transformações pode ser obtido com os operadores lineares bidimensionais, por exemplo, rotação, escalamento, reflexão, cisalhamento, etc. Entretanto, a origem do sistema de coordenadas não é alterada por nenhuma destas transformações,

[2] Um espaço vetorial de dimensão finita, munido de produto interno.

[3] Da forma $\lambda I, \lambda \in \Re$ e $I$ é a matriz identidade.

[4] Introduzida em Computação Gráfica por Roberts em 1966.

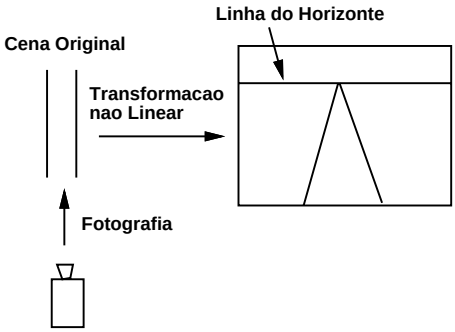



Figura 5.3: Transformação Projetiva.

impossibilitanto que o efeito de translação seja representado por um operador linear bi-dimensional. Além disso, um paralelogramo não pode ser transformado num quadrilátero arbitrário por um operador linear. Estes problemas podem ser tratados, de forma unificada, com a introdução de coordenadas homogêneas e o uso da Geometria Projetiva.
**Definição:** O plano projetivo $RP^2$ é conjunto das retas do $\Re^3$ que passam pela origem, a menos da própria origem. Um ponto no plano projetivo é um conjunto

$$P = \{\lambda(x, y, z);\ \lambda \neq 0,\ (x, y, z) \neq (0, 0, 0)\}.$$

Este ponto é denotado por $P = [x, y, z]$ em coordenadas homogêneas.

Considerando o plano $z = 1$ como sendo o plano afim Euclideano mergulhado no plano projetivo, então, qualquer ponto

$$P = [x, y, z] \in RP^2,\ z \neq 0 \text{ pode ser escrito como: } P = \left[\frac{x}{z}, \frac{y}{z}, 1\right],$$

que representa a interseção da reta $\lambda(x, y, z)$ com o plano $z = 1$ ($\lambda = 1/z$). Os pontos $[x, y, 0]$ do plano projetivo não possuem representantes no plano $z = 1$. Tem-se, desta forma, uma partição natural do plano projetivo em dois conjuntos:

$$RP^2 = \{[x, y, 1]\} \cup \{[x, y, 0]\}.$$

Os pontos $[x, y, 0]$ são os pontos ideais do plano projetivo. É interessante verificar que, nesse modelo, duas retas paralelas $l$ e $m$ no plano afim se interceptam em um ponto ideal, uma vez que a interseção de dois planos, que passam pela origem, contendo $l$ e $m$, é uma reta no plano $z = 0$ (um ponto ideal), conforme pode ser visto na figura 5.4.

Uma reta no plano projetivo é o conjunto dos pontos $[x, y, z]$ que satisfazem a uma equação linear $ax + by + cz = 0$. Note-se que, se a equação é satisfeita por um ponto $(x_0, y_0, z_0) \neq (0, 0, 0)$, então, ela também é satisfeita por qualquer ponto com coordenadas homogêneas $\lambda(x_0, y_0, z_0)$. Isto significa que o ponto $[x_0, y_0, z_0]$ do plano projetivo pertence à reta. No modelo afim do plano projetivo, a equação da reta projetiva representa um plano em $\Re^3$ passando pela origem e, portanto, se esse plano contém o ponto $(x_0, y_0, z_0)$, então, ele também contém a reta que passa por esse ponto e pela origem.

Indica-se por $\pi$ a aplicação que associa a cada ponto $v = (x, y, z)$ no espaço afim $\Re^3$, o ponto $[x, y, z]$ do plano projetivo (isto é, $\pi(x, y, z) = [x, y, z]$ é a reta do espaço Euclideano definida pelo vetor $v$). Essa aplicação é chamada de aplicação quociente.

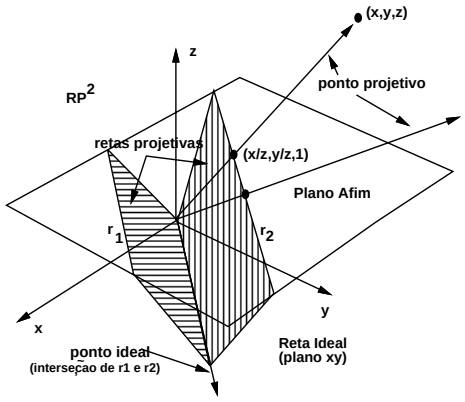



Figura 5.4: Plano Projetivo.

Se $T$ é uma operador linear invertível do $\Re^3$, então $T$ transforma retas em retas e deixa a origem $(0, 0, 0)$ fixa. Desse modo, $T$ define naturalmente uma transformação no plano projetivo. Essa transformação é chamada de transformação projetiva induzida por $T$ e será indicada por $\overline{T}$. Se $[x, y, z]$ são as coordenadas homogêneas do ponto $P$ e $M$ é a matriz $3 \times 3$ do operador $T$, então: $\overline{T}(P) = M(x, y, z)^T$. Diz-se que $M$ é a matriz da transformação projetiva $\overline{T}$. Utilizando coordenadas homogêneas, pode-se representar uma transformação afim bi-dimenional ($A(u) = Tu + v$) por uma matriz $3 \times 3$.

A matriz projetiva $M$, $3 \times 3$, genérica pode ser dividida, para efeito de estudo, em quatro partes:

$$M = \left( \begin{array}{cc|c} a & c & m \\ b & d & n \\ \hline p & q & s \end{array} \right).$$

Colocando $a = d = s = 1$ e $b = c = p = q = 0$, obtém-se a matriz de translação pura:

$$\begin{pmatrix} 1 & 0 & m \\ 0 & 1 & n \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x+m \\ y+n \\ 1 \end{pmatrix}.$$

Com $s = 1$ e $m = n = p = q = 0$, obtém-se escala, rotação, reflexão e cisalhamento, produzindo os mesmos efeitos dos operadores lineares bi-dimensionais:

$$\begin{pmatrix} a & c & 0 \\ b & d & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} ax+cy \\ bx+dy \\ 1 \end{pmatrix}.$$

Em ambos os casos, vê-se que pontos do plano afim são levados em pontos do plano afim e que pontos ideais são levados em pontos ideais.

Para mostrar o efeito de $p, q \neq 0$ na terceira linha da matriz, considere-se o seguinte:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} \lambda x \\ \lambda y \\ \lambda z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ p & q & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ px + qy + 1 \end{pmatrix}.$$

Agora, um ponto do plano afim, expresso em coordenadas homogêneas, é levado para um ponto do espaço tri-dimensional definido por $z = px + qy + 1$. Aplicando esta mesma transformação a um ponto ideal obtém-se:

$$M(x, y, 0)^T = (x, y, px + qy)^T.$$

Observando as equações acima, vê-se que não ocorre o efeito de invariância dos casos anteriores: pontos ideais podem ser transformados em pontos do plano afim e vice-versa. Geometricamente, se um ponto ideal é transformado em um ponto $P_0$ do plano afim, então a família de retas paralelas, que se interceptam nesse ponto ideal, são transformadas em uma família de retas incidentes no ponto $P_0$ (fig. 5.5). O ponto $P_0$ é chamado de ponto de fuga da transformação. Um ponto de fuga correspondendo a uma direção paralela a um dos eixos coordenados é chamado ponto de fuga principal. Como no plano afim existem no máximo duas direções ortogonais, podem-se ter transformações projetivas com no máximo dois pontos de fuga principais. Cada um desses pontos de fuga é a imagem do ponto ideal que corresponde a estas direções: $[x, 0, 0]$ e $[0, y, 0]$. A existência desses pontos de fuga é controlada pelos elementos $p$ e $q$, correspondendo às direções $x$ e $y$, respectivamente.

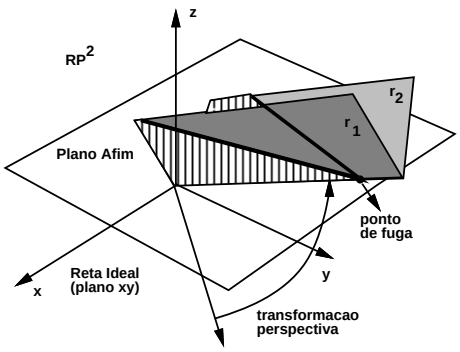



Figura 5.5: Ponto Ideal Transformado em Ponto Real.

Entretanto, interessa obter o resultado desta transformação nos pontos do plano afim. Isto é feito considerando a interseção com o plano $z = 1$:

$$[x, y, z] = \left[\frac{x}{z}, \frac{y}{z}, 1\right], \; z = px + qy + 1.$$

A interseção só existe se $z \neq 0$.

O efeito do elemento $s$ corresponde a um escalamento global. De fato,

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} \lambda x \\ \lambda y \\ \lambda z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & s \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ s \end{pmatrix}.$$

Aqui ocorre uma transformação que não altera os pontos ideais, mas os pontos do plano afim não ficam invariantes. O ponto do plano afim que representa o ponto transformado é $(x/s, y/s, 1)$. Neste caso, tem-se uma expansão ou compressão uniforme de todos os pontos do plano afim.

## 5.6 Composição de Transformações Projetivas bi-dimensionais

Uma composição de transformações é obtida pré-multiplicando-se as matrizes que representam os operadores correspondentes. Como a multiplicação de matrizes não é uma operação comutativa, conclui-se que a ordem em que os operadores são aplicados influi no resultado final.

Até agora, foram consideradas apenas rotações ao redor da origem. Coordenadas homogêneas proveêm uma maneira de obter uma rotação, em torno de um ponto qualquer, pela composição de transformações. Em geral, uma rotação deste tipo pode ser obtida em três passos:

- Transladando o objeto e o centro da rotação para a origem.
- Executando a rotação desejada.
- Transladando o resultado de volta ao centro de rotação original.
- $[R_p] = [T^{-1}][R_o][T]$

Da mesma forma, uma reflexão em relação a uma linha arbitrária é obtida em cinco passos, a partir da composição das transformações correspondentes:

- Transladando a linha e o objeto de maneira a que a linha passe pela origem.
- Rotacionando o objeto e a linha ao redor da origem até que a linha coincida com um dos eixos coordenados.
- Refletindo em relação ao eixo coordenado escolhido.
- Aplicando a rotação inversa ao redor da origem.
- Transladando de volta para a localização original.
- $[Rflt_l] = [T^{-1}][R^{-1}][Rflt_x][R][T]$

## 5.7 Transformações tri-dimensionais

A capacidade de representar e exibir um objeto tri-dimensional é fundamental para perceber a forma deste objeto. Além disso, a capacidade de rodar, transladar e projetar vistas do objeto é, em muitos casos, fundamental a esta percepção.

Baseado na experiência prévia do caso bi-dimensional, vai-se introduzir de imediato o espaço projetivo. De um ponto de vista sintético, a cada família de planos paralelos em $\Re^3$ associa-se uma reta ideal, que é a interseção dessa família. Note-se que a cada família de retas paralelas em cada plano está associado um ponto ideal nessa reta ideal.

O modelo analítico do espaço projetivo pode ser introduzido de modo semelhante ao modelo do plano projetivo. Considere-se o espaço $\Re^4 = \{(x_1, x_2, x_3, x_4);\ x_i \in \Re\}$ e nele mergulhe-se o espaço $\Re^3$ como sendo o hiperplano $x_4 = 1$. O espaço $RP^3$ é o conjunto das retas

$$\{\lambda(x, y, z, w);\ \lambda \neq 0,\ (x, y, z, w) \neq (0, 0, 0, 0)\} \text{ de } \Re^4, \text{ excluída a origem.}$$

A cada ponto $P$ estão associadas as suas coordenadas homogêneas, $P = [x, y, z, w]$, e o espaço projetivo possui, portanto, uma decomposição natural:

$$RP^3 = \{[x, y, z, 1]\} \cup \{[x, y, z, 0]\},$$

que corresponde exatamente aos pontos do espaço afim e aos pontos ideais no infinito.

As transformações projetivas são introduzidas de maneira análoga. Dado um operador linear invertível $T : \Re^4 \to \Re^4$, sabe-se que $T$ transforma retas em retas e deixa a origem fixa. Assim, $T$ define uma transformação $\overline{T} : RP^3 \to RP^3$. Se $M$ é a matriz $4 \times 4$ do operador $T$ e $P = [x, y, z, w]$ é um ponto em $RP^3$, então, $\overline{T}(P)$ tem coordenadas homogêneas dadas por: $P^* = M(x, y, z, w)^T$.

Para se estudar as transformações projetivas de $RP^3$, do ponto de vista geométrico, vai-se dividir a matriz de transformação em quatro partes:

$$\left(\begin{array}{ccc|c} & & & 3 \\ & 3 \times 3 & & \times \\ & & & 1 \\ \hline -- & -- & -- & -- \\ & 1 \times 3 & & 1 \times 1 \end{array}\right).$$

A submatriz $3 \times 3$ produz uma transformação linear na forma da combinação de um escalamento, cisalhamento, reflexão e rotação. A submatriz $3 \times 1$ produz uma translação, a submatriz $1 \times 3$ produz uma transformação perspectiva e a submatriz $1 \times 1$ produz um escalamento global.

A operação de rotação ao redor de um eixo coordenado é obtida apenas com a submatriz $3 \times 3$. Por exemplo, rodar um objeto ao redor do eixo $x$ não altera as coordenadas $x$ dos seus pontos, pois a rotação acontece em planos perpendiculares ao eixo $x$. Este fato vale para os outros eixos coordenados. A rotação em cada um desses planos é governada pela matriz de rotação bi-dimensional. A partir desta matriz, e usando o fato que uma rotação ao redor do eixo $x$ não altera as coordenadas $x$, pode-se escrever a matriz $4 \times 4$ de rotação,

por um ângulo $\theta$, ao redor do eixo $x$:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\theta) & -sin(\theta) & 0 \\ 0 & sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

A rotação é positiva (sentido anti-horário) quando se olha para a origem, a partir do lado positivo do eixo de rotação. Analogamente, a matriz de rotação, por um ângulo $\gamma$, ao redor do eixo $z$ é:

$$\begin{pmatrix} cos(\gamma) & -sin(\gamma) & 0 & 0 \\ sin(\gamma) & cos(\gamma) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Para a rotação, por um ângulo $\phi$, ao redor do eixo $y$ a matriz é:

$$\begin{pmatrix} cos(\phi) & 0 & sin(\phi) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(\phi) & 0 & cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Neste último caso, o sinal dos senos foram trocados para manter a convenção da direção positiva de rotação.

A rotação ao redor de um eixo arbitrário do espaço é obtida, em coordenadas homogêneas, com uma composição de transformações (o eixo é especificado por um ponto $(x_0, y_0, z_0)$ e uma direção $(c_x, c_y, c_z)$):

- Transladando o ponto $(x_0, y_0, z_0)$ para a origem.
- Rotacionando o eixo de rotação de forma a que coincida com o eixo $z$. Em geral, este passo envolve duas rotações; uma em torno do eixo $x$ e a outra em torno do eixo $y$.
- Rotacionando em torno de $z$ pelo ângulo desejado.
- Rotacionando pelo inverso das rotações aplicadas ao eixo de rotação.
- Transladando de volta ao ponto $(x_0, y_0, z_0)$.
- $[R_e] = [T^{-1}][R_x^{-1}][R_y^{-1}][R_z][R_y][R_x][T]$

Algumas orientações de um objeto tri-dimensional não podem ser obtidas usando rotações puras; elas requerem reflexões. Em 3D, uma reflexão ocorre em relação a um plano. Uma reflexão em relação ao plano $xy$ altera somente as coordenadas $z$ (seus sinais são trocados). A matriz de reflexão é idêntica a matriz identidade, a menos do elemento (3,3), que é negativo. O mesmo vale para os planos $xz$ e $yz$, onde o elemento negativo é o elemento (2,2) e (1,1), respectivamente.

Uma reflexão em relação a um plano arbitrário, também pode ser obtida, em coordenadas homogêneas, por uma composição de transformações. Os passos necessários são:

- Transladar um ponto qualquer do plano de reflexão para a origem.

- Rodar o vetor normal ao plano de reflexão, em relação à origem, até que coincida com o eixo $z$. Também envolve, tipicamente, duas rotações; uma em relação a $x$ e a outra em relação a $y$.

- Refletir o objeto em relação ao plano $z = 0$.

- Executar as transformações inversas daquelas aplicadas ao plano de reflexão.

- $[Rflt_p] = [T^{-1}][R_x^{-1}][R_y^{-1}][Rflt_z][R_y][R_x][T]$

## 5.8 Transformação Perspectiva

Chama-se de transformação perspectiva a uma transformação projetiva cuja imagem de algum ponto ideal $P_0$ é um ponto $P$ do espaço afim. Neste caso, as retas paralelas na direção do ponto $P_0$ são transformadas em retas incidentes a $P$. Este tipo de transformação é muito importante no processo de visualização.

Quando algum elemento da submatriz $(1 \times 3)$ é diferente de 0, ocorre uma transformação perspectiva. Uma transformação perspectiva com um único ponto de fuga, sobre o eixo $z$, é dada por:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & r & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \\ rz+1 \end{pmatrix}.$$

Aqui, um ponto do espaço afim é levado para o hiperplano $w = rz + 1 \neq 1$. As coordenadas homogêneas usuais são obtidas dividindo-se por $w : [x', y', z', 1] = [x/w, y/w, z/w, 1]$. Note-se que os pontos do espaço afim com coordenada $z = 0$ não são afetados por esta transformação e aqueles com coordenada $z = -1/r$ são transformados em pontos ideais (fig. 5.6).

Uma projeção perspectiva em algum plano de visão bi-dimensional é obtida pré-multiplicando-se, uma matriz de projeção ortográfica[5], à matriz de transformação perspectiva (projetanto no plano $z = 0$, o efeito é descartar a coordenada $z$). Escolhendo-se um centro de projeção $z_c$, sobre o eixo $z$, e fazendo $r = -1/z_c$, obtém-se o mesmo resultado de uma projeção sobre o plano $z = 0$.

Aplicando a transformação perspectiva ao ponto ideal $[0, 0, 1, 0]$, que corresponde à interseção das retas paralelas ao eixo $z$, obtém-se $[0, 0, 1, r]$. O ponto $[x', y', z', 1] = [0, 0, 1/r, 1]$ representa a imagem deste ponto ideal. Dessa forma, vê-se que as retas paralelas à direção $z$ convergem para um ponto de fuga, de coordenada $1/r$, sobre o eixo $z$. Isto significa que o semi-espaço infinito $0 \leq z \leq \infty$ é transformado no semi-espaço finito $0 \leq z' \leq 1/r$.

Quando mais de um elemento da submatriz $(1 \times 3)$ é não nulo, têm-se transformações perspectivas com dois ou três pontos de fuga. A transformação perspectiva com três pontos

[5] Esta matriz é singular (não é invertível), pois possui as terceiras linha e coluna, nulas.

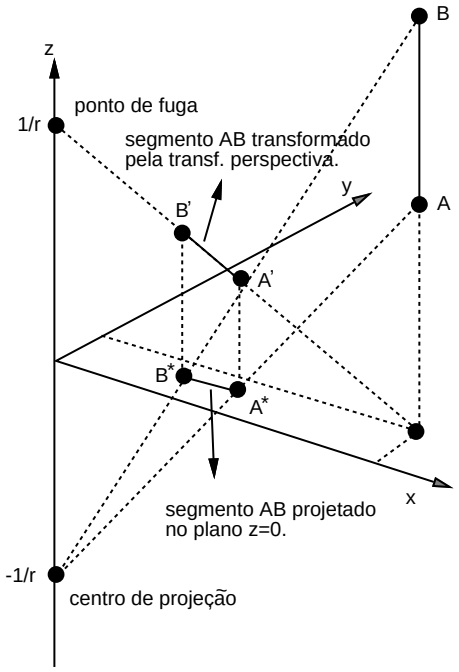



Figura 5.6: Perspectiva com um ponto de fuga.

de fuga é dada por:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ p & q & r & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \\ w = px + qy + rz + 1 \end{pmatrix}, \qquad \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} x/w \\ y/w \\ z/w \\ 1 \end{pmatrix}.$$

A transformação perspectiva com três pontos de fuga tem três centros de projeção: um sobre o eixo $x$, com coordenadas $[-1/p, 0, 0, 1]$; outro sobre o eixo $y$, com coordenadas $[0, -1/q, 0, 1]$; e o terceiro sobre o eixo $z$, com coordenadas $[0, 0, -1/r, 1]$. Os pontos de fuga se localizam sobre o eixo $x$, com coordenadas $[1/p, 0, 0, 1]$; sobre o eixo $y$, com coordenadas $[0, 1/q, 0, 1]$; e sobre o eixo $z$, com coordenadas $[0, 0, 1/r, 1]$. O mesmo resultado é obtido pela aplicação, em cascata, de três transformações perspectivas com um único ponto de fuga, um em cada eixo coordenado.

Transformações perspectivas com dois pontos de fuga podem ser obtidas pela composição de uma transformação perspectiva, com um único ponto de fuga, com uma transformação de rotação ao redor de um eixo coordenado perpendicular ao eixo coordenado que contém o centro de projeção. Se forem usadas duas rotações, obtém-se uma transformação perspectiva com três pontos de fuga: $[T_{3pf}] = [P_{rz}][R_x][R_y]$.

Os pontos de fuga correspondendo às direções não paralelas aos eixos principais podem ser encontrados calculando, diretamente, a interseção de dois segmentos (originalmente paralelos à direção desejada) transformados, ou encontrando a imagem do ponto ideal correspondente.

## 5.9 Exercícios

**5.1** *Faça um estudo comparativo das Geometrias Euclideana e Afim, incluindo uma discussão dos principais conceitos, tipos de transformações e invariantes.*

**5.2** *Explique as principais vantagens do uso da Geometria Projetiva em Computação Gráfica.*

**5.3** *Mostre, a partir de um exemplo 2D e um 3D, que transformações geométricas não são comutativas.*

**5.4** *Discuta um método que permita rodar um objeto 3D em torno de um eixo de rotação arbitrário e refletí-lo em relação a um plano arbitrário.*

**5.5** *Prove que uma transformação linear preserva paralelismo. Em seguida, construa uma transformação projetiva que leve um quadrado unitário num quadrilátero arbitrário (em 2D).*

**5.6** *Dê uma justificativa geométrica para o fato de uma transformação projetiva transformar uma cônica em outra cônica qualquer, por exemplo, um círculo em uma hipérbole.*

**5.7** *Verifique que a transformação afim*

$$\begin{aligned} x &= x_1 & \qquad x_1 &= x \\ y &= y_1 - x_1 & \qquad y_1 &= x + y \end{aligned}$$

*leva o círculo* $x^2 + y^2 - 1$ *na elipse* $2x_1^2 - 2x_1y_1 + y_1^2 - 1$.

**5.8** *Verifique que a transformação projetiva*

$$\begin{aligned} x &= x_1 - y_1 & \qquad x_1 &= (w + x)/2 \\ y &= w_1 & \qquad y_1 &= (w - x)/2 \\ w &= x_1 + y_1 & \qquad w_1 &= y \end{aligned}$$

*leva o círculo* $x^2 + y^2 - w^2$ *na hipérbole* $w_1^2 - 4x_1y_1$.

# Capítulo 6

# Modelagem Geométrica

A Modelagem Geométrica ocupa uma posição central em Computação Gráfica devido ao seu imapcto sobre os processos industriais automatizados de projeto e manufatura (sistemas CAD/CAM). Desde o seu surgimento, na década de 60, esta área vem evoluindo sensivelmente. Inicialmente, o problema que se colocava eram os sistemas de desenho, tanto bi como tri-dimensionais[1]. Estes sistemas utilizavam modelos bastante ingênuos, que nada mais eram do que conjuntos de pontos ligados por segmentos de reta, conhecidos como modelos de arame (*wire frame models*).

O principal problema dos modelos de arame é que eles não conseguem captar a noção de sólido[2], que é caracterizado por propriedades físicas (como massa, volume, área, momento de inércia, etc...) e, por conseguinte, não atendem às demandas da indústria. Na realidade, faltava, nessa época, uma conceituação precisa do que é um objeto sólido.

## 6.1 Esquemas de Representação

O conceito de sólido é melhor compreendido a partir do paradigma dos quatro universos[3] (fig. 6.1), onde fica clara a necessidade de um modelo matemático que caracterize, precisamente, um objeto sólido. A base teórica para esta conceituação é encontrada na geometria diferencial e na topologia diferencial. Um estudo da teoria necessária destas duas disciplinas da matemática foge ao escopo deste texto. Assim, vão-se discutir apenas representações para um sólido, onde as características importantes para a validade da representação serão apontadas.

### 6.1.1 Descrição de Sólidos

Informalmente, assuma-se que um sólido é um conjuntos de pontos. Este conjunto de pontos pode ser descrito, basicamente, de duas formas:

[1] O trabalho pioneiro de Ivan Sutherland, que criou o Sketchpad, um sistema de desenho bi-dimensional, é o marco inicial desta área.

[2] A detecção de colisão, por exemplo, necessita do conceito de sólido.

[3] Introduzido em 1980 por Ari Requicha, um pesquisador português radicado nos USA.

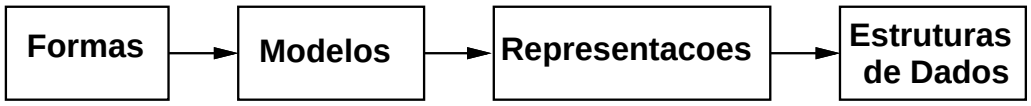



Figura 6.1: Modelo Conceitual.

- por coordenadas (parametricamente);
- por densidade (implicitamente).

Estes dois tipos de descrição originam dois tipos de representação de sólidos a saber:

- Representação por bordo, B-rep (*Boundary representation*);
- Representação implícita, CSG (*Constructive Solid Geometry*).

## 6.2 Representação por Bordo

Na representação por bordo ou por fronteira, o sólido é definido indiretamente, através da superfície, compacta[4] e sem bordo, que o delimita (forma a sua fronteira). Esta superfície é descrita parametricamente por um mapeamento, $\varphi : U \subset \Re^2 \rightarrow \Re^3$, chamado de uma parametrização da superfície ou carta local (fig. 6.2).

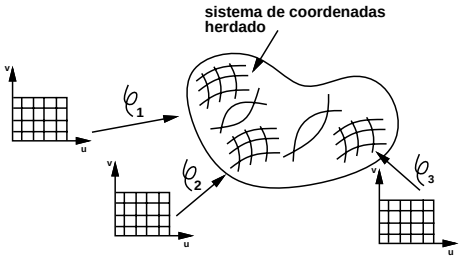



Figura 6.2: Parametrização de uma Superfície.

Desta forma, a parametrização

$$\varphi(u,v) = (\varphi_1(u,v),\ \varphi_2(u,v),\ \varphi_3(u,v)) = (x,y,z)$$

associa a cada ponto do plano, um ponto do espaço tri-dimensional. Para que o sólido seja bem definido, exige-se que a superfície não apresente auto-interseção e que o vetor normal, $(\partial\varphi/\partial u \times \partial\varphi/\partial v)$, não se anule em algum ponto da superfície. Isto porque a normal é

[4] Fechada e limitada.

utilizada para determinar o interior e o exterior do sólido, a partir de alguma convenção, por exemplo, que a normal em qualquer ponto da superfície aponte, localmente, para o exterior do sólido.

A parametrização estabelece um sistema de coordenadas sobre a superfície, herdado de um sistema de coordenadas no plano. Em geral, é impossível cobrir (descrever) toda a superfície com uma única parametrização. Por isso, são utilizadas descrições por partes, onde empregam-se várias parametrizações que definem pedaços de superfície que vão sendo colados uns aos outros (fig. 6.2), formando um atlas.

A parametrização de uma esfera de raio um, centrada na origem, é dada por exemplo:

$$f(\theta, \phi) = (cos\theta sin\phi,\ sin\theta sin\phi,\ cos\phi).$$

O vetor normal é dado em cada ponto por:

$$\frac{\partial f}{\partial \theta} \times \frac{\partial f}{\partial \phi} = (-sin\theta sin\phi,\ cos\theta sin\phi,\ 0) \times (cos\theta cos\phi,\ sin\theta cos\phi,\ -sin\phi).$$

Se $\phi = \pi$ ou $\phi = 0$ a normal à esfera não está definida, por esta parametrização, nos polos. Assim, o domínio $U$ desta parametrização é dado por:

$$U = \{(\theta, \phi) \in \Re^2;\ 0 < \phi < \pi;\ 0 \leq \theta < 2\pi\}.$$

### 6.2.1 Representação Linear por partes

Uma superfície parametrizada e geometricamente complexa pode ser aproximada por uma superfície linear por partes. Um método bastante empregado consiste em poligonizar o domínio da parametrização (em geral são utilizados triângulos ou quadriláteros). Cada vértice no domínio poligonal é levado para a superfície pela parametrização e ligado aos vértices adjacentes. Este processo cria uma superfície poligonizada, chamada de malha poligonal. Com isto, obtém-se uma geometria aproximada mas, em compensação, bastante simples.

Uma malha poligonal nada mais é do que uma decomposição celular[5] poligonal definida por uma coleção conexa de vértices, arestas e polígonos, tal que cada aresta é compartilhada por no máximo dois polígonos e a interseção de dois polígonos quaisquer é uma aresta, um vértice ou é vazia. Algumas operações básicas com malhas poligonais correspondem a:

- achar todas as arestas que incidem em um vértice;
- achar os polígonos que compartilham uma aresta ou um vértice;
- achar as aresta que delimitam um polígono;
- exibir a malha.

Existem quatro formas básicas para codificar uma malha poligonal:

- codificação explícita;

[5] Um complexo afim homeomorfo à superfície.

- codificação com ponteiros para lista de vértices;
- codificação com ponteiros para lista de arestas;
- codificação com arestas aladas (*winged-edge*).

### Codificação Explícita

Nesta codificação, cada polígono armazena explicitamente uma lista de coordenadas de vértices: $P = \{(x_1, y_1, z_1), (x_2, y_2, z_2), ..., (x_n, y_n, z_n)\}$. Esta codificação é adequada para armazenar um único polígono. No entanto, produz redundância de informação, quando o objetivo é armazenar vários polígonos, uma vez que são armazenados vértices duplicados. Além disso, cada aresta compartilhada é desenhada duas vezes na exibição da malha, o que é problemático quando se utilizam plotadoras ou filme. A execução de *querys* também é complicada, porque os relacionamentos de adjacência entre vértices, arestas e polígonos não são armazenados explicitamente (obriga a execução de algoritmos geométricos).

### Codificação com Ponteiros para Lista de Vértices

Neste tipo de codificação, os vértices são armazenados separadamente, em uma lista de vértices. Cada polígono faz referência aos seus vértices, na lista de vértices, por meio de ponteiros (fig. 6.3).

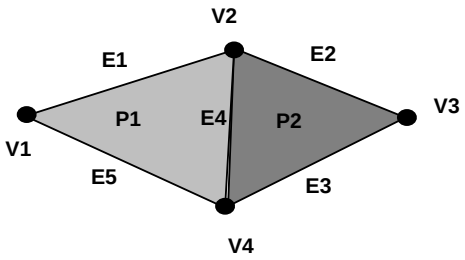



Figura 6.3: Codificação com Ponteiros para Lista de Vértices.

- $V = \{V_1 = (x_1, y_1, z_1), V_2 = (x_2, y_2, z_2), V_3 = (x_3, y_3, z_3), V_4 = (x_4, y_4, z_4)\}$;
- $P_1 = \{V_1, V_2, V_4\}$;
- $P_2 = \{V_4, V_2, V_3\}$.

Esta codificação proporciona uma maior economia de espaço, pois cada vértice é armazenado uma única vez. Além disso, as coordenadas de um vértice podem ser alteradas facilmente. Porém, ainda é dificíl determinar os polígonos que compartilham uma aresta e as arestas compartilhadas ainda são desenhadas duas vezes.

**Codificação com Ponteiros para Lista de Arestas**

Se, além de uma lista de vértices, for introduzida também uma lista de arestas, os polígonos agora fazem referência às suas arestas, na lista de arestas, através de ponteiros (fig. 6.3).

- $V = \{V_1 = (x_1, y_1, z_1), V_2 = (x_2, y_2, z_2), V_3 = (x_3, y_3, z_3), V_4 = (x_4, y_4, z_4)\}$;
- $E_1 = \{V_1, V_2, P_1, \lambda\}$;
- $E_2 = \{V_2, V_3, P_2, \lambda\}$;
- $E_3 = \{V_3, V_4, P_2, \lambda\}$;
- $E_4 = \{V_2, V_4, P_1, P_2\}$;
- $E_5 = \{V_4, V_1, P_1, \lambda\}$;
- $P_1 = \{E_1, E_4, E_5\}$;
- $P_2 = \{E_2, E_3, E_4\}$.

Agora, desenham-se todas as arestas a partir da lista de arestas e não mais percorrendo a fronteira dos polígonos. A determinação das arestas que incidem em um vértice ainda requer um algoritmo geométrico. Se for introduzido para cada aresta as referências para os dois polígonos que a compartilham ($\lambda$, se houver apenas um), fica imediata a determinação dos polígonos que incidem na aresta.

**Codificação com Arestas Aladas**

A codificação com arestas aladas surgiu em 1974 e foi um marco no desenvolvimento de codificações para representação por fronteira. Criada por Baungart, ela armazena uma série de informações na estrutura associada a arestas, pois o número de campos necessário é fixo (uma aresta é delimitada por dois vértices e delimita dois polígonos, chamados faces). São mantidas uma lista de vértices, uma lista de arestas e uma lista de faces. Cada aresta possui, então, ponteiros para as duas faces que a compartilham, para o seu vértice inicial e para as quatro arestas que a sucedem e antecedem no ciclo ordenado de arestas ao redor das duas faces (fig. 6.4).

Todos os nove tipos de relacionamentos de adjacência que podem ser formados a partir de vértices, faces e arestas são obtidos, diretamente, da estrutura de dados. A atualização da estrutura de dados neste tipo de codificação é uma tarefa complexa e é feita, normalmente, com o emprego dos chamados operadores de Euler, para garantir que após cada atualização o modelo continua satisfazendo a equação de Euler: $V - A + F = 2$.

## 6.3 Representação Implícita

Na representação implícita, o sólido é definido a partir de um conjunto de valores de densidade que caracteriza os seus pontos. Por exemplo, uma mesa é caracterizada pelo conjunto

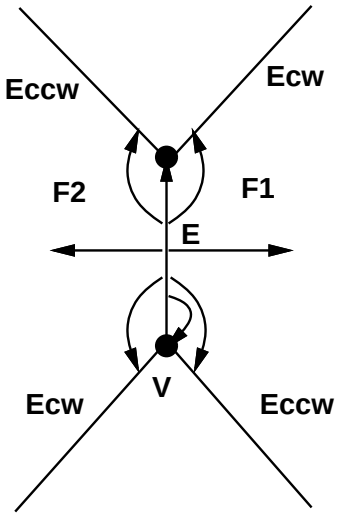



Figura 6.4: Codificação por Arestas Aladas.

de pontos do ambiente com densidade da madeira, enquanto o seu complemento é formado pelo conjunto de pontos com densidade do ar. O conjunto dos pontos do $\Re^n$ que estejam associados a um certo intervalo de densidades forma o conjunto de pontos que define o sólido. Este tipo de representação descreve a superfície dos objetos, implicitamente, por uma equação do tipo: $F(X) = c;\ X \in \Re^n,\ c \in \Re$. A função $F : \Re^n \to \Re$ é de classe $C^k$ e é chamada de função implícita.

Em geral, uma superfície definida de forma implícita pode apresentar auto-interseção. A pergunta é: $F(x, y, z) \to \Re$ define implicitamente $z = f(x, y)$ em algum domínio razoável? Em Computação Gráfica não se deseja que uma curva ou superfície que delimita a fronteira de um sólido se auto-intercepte. A verificação deste tipo de situação é feita com base no teorema da função implícita:

**Teorema 6.1** *Seja $F : \Re^n \to \Re$ definida em um aberto $U$. Se $F$ possui derivadas parciais contínuas em $U$ (ou seja, $F$ é diferenciável em $U$) e $\nabla F \neq 0$ em $U$, então $F$ é uma subvariedade de dimensão $n-1$ do $\Re^n$ (uma superfície sem auto-interseção).*

Um valor $c \in \Re$ é dito um valor regular de $F$ se $F^{-1}(c)$ não contém pontos onde o gradiente de $F$ se anula (chamados de pontos singulares):

$$\forall p \in F^{-1}(c) \Rightarrow \nabla F_p = \left(\frac{\partial F}{\partial x}, \frac{\partial F}{\partial y}, \frac{\partial F}{\partial z}\right) |_p \neq 0.$$

Neste texto interessam apenas os casos em que $n = 2$ ou $n = 3$, onde se têm curvas e superfícies implícitas, respectivamente.

Considere-se a função $F(x, y) = x^2 + y^2$ que define um parabolóide no $\Re^3$. Fixando um valor constante $c$ para $F$, obtêm-se as curvas de nível da função que, neste caso, são círculos, se $c > 0$. Esta é uma forma de definir um círculo implicitamente. Note-se que

$\nabla F = (2x, 2y)$ se anula na origem. Neste caso, 0 não é um valor regular de $F$ e, por conseguinte, $F(x, y) = 0$ não define uma superfície implícita[6]. Pode-se mostrar que o vetor gradiente é perpendicular à curva de nível em cada ponto (faça-o!).

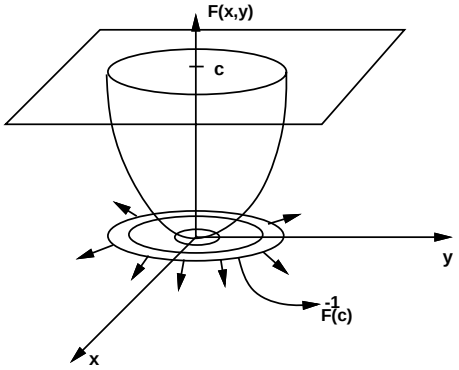



Figura 6.5: Círculo Definido Implicitamente.

Um outro exemplo são cascas esféricas definidas implicitamente por uma função

$$F(x, y, z) = x^2 + y^2 + z^2.$$

Para todo $k > 0$, $F^{-1}(k)$ representa a superfície de uma esfera no $\Re^3$. Novamente, 0 não é um valor regular de $F$, pois $F^{-1}(0) = (0, 0, 0)$ e $\nabla F = (2x, 2y, 2z)$ se anula na origem.

Um exemplo mais interessante é dado pela função abaixo:

$$\begin{aligned} F(x, y) &= y^2 - x^2 - x^3, \\ \nabla F &= (2y, -3x^2 - 2x), \text{ ou na forma paramétrica,} \\ x(t) &= t^2 - 1, \\ y(t) &= t(t^2 - 1). \end{aligned} \tag{6.1}$$

Note-se que a curva de nível 0 de $F$ é um laço, que apresenta uma singularidade na origem (fig. 6.6):

$$z = F(x, y) = y^2 - x^2 - x^3 = 0.$$

Se olhar-se agora para F(x,y) como superfície de nível 0 da função

$$\begin{aligned} H : \Re^3 \to \Re, \qquad H(x, y, z) &= -z + y^2 - x^2 - x^3, \\ \nabla H &= (-3x^2 - 2x, 2y, -1), \\ \nabla H_{(0,0,0)} &= (0, 0, -1), \end{aligned}$$

[6] $F^{-1}(0) = (0, 0)$.

vê-se que todos os pontos são regulares, pois o gradiente de $H$ é diferente de $(0,0,0)$ em todos os pontos do $\Re^3$. Conclui-se então que o gráfico de $F$ no $\Re^3$ é realmente gráfico de uma função.

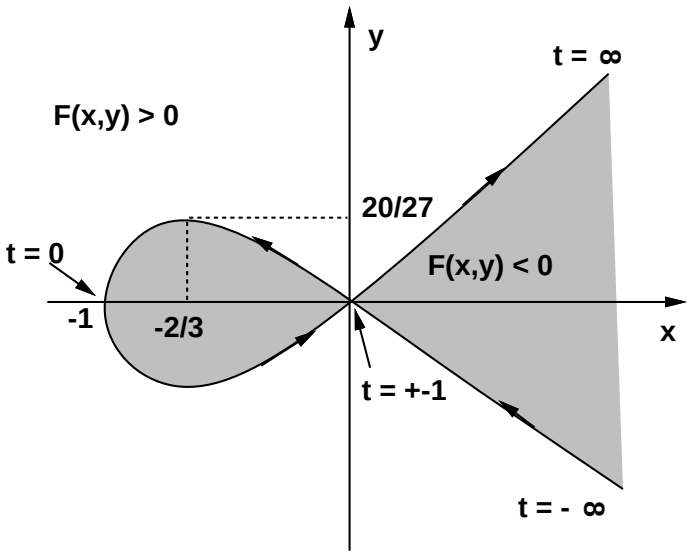



Figura 6.6: Curva do Laço.

### 6.3.1 Objeto Implícito

Até aqui foram dados exemplos de superfícies implícitas. Na realidade, necessita-se do conceito de objeto implícito, para que se possa construir uma representação para sólidos.
**Definição:** Um subconjunto $O \subset \Re^n$ é chamado de objeto implícito (fig. 6.7) se existe $F : U \to \Re,\ O \subset U$, e existe um subconjunto

$$V \subset \Re \ \text{ tal que } \ O = F^{-1}(V) \ \text{ ou } \ O = \{p \in U,\ F(p) \in V\}.$$

Um objeto implícito é regular se a função $F$ satisfaz a condição de regularidade. Um objeto implícito é válido se ele define uma superfície no $\Re^n$.
Seja $p = (x, y, z)$. Então, vai-se considerar que sendo $F(p)$:

- $> 0 \ \Rightarrow p \in$ exterior de $O$;
- $= 0 \ \Rightarrow p \in$ fronteira de $O$;
- $< 0 \ \Rightarrow p \in$ interior de $O$.

A função $F$ faz uma classificação dos pontos do espaço. Dado qualquer ponto, $F$ permite decidir se o ponto está no interior, na fronteira ou no exterior do objeto.

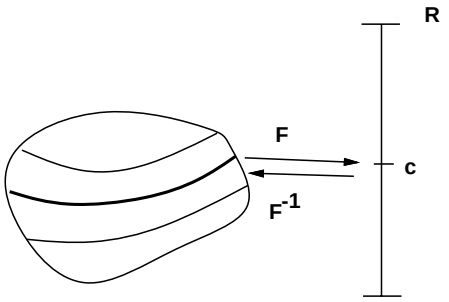



Figura 6.7: Objeto Implícito.

### 6.3.2 Esquema de Representação CSG

Um método efetivo para construir objetos complexos, a partir de objetos simples, é através de operações CSG (*Constructive Solid Geometry*). Este esquema é baseado em operações *booleanas* regularizadas de conjuntos de pontos: $\cup$, $\cap$, / (união, interseção e diferença).

Um objeto é regular se o fecho do interior do seu conjunto de pontos é igual ao próprio conjunto de pontos. Um modelo CSG é codificado por uma árvore, onde os nós contêm operações de conjunto ou transformações geométricas e as folhas contêm objetos primitivos.

Um sólido CSG é definido como um conjunto de pontos do $\Re^n$ que satisfaz $F(X) \leq 0$ para algum $F : \Re^3 \rightarrow \Re$ de classe $C^1$. As operações *booleanas* são definidas por (prove!):

- $F_1 \cup F_2 = min(F_1, F_2)$;
- $F_1 \cap F_2 = max(F_1, F_2)$;
- $F_1/F_2 = F_1 \cap \overline{F_2} = max(F_1, -F_2)$.

## 6.4 Conversão entre Representações

A conversão entre as representações por bordo e implícita é um tópico importante porque ambas apresentam vantagens e desvantagens. Na representação por bordo as interseções de superfícies estão armazenadas explicitamente, além de ser fácil exibir um ponto sobre uma superfície. Em contra partida, a determinação da inclusão de um ponto em um objeto requer algoritmos não triviais, especialmente em dimensão três. A avaliação de operações *booleanas* é bastante trabalhosa do ponto de vista algoritmico.

Na representação implícita, a determinação da inclusão de um ponto é imediata, bastando avaliar o sinal da função de densidade no ponto. Porém, exibir um ponto sobre a superfície de um objeto implica em resolver uma equação que pode ser complicada, mas a avaliação de operações *booleanas* é simples.

Estas considerações apontam na direção de sistemas com multi-representação. Determinados algoritmos são mais facilmente implementados a partir de uma certa representação. Assim, caso exista conversão entre as representações, cada algoritmo pode se valer da re-

presentação mais apropriada. A conversão se encarrega de manter ambas as representações atualizadas (fig. 6.8).

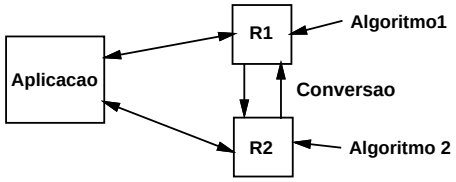



Figura 6.8: Multi-representação.

A conversão basicamente pode ser efetuada de duas formas: exata ou aproximada. Por exemplo, um círculo definido implicitamente pela equação $x^2 + y^2 = 1$ possui uma descrição paramétrica equivalente:

$$\varphi(\theta) = (cos(\theta),\ sin(\theta)); \qquad 0 \leq \theta < 2\pi.$$

Porém, em geral, não existe conversão exata (é fácil converter de forma exata curvas algébricas de grau 2 e 3 ou monóides).

A conversão de uma representação implícita para uma representação por bordo é designinada pelo termo — avaliação do bordo. Esta conversão é feita de forma aproximada e envolve a poligonização de superfícies implícitas[7]. No caso em questão, seja $M$ uma superfície implícita. Construa-se uma triangulação $T$ suficientemente fina do espaço Euclideano, de forma que, para todo simplexo $\sigma$ de $T$, $M \cap \sigma$ é homeomorfo a um disco $m$-dimensional. Exige-se que a triangulação $T$ seja uma "boa triangulação", que satisfaça às seguintes condições:

- O diâmetro de cada simplexo $\sigma$ de $T$ é $\epsilon/2$;
- $T$ não possui vértices sobre $M$;
- $M$ intercepta o esqueleto de ordem 1 de $T$ transversalmente (não há tangência);
- Cada aresta de $T$ intercepta $M$ em no máximo um ponto.

A primeira condição é satisfeita se for utilizada, por exemplo, a triangulação CFK (Coxeter-Frendenthal). As outras condições podem ser obtidas a partir de uma pequena perturbação na triangulação. Com uma boa triangulação, a interseção de uma superfície no $\Re^3$ com um simplexo (no caso, um tetraedro) se dá em três ou quatro pontos (fig. 6.9a).

O problema se reduz, então, em determinar, para cada aresta de um simplexo, se $M$ a intercepta e, se for o caso, encontrar a interseção. Avaliando-se a função implícita nos

[7] Toda superfície, tanto paramétrica como implícita, conforme definida neste texto, admite uma poligonização.

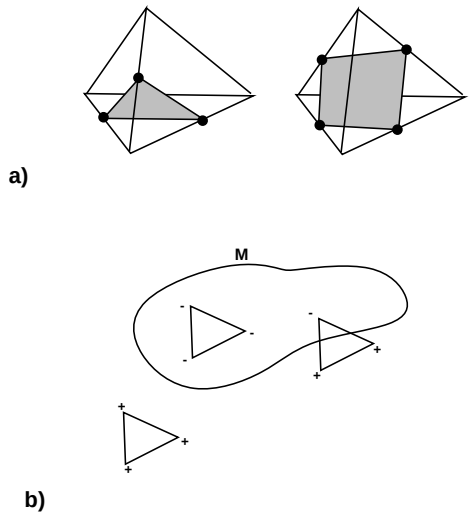



Figura 6.9: Interseção de $M$ com um Tetraedro.

vértices de uma aresta e se, em cada vértice, os sinais forem opostos significa que há interseção (fig. 6.9b). Assim, $\varphi(t) = tv_1 + (1-t)v_0$, que é a equação paramétrica do suporte da aresta, é usada para encontrar as raízes da equação $F(\varphi(t)) = 0,\ 0 \leq t \leq 1$. Os pontos de interseção devem ser ligados na ordem apropriada.

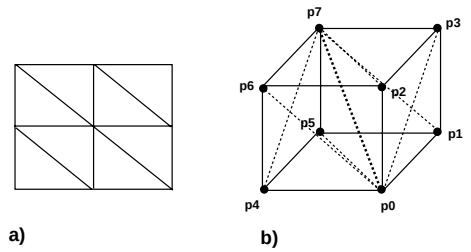
Uma triangulação CFK é obtida no $\Re^2$ dividindo-se a região de interesse em quadrados e traçando-se uma diagonal em cada quadrado (fig. 6.10a). No $\Re^3$, a região é dividida em cubos. Uma diagonal é traçada e projetada em cada face do cubo. Adiciona-se, então, a cada simplexo 2D, o vértice da diagonal que não pertence ao simplexo (fig. 6.10b). Em geral, no $\Re^n$, um hipercubo tem $n!$ simplexos de dimensão $n$.

A geração de imagens médicas, a partir de dados colhidos através de aparelhos de tumografia computadorizada, envolve, justamente, a conversão de uma representação implícita (estes aparelhos retornam as densidades dos diferentes tecidos) para uma representação por bordo (neste processo, o bordo é aproximado por uma superfície poligonal composta por milhares de triângulos).

Note-se que a conversão CSG $\rightarrow$ B-rep é mais complicada, pois estão envolvidos vários objetos implícitos. Neste caso, interessa apenas a fronteira externa. Os polígonos devem ser recortados e aqueles contidos no interior do objeto final devem ser descartados.

A conversão da representação por bordo para a implícita se dá através de uma subdivisão espacial. O espaço é particionado em células e, em seguida, são enumeradas as células que interceptam o objeto (não existe ocupação parcial). As estruturas quadtree e octree, encontradas na literatura, não são esquemas de representação propriamente ditos, mas apenas estruturas de dados para a decomposição celular do espaço.

Uma octree é codificada como uma árvore na qual a raíz corresponde a um cubo do $\Re^3$ que contém todo o modelo. Em seguida, este cubo é subdividido em oito cubos congruentes,




- $(p_0, p_1, p_3, p_7)$ $(p_0, p_1, p_5, p_7)$
- $(p_0, p_2, p_3, p_7)$ $(p_0, p_2, p_6, p_7)$
- $(p_0, p_4, p_5, p_7)$ $(p_0, p_4, p_6, p_7)$

Figura 6.10: Triangulação CFK.

chamados octantes, por três planos ortogonais aos eixos coordenados. Cada octante é então classificado em:

- cheio - completamente contido no interior do objeto;
- vazio - completamente contido no exterior do objeto;
- incompleto - uma parte está contida no interior e a outra no exterior do objeto.

Octantes incompletos são subdivididos, recursivamente, até um nível máximo. Neste ponto, os octantes incompletos são considerados cheios.

## 6.5 Ambigüidade e Unicidade de Representações

Retornando ao paradigma dos quatro universos, define-se que uma representação é única quando um modelo possui uma, apenas uma, representação. Em geral, a condição de unicidade é de difícil obtenção. Tanto a representação implícita como a representação por bordo não são representações que apresentam unicidade (dê exemplos!).

O conceito de ambigüidade é mais importante pois estabelece que, dada uma representação, existe um único modelo associado. Uma representação ambígüa é catastrófica, pois não se consegue determinar qual o modelo que ela representa (tornaria uma máquina de controle numérico, inviável). A representação *wire-frame* é extremamente ambígüa.

Um dos problemas que se coloca na área de Modelagem Geométrica diz respeito à obtenção de algoritmos robustos para a conversão (aproximada) entre representações distintas.

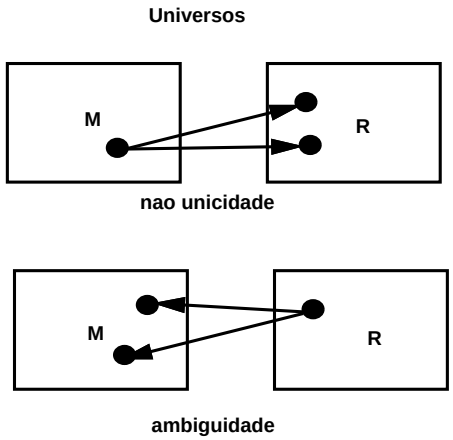



Figura 6.11: Unicidade e Ambigüidade de Representação.

## 6.6 Exercícios

**6.1** *Discuta a obtenção dos relacionamentos de adjacência, entre vértices, arestas e faces, a partir das seguintes estruturas de dados:*

- *lista de faces;*
- *ponteiros para lista de vértices;*
- *ponteiros para listas de arestas;*
- *winged edge.*

**6.2** *Considerando apenas vértices, arestas e faces é possível formar 9 tipos de relacionamentos de adjacência diferentes. Por exemplo, F(A) refere-se ao conjunto de arestas em torno de uma face, F(V) ao conjunto de vértices em torno da face, etc. Quais destes conjuntos podem ser ordenados?*

**6.3** *Como alterar a estrutura de dados baseada em ponteiros para lista de arestas de forma a que a determinação das faces que compartilham uma aresta possa ser executada eficientemente?*

**6.4** *Prove as seguintes igualdades:*

- $F_1 \cap F_2 = max(F_1, F_2)$,
- $F_1 \cup F_2 = min(F_1, F_2)$,
- $F_1 - F_2 = max(F_1, -F_2)$,

*onde $F_i : \Re^n \to \Re$ são funções que determinam um objeto implícito como a imagem inversa do ponto zero (assumindo que zero é um ponto regular de $F_i$). Qual a importância deste resultado?*

**6.5** *Construa um programa para gerar uma aproximação poligonal para uma superfície implícita de dimensão 1 (curva), compacta e sem bordo.*

**6.6** *Estenda a implementação do exercício anterior de forma a a aproximar também superfícies implícitas de dimensão 2.*

**6.7** *Discuta as vantagens e desvantagens da representação por bordo, que usa uma descrição paramétrica do sólido, e volumétrica, que usa uma descrição implícita do sólido. Que tipos de aplicação se adequam mais a cada uma destas representações? Por que?*

**6.8** *Construa uma parametrização, $\phi : \Re^2 \to \Re^3$, para um triângulo no espaço, dado pelas coordenadas dos seus três vértices.*
*Sugestão: utilize coordenadas baricêntricas.*

**6.9** *Um esquema de representação para poli-linhas usando segmentos de reta associa a uma poli-linha L o conjunto $\{(x_0, y_0, x_1, y_1), (x_1, y_1, x_2, y_2), ..., (x_{n-1}, y_{n-1}, x_n, y_n)\}$ onde $(x_i, y_i, x_{i+1}, y_{i+1})$ representam as coordenadas iniciais e finais de cada segmento de reta que compõe a poli-linha.*

*a) Discuta essa representação em relação às propriedades de validade, ambigüidade e unicidade.*

*b) Defina uma noção de proximidade entre poli-linhas e discuta a representação em relação a este conceito.*

**6.10** *Um polígono P no espaço é uma sequência $V_1, V_2, ..., V_k$ de pontos distintos, situados no mesmo plano, tal que $V_1 = V_k$. O esquema de representação LV associa a P a n-tupla $LV(P) = (V_1, V_2, ..., V_{k-1})$.*

*a) Especifique um tipo abstrato de dados (TAD) para implementar essa representação.*

*b) Relacione as operações associadas a esse TAD com operações no modelo.*

*c) Investigue as possíveis inconsistências topológicas e geométricas que podem ocorrer nessa representação.*

*d) Descreva procedimentos para identificar as inconsistências no item anterior.*

**6.11** *Considere o universo dos polígonos convexos de n lados e elabore métodos de representação geométrica para descrever os elementos desse universo.*

*a) Usando um esquema construtivo.*

*b) Usando um esquema de decomposição.*

*c) Usando um esquema de aproximação.*

*d) Discuta a conversão entre estes sistemas.*

**6.12** *Estenda o exercício anterior usando métodos de representação algoritmica.*

*a) Defina os parâmetros do modelo.*

*b) Inclua pelo menos duas representações geométricas.*

*c) Descreva a implementação das operações de:*

*Visualização em aramado.*

*Cálculo de área.*

*Escalamento.*

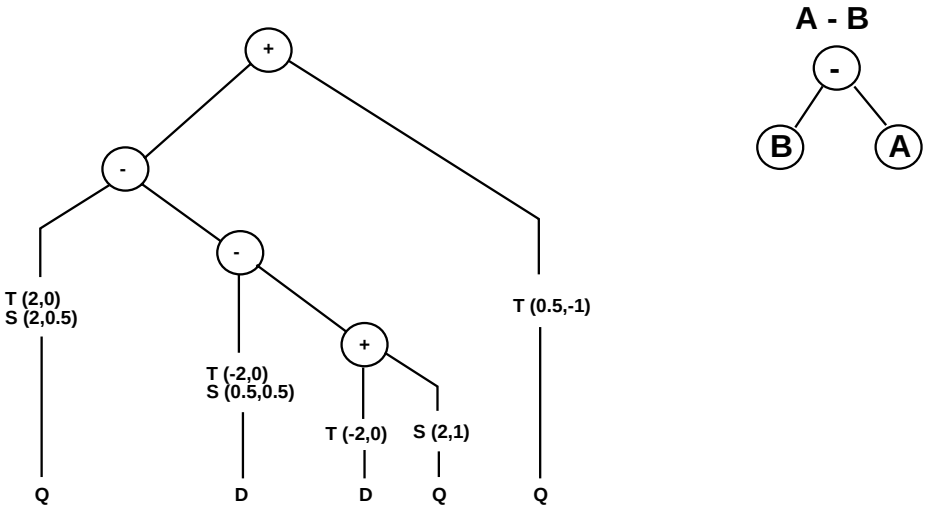
*Teste de inclusão de um ponto.*

**6.13** *Considere um sistema de representação CSG bidimensional que utiliza as seguintes primitivas:*

$$\textit{Disco (D):}\ \{(x,y) \in \Re^2;\ x^2 + y^2 \leq 1\};$$

$$\textit{Quadrado (Q):}\ \{(x,y) \in \Re^2;\ |x| \leq 1\ e\ |y| \leq 1\}.$$

*Desenhe o objeto geométrico que é representado pela árvore CSG da figura 6.12:*




T = Translação + = União de Conjuntos

S = Escalamento - = Diferença de Conjuntos

Figura 6.12: Árvore CSG.

**6.14** *Em um levantamento topográfico, a altura do terreno em relação ao nível do mar (nível 0) é determinada pela amostragem mostrada na figura 6.13.*

*a) Construa uma aproximação poligonal que delimite a área do terreno que está acima do nível do mar.*

*b) Explique o problema do item a) sob o ponto de vista da representação de modelos em Modelagem Geométrica.*

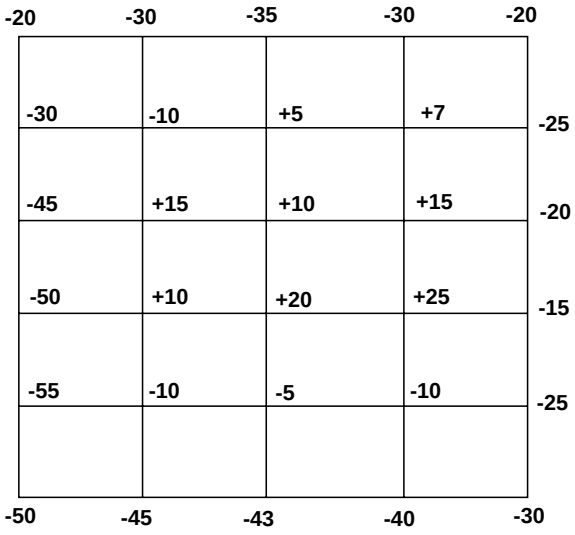



Figura 6.13: Levantamento Topográfico.

**6.15** *Considere o círculo definido implicitamente por $x^2+y^2-1=0$ e o conjunto das retas que passam pelo ponto $p=(-1,0)$, conforme indicado na figura 6.14. Uma forma de converter da forma implícita para a paramétrica consiste em parametrizar as retas que passam por $p$, $r(t): y=tx+t$, e encontrar os pontos de interseção com o círculo (substituindo $y$ na equação do círculo e resolvendo para $x$). Das duas raízes, uma corresponde à coordenada $x$ do ponto $p$ e a outra corresponde ao ponto (variável) $x(t)$ no qual $r(t)$ intersepta o círculo. Derive então a forma paramétrica:*

$$\begin{aligned} x(t) &= \frac{1-t^2}{1+t^2}, \\ y(t) &= \frac{2t}{1+t^2}. \end{aligned}$$

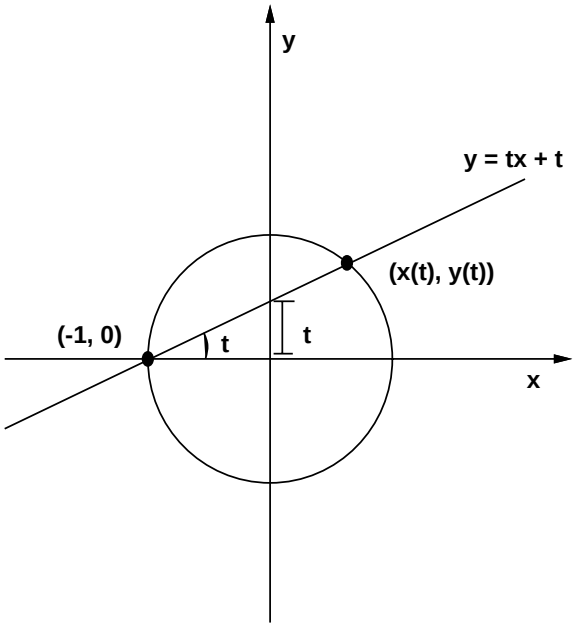



Figura 6.14: Conversão Implícita - Paramétrica.

**6.16** *Para uma cônica geral, o ponto p do exercício anterior pode não ser tão óbvio. Para estes casos, é mais fácil considerar a equação homogeneizada ($[x, y, w] = (x/w, y/w, 1)$):*

$$a_{11}x^2 + a_{22}y^2 + a_{33}w^2 + 2a_{12}xy + 2a_{13}xw + 2a_{23}yw = 0.$$

*Se a cônica contém um dos pontos fundamentais ($(0, 0, 1)$, $(0, 1, 0)$ ou $(0, 0, 1)$) então a equação é linear na variável correspondente, que pode ser explicitada em função das outras duas. Considerando-as coordenadas projetivas parametrizadas, tem-se uma parametrização da cônica.*

*a) Considere o círculo $x^2+y^2-2xw$ que contém a origem $(0, 0, 1)$. Derive a parametrização projetiva da forma projetiva da cônica:*

$$\begin{aligned} x &= s \\ y &= t \\ w &= (s^2 + t^2)/2s. \end{aligned}$$

*b) Coloque $s = 1$ e obtenha coordenadas afins parametrizadas. Divida por $w$ e obtenha uma parametrização afim da forma afim do círculo (fig. 6.15):*

$$\begin{aligned} x(t) &= 2/(1 + t^2) \\ y(t) &= 2t/(1 + t^2). \end{aligned}$$

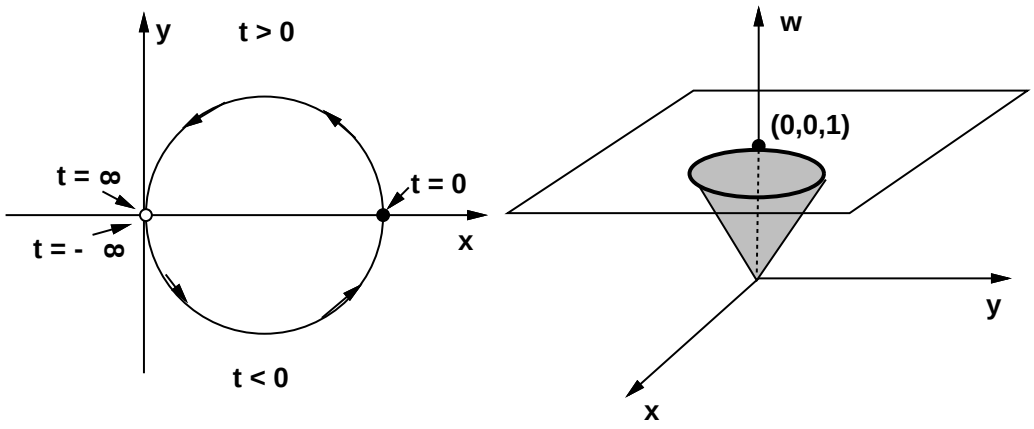



Figura 6.15: Parametrização de Cônicas.

**6.17** *Considere a cônica* $x^2+6xy+5y^2-2x-2y-1=0$. *Na forma homogênea,* $x^2+6xy+5y^2-2xw-2yw-w^2=0$, *vê-se que ela não contém nenhum dos pontos fundamentais, uma vez que possui os termos quadrados em* $x, y$ *e* $w$. *A estratégia é então encontrar um ponto no infinito (correspondendo a* $w=0$*), e levá-lo para* $(0,1,0)$. *Parametriza-se então a cônica transformada e aplica-se a transformação inversa à parametrização obtida.*

*a) Ache os dois pontos no infinito resolvendo* $a_{11}x^2+2a_{12}xy+a_{22}y^2=0$, *fazendo* $y=a_{11}$ *e resolvendo uma equação do segundo grau para encontrar duas coordenadas* $x$.

*b) Seja* $(u,v,0)$ *um ponto no infinito. Aplicando a transformação*

$$\begin{aligned} x &= x_1+uy_1, \\ y &= vy_1, \\ w &= w_1, \end{aligned}$$

*que leva* $(u,v,0)\to(0,1,0)$, *encontre a equação afim da cônica transformada.*

*c) Parametrizando a cônica resultante e aplicando a transformação inversa, obtenha a parametrização da cônica original:*

$$\begin{aligned} x(t) &= \frac{5t^2-2t-1}{4t}, \\ y(t) &= \frac{1+2t-t^2}{4t}. \end{aligned}$$

*d) Entenda o que foi feito desenhando cada passo no plano projetivo. No caso presente, a interpretação geométrica é que a cônica original, que representa uma hipérbole, é levada em uma elipse por uma rotação.*

**6.18** *Um ponto p de uma curva algébrica de grau n tem multiplicidade m se um número infinito de retas por p interceptam a curva em $n-m$ pontos adicionais (considerando pontos complexos e interseções no infinito).*

*Nem todas as cúbicas irredutíveis possuem uma forma paramétrica racional. Aquelas que têm um ponto singular devem tê-lo com multiplicidade 2. Geometricamente, a parametrização de uma cúbica é análoga à parametrização de uma cônica; consideram-se as retas que passam pelo ponto singular duplo e calculam-se as interseções com a cúbica. Só pode haver mais uma única interseção.*

*a) Considere a cúbica $y^2 - x^2 - x^3$. Mostre que a origem é um ponto singular duplo.*

*b) Parametrizando as retas $r(t) : y = tx$ que passam pela origem, obtenha a parametrização apresentada na seção 6.3.*

**6.19** *O teorema de Bezout diz que duas curvas algébricas de grau m e n se interceptam em exatamente mn pontos ou têm uma componente comum.*

*a) Explique o método de parametrização de cônicas e cúbicas, baseado na interseção de retas com a curva, em face ao teorema de Bezout.*

*b) Uma curva algébrica possui um número finito de pontos múltiplos. Um monóide é uma curva algébrica de grau n com um ponto de multiplicidade $n-1$. Generalize o método de parametrização do item anterior para monóides.*

***6.20*** *Suponha que se deseja ter apenas planos com coeficientes racionais. Uma rotação de um ângulo $\alpha$ em torno de um eixo qualquer pode fazer com que o plano transformado passe a ter coeficientes irracionais.*

*a) Dado $\alpha$ qualquer encontre um novo ângulo $\alpha'$ tal que $(u, v) = (cos(\alpha'), sin(\alpha'))$ seja racional.*
*Sugestão: use a parametrização racional do círculo unitário do exercício 6.15, onde $t = m/n = tan(\alpha'/2)$.*

*b) Encontre os novos coeficientes (racionais) do plano rodado em torno do eixo z.*

*c) Componha três rotações racionais para obter a rotação racional em torno de um eixo arbitrário.*

# Capítulo 7

# Sistemas de Modelagem

Os Sistemas de Modelagem são sistemas, geralmente interativos, que permitem a criação e manipulação de modelos, no computador, por um usuário. Ao longo do tempo, vários sistemas de modelagem surgiram com os mais diferentes propósitos. Sistemas CAD/CAM, por exemplo, tornaram-se viáveis a partir do surgimento de sistemas de modelagem poderosos, robustos e de fácil utilização (será?).

Um sistema de modelagem pode ser caracterizado do seguinte modo (fig. 7.1):

- pela sua interface com o usuário;
- por um conjunto de operações utilizadas para construir os modelos;
- pelo conjunto de técnicas de modelagem que irão implementar, efetivamente, as operações sobre os modelos.
- e por um esquema de representação interna (arquitetura);

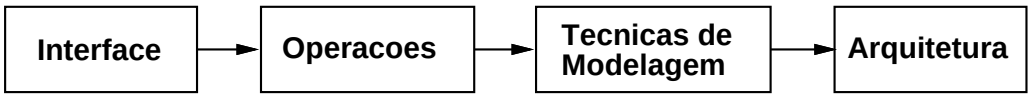



Figura 7.1: Modelo Conceitual.

## 7.1 Operações com Modelos

Existem, basicamente, três classes de operações que podem ser utilizadas na construção de um modelo:

- Operações Topológicas

- Operações de Combinação
  - Operações de Conjunto
  - Blending
- Operações Geométricas
  - Movimentos Rígidos
  - Deformações

A classe de operações topológicas é muito utilizada para construir representações por bordo, e deve garantir a consistência topológica do modelo resultante. Em geral, utilizam-se operadores de Euler que constroem modelos que satisfazem à fórmula de Euler. As operações topológicas costumam ser empregadas para "costurar" as diversas partes do modelo à medida que elas vão sendo criadas. Elas atuam, diretamente, sobre a topologia do modelo, construindo o seu grafo de adjacências.

As operações de combinação constroem um modelo a partir da combinação de dois outros modelos pré-existentes. As operações de conjunto utilizam as operações *booleanas* para construir modelos complexos a partir de modelos simples, que são instanciados e posicionados, apropriadamente, no espaço. Já as operações de blending são utilizadas para combinar dois objetos implícitos, por meio do blending (da ponderação) das funções implícitas que os definem. As operações de combinação são excenciais numa representação CSG.

As transformações geométricas atuam diretamente sobre a geometria dos modelos, reposicionando-os, através de movimentos rígidos, ou deformando-os com o uso de transformações de deformação.

## 7.2 Técnicas de Modelagem

O processo de construção de uma representação, a partir de uma sequência de dados fornecidos por um usuário comum, constitui o que se costuma chamar de uma técnica de modelagem. É intuitivo que existam técnicas de modelagem mais convenientes à construção de um certo tipo de representação.

A técnica de modelagem mais adequada a uma representação CSG é aquela que utiliza o paradigma de modelagem com massa. A partir de formas básicas que vão sendo combinadas, o usuário pode ir esculpindo o seu objeto. As ferramentas utilizadas são as operações *booleanas*, que costumam ser desgnidadas como operações CSG.

O conjunto de operações CSG pode ser visto como uma técnica de modelagem, que também pode ser empregada numa representação por bordo. Para representações baseadas em modelos paramétricos, no entanto, existem outras técnicas de modelagem que podem ser aplicadas na construção das suas partes. Desse modo, os pedaços de superfície vão sendo criados e colados uns aos outros, montando-se um grafo de adjacências. As principais técnicas são:

- superfícies de revolução;
- superfícies com forma livre;
- *lofting*;

- varredura (*sweep*).

  superfícies tubulares

  superfícies de extrusão

### 7.2.1 Revolução

Uma superfície de revolução é criada a partir de uma curva plana (possivelmente poligonal), chamada de perfil, que é rotacionada em torno de um eixo de revolução.

### 7.2.2 Formas Livre

Uma superfície com forma livre é esculpida através da manipulação de uma malha de controle poligonal, que define uma superfície algébrica descrita através dos seus coeficientes em relação a uma base qualquer, por exemplo, base de Bézier ou B-spline.

### 7.2.3 Lofting

A operação de *lofting* constrói a superfície interpolando as suas seções transversais ao longo de um eixo. Esta técnica é bastante utilizada na construção de imagens médicas. Um caso particular é aquele no qual se aplica uma translação aos vértices de uma das faces do modelo, criando um levantamento da face.

### 7.2.4 Varredura

A operação de *sweep*, ou varredura, cria uma superfície a partir do deslocamento de uma curva $\alpha$ contida em um plano $\pi$ que intersepta um caminho $\gamma$ transversalmente. Deslocando-se o plano $\pi$ ao longo do caminho e mantendo-se constante o ângulo entre a sua normal e o vetor tangente à curva $\gamma$, obtém-se uma superfície gerada pela curva $\alpha$, que é chamada de superfície de translação. A curva $\gamma$ é chamada de curva guia e a curva $\alpha$ é chamada de seção. Dois casos particulares importantes são as superfícies tubulares, onde o plano $\pi$ é ortogonal à curva $\gamma$ e a curva $\alpha$ é um círculo com centro em $\gamma$, e as superfícies de extrusão, onde a curva $\gamma$ é uma reta e o plano $\pi$ é ortogonal à $\gamma$.

## 7.3 Objetos Compostos

É muito comum, quando da criação do modelo de um objeto complexo, utilizarem-se composições de objetos mais simples. Pode-se definir então um objeto composto como sendo formado pelo agrupamento de dois ou mais objetos primitivos. O processo de criação de um objeto composto é bastante similar àquele empregado por uma criança que manipula um jogo de montagem de objetos a partir de algumas peças (blocos) básicas. No caso do computador, a montagem é feita a partir dos primitivos, que são instanciados e posicionados (por meio de transformações geométricas) no local apropriado. Este processo cria uma hierarquia de objetos, que pode ser representada por uma árvore binária. As folhas estão associadas à objetos primitivos e os nós internos correspondem à operações sobre objetos ou transformações geométricas.

A visualização de um objeto composto pode ser implementada através do percorrimento da árvore. Deve-se notar que as transformações geométricas devem ser acumuladas à medida que se avança no percurso. Pode-se imaginar que cada objeto primitivo está definido no seu próprio sistema de coordenadas e que as transformações geométricas executam uma mudança de sistema de coordenadas: para o sistema de coordenadas do objeto composto. Para isto, normalmente, utiliza-se uma pilha de transformações geométricas, onde a transformação (possivelmente composta) corrente encontra-se no topo. As operações definidas sobre esta pilha são as seguintes:

- empilhar — pushmatrix
- desempilhar — popmatrix
- multiplicar (compor) uma transformação à transformação corrente — multmatrix
- copiar uma transformação para o topo da pilha (torná-la corrente) — loadmatrix
- recuperar uma cópia da transformação corrente — getmatrix

Hierarquias são muito importantes também na criação de objetos articulados. Neste caso, cada ponto de articulação possui uma matriz de transformação associada, que irá agir sobre todos os sub-objetos influenciados pela articulação em questão.

## 7.4 Representação Interna

A análise feita no capítulo 6 indica que não existe um esquema de representação ideal. Se, por um lado, um esquema CSG apresenta facilidade de edição dos modelos, e de execução de operações *booleanas*, por outro lado, ele torna difícil o estudo das propriedades geométricas do modelo, bem como a definição de outras operações com os modelos. A representação B-rep, por sua vez, tem um controle muito bom sobre a geometria do modelo, porém apresenta dificuldade quanto a implementação das operações *booleanas*. Várias linhas de pesquisa, atualmente, indicam a direção de sistema mistos de representação. Os sistemas mistos se classificam em: híbridos e de multi-representação.

Nos sistemas híbridos, as várias representações do modelo convivem simultaneamente e tanto a criação quanto a manipulação do modelo podem ser feitas em qualquer das representações. Isso acarreta a necessidade de manter-se a consistência interna do modelo, de modo a que as operações feitas com uma representação se reflitam nas outras. Nesse caso, devem-se desenvolver algoritmos robustos, que permitam a passagem de um esquema de representação para outro. Em geral, esta é uma tarefa difícil e, às vezes, impossível.

Nos sistema de multi-representação, tem-se uma representação principal do modelo e várias representações secundárias. Estas últimas são utilizadas para resolver os problemas específicos que não afetam a estrutura do modelo. Todas as operações de modificação do modelo devem ser feitas na representação principal.

## 7.5 Exercícios

**7.1** *Desenvolva algoritmos para construir a representação por fronteira de objetos poliedrais usando as seguintes técnicas de modelagem:*

*a) Varredura (Sweep)*

*b) Extrusão*

*c) Revolução*

*Especifique os parâmetros de cada tipo de modelo.*

**7.2** *Especifique uma linguagem para a implementação de um Sistema de Modelagem CSG com:*

*a) Primitivas: esferas.*

*b) Operações: união, interseção e diferença.*

# Capítulo 8

# Visualização

A visualização é uma disciplina da Computação Gráfica que tem como objetivo produzir imagens, em um dispositivo de exibição, a partir de um modelo representado de alguma forma no computador. A grosso modo, e dependendo do modelo utilizado, tem-se a visualização bi-dimensional, a visualização de superfícies tri-dimensionais e a visualização volumétrica. Aqui, vai-se tratar apenas da visualização de superfícies tri-dimensionais.

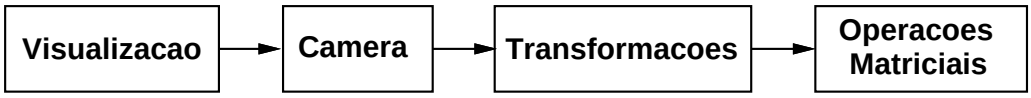



Figura 8.1: Modelo Conceitual.

Pode-se definir a visualização tri-dimensional como o processo de geração de imagens de uma cena tri-dimensional com um determinado grau de realismo. Essencialmente, este processo implica na solução de dois tipos de problemas distintos: a determinação, a partir de uma descrição da cena, das áreas visíveis de cada objeto na imagem final e o cálculo da função de iluminação correspondendo a estas áreas visíveis, para a obtenção da cor dos elementos da imagem (*shading*).

A função de iluminação está diretamente ligada à semântica dos dados e, em conseqüência, com as características de cada classe de aplicação. No caso mais geral da síntese de imagens foto-realistas, a função de iluminação se tratuz no fenômeno físico de reflexão e refração da luz. Já em outras aplicações, como por exemplo na área de engenharia, a função de iluminação pode servir para revelar propriedades intrínsecas dos objetos, tais como curvatura, tensão estrutural e temperatura.

A descrição da cena é contituída pela especificação dos elementos da cena e por uma transformação de visualização. Os elementos da cena são primordialmente os objetos a serem visualizados, mas incluem, também, os objetos que irão contribuir indiretamente no resultado final da imagem, como as fontes de luz. A transformação de visualização é especificada através de uma câmara virtual e do suporte de exibição.

O processo de visualização divide-se, basicamente, nas seguintes etapas (não necessariamente nesta ordem):

- Seleção das partes dos objetos contidas no campo de observação da câmara virtual (recorte).
- Cálculo da função de iluminação.
- Projeção das superfícies de cada objeto no plano da imagem e o mapeamento destas projeções em áreas correspondentes no suporte de exibição (transformação de visualização).
- Determinação das áreas visíveis dos objetos do ponto de vista da câmara virtual (determinação da visibilidade: ordenação e rasterização).
- Cálculo da função de colorização (amostragem).

As transformações de visualização, no seu conjunto, objetivam mapear as superfícies dos objetos em cena para o suporte de exibição. A sua principal finalidade é propiciar a execução mais eficiente das diversas etapas do processo de visualização. Para atingir-se este objetivo, os objetos devem ser transformados para sistemas de coordenadas (ou espaços de referência) nos quais se torne mais natural e conveniente a realização das tarefas inerentes a cada etapa do processo.

## 8.1 Espaços de Referência

Os espaços de referência são sistemas de coordenadas que estão associados da seguinte maneira às etapas da visualização:

- Espaço do Objeto - Modelagem;
- Espaço da Cena - Posicionamento;
- Espaço da Câmara - Iluminação;
- Espaço Normalizado - Recorte;
- Espaço da Ordenação - Visibilidade;
- Espaço da Imagem (ou da tela) - Rasterização.

A seguir, são descritos cada um desses espaços de referência e as suas propriedades.

**•Espaço do Objeto**
Corresponde aos sistemas de coordenadas nos quais os objetos primitivos estão descritos. Geralmente, o objeto está posicionado na origem, possui dimensões que são múltipos inteiros da unidade e está alinhado com os eixos principais, sendo chamado, neste caso, de um objeto padronizado. Note-se que é muito mais fácil manipular a geometria de um objeto padronizado, de forma a reorientá-lo, reposicioná-lo ou aplicar-lhe um fator de escala.

**•Espaço da Cena**
É um sistema de coordenadas global. Nele são posicionados e orientados, uns em relação aos outros, todos os elementos da cena, incluindo a câmara virtual. As dimensões estão normalmente em um padrão de escala natural da aplicação.

**•Espaço da Câmara**
É um espaço isométrico ao espaço da cena. A câmara está na origem do sistema de coordenadas, dois eixos principais são perpendiculares à direção de visão e o plano de projeção está sobre o terceiro eixo.

**•Espaço Normalizado**
É um sistema de coordenadas que privilegia o observador. A câmara está centrada na origem deste sistema. A pirâmide de visão é normalizada para o recorte.

**•Espaço de Ordenação**
É o sistema de coordenadas utilizado para a determinação da visibilidade. Esse sistema de coordenadas é obtido através de uma transformação projetiva do sistema de coordenadas normalizadas, de modo a trocar a projeção perspectiva pela projeção paralela ortográfica, preservando a planaridade, linearidade e ordenação dos objetos em relação à profundidade da cena. É importante manter a precisão em relação ao eixo $z$. Ângulos não são preservados. Ocorre uma compressão não linear na direção $z$.

**•Espaço da Imagem**
É o sistema de coordenadas do dispositivo. Algumas de suas características são: formato dos pixels, razão de aspecto, resolução espacial e de cor e o fator gama.

## 8.2 Especificação da Visualização

As transformações de visualização são especificadas através da câmara virtual e do suporte de exibição. A câmara virtual define os parâmetros de observação da cena no espaço tri-dimensional, enquanto que o suporte de exibição define os parâmetros relativos ao formato da imagem.

Há várias maneiras de se definir os parâmetros da câmara virtual. Aqui, vai-se adotar um modelo genérico, derivado do trabalho de Alvy Ray Smith na Lucas Film[1]. Neste modelo de visualização tri-dimensional, especifica-se um volume de visão, contra o qual os objetos são recortados, um tipo e um plano de projeção e uma *viewport* no suporte de exibição.

Existem diversos tipos de projeção cuja finalidade é transformar pontos de um espaço de dimensão $n$ em pontos de um espaço de dimensão menor do que $n$. No caso em questão, são pontos do espaço tri-dimensional sendo levados em pontos do espaço bi-dimensional. A classe de projeção que normalmente é utilizada em Computação Gráfica é conhecida como projeções geométricas planares, porque a projeção é sobre um plano e utiliza projetores retilíneos. Em cartografia é muito comum a utilização de outros tipos de projeção.

As projeções geométricas planares, daqui para frente simplesmente projeções, podem ser

[1] Transformation Tutorial Notes, Technical Memo no.78 - Lucasfilm Ltda - May 11,1983.

divididas em duas categorias: perspectiva e paralela (ortográfica ou oblíqua). A distinção é em relação a distância do centro de projeção ao plano de projeção. Se esta distância for finita tem-se a projeção perspectiva, caso contrário, a projeção paralela.

Contrastando com a projeção paralela, na projeção perspectiva linhas paralelas convergem, o tamanho do objeto é reduzido à medida que a sua distância ao centro de projeção aumenta e ocorre um encurtamento não uniforme das linhas do objeto em função da orientação e da distância do objeto ao centro de projeção. Todos estes efeitos ajudam na percepção de profundidade do sistema visual humano (embora a forma do objeto não seja preservada). Por este motivo, vão-se considerar aqui apenas as projeções perspectivas.

O plano de projeção é completamente arbitrário, sendo definido pela sua normal (VPN) e pela sua distância (View Distance) ao centro de projeção (COP). Uma vez determinado o plano de projeção, especifica-se sobre ele uma área retangular, chamada janela, que, juntamente com o COP, determina uma pirâmide semi-infinita. A janela é definida sobre o plano de projeção a partir de dois eixos ortogonais que, juntamente com o VPN, formam o sistema de coordenadas da câmara (VRC). A origem deste sistema é a projeção do COP no plano de projeção. Os dois eixos coordenados sobre o plano de projeção ficam determinados por um vetor, chamado VUP, cuja projeção sobre o plano de projeção determina, diretamente, o primeiro deles. Chamando o eixo indeterminado de $u$, o eixo na direção do VPN, de $n$ e o eixo na direção da projeção de VUP, de $v$, tem-se que $u = n \times v$, de forma que $u$, $v$ e $n$ formam um sistema de mão esquerda. A janela é especificada, então, por suas coordenadas mínimas e máximas ao longo de $u$ e $v$.

Para limitar o volume de visão, especifica-se um par de planos - frontal e traseiro, paralelos ao plano de projeção, distando $F$ (near) e $B$ (far) em relacão ao COP, ao longo da linha que une o COP ao centro da janela. Para que o volume de visão seja positivo, exige-se $F < B$. O volume de visão é então um tronco de pirâmide (fig. 8.2).

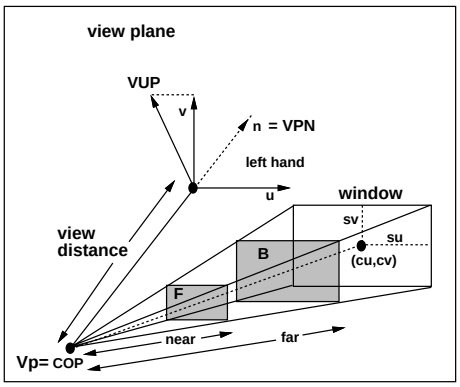



Figura 8.2: Sistema de Coordenadas da Câmara.

## 8.3 Transformações de Visualização

Uma vez especificados os parâmetros de visualização, uma série de transformações devem ser aplicadas aos objetos, de forma a mapeá-los para espaços mais apropriados a execução dos algoritmos de visualização. Estas transformações podem ser convenientemente realizadas por meio de coordenadas homogêneas e transformações projetivas.

Toda transformação linear bijetiva do $\Re^4$ induz, naturalmente, uma transformação no espaço projetivo (capítulo 5). Esta transformação é chamada de transformação projetiva induzida e é representada por uma matriz $4 \times 4$. Uma sequência de transformações pode ser agrupada, pela concatenação das matrizes individuais de cada transformação, em uma única matriz. Esta propriedade permite concentrar as transformações de visualização em matrizes que fazem o mapeamento de um espaço de referência para o outro.

Assumindo que os objetos já foram transformados para o espaço da cena, estes serão subseqüentemente transformados para o espaço da câmara (uma translação seguida de uma mudança de base).

- Translada-se o COP $(\ V_{px} \quad V_{py} \quad V_{pz} \quad 1\ )^T$ para a origem;
- Roda-se o VRC de forma a que o eixo $n$ coincida com o eixo $z$, que o eixo $u$ coincida com o eixo $x$ e que o eixo $v$ coincida com o eixo $y$ (troca-se o sistema de coordenadas: $x, y, z \rightarrow u, v, n$).

Isto é feito pela transformação representada pela matriz $V$, a seguir. A matriz $A$, não mostrada, representa a translação e a matriz $B$ faz a mudança de base, podendo ser facilmente determinada a partir da relação abaixo:

$$B\,(\ u \quad v \quad n\ ) = I \Rightarrow B = (\ u \quad v \quad n\ )^{-1} = (\ u \quad v \quad n\ )^T,$$

$$V = BA = \begin{pmatrix} u_x & u_y & u_z & -u.V_p \\ v_x & v_y & v_z & -v.V_p \\ n_x & n_y & n_z & -n.V_p \\ 0 & 0 & 0 & 1 \end{pmatrix};\quad V^{-1} = \begin{pmatrix} u_x & v_x & n_x & V_{px} \\ u_y & v_y & n_y & V_{py} \\ u_z & v_z & n_z & V_{pz} \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

O próximo passo é calcular a transformação de normalização para que o volume de visão seja transformado no volume canônico (pirâmide normalizada), definido pelos planos:

$$x = z,\quad x = -z,\quad y = z,\quad y = -z,\quad z = z_{min},\quad z = 1.$$

As etapas necessárias são:

- Aplicar um cizalhamento de forma a que o segmento de reta que une o COP ao centro da janela seja posicionado sobre o eixo $z$ (fig. 8.3).
- Aplicar fatores de escala, em relação aos três eixos, para que o volume de visão seja mapeado no volume canônico (fig. 8.4).

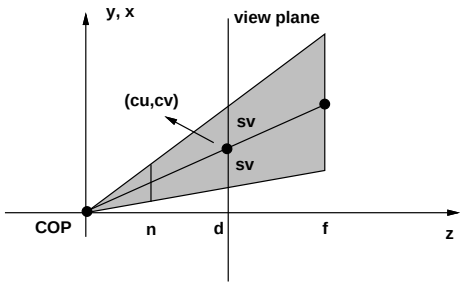



Figura 8.3: Cizalhamento.

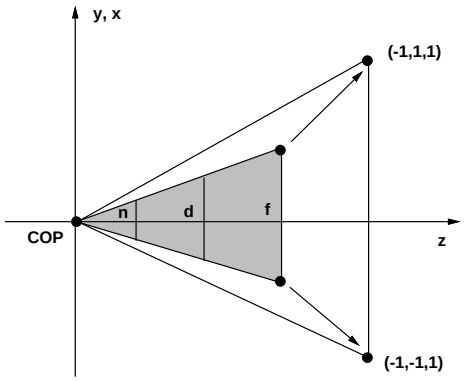



Figura 8.4: Escala.

A matriz $C$ faz o cizalhamento baseado nas seguintes relações:

$$x' = x - \frac{c_u}{d}z; \qquad y' = y - \frac{c_v}{d}z; \qquad z' = z.$$

Assim, o centro da janela fica sobre o eixo $z$: $(\, c_u \quad c_v \quad d\,) \rightarrow (\, 0 \quad 0 \quad d\,)$,

$$C = \begin{pmatrix} 1 & 0 & -c_u/d & 0 \\ 0 & 1 & -c_v/d & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

As equações dos planos que limitam a pirâmide são dadas, respectivamente, por:

$$x = \pm\frac{s_u}{d}z; \qquad y = \pm\frac{s_v}{d}z; \qquad z = 1.$$

A matriz $D$, abaixo, transforma os pontos da seguinte forma:

- $(\, (s_u/d)f \quad (s_v/d)f \quad f\,) \rightarrow (\, 1 \quad 1 \quad 1\,)$;

- $(\,-(s_u/d)f \quad (s_v/d)f \quad f\,) \rightarrow (\,-1 \quad 1 \quad 1\,)$;
- $(\,-(s_u/d)f \quad -(s_v/d)f \quad f\,) \rightarrow (\,-1 \quad -1 \quad 1\,)$;
- $(\,(s_u/d)f \quad -(s_v/d)f \quad f\,) \rightarrow (\,1 \quad -1 \quad 1\,)$;
- $(\,0 \quad 0 \quad n\,) \rightarrow (\,0 \quad 0 \quad z_{min}\,), \quad z_{min} = n/f$;
- $(\,0 \quad 0 \quad d\,) \rightarrow (\,0 \quad 0 \quad d/f\,)$;
- $(\,0 \quad 0 \quad f\,) \rightarrow (\,0 \quad 0 \quad 1\,)$.

$$D = \begin{pmatrix} d/(s_u f) & 0 & 0 & 0 \\ 0 & d/(s_v f) & 0 & 0 \\ 0 & 0 & 1/f & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

$N$ é a matriz resultante, que leva os pontos para o espaço normalizado.

$$N = DC = \begin{pmatrix} d/(s_u f) & 0 & -c_u/(s_u f) & 0 \\ 0 & d/(s_v f) & -c_v/(s_v f) & 0 \\ 0 & 0 & 1/f & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$N^{-1} = \begin{pmatrix} s_u f/d & 0 & c_u f/d & 0 \\ 0 & s_v f/d & c_v f/d & 0 \\ 0 & 0 & f & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

O último passo é encontrar a transformação projetiva da pirâmide normalizada no prisma retangular unitário:

$$-1 \leq x \leq 1, \quad -1 \leq y \leq 1, \quad 0 \leq z \leq 1.$$

Esta etapa é executada em três passos:

1) Translada-se $z_{min}$ para a origem (fig 8.5);

2) Escala-se na direção $z$ para que o plano traseiro coincida com o plano $z = 1$;

3) Aplica-se a transformação perspectiva (fig. 8.6):

$$\text{translação} = T_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -z_{min} \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

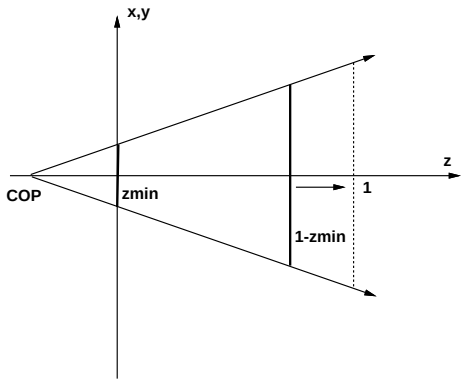



Figura 8.5: Transformação Perspectiva - Passo 1).

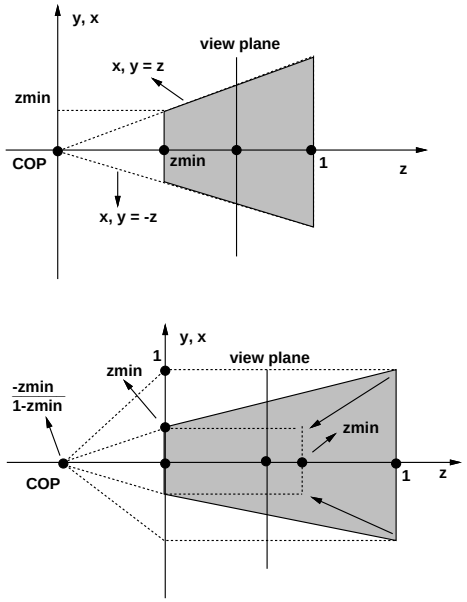



Figura 8.6: Transformação Perspectiva - Passos 2) e 3).

$$\text{escala} = T_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/(1-z_{min}) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$\text{perspectiva} = T_3 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & (1-z_{min})/z_{min} & 1 \end{pmatrix},$$

$$T_3T_2T_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/(1-z_{min}) & -z_{min}/(1-z_{min}) \\ 0 & 0 & 1/z_{min} & 0 \end{pmatrix}.$$

A matriz $P$, que leva os pontos para o espaço da ordenação, difere da matriz acima por um fator de escala $1/z_{min}$ em $x$, $y$ e $z$. Este fator de escala é necessário porque a transformação perspectiva empregada não altera os pontos sobre o plano $z = 0$, logo o tronco de pirâmide foi levado no prisma com dimensões $2z_{min}$, em $x$ e $y$, e $z_{min}$ em $z$.

$$S * T_3T_2T_1 = \begin{pmatrix} 1/z_{min} & 0 & 0 & 0 \\ 0 & 1/z_{min} & 0 & 0 \\ 0 & 0 & 1/(z_{min}(1-z_{min})) & -1/(1-z_{min}) \\ 0 & 0 & 1/z_{min} & 0 \end{pmatrix}.$$

Por fim, multiplicando-se esta última matriz por $z_{min}$ obtém-se a matriz $P$. Isto pode ser feito uma vez que a multiplicação de uma matriz que representa uma transformação projetiva por um escalar não altera a transformação (no final a divisão por $w$ "compensa"o efeito).

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/(1-z_{min}) & -z_{min}/(1-z_{min}) \\ 0 & 0 & 1 & 0 \end{pmatrix}; \quad P^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1-1/z_{min} & 1/z_{min} \end{pmatrix}.$$

Neste processo, a origem é levada para o infinito, ou seja, o COP passa a estar no infinito, de maneira que projetar neste espaço significa executar uma projeção paralela ortográfica (ignorar a coordenada $z$).

As transformações projetivas utilizadas nas etapas anteriores se caracterizavam por manter os pontos do espaço afim[2], e os pontos ideais[3], invariantes, ou seja, pontos do espaço afim eram transformados em pontos do espaço afim e pontos ideais em pontos ideais. No entanto, o último passo exige um tipo de transformação projetiva, chamada transformação perspectiva, que não possui esta característica de invariância (a imagem de pelo menos um ponto ideal é um ponto do espaço afim, o que cria um ponto de fuga). Um ponto $P = (x_n, y_n, z_n, 1)$ do espaço afim é transformado em um ponto $P' = (x, y, z, w)$ do espaço projetivo. Para obter o ponto correspondente do espaço afim é necessário dividir cada coordenada de $P'$ por $w$: $(x_o, y_o, z_o, 1) = (x/w, y/w, z/w, 1)$. Como $z_o$ não varia linearmente com $z_n$ — à medida que $z_n$ se aproxima do plano traseiro, $z_o$ se aproxima de 1 mais rapidamente — surgem problemas quando se deseja interpolar quantidades, afora posições, no espaço de Ordenação (por exemplo, na colorização de Gouraud, onde é feita uma interpolação de intensidades).

Como já foi dito, sem justificativa, o espaço de Ordenação não é o espaço no qual a função de iluminação deve ser avaliada. Isto porque, esta avaliação envolve, entre outras coisas, o produto escalar entre a normal e a direção da fonte de luz, no ponto a ser iluminado.

[2] Pontos da forma $(x, y, z, 1)$.
[3] Pontos da forma $(x, y, z, 0)$.

Uma vez que as transformações de visualização não são isométricas (envolvem cizalhamento, fatores de escala distintos em cada direção e perspectiva), os ângulos não são preservados e, conseqüentemente, o produto escalar também não. Por este motivo, avalia-se a função de iluminação no espaço da Cena, ou em qualquer espaço isométrico a ele, por exemplo, o espaço da Câmara. Na colorização de Phong, por exemplo, é necessário fazer uma interpolação de normais em cada pixel. Neste caso, é aplicada a transformação inversa, de forma a mapear o ponto de volta ao espaço da Câmara e alí avaliar a função de iluminação.

Um questão importante é como aplicar transformações projetivas à normais. Considerando o caso simples em que a superfície é poligonal, isto significa transformar a normal a um plano. Sendo $N = (\ A \quad B \quad C \quad D\ )$, um plano é definido como o conjunto dos pontos $P = (\ x \quad y \quad z \quad 1\ )$ tais que $N.P = 0$ ou, na forma de matriz, $N^T.P = 0$. Supondo-se que todos os pontos do plano foram transformados por uma matriz $M$, para manter $N^T.P = 0$ para os pontos transformados, deve-se transformar $N$ por uma matriz $Q$, a ser determinada, tal que: $(Q.N)^T.M.P = 0$. Reescrevendo esta expressão como $N^T.Q^T.M.P = 0$, conclui-se que $Q^T.M$ deve ser um múltiplo da matriz identidade. Se o fator multiplicativo for 1, chega-se a $Q = (M^{-1})^T$, significando que o vetor normal $N^{'}$ do plano transformado por $M$ é dado por: $N^{'} = (M^{-1})^T.N$. Esta expressão tem aplicação importante em modelagem geométrica, onde é comum aplicarem-se deformações aos objetos e as normais (que costumam fazer parte do banco de dados) devem sofrer as mesmas deformações.

### 8.3.1 Mapeamento para Viewport

O programador especifica uma *viewport* 3D, contida no espaço de coordenadas do dispositivo,

$$0 \leq x \leq X_{max}, \quad 0 \leq y \leq Y_{max}, \quad 0 \leq z \leq Z_{max},$$

na qual o volume de visão é mapeado.
Dada uma *viewport* 3D, o mapeamento do volume de visão, para as coordenadas do dispositivo, é feito em quatro etapas[4] (fig. 8.7):

1) Translada-se o volume de visão canônico de forma a que o vértice $(\ -1 \quad -1 \quad 0 \quad 1\ )$ vá para a origem e aplica-se um fator de escala de 0.5 em $x$ e $y$.

2) Escala-se o volume para as dimensões da *viewport* 3D.

3) Translada-se o novo volume para o canto inferior esquerdo da *viewport* 3D.

4) Arredonda-se para o pixel virtual mais próximo.

A matriz $K$ faz a translação e a escala inicial:

$$K = \begin{pmatrix} 0.5 & 0 & 0 & 0.5 \\ 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

[4] Em alguns dispositivos, as coordenadas $(0, 0)$ correspondem ao canto superior esquerdo da tela, devendo-se, neste caso, fazer $Y_{max} - y$.

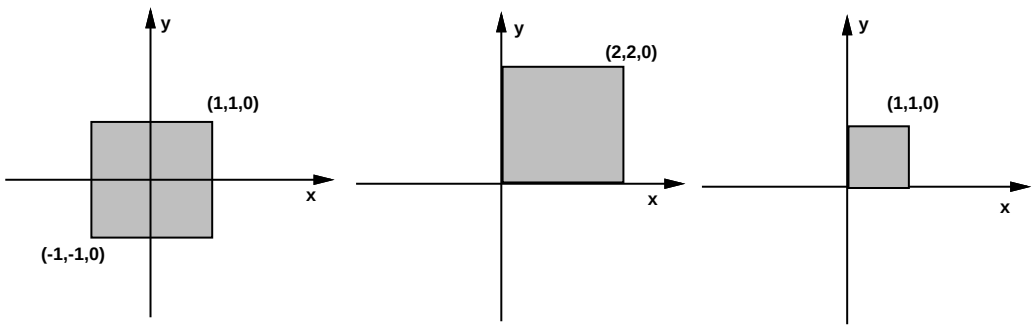



Figura 8.7: Mapeamento para Viewport.

A matriz $L$ executa o mapeamento de NDC para as coordenadas do dispositivo:

$$L = \begin{pmatrix} X_{max} - X_{min} & 0 & 0 & X_{min} \\ 0 & Y_{max} - Y_{min} & 0 & Y_{min} \\ 0 & 0 & Z_{max} - Z_{min} & Z_{min} \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

A matriz $M$ executa o arredondamento:

$$M = \begin{pmatrix} 1 & 0 & 0 & 0.5 \\ 0 & 1 & 0 & 0.5 \\ 0 & 0 & 1 & 0.5 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

A matriz $D$ é a matriz resultante, onde $\Delta_i = (i_{max} - i_{min})$ e $\nabla_i = (i_{max} + i_{min})$.

$$D = MLK = \begin{pmatrix} \Delta_x/2 & 0 & 0 & (\nabla_x + 1)/2 \\ 0 & \Delta_y/2 & 0 & (\nabla_y + 1)/2 \\ 0 & 0 & \Delta_z & Z_{min} + 0.5 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$D^{-1} = \begin{pmatrix} 2/\Delta_x & 0 & 0 & -(\nabla_x + 1)/\Delta_x \\ 0 & 2/\Delta_y & 0 & -(\nabla_y + 1)/\Delta_y \\ 0 & 0 & 1/\Delta_z & -(Z_{min} + 0.5)/\Delta_z \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Esta é a última etapa nas transformações de visualização. O mapeamento completo, do espaço da cena para o espaço da imagem, é dado então pela matrix $DPNV$. Após aplicar-se esta matriz a um ponto, basta truncar as coordenadas resultantes para obterem-se as coordenadas inteiras de tela correspondentes (isto pode ser feito, em C, pela função floor). O mapeamento inverso, do espaço da imagem para o espaço da cena, é dado pela matrix $V^{-1}N^{-1}P^{-1}D^{-1}$.

## 8.4 Exercícios

**8.1** *Quais são as principais operações de visualização?*

**8.2** *Enumere os espaços de referência para especificação das operações de visualização e descreva as suas características. Explique por que eles são adequados às respectivas operações.*

**8.3** *Descreva os parâmetros que especificam uma câmara virtual.*

**8.4** *Discuta o processamento das transformações de visualização para modelos paramétricos e implícitos.*

**8.5** *Mostre que no caso da projeção paralela a matriz $PN$ deve ser substituída pela matriz $O$ dada abaixo:*

$$O = \begin{pmatrix} 1/s_u & 0 & 0 & -c_u/s_u \\ 0 & 1/s_v & 0 & -c_v/s_v \\ 0 & 0 & 1/(f-n) & -n/(f-n) \\ 0 & 0 & 0 & 1 \end{pmatrix}; \quad O^{-1} = \begin{pmatrix} s_u & 0 & 0 & c_u \\ 0 & s_v & 0 & c_v \\ 0 & 0 & f-n & n \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

**8.6** *Um plano de recorte traseiro no infinito acarreta problemas nas transformações apresentadas neste capítulo. Por exemplo, a matrix $N$ fica com todos os elementos nulos exceto $N[4][4] = 1$ (o que isso significa?). Neste caso, é necessário que se defina um novo volume de visão canônico para o recorte. Recomenda-se deixar o plano traseiro no infinito, ao invés de mapeá-lo para o plano $z = 1$, mapeando-se o plano de visão $z = d$ no plano $z = 1$.*

*a) Derive um mapeamento que leve $z_{min}$ em 0 e deixe $z = 1$ inalterado ($z_{min} = n/d$), em seguida mapeie o plano traseiro (no infinito) no plano $z = d/(d-n)$. Então, pré-multiplicando por $(d-n)/d$, obtenha:*

$$N_\infty = \begin{pmatrix} 1/s_u & 0 & -c_u/(s_u d) & 0 \\ 0 & 1/s_v & -c_v/(s_v d) & 0 \\ 0 & 0 & 1/d & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}; \quad P_\infty = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -n/d \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

*b) No caso da projeção paralela, derive o mapeamento que leva o plano frontal no plano $z = 0$ e o plano de visão no plano $z = 1$, para obter:*

$$O_\infty = \begin{pmatrix} 1/s_u & 0 & 0 & -c_u/s_u \\ 0 & 1/s_v & 0 & -c_v/s_v \\ 0 & 0 & 1/(d-n) & -n/(d-n) \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

# Capítulo 9

# Recorte

O recorte é uma operação muito importante no contexto da Computação Gráfica devido à variedade de suas aplicações. Uma definição precisa de recorte é a seguinte:
**Definição:** Dada uma superfície $M$ fechada de co-dimensão 1 do $\Re^n$, o complemento de $M$, $\Re^n - M$, possui duas componentes conexas. Se $S$ é um subconjunto do $\Re^n$, chama-se de recorte de $S$ por $M$ à operação que consiste em determinar os subconjuntos de $S$ que estão em cada uma das componentes conexas de $M$ (fig. 9.1).

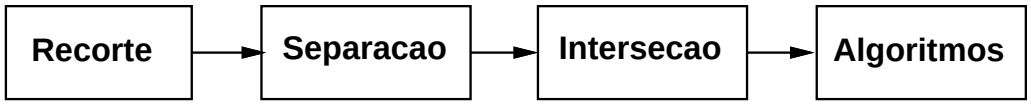



Figura 9.1: Modelo Conceitual.

Quando $n = 3$, $M$ é uma superfície fechada do $\Re^3$ e diz-se que o recorte é tri-dimensional. Qundo $n = 2$, $M$ é uma curva fechada do $\Re^2$ e diz-se que o recorte é bi-dimensional. A maioria das aplicações de Computação Gráfica usa recorte bi-dimensional, onde $S$ é uma curva poligonal ou simplesmente um retângulo, ou recorte tri-dimensional, onde a superfície $S$ é um poliedro, um prisma retangular ou uma pirâmide.

Dentre as aplicações do recorte uma das mais importantes, pelo papel que desempenha no processo de visualização, é a determinação dos objetos contidos na pirâmide de visão. Esta operação precisa ser realizada com a maior eficiência nos algoritmos de visualização, para uma classe bem definida de primitivas geométricas. Os únicos algoritmos que não utilizam o recorte explicitamente são aqueles baseados no traçado de raios (*ray tracing*). Outras aplicações do recorte dizem respeito às áreas de visibilidade, realismo, simulação, seleção (*pick*) de primitivas, modelagem e sistemas interativos.

Na área dos algoritmos de visibilidade, pode-se citar o método de Weiler-Atherton que utiliza o recorte recursivo de polígonos para resolver o problema da visibilidade. Vários algoritmos para a visualização de sombras se resumem, essencialmente, em operações de recorte e projeção.

O recorte pode ser empregado como um método padrão para implementar a operação de *pick*. Para isto, um pequeno retângulo circundando o cursor, chamado janela de *pick*, é usado para determinar que primitivas estão na vizinhança do cursor. A maioria das primitivas são trivialmente rejeitadas e, dentre as aceitas, aquela de mais alta prioridade é a selecionada.

Na modelagem sólida, o recorte é empregado de maneiras diversas, dentre as quais se destaca o seu uso em operações CSG. Sistemas de interface com o usuário baseados em janela utilizam recorte no processo de superposição de várias janelas.

O recorte aplicado aos algoritmos de visibilidade tem como finalidade primária eliminar todos os objetos fora do campo de visão da câmara virtual. Essa operação se faz necessária por motivo de eficiência e, principalmente, para evitar que objetos atrás do centro de projeção sejam mapeados erroneamente no plano de projeção.

Existem outras técnicas que, em alguns casos, podem ser aliadas ao recorte na execução desta tarefa. Um exemplo é a eliminação de polígonos de superfícies fechadas, orientados na direção oposta a da câmara virtual que, por este motivo, não são visíveis.

Os algoritmos de recorte podem operar com uma configuração específica ou arbitrária. No primeiro caso, a geometria da superfície de recorte é pré-definida, de modo a simplificar a execução do algoritmo. As transformações de visualização deformam a pirâmide de visão com este intuito.

Dependendo do sistema de visualização, o recorte pode operar com tipos diferentes de dados geométricos. Os tipos mais comuns são segmentos de reta e polígonos. Outras primitivas geométricas, tais como superfície algébricas e paramétricas, requerem um processo mais complexo de recorte e podem, alternativamente, ser aproximadas por polígonos, antes da visualização.

O método empregado no recorte pode envolver uma solução direta ou recursiva. No primeiro caso, a interseção é calculada analiticamente, enquanto que, no segundo, o objeto a ser recortado é subdividido recursivamente, até que as suas partes estejam inteiramente dentro ou fora do volume de recorte.

O cálculo do recorte pode ser exato ou aproximado. Muitas vezes, não é necessário um cálculo preciso da interseção. Métodos recursivos de recorte, geralmente, operam dentro de uma determinada tolerância.

## 9.1 Recorte de Segmentos de Reta

O algoritmo mais conhecido de recorte foi criado por Cohen & Sutherland. A partir dele, vários outros algoritmos foram desenvolvidos visando uma maior eficiência. Esta classe de algoritmos utiliza uma região de recorte retangular, alinhada com os eixos principais do sistema de coordenadas.

O algoritmo de Cohen-Sutherland é baseado na classificação das extremidades do segmento de reta, determinando, em primeiro lugar, se o segmento pode ser trivialmente aceito ou rejeitado. Caso não possa, ele é recortado em uma das arestas do retângulo, sendo dividido em dois segmentos, sendo, pelo menos um, trivialmente rejeitado. Assim, o segmento é recortado iterativamente sempre testando-se a sua trivial aceitação ou rejeição, havendo uma subdivisão caso o teste falhe, até que esteja completamente dentro do retângulo de recorte ou seja rejeitado. O algoritmo é particularmente eficiente no caso de um retângulo

muito grande que contém quase todos os segmentos, ou no caso de um retângulo muito pequeno, onde ocorre a situação inversa (quase todos os segmentos são rejeitados).

O grande mérito do algoritmo está na forma de testar se um segmento pode ser trivialmente aceito ou rejeitado. Para isto, no caso bi-dimensional, as arestas do retângulo são prolongadas, dividindo o plano em nove regiões, conforme explicitado na figura 9.2. Cada uma dessas regiões possui um código de 4 bits, determinado pelo lado em que a região está em relação aos semi-planos criados pelas arestas do retângulo (1 significa que a região não está no mesmo semi-plano do retângulo de recorte).

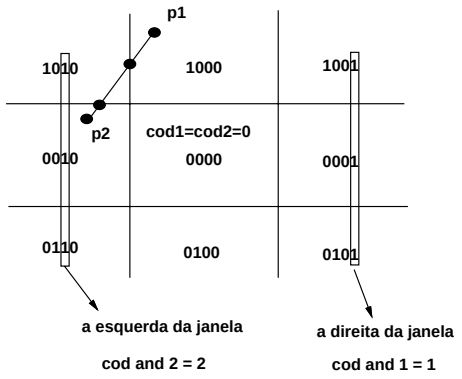



Figura 9.2: Regiões no Recorte.

Supondo-se a ordem aresta superior, inferior, esquerda e direita, a extremidade $(x, y)$ do segmento pode ser classificada testando se:

$$y > y_{max}, \quad y < y_{min}, \quad x < x_{min}, \quad x > x_{max}.$$

Um código 0001 significa que o ponto está abaixo da aresta superior, acima da aresta inferior, à direita da aresta esquerda e à direita da aresta direita. Se o código de ambas as extremidades for 0000 significa que o segmento pode ser trivialmente aceito. Se a operação lógica AND entre os bits correspondentes de cada código não resultar em 0000, o segmento pode ser trivialmente rejeitado:

$$(c_1 \ \& \ c_2) \ != 0 \rightarrow \text{ rejeito,}$$

$$(c_1 == 0 \ \&\& \ c_2 == 0) \rightarrow \text{ aceito.}$$

A interseção de um segmento com a reta $y = \pm Y$ pode ser facilmente calculada parametricamente:

$$y = (1-t)P_1 + tP_2 \ \text{ ou } \ y = P_1.y + t(P_2.y - P_1.y) \Rightarrow t_Y = \pm\frac{Y - P_1.y}{(P_2.y - P_1.y)}.$$

Assim, $y = \pm Y$ e $x = P_1.x + t_Y(P_2.x - P_1.x)$. Os códigos de operação também são facilmente determinados:

$$cod = (P_1.y > Y) \ ? \ 8 \ : \ ((P_1.y < -Y) \ ? \ 4 \ : \ 0);$$

$$cod \wedge = (P_1.x > X) \ ? \ 1 \ : \ ((P_1.x < -X) \ ? \ 2 \ : \ 0).$$

O algoritmo de Cohen-Sutherland pode ser aplicado ao caso tri-dimensional. Neste caso, tem-se uma superfície de recorte que é um paralelepípedo ou um tronco de pirâmide. Cada face é prolongada dando origem a 27 regiões. O código passa a ter 6 bits. No máximo seis interseções são calculadas, uma para cada lado do volume de visão.

É possível adotar-se um único algoritmo de recorte tanto para a projeção paralela, como para a perspectiva, desde que, no caso da projeção perspectiva, o recorte seja feito após a transformação perspectiva, em coordenadas homogêneas. Além da vantagem óbvia de poder ser utilizado um único algoritmo (otimizado) de recorte, certas transformações projetivas (pouco usadas) e *splines* racionais podem produzir um valor negativo para a coordenada homogênea $w$. Neste caso, apenas o recorte em coordenadas homogêneas irá produzir o resultado correto. O volume de visão canônico da projeção paralela é dado por:

$$-1 \leq x_o \leq 1, \quad -1 \leq y_o \leq 1, \quad 0 \leq z_o \leq 1.$$

As inequações correspondentes em coordenadas homogêneas são:

$$-1 \leq \frac{x}{w} \leq 1, \quad -1 \leq \frac{y}{w} \leq 1, \quad 0 \leq \frac{z}{w} \leq 1,$$

que podem ser reescritas como:

$$-w \leq x \leq w, \quad -w \leq y \leq w, \quad 0 \leq z \leq w : \ w > 0,$$

$$-w \geq x \geq w, \quad -w \geq y \geq w, \quad 0 \geq z \geq w : \ w < 0.$$

No caso do recorte de segmentos de reta, somente o caso $w > 0$ (normalmente $w = 1$) interessa, uma vez que antes da transformação perspectiva todos os pontos possuem $w > 0$.

## 9.2 Recorte de Polígonos

O algoritmo de recorte de polígonos tri-dimensionais, de Sutherland-Hodgman, foi um marco no desenvolvimento da Computação Gráfica. Até então, os algoritmos de recorte de polígonos eram estruturados de maneira semelhante aos algoritmos de recorte de segmentos de reta. Cada aresta era analisada em relação à região de recorte como um todo. Isto acarretava na perda da conectividade do polígono recortado. A solução deste problema topológico foi inverter a ordem das operações, considerando a interseção de cada plano de recorte em relação ao polígono como um todo. O problema deste algoritmo é que ele só funciona se a região de recorte for convexa e as partes desconexas do polígono resultante são ligadas por um segmento de reta.

Um algoritmo genérico de recorte de polígonos contra polígonos foi desenvolvido por Weiler e utiliza uma estrutura de dados baseada em topologia. O propósito deste algoritmo é permitir a sua utilização na implementação de operações CSG.

## 9.3 Exercícios

**9.1** *Qual o propósito do recorte na visualização?*

**9.2** *Discuta os principais conceitos envolvidos no recorte.*

**9.3** *Enumere os tipos básicos de algoritmos de recorte.*

**9.4** *Explique o esquema de classificação de pontos do algoritmo de Sutherland-Cohen.*

**9.5** *Compare as estratégias do algoritmo de recorte de linhas de Sutherland-Cohen contra o algoritmo de recorte de polígonos de Sutherland-Hodgman.*

**9.6** *No contexto de um sistema de visualização tri-dimensional, discuta as vantagens do uso de um algoritmo de recorte 3D para os objetos contra um recorte 2D aplicado às projeções dos objetos.*

**9.7** *Crie um algoritmo de recorte (e implemente-o) que ao invés de aceitar as partes dos segmentos de reta contidas na janela faz o oposto; aceita as partes dos segmentos fora da janela. Dê uma aplicação deste novo algoritmo.*

**9.8** *Crie um algoritmo de recorte especializado em:*

*a) Retângulos alinhados com os eixos coordenados.*

*b) Retângulos genéricos.*

*c) Circunferências.*

*d) Triângulos.*

**9.9** *Implemente o algoritmo de recorte, em coordenadas homogêneas, para segmentos de reta.*

# Capítulo 10

# Rasterização

A descrição dos objetos em cena é dada, geometricamente, em uma forma vetorial (pontos, polígonos, curvas e superfícies paramétricas ou implícitas, etc.). Esses objetos definem superfícies e curvas que compõem a partição da tela virtual em regiões visíveis, nas quais a função de iluminação será calculada.

O problema que se coloca é a conversão de dados no formato vetorial para o formato matricial — que é o formato aceito por grande parte dos dispositivos gráficos — o que, matematicamente, corresponde à discretizaçao da função de iluminação a partir da partição uniforme da tela virtual em um certo número finito de elementos chamados *pixels*, com o entuito de obter-se uma representação finita. A rasterização enumera então os pontos de uma região digital da tela virtual, associada às regiões visíveis da cena, que correspondem aos pontos nos quais a função de iluminação deve ser amostrada.

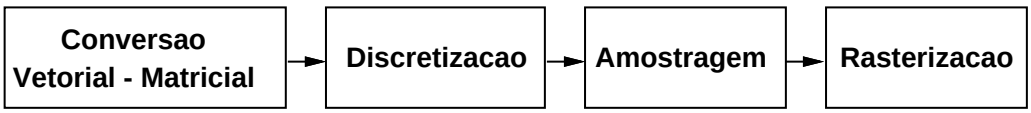



Figura 10.1: Modelo Conceitual.

O processo de rasterização consiste, basicamente, em uma enumeração da região digital correspondente a essas regiões visíveis (fig. 10.2).

A forma de rasterização define a estrutura computacional do processo de enumeração dos elementos da imagem (pixels). Ela pode ser incremental, por subdivisão ou analítica:

- A rasterização incremental possui uma etapa de inicialização — onde são feitos os cálculos dos valores iniciais e dos incrementos das funções associadas à geometria do objeto — e uma fase de iteração, que consiste no cálculo incremental dessas funções, fornecendo os valores relativos a cada pixel enumerado por esse processo.
- A rasterização por subdivisão consiste, essencialmente, na divisão recursiva da su-

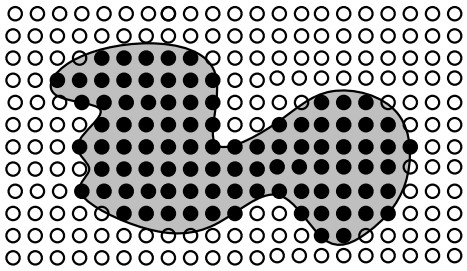


Figura 10.2: Processo de Enumeração.

perfície, o que, normalmente, é feito em quatro etapas: determinação da projeção da superfície no plano da imagem; teste de parada da subdivisão; divisão da superfície; e enumeração dos pixels. Como exemplo, cita-se o algoritmo para a determinação da visibilidade de superfícies, de Warnock.

- A rasterização analítica tem uma estrutura algoritmica bastante simples: para todo pixel potencialmente contido na projeção da superfície na tela virtual, calcula-se a interseção da superfície com um raio que parte do centro de projeção e passa pelo pixel em questão. Se a interseção não for vazia esse elemento será ocupado. O cálculo da interseção se reduz ao uso de algum método numérico que determine o ponto onde uma semi-reta intercepta a superfície do objeto[1].

A rasterização estabelece uma relação entre os objetos da cena e as regiões da imagem digital. Esse processo é, essencialmente, geométrico. Entretanto, para garantir resultados corretos, precisa-se levar em conta o método que será utilizado para se obter a amostragem da função de intensidade de cor.

## 10.1 Rasterização de Elementos Lineares

A rasterização de elementos lineares (segmentos de reta, polígonos, etc.) é bastante utilizada em Computação Gráfica, pois grande parte dos modelos utilizam uma representação B-rep poligonal. O algoritmo pioneiro dessa família foi introduzido por Bresenham, para a rasterização de um segmento de reta.

Os algoritmos de rasterização de elementos lineares empregam, principalmente, formas incrementais de rasterização.

### 10.1.1 Rasterização de Segmentos de Reta

Os algoritmos para rasterização de segmentos computam as coordenadas dos pixels que estão sobre, ou próximos a, uma linha ideal, infinitamente fina, restringida a um reticulado 2D na resolução do dispositivo.

[1] No caso de superfícies algébricas, o problema se reduz, após a substituição, à determinação dos zeros de um polinômio de uma variável.

A abordagem mais simples consiste em calcular a inclinação $m = dy/dx$ para — incrementando $x$ de uma unidade, a partir do ponto inicial — calcular, para cada $x_i$, o valor de $y = mx_i + B$ e acender o pixel na posição $(x_i, \text{round}(y_i))$. Este método escolhe, em cada passo, o pixel mais próximo à linha, ou seja, aquele cuja distância à linha é menor. O seu incoveniente é que a cada iteração são necessárias operações de ponto flutuante: multiplicação, adição e chamada à função round. A multiplicação pode ser eliminada, notando-se que (fig. 10.3):

$$y_{i+1} = mx_{i+1} + B = m(x_i + \Delta x) + B = y_i + m\Delta x,$$

e se $\Delta x = 1$, então $y_{i+1} - y_i = m$. Isto significa que um incremento em $x$ de uma unidade muda $y$ de $m$, que é a inclinação da linha. Para todos os pontos $(x_i, y_i)$ sobre a linha, tem-se que se $x_{i+1} = x_i + 1$, então $y_{i+1} = y_i + m$[2].

A inicialização corresponde a atribuir os valores inteiros da extremidade esquerda a $(x_0, y_0)$. Se $|m| > 1$, um passo unitário em $x$ cria um passo $> 1$ em $y$. Neste caso, os papéis de $x$ e $y$ devem ser invertidos, ou seja, $y$ é incrementado de uma unidade e $x$ por $\Delta x = 1/m$.

Na implementação deste algoritmo, as variáveis $m$ e $y$ devem ser reais, porque a inclinação é uma fração. Além do mais, a avaliação da função round toma tempo.

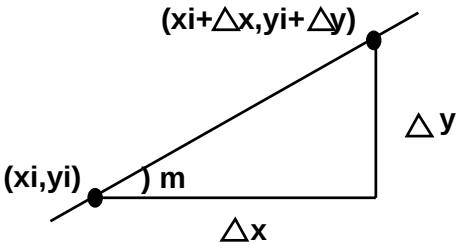



Figura 10.3: Coeficiente Angular de um Segmento.

O algoritmo do ponto médio visa eliminar estes problemas. Ele parte do princípio de que, uma vez selecionado o pixel $P = (x_p, y_p)$, a escolha do próximo pixel se restringe entre o pixel uma coluna para a direita, na mesma linha, chamado $E$, e o pixel uma coluna para a direita e uma linha para cima, chamado $NE$. Sendo $Q$ o ponto de interseção do segmento com a linha do reticulado em $x = x_p + 1$, o algoritmo verifica de que lado o ponto médio $M = (x_p + 1, y_p + 1/2)$ está. Se estiver acima do segmento, escolhe-se $E$. Se estiver abaixo, escolhe-se $NE$, conforme explicitado na figura 10.4.

Representando-se a linha de forma implícita, tem-se:

$$F(x, y) = ax + by + c = 0.$$

Na forma explícita, com inclinação $0 \leq dy/dx \leq 1$, escreve-se:

$$y = \frac{dy}{dx}x + B, \qquad dy = y_1 - y_0 > 0, \qquad dx = x_1 - x_0 > 0,$$

[2]Os valores de $x$ e $y$ estão definidos em termos dos seus valores anteriores.

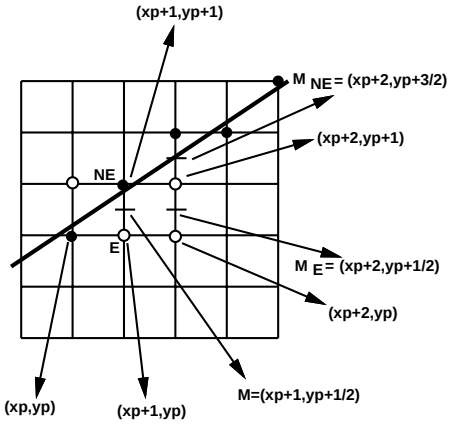



Figura 10.4: Algoritmo do Ponto Médio.

ou $xdy - ydx + Bdx = 0$. Igualando-se os termos, obtém-se: $a = dy$, $b = -dx$ e $c = Bdx$. Um ponto $y$, acima do segmento, comparado com o ponto $y' = (dy/dx)x + B$, sobre o segmento, fornece:

$$y > y' = \frac{dy}{dx}x + B \qquad \text{ou} \qquad ydx > xdy + Bdx \Rightarrow 0 > xdy - ydx + Bdx,$$

ou seja: $F(x, y) = xdy - ydx + Bdx < 0$ para pontos acima do segmento. Para pontos sobre o segmento, $F(x, y) = 0$ e para pontos abaixo do segmento, $F(x, y) > 0$. A figura 10.5 dá uma interpretação geométrica, considerando o gráfico da função $F(x, y)$ no $\Re^3$ e a reta definida implicitamente como a curva de nível $F(x, y) = 0$.

Para aplicar o critério do ponto médio, é necessário calcular

$$F(M) = F(x_p + 1,\ y_p + 1/2)$$

e testar o sinal. Definindo-se uma variável

$$d = F(x_p + 1,\ y_p + 1/2) = a(x_p + 1) + b(y_p + 1/2) + c,$$

escolhe-se o pixel $NE$, se $d > 0$ e o pixel $E$, se $d < 0$. Se for escolhido $NE$, a variável $d$ deve ser avaliada, na próxima iteração, no ponto $(x_p + 2,\ y_p + 3/2)$:

$$d_{NE} = F\left(x_p + 2,\ y_p + \frac{3}{2}\right) = a(x_p + 2) + b\left(y_p + \frac{3}{2}\right) + c.$$

Subtraindo o valor anterior de $d$ de $d_{NE}$, obtém-se:

$$d_{NE} = d_{old} + a + b = d_{old} + dy - dx.$$

Se for escolhido o pixel $E$, a variável $d$ deve ser avaliada no ponto $(x_p + 2,\ y_p + 1/2)$:

$$d_E = F\left(x_p + 2,\ y_p + \frac{1}{2}\right) = a\left(x_p + 2\right) + b\left(y_p + \frac{1}{2}\right) + c.$$

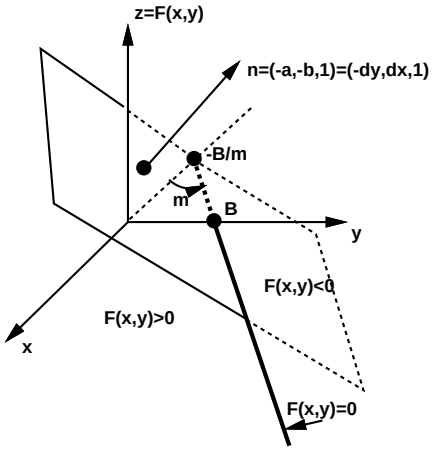



Figura 10.5: Gráfico da função $F(x, y)$.

Subtraindo o valor anterior de $d$ de $d_E$, obtém-se:

$$d_E = d_{old} + a = d_{old} + dy.$$

Em ambos os casos, o valor de $d$, no próximo passo, pode ser obtido incrementalmente, a partir do seu valor no passo corrente. O valor inicial de $d$ é calculado a partir do ponto inicial $(x_0, y_0)$[3]:

$$d_0 = F\left(x_0 + 1,\ y_0 + \frac{1}{2}\right) = a\,(x_0 + 1) + b\left(y_0 + \frac{1}{2}\right) + c$$

$$= ax_0 + by_0 + c + a + \frac{b}{2} = F(x_0, y_0) + a + \frac{b}{2} = a + \frac{b}{2} = dy - \frac{dx}{2}.$$

Para evitar a divisão por 2, redefine-se a função $F$ original:

$$F(x, y) = 2(ax + by + c).$$

Isto multiplica as constantes e a variável $d$ por 2, mas não altera o sinal de $d$.

- $d_0 = 2dy - dx$,
- $d_E = 2dy$, se $d \leq 0$;
- $d_{NE} = 2dy - 2dx$, se $d > 0$.

Tudo o que foi feito, até agora, supunha que $0 \leq dy/dx \leq 1$. Se $dy/dx > 1$, os papéis de $x$ e $y$ são invertidos. Dado que foi escolhido o pixel $P = (x_p, y_p)$, a escolha do próximo pixel se restringe entre o pixel uma linha para cima, na mesma coluna, chamado $N$, e o pixel uma linha para cima e uma coluna para direita, chamado $NE$. Sendo $Q$ o ponto de interseção

[3] O ponto $(x_0, y_0)$ está sobre a linha, logo $F(x_0, y_0) = 0$.

do segmento com a coluna do reticulado em $y = y_p + 1$, o algoritmo verifica de que lado o ponto médio $M = (x_p + 1/2, y_p + 1)$ está. Se estiver à direita do segmento, escolhe-se $N$. Se estiver à esquerda, escolhe-se $NE$. Um ponto $x$, à direita do segmento, comparado com o ponto $x' = (dx/dy)y - B$, sobre o segmento, fornece:

$$x > x' = \frac{dx}{dy}y - B \qquad \text{ou} \qquad xdy > ydx - Bdx \Rightarrow 0 < xdy - ydx + Bdx,$$

ou seja: $F(x, y) = xdy - ydx + Bdx > 0$ para pontos à direita do segmento. Para pontos sobre o segmento, $F(x, y) = 0$ e para pontos à esquerda do segmento, $F(x, y) < 0$.

Se for escolhido $NE$, a variável $d$ deve ser avaliada, na próxima iteração, no ponto $(x_p + 3/2,\ y_p + 2)$:

$$d_{NE} = F\left(x_p + \frac{3}{2},\ y_p + 2\right) = a\left(x_p + \frac{3}{2}\right) + b(y_p + 2) + c.$$

Subtraindo o valor anterior de $d$ de $d_{NE}$, obtém-se:

$$d_{NE} = d_{old} + a + b = d_{old} + dy - dx.$$

Se for escolhido o pixel $N$, a variável $d$ deve ser avaliada no ponto $(x_p + 1/2,\ y_p + 2)$:

$$d_N = F\left(x_p + \frac{1}{2},\ y_p + 2\right) = a\left(x_p + \frac{1}{2}\right) + b(y_p + 2) + c.$$

Subtraindo o valor anterior de $d$ de $d_N$, obtém-se: $d_N = d_{old} + b = d_{old} - dx$. do seu valor no passo corrente. O valor inicial de $d$ é calculado a partir do ponto inicial $(x_0, y_0)$:

$$d_0 = F\left(x_0 + \frac{1}{2},\ y_0 + 1\right) = a\left(x_0 + \frac{1}{2}\right) + b(y_0 + 1) + c$$

$$= ax_0 + by_0 + c + \frac{a}{2} + b = F(x_0, y_0) + \frac{a}{2} + b = \frac{a}{2} + b = \frac{dy}{2} - dx.$$

$F(x, y) = 2(ax+by+c)$. Isto multiplica as constantes e a variável $d$ por 2, Multiplicando-se por 2 a função $F$, obtém-se:

- $d_0 = dy - 2dx$,
- $d_N = -2dx$, se $d \geq 0$;
- $d_{NE} = 2dy - 2dx$, se $d < 0$.

Para as outras inclinações, $-1 \leq dy/dx < 0$ ou $dy/dx < -1$, basta notar que a variável $d$ deve ser calculada em $(x_p + 1, y_p - 1/2)$ ou $(x_p - 1/2, y_p + 1)$, respectivamente, e que o critério para a escolha entre $E, N$ ou $SE, NW$ fica invertido, assim como a escolha da variável incrementada (fig. 10.6).

- $d_0 = 2dy + dx$;
- $d_E = 2dy$, se $d \geq 0$,

- $d_{SE} = 2dy + 2dx$, se $d < 0$.
- $d_0 = -dy - 2dx$;
- $d_N = -2dx$, se $d \leq 0$,
- $d_{NW} = -2dy - 2dx$, se $d > 0$.

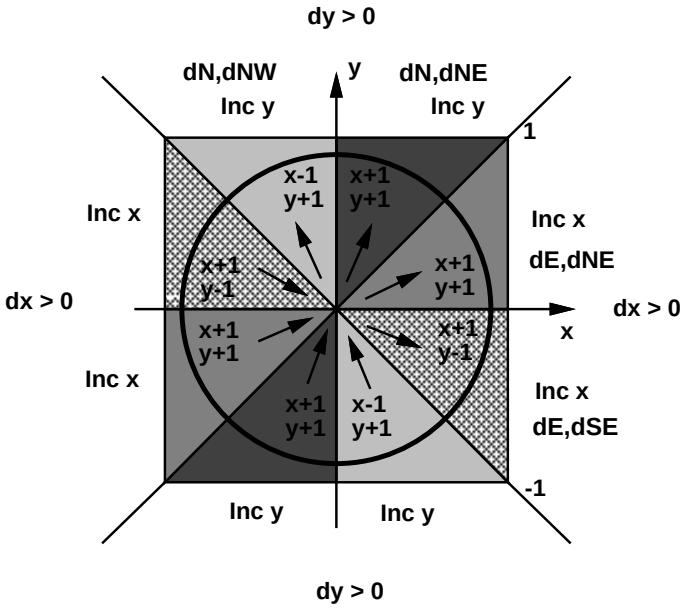



Figura 10.6: Influência das Inclinações na Rasterização.

Para garantir que as linhas $P_0P_1$ e $P_1P_0$ tenham a mesma aparência, costuma-se traçar a linha sempre da esquerda para a direita (do ponto com a menor coordenda $x$ para o ponto com a maior) ou de baixo para cima, de acordo com a inclinação.

A intensidade de uma linha varia com a inclinação, pois a diagonal de um quadrado é $\sqrt{2}$ vezes mais extensa que os lados. Assim, a intensidade é máxima, valendo $I$, para linhas horizontais ou verticais, decaindo até o valor $I/\sqrt{2}$ para uma linha com inclinação 1, assumindo que o mesmo número de pixels é usado para traçar ambas as linhas. Para dispositivos com mais de duas intensidades para os pixels, esta discrepância pode ser compensada, variando-se a intensidade em função da inclinação.

Podem-se rasterizar PolyLines da mesma forma, um segmento por vez, mas os vértices comuns devem ser acendidos uma única vez, pois no modo xor estes pontos teriam a cor do fundo e, no caso de um filme, estes pontos teriam uma intensidade dobrada. Da mesma forma, segmentos muito próximos, ou que se cruzam, apresentam o mesmo tipo de problema.

### 10.1.2 Preenchimento de Áreas

A tarefa de preencher áreas consiste em enumerar os pixels que se encontram no interior da área, para que sejam atribuídas as cores aproriadas. O processo de enumeração é feito analisando segmentos horizontais — chamados linhas de varredura (*scans*) — e preenchendo, da esquerda para a direita, grupos de pixels adjacentes — chamados segmentos de varredura (*spans*) — contidos na interseção da linha de varredura com a área (fig. 10.7).

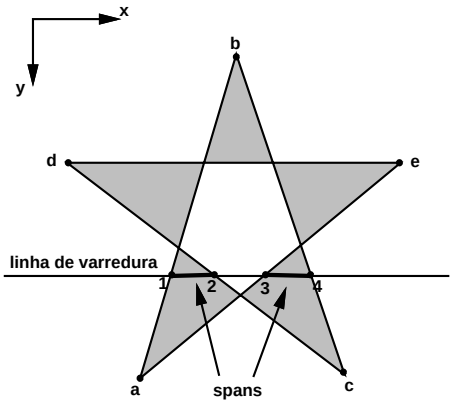



Figura 10.7: Preenchimento de Polígonos.

O exemplo mais simples corresponde a uma área retangular preenchida com uma cor única. Neste caso, existe uma forte coerência espacial, que determina que dentro do mesmo segmento de varredura a cor não muda e que linhas de varredura que interceptam o retângulo determinam segmentos de varredura idênticos. Para evitar que a aresta comum a dois retângulos seja desenhada duas vezes, costuma-se considerar que um pixel em uma aresta não faz parte do retângulo se o semi-plano, definido pela aresta e contendo o retângulo, estiver abaixo ou à esquerda da aresta (fig. 10.8). Assim, as arestas direita e superior do retângulo não são desenhadas[4].

O caso geral de preenchimento de polígonos (côncavos, convexos, com furos e não necessariamente simples) pode ser tratado de maneira similar: calculando segmentos de varredura entre as arestas direita e esquerda do polígono. Os extremos dos segmentos de varredura são calculados incrementalmente, a partir da interseção da linha de varredura anterior com as arestas:

- Encontre a interseção da linha de varredura com todas as arestas.
- Ordene as interseções em ordem crescente de coordenada $x$.

[4] O vértice no canto inferior esquerdo continua sendo desenhado duas vezes.

Figura 10.8: Preenchimento de Retângulos.

- Preencha todos os pixels entre pares consecutivos de interseções (1 - 2, 3 - 4, ...)[5].

As interseções podem ser calculadas, incrementalmente, adaptando-se o algoritmo de rasterização de linhas. Para garantir que apenas pontos interiores ao polígono são desenhados, quando um valor fracionário é calculado, arredonda-se para cima, no caso de arestas de entrada, ou para baixo, no caso de arestas de saída do polígono. Se a extremidade esquerda de um segmento de varredura é inteira, ela é definida como interior ao polígono. No caso da extremidade direita, ela é definida como externa.

O ponto de maior coordenada $y$ de uma aresta não é considerado (será desenhado apenas se for o ponto de menor coordenada $y$ da próxima aresta), garantindo que os vértices não são desenhados duas vezes. Com este critério, arestas horizontais podem ser descartadas.

Por questão de eficiência, na determinação das interseções, a implementação do algoritmo utiliza uma tabela de arestas (ET), que contém todas as arestas ordenadas em ordem crescente de coordenada $y$ mínima ($y_{min}$). Normalmente, existem tantas entradas quanto linhas de varredura. Em cada posição da tabela existe uma lista de arestas, ordenadas em ordem crescente de coordenada $x_{min}$ (correspondente a $y_{min}$), com as arestas com o mesmo valor $y_{min}$. Em cada registro desta lista estão armazenadas a coordenada $y_{max}$ da aresta, a coordenada $x_{min}$ e o incremento em $x$, $1/m$ (o inverso da inclinação).

Existe uma outra estrutura de dados, chamada AET, que contém as arestas ativas (aquelas interceptadas pela linha de varredura corrente). Ao invés de $x_{min}$, existe o campo $x$, que contém a coordenada $x$ da interseção da aresta com a linha de varredura corrente. Nesta lista, as arestas estão ordenadas em ordem crescente de coordenada $x$. O algoritmo segue a seguinte sequência:

- Atribua a $y$ a menor coordenada $y_{min}$ armazenada na ET.
- Inicialise a AET (vazia).
- Repita até que a AET e a ET estejam vazias:

  Remova da AET as arestas com $y_{max} = y$.

  Mova para a AET as arestas da ET cujo $y_{min} = y$, fazendo $x = x_{min}$.

  Ordene a AET em ordem crescente de $x$.

  Preencha os pixels entre pares consecutivos de coordenadas $x$ da AET.

[5] Interseções ímpares são pontos de entrada e as pares são pontos de saída do polígono.

Incremente $y$ de uma unidade.

Para cada aresta da AET, atualize $x$ de acordo com o novo $y$.

A figura 10.9 mostra o preenchimento das duas tabelas para o polígono da figura 10.7. No caso da AET, considere-se a linha de varredura mostrada naquela figura.

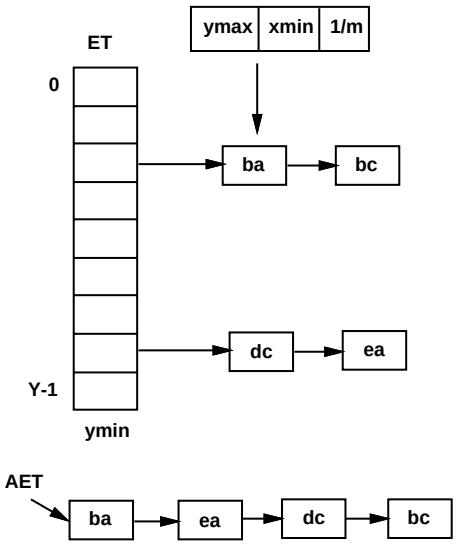



Figura 10.9: Tabela de Arestas e Tabela de Arestas Ativas.

Note-se que como o algoritmo não considera o ponto de maior coordenada $y$ em cada aresta, não é necessário nenhum cuidado especial quando a linha de varredura intercepta um vértice do polígono. Normalmente, a determinação de ponto em polígono é feita disparando-se um tiro horizontal, com origem no ponto a ser testado, e contando-se o número de interseções com a fronteira do polígono. Se o número for impar o ponto está dentro e se for par está fora. No caso do tiro passar exatamente por um vértice, conta-se uma interseção apenas se o vértice corresponder ao mínimo (ou máximo, faça uma escolha) da aresta. Arestas horizontais são simplesmente ignoradas (fig. 10.10).

## 10.2 Rasterização de Elementos não Lineares

Podem-se utilizar, na descrição geométrica das superfícies do modelo, elementos não lineares, tais como superfícies parametrizadas, superfícies implícitas, etc. É possível obter-se uma aproximação B-rep poligonal desses elementos, de modo a efetuar o processo de visualização dos modelos. No entanto, é útil, muitas vezes, executar o processo de rasterização diretamente da equação, em geral não linear, que define a superfície do modelo. Essas técnicas de rasterização empregam, em geral, os métodos por subdivisão ou analíticos.

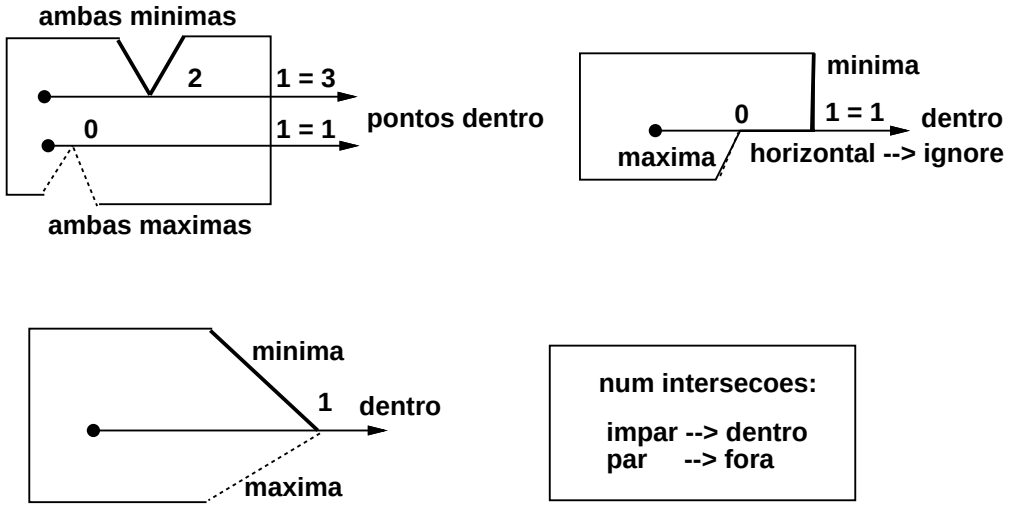



Figura 10.10: Ponto em Polígono.

Aqui, vai-se apresentar um algoritmo incremental para rasterização de círculos e elipses no plano, com eixos alinhados aos eixos coordenados.

### 10.2.1 Rasterização de Círculos

Considere-se o círculo padrão, centrado na origem (0,0):

$$F(x, y) = x^2 + y^2 - R^2,$$

onde $R$ é o raio do círculo. O algoritmo utiliza o fato de que há uma simetria entre os arcos situados nos oito octantes (fig. 10.11). Vai-se considerar, então, apenas o segundo octante.

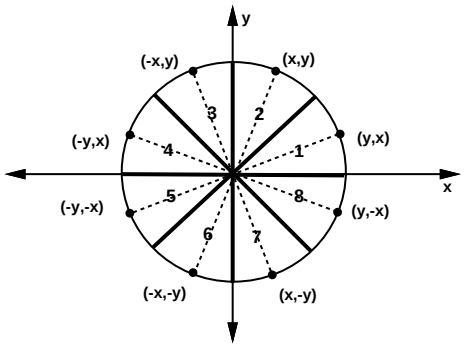



Figura 10.11: Simetria em um Círculo.

Como acontece em todo algoritmo baseado no ponto médio, existe uma função implícita — de decisão ou de densidade — que é avaliada no ponto médio entre dois pixels e que é usada para determinar se o ponto médio está dentro ou fora do círculo. No segundo octante, se o pixel corrente está em $(x_p, y_p)$, a função $d$ é avaliada no ponto $(x_p + 1,\ y_p - 1/2)$ — o ponto médio entre $E$ e $SE$ (fig. 10.12). Se o próximo pixel for o pixel $E$, o próximo ponto médio estará em $(x_p + 2,\ y_p - 1/2)$. Assim:

$$d_{old} = F\left(x_p + 1,\ y_p - \frac{1}{2}\right) = (x_p + 1)^2 + \left(y_p - \frac{1}{2}\right)^2 - R^2,$$

$$d_E = F\left(x_p + 2,\ y_p - \frac{1}{2}\right) = (x_p + 2)^2 + \left(y_p - \frac{1}{2}\right)^2 - R^2,\ \text{se } d < 0.$$

Verifica-se que $d_E = d_{old} + (2x_p + 3)$. Se o próximo pixel for o pixel $SE$, tem-se:

$$d_{SE} = F\left(x_p + 2,\ y_p - \frac{3}{2}\right) = (x_p + 2)^2 + \left(y_p - \frac{3}{2}\right)^2 - R^2,\ \text{se } d \geq 0.$$

Assim, $d_{SE} = d_{old} + (2x_p - 2y_p + 5)$. A condição inicial é calculada assumindo-se $R$ inteiro e que o círculo inicia em $(0, R)$. O primeiro ponto médio é calculado em $(1,\ R - 1/2)$:

$$d_0 = F\left(1,\ R - \frac{1}{2}\right) = 1 + \left(R - \frac{1}{2}\right)^2 - R^2 = \frac{5}{4} - R.$$

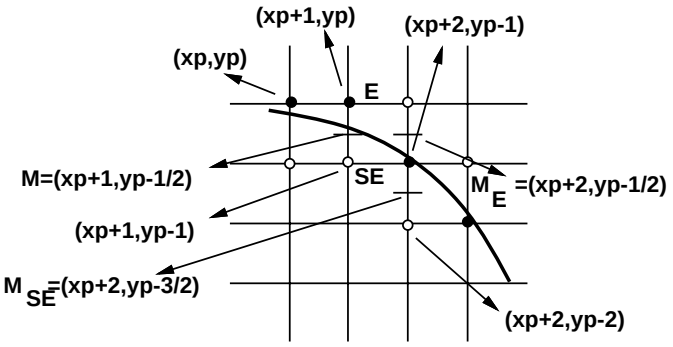



Figura 10.12: Algoritmo do Ponto Médio para Círculos.

A função de decisão $d$ assume valores negativos em pontos interiores ao círculo; valores positivos em pontos exteriores; e vale zero em pontos sobre o círculo. Isto pode ser visto, a partir da figura 10.13, interpretando o gráfico da função $F$ no $\Re^3$, que é um parabolóide. O círculo corresponde à curva de nível $F(x, y) = 0$. Conclui-se, então, que se $d < 0$ escolhe-se o pixel $E$ e se $d > 0$ escolhe-se o pixel $SE$. Se $d = 0$ a escolha é arbitrária, por exemplo, escolha-se o pixel $E$.

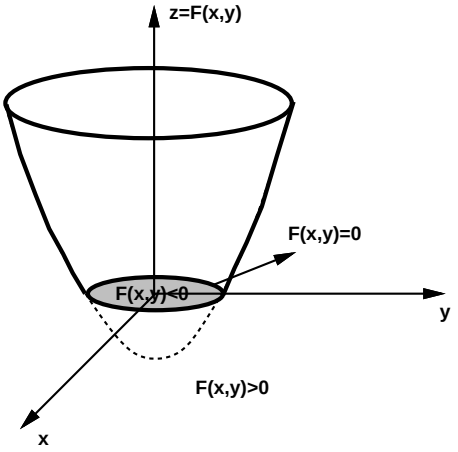



Figura 10.13: Gráfico da Função $F(x, y)$.

Pode-se fazer uma mudança de variável $h = d - 1/4$ ou $d = h + 1/4$. Desta forma, $h_0 = 1 - R$ e o teste $d < 0$ é substituído por $h < -1/4$. Como $h$ inicia com um valor inteiro e é sempre incrementado por valores inteiros, pode-se utilizar o teste $h < 0$ sem problema. Os incrementos não são alterados pela adição da constante $1/4$. Assim, obtém-se um algoritmo que pode ser implementado, inteiramente, com aritmética inteira.

### 10.2.2 Rasterização de Elipses

Considere-se a elipse padrão, centrada na origem (0,0):

$$F(x, y) = b^2x^2 + a^2y^2 - a^2b^2,$$

onde $2a$ é o comprimento do eixo maior, sobre o eixo $x$ e $2b$ o comprimento do eixo menor, sobre o eixo $y$. O algoritmo utiliza o fato de que há uma simetria entre os arcos situados nos quatro quadrantes. Considerando, então, apenas o primeiro quadrante, este é dividido em duas regiões. A fronteira destas regiões é a linha que parte da origem e passa pelo ponto da elipse cuja tangente tem inclinação -1. Neste ponto o vetor gradiente tem inclinação $+1$, ou seja, sua componente $x$ iguala a componente $y$. Na região 1, adjacente ao eixo $y$, a componente $y$ é maior do que a $x$ e na região 2 a componente $x$ é maior do que a $y$.

Como acontece em todo algoritmo baseado no ponto médio, existe uma função implícita — de decisão ou de densidade — que é avaliada no ponto médio entre dois pixels e que é usada para determinar se o ponto médio está dentro ou fora da elipse. Na região 1, se o pixel corrente está em $(x_p, y_p)$, a função $d_1$ é avaliada no ponto $(x_p + 1,\ y_p - 1/2)$ — o ponto médio entre $E$ e $SE$. Se o próximo pixel for o pixel $E$, o próximo ponto médio estará em $(x_p + 2,\ y_p - 1/2)$. Assim:

$$d_{old} = F\left(x_p + 1,\ y_p - \frac{1}{2}\right) = b^2(x_p + 1)^2 + a^2\left(y_p - \frac{1}{2}\right)^2 - a^2b^2,$$

$$d_E = F\left(x_p + 2,\ y_p - \frac{1}{2}\right) = b^2(x_p+2)^2 + a^2\left(y_p - \frac{1}{2}\right)^2 - a^2b^2,\ \text{se}\ \ d_1 < 0.$$

Verifica-se que $d_E = d_{old} + b^2(2x_p + 3)$. Se o próximo pixel for o pixel $SE$, tem-se:

$$d_{SE} = F\left(x_p + 2,\ y_p - \frac{3}{2}\right) = b^2(x_p+2)^2 + a^2\left(y_p - \frac{3}{2}\right)^2 - a^2b^2,\ \text{se}\ d_1 \geq 0.$$

Assim, $d_{SE} = d_{old} + b^2(2x_p + 3) + a^2(-2y_p + 2)$. Na região 2, se o pixel corrente está em $(x_p, y_p)$, a função $d_2$ é avaliada no ponto $(x_p + 1/2, y_p - 1)$:

$$d_{old} = F\left(x_p + \frac{1}{2},\ y_p - 1\right) = b^2\left(x_p + \frac{1}{2}\right)^2 + a^2(y_p-1)^2 - a^2b^2,$$

$$d_S = F\left(x_p + \frac{1}{2},\ y_p - 2\right) = b^2\left(x_p + \frac{1}{2}\right)^2 + a^2(y_p-2)^2 - a^2b^2,\ \text{se}\ d_2 \geq 0,$$

$$d_{SE} = F\left(x_p + \frac{3}{2},\ y_p - 2\right) = b^2\left(x_p + \frac{3}{2}\right)^2 + a^2(y_p-2)^2 - a^2b^2,\ \text{se}\ d_2 < 0.$$

Assim, $d_S = d_{old} + a^2(-2y_p + 3)$ e $d_{SE} = d_{old} + b^2(2x_p + 2) + a^2(-2y_p + 3)$. A condição inicial é calculada assumindo-se valores inteiros $a$ e $b$ e que a elipse inicia em $(0, b)$. O primeiro ponto médio é calculado em $(1,\ b - 1/2)$:

$$d_{10} = F\left(1,\ b - \frac{1}{2}\right) = b^2 + a^2\left(b - \frac{1}{2}\right)^2 - a^2b^2 = b^2 + a^2\left(-b + \frac{1}{4}\right).$$

A cada iteração na região 1, deve-se testar se é necessário trocar de região. Isto é feito, avaliando-se o gradiente no ponto médio entre $E$ e $SE$. Quando o ponto médio passa para a região 2[6], a função de decisão passa a ser $d_2$, que é inicializada, a partir do último pixel na região 1, no ponto $(x_p + 1/2, y_p - 1)$. Quando o valor $y$ do pixel for 0, o algoritmo termina.

$$d_{20} = F\left(x_p + \frac{1}{2},\ y_p - 1\right) = b^2\left(x_p + \frac{1}{2}\right)^2 + a^2(y_p-1)^2 - a^2b^2.$$

## 10.3 Amostragem

Quando foi discutido o paradigma dos quatro universos, frisou-se que, na passagem do universo matemático para o universo de representação, era-se obrigado a fazer uma discretização do modelo para obter-se uma representação finita. Para discretizar uma função[7], deve-se colher um número finito de amostras, que servirão para caracterizá-la, conforme pode ser visto na figura 10.14.

Para obter-se novamente a função original deve-se fazer algum tipo de interpolação dos dados amostrados. Esse processo de interpolação é conhecido como a reconstrução da função ou do sinal. A reconstrução pode ser entendida como um processo de descrever uma função

[6] $a^2(y_p - 1/2) \leq b^2(x_p + 1)$.

[7] Designignada, em engenharia, como um sinal.

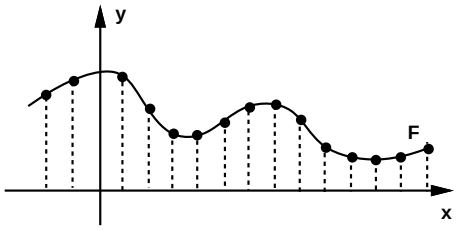



Figura 10.14: Amostragem de um Sinal.

em relação a uma base. Por exemplo, os multiplicadores de Lagrange constituem um método de interpolação que utiliza uma base polinomial. Uma interpolação linear utliza a função chapéu (fig. 10.15) que é uma função linear por partes dada por:

$$L(x) = 1 - x \;\text{ se }\; 0 \leq x \leq 1; \qquad x + 1 \;\text{ se }\; -1 \leq x < 0; \qquad 0 \;\text{ se }\; x < -1 \;\text{ ou }\; x > 1.$$

Assim, $p(x) = \sum_{j=0}^{n} y_j L(x-j)$; $y_j = f(j)$, conforme ilustrado na figura 10.15. Um monitor de vídeo reconstrói uma imagem digital utilizando uma base Gaussiana.

## 10.4 Reconstrução Exata

A pergunta apropriada neste momento é: será que sempre é possível reconstruir a função original de forma exata? Esta pergunta é respondida pelo teorema de amostragem de Shannon. Se o sinal for de banda limitada ele pode ser reconstruído de forma exata a partir de um conjunto de amostras uniformes tomadas pelo menos a uma taxa duas vezes maior do que a maior freqüência presente no sinal. Este é o chamado limite de amostragem de Nyquist. O processo de reconstrução envolve a teoria de transformada de Fourier e não será abordado nexte texto.

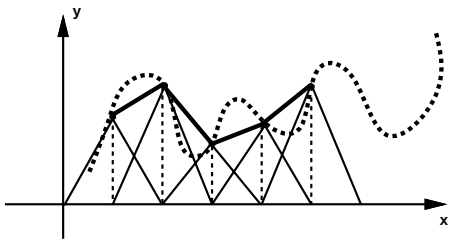



Figura 10.15: Interpolação Linear.

## 10.5 *Aliasing*

Quando o sinal não é de banda limitada, ou a amostragem é feita abaixo do limite de Nyquist, pode acontecer que uma componente de alta freqüência presente no sinal seja reconstruída como uma componente de baixa freqüência, dita um alias da componente original. Este fenômeno é designado por *aliasing* (fig. 10.16). As técnicas de anti-aliasing, que visam minimizar o problema, ou aumentam a taxa de amostragem ou filtram as altas freqüências presentes no sinal.

Figura 10.16: *Aliasing* de um Sinal.

Em áudio limitam-se as freqüências presentes em uma música de acordo com a acuidade do ouvido humano, que detecta sons com freqüências que variam de 16Hz a 22KHz. Isto é o que torna uma gravação em CD aceitável[8].

Quando o sinal de interesse representa uma imagem contínua (um sinal bi-dimensional),

$$S : \Re^2 \rightarrow \zeta(\Re^3),$$

ele pode conter freqüências infinitas, que são percebidas visualmente (por exemplo, um tabuleiro de xadrez visto em perspectiva). Uma solução é filtrar as altas freqüências, o que substitui o aliasing por ruído.

## 10.6 Tipos de Amostragem

Existem dois tipos básicos de amostragem: pontual e por área. Na amostragem pontual é colhida uma única amostra por pixel. Como os pixels possuem área, surge o problema de escolher o ponto dentro do pixel que define a amostra, pois, em função disto, podem ser amostrados objetos diferentes (sinais diferentes). Na amostragem por área ou analítica, os objetos são recortados contra a área de um pixel e é feita uma estimativa da área de cada parte de cada objeto contido no pixel (fig. 10.17). A cor do pixel é dada pela média das cores das partes, ponderada pela área de cada parte:

$$C_{pixel} = \frac{A_1 c_1 + A_2 c_2 + ... + A_n c_n}{A_1 + A_2 + ... + A_n}.$$

Este processo equivale a aplicar um filtro de passa baixa na imagem seguido de uma amostragem pontual.

[8] Embora alguns puristas digam que a qualidade sonora fica comprometida, por causa da eliminação dos harmônicos.

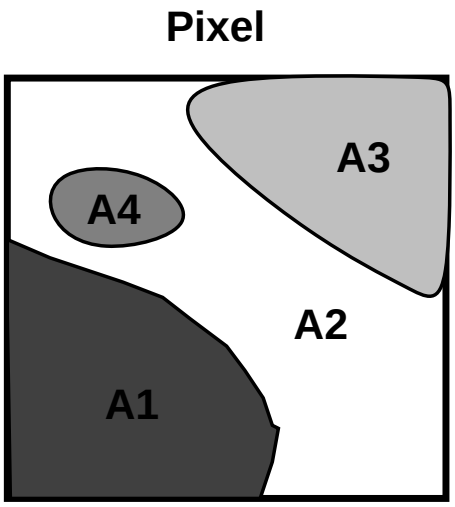



Figura 10.17: Amostragem por Área.

A técnica de superamostragem subdivide os pixels em sub-pixels. Em cada subpixel é feita uma amostragem pontual e a cor do pixel é dada pela média das cores dos sub-pixels:

$$C_{pixel} = \sum_{i,j=1}^{m,n} \frac{I(i,j)}{mn},$$

onde $I(i,j)$ define a cor do sub-pixel $(i,j)$. No limite, quando o número de sub-pixels vai para infinito $(m,n \to \infty)$, a superamostragem é equivalente a amostragem por área[9].

Quando existe o canal alpha, que define a opacidade do pixel, é possível usá-lo para compor imagens com anti-aliasing embutido.

[9] Esta é a idéia por trás do método de Monte Carlo.

## 10.7 Exercícios

**10.1** *Quais são os principais métodos de rasterização? Descreva as suas estratégias.*

**10.2** *Discuta as relações entre rasterização e amostragem.*

**10.3** *Qual é o algoritmo de rasterização mais adequado para:*

*a) Modelos Implícitos*

*b) Modelos Poligonais*

*c) Retalhos Bi-cúbicos*

*Justifique a sua resposta.*

**10.4** *Implemente o algoritmo do ponto médio para rasterização de linhas, inclusive suportando estilos de linha (tracejado, pontilhado, traço ponto, etc.) e espessura de linha. Em seguida:*

*a) Explique o funcionamento do algoritmo.*

*b) Em que caso a rasterização depende da direção do segmento? Como contornar esse problema?*

*c) Discuta maneiras de "casar", adequadamente, linhas com estilo e com espessura.*

*Sugestão: utilize dois bytes (16 bits) para especificar o padrão (estilo) da linha. Assim,* 1111000011110000 *especifica um traço com quatro pontos e um espaço de quatro pontos, criando uma linha tracejada.*

**10.5** *Implemente o algoritmo do ponto médio para rasterização de círculos. Assuma que o centro e o raio do círculo são arbitrários, porém inteiros.*

**10.6** *Demonstre que o algoritmo do ponto médio para rasterização de linhas força uma escolha apropriada de sinal, que produz um movimento correto, mesmo em casos degenerados, nos quais o pixel selecionado* $(x, y)$ *e a interseção real para a próxima abcissa* $(x + 1)$ *não acontece entre os valores de ordenada* $y$ *e* $y + 1$*, por exemplo (fig. 10.18):*

*a) Entre* $y$ *e* $y - 1$*, como no ponto* $(2, 1)$ *na linha de* $(0, 0)$ *a* $(7, 2)$

*b) Entre* $y + 1$ *e* $y + 2$*, como no ponto* $(2, 1)$ *na linha de* $(0, 0)$ *a* $(7, 5)$*.*

**10.7** *O algoritmo do ponto médio assume que as extremidades dos segmentos possuem coordenadas inteiras. Crie um algoritmo que elimine esta restrição.*

**10.8** *Descreva um procedimento para rasterização recursiva de um círculo unitário:*

$$S^1 = \{(x, y) \in \Re^2; x^2 + y^2 = 1\}.$$

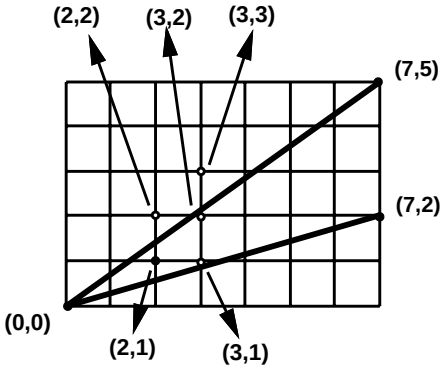



Figura 10.18: Casos Degenerados no Algoritmo do Ponto Médio para Retas.

**10.9** *Desenvolva um procedimento incremental para a rasterização de triângulos.*

**10.10** *Analise a rasterização incremental de triângulos, polígonos convexos, e polígonos arbitrários.*

**10.11** *Desenvolva um procedimento por teste exaustivo para a rasterização de esferas descritas de forma implícita.*

**10.12** *Discuta um método de anti-aliasing por super amostragem adaptativa em ray tracing.*
*Sugestão: divida inicialmente o pixel em quatro sub-pixels.*

**10.13** *Analise o princípio de construção da Summed Area Table baseado nas propriedades das integrais.*

# Capítulo 11

# Visibilidade

A questão da visibilidade foi um dos aspectos centrais da Computação Gráfica durante uma parte do seu desenvolvimento: foi o primeiro passo na direção do fotorealismo e a sua solução tinha relações profundas com as outras operações do processo de visualização. Este papel privilegiado se justifica na medida em que os algoritmos de visibilidade precisam estruturar as operações de visualização para a obtenção da imagem (fig. 11.1):

- As transformações de visualização devem ser realizadas numa determinada ordem, de modo a levar os objetos para um sistema de coordenadas que permita uma ordenação eficiente, levando em conta a transformação perspectiva.
- A determinação da região digital da tela virtual associada às regiões visíveis (rasterização) pode ser combinada de diversas formas com as etapas de um algoritmo de visibilidade.
- Uma vez determinadas as superfícies visíveis, o cálculo da função de iluminação indicará a cor dos elementos correspondentes da imagem (pixels).

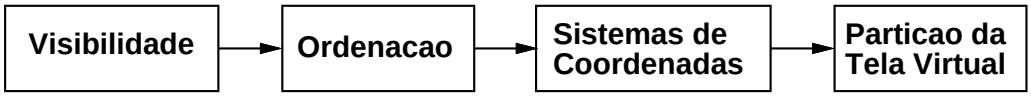



Figura 11.1: Modelo Conceitual.

A visibilidade se coloca de maneira bastante diferenciada em função da natureza da imagem a ser visualizada. Se a imagem é composta apenas por linhas, empregam-se os chamados algoritmos de linhas escondidas; se a visualização de superfícies é importante, empregam-se os algoritmos de superfícies escondidas.

Atualmente, com o avanço da indústria de equipamentos gráficos, várias técnicas de visibilidade são incorporadas, diretamente, no *hardware* e os algoritmos de linhas escondidas costumam ser usados apenas em certos tipos de desenhos técnicos de engenharia.

## 11.1 Classificação dos Algoritmos

O problema da visibilidade consiste, essencialmente, na determinação das superfícies mais próximas ao observador, que, conseqüentemente, estarão visíveis. Este problema envolve, basicamente, uma ordenação parcial: até a primeira superfície opaca em cada vizinhança da imagem.

Os algoritmos de visibilidade podem ser divididos entre os que operam na precisão do espaço da cena e aqueles que operam na precisão do espaço da imagem:

- Os que operam com precisão da cena utilizam, normalmente, números reais com ponto flutuante. Pode-se dizer que eles calculam a solução exatamente e o resultado é uma lista ordenada das faces a serem projetadas no plano da tela virtual.

- Os que operam com precisão de imagem estão interessados na solução correta do problema para um determinado nível de resolução. Neste caso, o algoritmo procura resolver o problema no âmbito de cada elemento da imagem (pixel), analisando as profundidades relativas em cada raio de visão, sendo a visibilidade adiada até o último momento.

Os algoritmos de visibilidade podem ser classificados de acordo com o método de ordenação utilizado para determinar as superfícies visíveis do ponto de vista da câmara virtual. A ordenação empregada nos algoritmos está intimamente ligada com a operação de rasterização, que determina a região digital da tela virtual correspondente aos objetos em cena (objetos que ocupam áres disjuntas da imagem são independentes em termos de visibilidade). Além disso, a rasterização pode ser vista como um processo de ordenação pelo qual faz-se uma enumeração espacial dos pixels ocupados por cada objeto. Essencialmente, a rasterização resulta na ordenação em $X$ e $Y$, enquanto a visibilidade na ordenação em $Z$ (profundidade), no sistema de coordenadas normalizadas.

As estruturas computacionais dos algoritmos de visibilidade fazem uso das seguintes seqüências de ordenação: $Z(XY)$; $(XY)Z$; $YXZ$ (os parênteses indicam operação de ordenação combinada). Essas três estruturas de ordenação correspondem a três tipos de algoritmos que resolvem:

- a visibilidade antes da rasterização (z-Sort, Partição do Espaço, Recorte Recursivo);
  Calculam a visibilidade exatamente, processando a ordenação em $Z$ globalmente, a nível de objetos ou faces dos objetos.

- a rasterização antes da visibilidade (z-Buffer, Subdivisão Recursiva, Traçado de Raios);
  Calculam a visibilidade na precisão da imagem, reduzindo o problema da ordenação para uma vizinhança dos pixels.

- a rasterização integrada com a visibilidade (Scanline);
  Utilizam uma solução intermediária, operando em segmentos unidimensionais associados às linhas da imagem (*spans*).

## 11.2 Transformação Perspectiva

A despeito do tipo de projeção utilizada, a determinação da visibilidade de dois pontos se resume em descobrir se um ponto obscurece o outro, ou seja, se $p_1$ e $p_2$ estão sobre o mesmo

projetor. A determinação da visibilidade deve ser feita no espaço tri-dimensional, antes que a projeção (que leva os pontos para o espaço bi-dimensional) destrua a informação de profundidade necessária às comparações.

As comparações de profundidade são realizadas, normalmente, após a transformação de normalização[1], de forma que os projetores são paralelos ao eixo $z$, na projeção paralela, ou emanam da origem, no caso da perspectiva. Para a projeção paralela, os raios estão sobre o mesmo projetor quando $x_1 = x_2, \quad y_1 = y_2$. Para a perspectiva, devem ser realizadas quatro divisões:

$$\frac{x_1}{z_1} = \frac{x_2}{z_2}, \qquad \frac{y_1}{z_1} = \frac{y_2}{z_2}.$$

Estas divisões podem ser evitadas se for utilizada a transformação perspectiva, que transforma a pirâmide de visão normalizada no prisma retangular unitário, levando o centro de projeção para o infinito na direção $-z$. A transformação perspectiva não altera as relações de profundidade, ou seja:
$z_1 < z_2 \rightarrow P(z_1) < P(z_2)$, como pode ser visto abaixo[2].

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/a & -z_{min}/a \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad z_{min} > 0, \quad a = (1 - z_{min}) > 0, \quad 0 < z_1 < z_2;$$

$$b = \left(\frac{1}{z_1}\frac{z_{min}}{a}\right) > c = \left(\frac{1}{z_2}\frac{z_{min}}{a}\right) \Rightarrow P(z_1) = \left(-b + \frac{1}{a}\right) < P(z_2) = \left(-c + \frac{1}{a}\right).$$

Com a transformação perspectiva, o observador passa a estar no infinito na direção $-z$. Assim, uma seqüência crescente de coordenadas $z$ positivas, ou uma seqüência crescente de coordenadas $z$ negativas, se afasta do observador.

## 11.3 Back-Face Culling

Se a superfície de um objeto é aproximada por uma superfície poligonal sem bordo, então as faces são polígonos e englobam completamente o volume do objeto. Assumindo-se que as normais de todas as faces apontam para fora do volume, tem-se que as faces, cujas normais apontam na direção contrária ao observador, estão numa parte do poliedro cuja visibilidade é bloqueada por outras faces mais próximas ao observador. Deve-se assumir, também, que o poliedro é convexo e não sofreu recorte contra o plano frontal de recorte.

Em coordenadas da cena, uma face obstruída pode ser identificada pela não negatividade do produto escalar da sua normal com o vetor definido pelo centro de projeção e qualquer ponto do polígono (fig. 11.2).

Se a transformação perspectiva foi aplicada, sendo utilizada uma projeção ortográfica no plano $xy$, então a direção de projeção é $(0, 0, +1)$. Neste caso, o teste de obstrução com o produto escalar se reduz em verificar se a normal à face possui uma coordenada $z$ positiva .

[1] Os métodos baseados no traçado de raios não utilizam transformação de normalização.

[2] A transformação $P$ assume que a pirâmide de visão normalizada está no semi-espaço $z$ positivo .

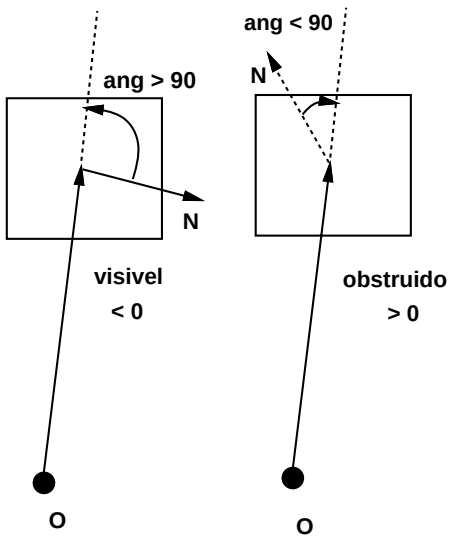



Figura 11.2: Back-Face Culling.

Se a cena é composta apenas por um único poliedro, então, *back-face culling* é o único método de visibilidade que precisa ser aplicado. Um projetor que atravessa um poliedro intercepta o mesmo número de faces visíveis e obstruídas. Por isso, um ponto qualquer na projeção do poliedro está na projeção do mesmo número de faces visíveis e obstruídas e o método de *back-face culling* reduz à metade o número de polígonos que precisam ser considerados pelos métodos de visibilidade com precisão de imagem.

## 11.4 z-Buffer

O algoritmo z-Buffer, criado por Catmull, é, talvez, o algoritmo de visibilidade mais simples de ser implementado tanto em *software* como em *hardware*. Este algoritmo requer que, além de um *frame buffer* que armazena os valores de cor para cada pixel, exista um z-Buffer, com o mesmo número de entradas, no qual o valor da coordenada $z$ de cada pixel é armazenado.

O z-Buffer é inicializado com um, que representa o valor de $z$ do plano de recorte traseiro e o *frame buffer* com a cor de fundo. O menor valor $z$ que pode ser colocado no z-Buffer corresponde ao valor do plano de recorte frontal. Os polígonos são rasterizados em qualquer ordem. Durante o processo de rasterização, se o ponto corrente do polígono sendo rasterizado possuir um valor $z$ menor do que o valor correspondente já armazenado no z-Buffer (significando que não está mais afastado do observador do que o ponto previamente armazenado), os seus valores de cor e de $z$ substituem os valores dos *buffers*.

Para calcular o valor $z$ de um pixel, deve-se resolver a equação do plano para a variável $z$, obtendo-se

$$z = -\frac{D + Ax + By}{C}.$$

No entanto, para todos os pixels na mesma linha ($y$ é constante), dado o valor $z_1$ do primeiro

pixel, então, o pixel em $(x + \Delta x)$ (normalmente, $\Delta x = 1$) tem valor

$$z = z_1 - \frac{A\Delta x}{C}.$$

Um cálculo incremental similar pode ser executado para descobrir o primeiro valor de $z$ na próxima linha ($x$ é constante):

$$z - z_1 = \frac{B\Delta y}{C}, \qquad \Delta y = 1.$$

O algoritmo do z-Buffer pode ser empregado com qualquer tipo de objeto, desde que um valor de cor e de $z$ possa ser determinado para cada ponto na sua projeção.

## 11.5 Scan-Line

Os algoritmos scan-line criam uma imagem por varredura[3], linha por linha. Eles operam na precisam da imagem determinando, incrementalmente, a interseção de cada objeto da cena com planos definidos pelo ponto de vista e pelas linhas da imagem (em geral, planos da forma $y = cte$) e se baseiam no tradicional algoritmo de rasterização de polígonos, que exploram a coerência de arestas e linhas. A visibilidade é resolvida para intervalos correspondentes a segmentos unidimensionais.

A diferença em relação ao processo de rasterização de um polígono é que, agora, existe mais de um polígono a ser considerado. A tabela de arestas ($ET$) contém todas as arestas de todas as projeções dos polígonos (arestas horizontais são ignoradas) e uma identificação, para cada aresta, do polígono que a possui. Existe, também, uma tabela de polígonos ($PT$) que contém os coeficientes da equação do plano do polígono, informação de cor e um *flag*, inicializado com falso, a ser usado durante a rasterização, para indicar se a linha de varredura corrente ($LC$) está dentro ou fora do polígono.

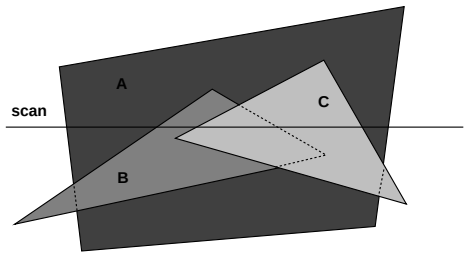



Figura 11.3: Três Polígonos Disjuntos.

A tabela de arestas ativas ($AET$) contém todas as arestas cortadas pela $LC$, ordenadas de acordo com a coordenada $x$ da interseção com a $LC$.

[3]Os processos de varredura buscam determinar a cor dos pixels, enumerando-os, um a um, a partir da primeira linha até a útima, e da esquerda para a direita, dentro da mesma linha.

Quando a primeira aresta da $AET$ é considerada, o *flag* do seu polígono é negado, tornando-se verdadeiro. Polígonos com *flag* verdadeiro são ditos ativos. A $AET$ é percorrida seqüencialmente e os *flags* dos polígonos correspondentes vão sendo negados, significando que, se a $LC$ estava dentro do polígono, saiu e vice versa.

Quando a $LC$ está dentro de um polígono e, ao cruzar uma aresta fica dentro de um outro polígono, o valor das coordenadas $z$, no ponto, indicam qual o polígono está mais próximo ao observador, passando a ser o polígono corrente ($PC$), e a sua cor é atribuída aos pixels da $LC$, deste ponto em diante.

Se, ao cruzar uma aresta, o *flag* do seu polígono torna-se falso, significa que a $LC$ saiu do polígono. Se for uma aresta do $PC$, então deve-se procurar qual o polígono, dentre os ativos, está mais próximo ao observador, passando a ser o novo $PC$. Se não for uma aresta do $PC$, e considerando-se que os polígonos não se penetram, não é necessário cálculo de coordenada $z$, uma vez que o $PC$ continua estando mais próximo ao observador.

Ao se passar à próxima linha, as arestas não mais cortadas são retiradas da $AET$, novas arestas, que passam a ser cortadas, são incluídas, a ordenação é refeita e o processo descrito acima é repetido.

## 11.6 z-Sort

O algoritmo de z-Sort se baseia em uma lista de prioridades, assumindo que uma imagem correta irá resultar se os objetos forem exibidos de acordo com as suas prioridades. Por exemplo, se os objetos não interferem na direção $z$, tudo o que é necessário é ordená-los em ordem decrescente de coordenada $z$, e depois exibí-los nesta ordem. Os objetos mais distantes ao observador (valores $z$ maiores ) são obstruídos pelos objetos mais próximos, à medida que os pixels destes se sobrepõem aos daqueles. Mesmo que haja interferência na direção $z$, ainda assim é possível determinar a ordem apropriada. Neste caso, se os objetos se sobrepõem ciclicamente ou se penetram, uma etapa de recorte é necessária.

- Ordene todos os polígonos de acordo com o valor da coordenada $z$ mínima de cada um.
- Resolva quaisquer ambigüidades causadas por uma sobreposição na direção $z$, recortando os polígonos.
- Rasterize os polígonos em ordem decrescente de coordenada $z$ mínima.

Alguns sistemas gráficos (GKS, PHIGS) permitem a atribuição explícita de prioridades, que tomam o lugar do valor $z$ mínimo. Assim, nunca há ambigüidade, uma vez que cada prioridade funciona como se os objetos estivessem em planos $z = cte$ diferentes. Esta versão simplificada do algoritmo é conhecida como algoritmo do pintor, fazendo uma analogia ao modo de como um pintor desenha os objetos mais próximos sobre aqueles mais distantes. O algoritmo do pintor pode ser aplicado ordenando-se os polígonos de acordo com a coordenada $z$ mínima de cada um, ou ordenando-se as coordenadas dos seus centróides. No entanto, isto nem sempre funciona, conforme pode ser visto na figura 11.4.

Seja $P$ o polígono (com maior $z$) no início da lista ordenada. Antes que este polígono seja rasterizado, deve-se testá-lo contra cada polígono $Q$, que interfira com ele na direção

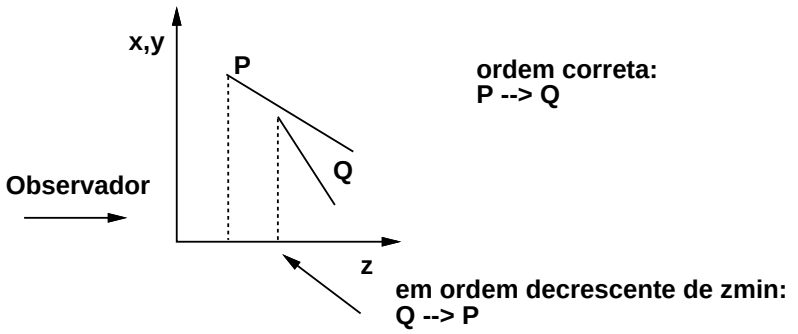



Figura 11.4: Problema Quando se Olha Apenas para um Ponto.

$z$[4], para provar que $P$ não obstrui $Q$, podendo, neste caso, ser rasterizado antes de $Q$. Abaixo estão colocados cinco testes, em ordem crescente de complexidade, de forma a que, assim que um deles seja satisfeito, tenha-se certeza de que $P$ não obscurece $Q$ e o próximo polígono $Q$, interferindo com $P$, é testado. Se todos os polígonos passarem, então $P$ pode ser rasterizado.

Os testes 1) e 2) são executados normalmente baseados apenas nas *bounding box 3D* dos polígonos. Já o teste 5) é bastante trabalhoso de ser implementado e costuma ser ignorado.

1) Os polígonos não interferem em $x$?

2) Os polígonos não interferem em $y$?

3) $P$ está completamente no semi-espaço, gerado pelo plano de $Q$, oposto ao do observador? (fig. 11.5a)

4) $Q$ está completamente no mesmo semi-espaço, gerado pelo plano de $P$, do observador? (fig. 11.5b)

5) As projeções de $P$ e $Q$ no plano $xy$ não se sobrepõem?

Se todos os testes falharem, e $Q$ não estiver marcado, assume-se, pelo momento, que $P$ obscurece $Q$ e testa-se se $Q$ pode ser rasterizado antes de $P$. Os testes 1,2 e 5 não precisam ser refeitos e os testes 3 e 4 são executados trocando-se $P$ por $Q$ e vice-versa. Caso um destes dois novos testes funcione, $Q$ é marcado[5] e movido para o início da lista, tornando-se o novo $P$. Caso contrário ($Q$ marcado ou os dois testes falharam), $P$ ou $Q$ deve ser recortado contra o plano do outro. O polígono original é descartado e os pedaços são inseridos na lista na posição apropriada.

[4] $z_{Q\_min} < z_{P\_max} \leq z_{Q\_max}$ ou $z_{P\_min} < z_{Q\_max} \leq z_{P\_max}$.

[5] A marcação é necessária para evitar um ciclo infinito, com $P$ e $Q$ se revezando no início da fila.

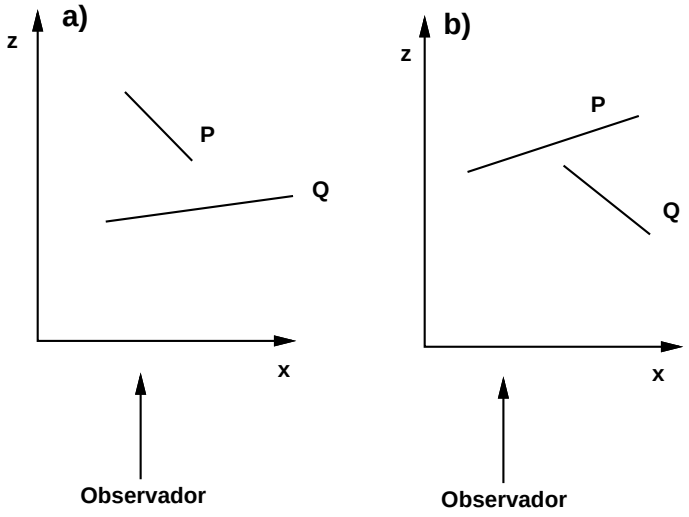



Figura 11.5: Teste 3 e 4.

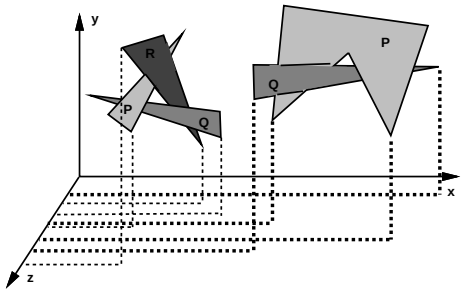



Figura 11.6: Casos em que um Ciclo Infinito Ocorre se não Houver Marcação.

## 11.7 Partição do Espaço - BSP

Este tipo de algoritmo é extremamente eficiente para calcular a visibilidade de um conjunto de polígonos 3D, a partir de um ponto de vista arbitrário. Inicialmente, uma etapa de pré-processamento, que gasta tempo e espaço, é executada e, a partir daí, sempre que a câmara virtual é reposicionada, os polígonos são exibidos em tempo linear.

O princípio básico por detrás do método é particionar o espaço usando um plano em relação ao qual, um conjunto de polígonos se situa em um lado, e o outro conjunto do lado oposto. Desta forma, o conjunto que estiver no mesmo semi-espaço do observador pode obstruir (mas não ser obstruído pelo) o outro conjunto. Cada um destes conjuntos pode ser particionado, recursivamente, por novos planos de separação. O objetivo é determinar a ordem correta para a rasterização dos polígonos.

A partição do espaço pode ser representada por uma árvore binária (BSP), cuja raíz

corresponde a um polígono qualquer (fig. 11.7). O plano do polígono na raíz ($PR$) é usado para particionar o espaço em dois semi-espaços. O primeiro semi-espaço contém todos os polígonos na frente de $PR$, em relação a sua normal. O segundo contém os polígonos atrás de $PR$. Se um polígono se situar em ambos os semi-espaços ele é recortado contra o plano de $PR$ e suas partes são associadas aos semi-espaços apropriados. Em cada um dos semi-espaços, um polígono qualquer é escolhido para ser o seu filho da frente e outro o de trás e cada um deles é usado para classificar, recursivamente, os polígonos em cada semi-espaço. O algoritmo termina quando cada nó contém apenas um polígono.

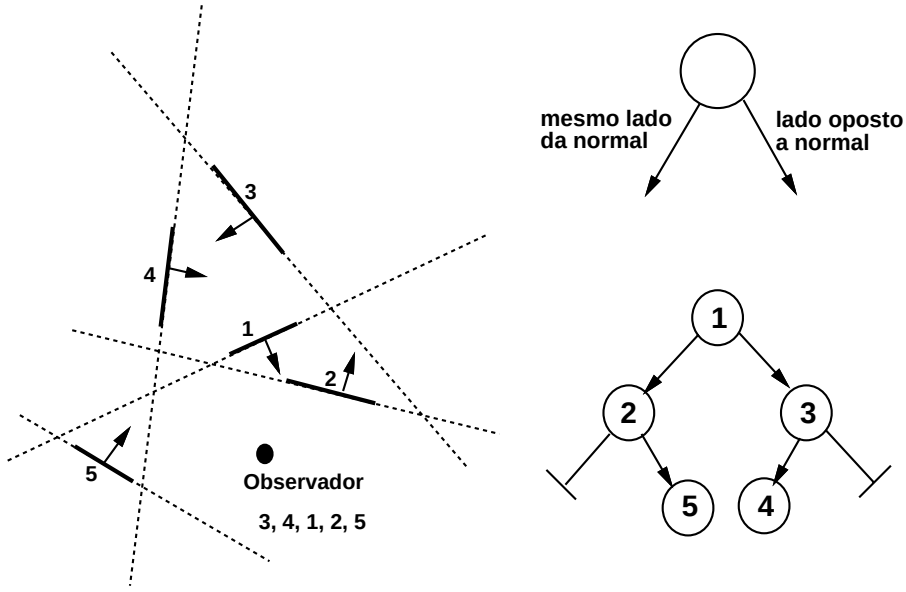



Figura 11.7: Binary Space Partition Tree.

Uma vez especificada a posição do observador, a árvore deve ser percorrida, a partir da raíz, de forma a que o conjunto de polígonos no semi-espaço oposto ao do observador seja exibido primeiro (o conjunto que pode ser obscurecido por $PR$), em seguida deve ser exibido $PR$ e, por fim, os polígonos no mesmo semi-espaço do observador (aqueles que podem obscurecer $PR$). Cada um dos filhos de $PR$ deve ser processado, recursivamente, da mesma forma.

Do mesmo modo que o algoritmo z-Sort, este algoritmo executa as etapas de ordenação e recorte de polígonos na precisão do objeto e utiliza, em precisão da imagem, a capacidade de sobre-escrita de um dispositivo *raster*.

## 11.8 Subdivisão de Área

Algoritmos de subdivisão de área seguem uma estratégia de dividir para conquistar, particionando, recursivamente, o quadrado, que corresponde à janela, no plano de projeção. O algoritmo de Warnock, inicialmente, divide este quadrado em 4 partes iguais. Em cada etapa do processo recursivo de subdivisão, a projeção de cada polígono se enquadra em uma dentre quatro possíveis relações com a área de interesse, conforme a figura 11.8:

- O polígono circunda, completamente, a área.
- O polígono intercepta a área.
- O polígono está completamente contido na área, dito contido.
- O polígono está completamente fora da área, dito disjunto.

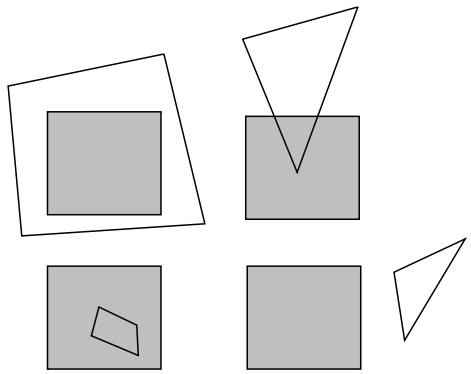


Figura 11.8: Quatro Relações de Polígonos com um Elemento de Área.

Polígonos disjuntos não têm influência alguma sobre a área de interesse. A diferença do polígono com a área (a parte do polígono interceptado que não está contida na área) também é irrelevante. A interseção do polígono com a área (a parte do polígono interceptado que está contida na área) é tratada da mesma forma que um polígono contido. Em quatro casos, uma decisão à respeito da área pode ser tomada, parando o processo de divisão da área:

- A interseção de todos os polígonos com a área é vazia. A área é preenchida com a cor do fundo.
- Existe apenas um polígono que intercepta a área ou apenas um polígono contido. Todos os outros são disjuntos. A área é preenchida, inicialmente, com a cor do fundo e depois a parte pertinente do polígono é rasterizada.
- Existe apenas um polígono que circunda a área. Todos os outros são disjuntos. A área é preenchida com a cor do polígono.

- Mais de um polígono intercepta, está contido ou circunda a área. Existe porém um polígono que circunda a área e está mais próximo ao observador. A área é preenchida com a cor deste polígono. A determinação da proximidade com o observador é realizada computando os valores das coordenadas $z$, dos planos dos polígonos em questão, nos quatro cantos da área. Se houver um polígono circundante cujos quatro valores calculados sejam menores que todos os outros, então este satisfaz o teste.

As áreas que não se enquadram nesses casos são divididas em mais quatro áreas iguais. Após esta subdivisão, os polígonos disjuntos ou circundantes à área original continuam a ser disjuntos ou circundantes, respectivamente.

Agora, qual o critério para parar o processo de subdivisão? Uma possibilidade é interrompê-lo quando a resolução do dispositivo for alcançada. Se, ao ser atingido o número máximo de subdivisões, nenhum dos quatro casos tiver ocorrido, então a profundidade dos polígonos pertinentes é calculada no centro do pixel. Aquele com menor coordenada $z$ define a colorização (o *shading*) da área.

## 11.9 Recorte Recursivo

Este algoritmo, criado por Weiler e Atherton, ao invés de recortar os polígonos contra áreas retangulares, como faz o algoritmo de subdivisão de área, utiliza um recorte geral: contra polígonos quaisquer (côncavos ou convexos e, possivelmente, com furos).

O primeiro passo do algoritmo ordena, por questão de eficiência, todos os polígonos em função de um critério baseado nas coordenadas $z$, por exemplo, máximo $z$ mínimo. Todos os polígonos são recortados contra o polígono $R$ mais próximo ao observador[6]. Nesse processo, são produzidas duas listas: $l_1$, com as partes, de cada polígono $p_i$, contidas em $R$ ($p_i \cap R$) e $l_2$, com as partes não contidas em $R$ ($p_i - R$). Todos os polígonos em $l_1$ que estão atrás de $R$ são eliminados, pois não são visíveis. Se um polígono em $l_1$ estiver mais próximo ao observador do que $R$ (estiver no mesmo semi-espaço que o observador, em relação ao plano de $R$), significa que a ordenação inicial não forneceu uma prioridade correta e ele é processado, recursivamente, para recortar os membros de $l_1$, novamente, contra ele. Quando este recorte recursivo termina, os polígonos em $l_1$ são exibidos. O algoritmo segue, então, processando os polígonos em $l_2$.

O recorte é sempre efetuado contra uma cópia do polígono original, e não contra um fragmento, uma vez que se supõe que é mais eficiente fazer o recorte desta forma, do que contra uma ou mais partes fragmentadas. Assim, sempre que um polígono é recortado, suas partes apontam para o polígono original.

O algoritmo utiliza, também, uma pilha, para poder tratar os casos de sobreposição cíclica, no qual um polígono está tanto na frente como atrás de outro. A pilha contém uma lista de polígonos que estão sendo utilizados como polígonos de recorte, mas cujo uso foi interrompido devido a um recorte recursivo. Se um polígono está na frente do polígono de recorte corrente mas está também na pilha, o processo de recursão pára, porque todas as suas partes, dentro e atrás deste polígono, já foram removidas.

[6] O recorte é feito considerando as projeções dos polígonos no plano de projeção ($z = 0$).

## 11.10 Traçado de raios

Os algoritmos de traçado de raios determinam a visibilidade de superfícies, traçando raios imaginários, a partir do ponto de vista, até um ponto de um objeto da cena (fig. 11.9). Um centro de projeção[7] e uma janela, sobre um plano de projeção arbitrário, são selecionados. A janela pode ser imaginada como estando particionada, formando um reticulado, cujos elementos correspondem aos pixels na resolução desejada. Para cada pixel, um raio é disparado do centro de projeção para o centro do pixel na cena. A cor do pixel corresponde a cor do objeto cuja interseção com o raio está mais próxima ao observador.

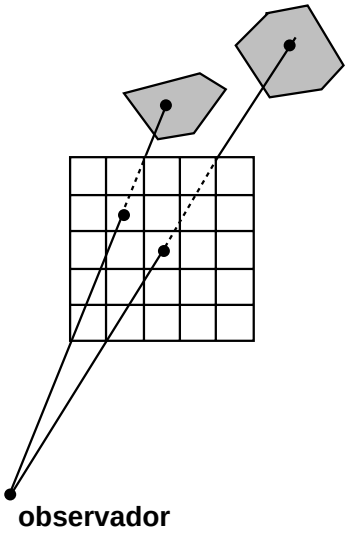



Figura 11.9: Traçado de Raios.

Algoritmos de traçado de raios podem ser adaptados para determinar a colorização (*shading*), sombras, transparências, avaliar operações *booleanas*, tratar reflexão e refração da luz. Para estender um algoritmo de traçado de raios para que seja capaz de tratar um novo tipo de superfície, basta escrever uma rotina que calcule a interseção de uma reta com a superfície.

## 11.11 Linhas Escondidas

Os algoritmos de linhas escondidas requerem que as linhas sejam arestas de polígonos e não, necessariamente, de poliedros. Estes algoritmos consideram apenas as linhas na fronteira de polígonos voltados para o observador[8] ($VO$).

O algoritmo de Appel define a invisibilidade quantitativa de um ponto, de uma linha,

[7] Também chamado ponto de vista ou observador.

[8] Cuja a componente $z$ da normal do polígono não é positiva .

como sendo igual ao número de polígonos, voltados para o observador, que o obstruem. Quando a linha passa por trás de um polígono $VO$, sua invisibilidade quantitativa é incrementada de uma unidade e quando ela sai de trás deste polígono, sua invisibidade quantitativa é decrementada de uma unidade. A linha está visível quando sua invisibilidade vale zero. Se não forem permitidos polígonos inter-penetrantes, a invisibilidade muda somente quando a linha passa por trás de uma linha de contorno[9].

Uma linha de contorno passa pela frente da aresta em consideração quando ela atravessa o triângulo formado pelo ponto de vista e os dois vértices da aresta. O ponto sobre a aresta onde ocorre o cruzamento é obtido projetando-se a linha de contorno sobre a aresta, o que pode ser obtido, recortando-se a aresta contra o plano definido pelo ponto de vista e a linha de contorno. O algoritmo requer que os polígonos sejam percorridos num sentido consistente, de forma a que o sinal da mudança de visibilidade possa ser determinado pelo sinal do produto vetorial entre a direção da aresta e a direção da linha de contorno.

Em primeiro lugar, o algoritmo computa a invisibilidade de um vértice, que funciona como a semente para o algoritmo. Isto é feito, calculando-se o número de interseções, entre o projetor que passa pela semente e os polígonos $VO$, mais próximos ao observador do que a semente. Em seguida, propaga-se este valor para as arestas que emanam da semente, incrementando ou decrementando este valor sempre que a aresta passar por trás ou pela frente de um polígono $VO$, respectivamente. Somente os segmentos com invisibilidade zero são desenhados. Quando um outro vértice é atingido, a invisibilidade neste ponto é propagada a todas as arestas que emanam deste vértice.

Vértices compartilhados por linhas de contorno necessitam de uma correção antes que se propague a sua invisibilidade. Isto porque, a invisibilidade pode mudar subtamente, uma vez que uma ou mais arestas emanando deste vértice podem ser obstruídas por um ou mais polígonos $VO$ compartilhando este vértice. Esta mudança pode ser levada em conta, testando-se a aresta contra os polígonos $VO$ que compartilham o vértice.

## 11.12 Exercícios

**11.1** *Que tipos de algoritmo de visibilidade são mais indicados a uma representação por bordo? E a uma representação implícita? Por que?*

**11.2** *No algoritmo de scan-line é realmente necessário reordenar as arestas da AET a cada nova linha de varredura ou basta apenas inserí-las ordenadamente? Em função da sua resposta, discuta a complexidade do algoritmo.*

**11.3** *Dê um exemplo em que, no algoritmo de z-sort, surge um ciclo infinito, onde os mesmos polígonos ficam se revezando no início da fila de prioridades. Qual a solução para este problema?*

**11.4** *Analise o problema da visibilidade ao longo de um raio para o caso de modelos representados em CSG.*

[9] Uma linha de contorno, ou de silhueta, é uma aresta compartilhada por um polígono voltado para o observador e outro de costas para o observador. Pode ser, também, uma aresta de um único polígono, que está voltado para o observador.

**11.5** *Discuta o cálculo das superfícies visíveis pelo método de A-buffer. Mostre quais são as aproximações utilizadas.*

# Capítulo 12

# Iluminação

O modelo de câmara virtual utilizado até agora serve para projetar a geometria da cena sobre a tela virtual e emprega as transformações de visualização para levar os objetos para sistemas de coordenadas apropriados às diversas etapas da visualização:

- recorte que é uma operação vetorial e serve para eliminar os objetos fora do campo de visão da câmara;
- visibilidade que é a responsável pela partição da tela virtual;
- e a rasterização (1D ou 2D, curvas ou superfícies) que descreve os objetos a partir dos pixels.

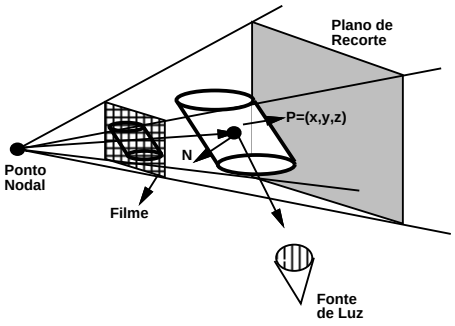



Figura 12.1: Modelo de Câmara Virtual.

Para completar o modelo da câmara virtual falta incorporar a iluminação da cena. Note-se que se trata de um modelo de câmara bastante geral que permite alterar até o plano de projeção[1] (o plano do filme de uma máquina fotográfica), o que numa câmara real é impossível.

[1] Numa câmara real, o filme está atrás do ponto nodal e por isso a imagem aparece invertida.

A criação de imagens realistas envolve, obrigatoriamente, a perfeita compreensão da interação da luz com a matéria e a utilização de modelos que, de alguma forma, simulem esta interação[2]. Determinados fenômenos de micro geometria que definem a textura de um objeto — e percebidos visualmente, por exemplo, como a rugosidade de uma superfície — são realizados a nível de visualização e não de modelagem.

A nível conceitual, pode-se imaginar uma função que a cada ponto do espaço associa a intensidade luminosa no ponto. Na prática, quando se utiliza um fotômetro para medir a iluminação de um ambiente, está-se calculando o valor desta função em um conjunto finito de pontos. Na geração de uma imagem, esta função deve ser projetada sobre a tela virtual para que se possa determinar a intensidade luminosa de cada pixel. Existe, então, uma função de colorização que associa a cada ponto da imagem a sua intensidade luminosa[3].

A função de iluminação depende do ponto em questão e de uma direção que define o observador. Assim, trata-se de uma equação a derivadas parciais com seis variáveis. Como a solução da equação de iluminação exata é complicada, utiliza-se uma equação (algébrica) aproximada que produz uma solução também aproximada. O modelo conceitual utilizado está explicitado na figura 12.2.

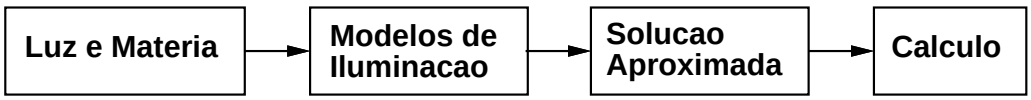



Figura 12.2: Modelo Conceitual.

Durante a rasterização, os pixels contidos no interior de um objeto são enumerados. Como os pixels possuem área, eles podem conter pedaços de vários objetos. Como retornar um valor de cor para a função de colorização sobre uma área? Este problema está relacionado com a necessidade de amostragem. O que deve ser determinado, então, é qual o método de amostragem a ser empregado. Este problema foi discutido na seção 10.3.

## 12.1 Modelos de Iluminação

Durante a interação da luz com um objeto, parte da energia é absorvida, parte é transmitida e parte é refletida na superfície do objeto. A componente refletida da energia luminosa incidente é que é a responsável pela sensação de cor produzida no cérebro de um ser humano. Quando a componente refletida perdeu energia de forma aproximadamente igual em todas as freqüências do espectro visível tem-se a cor cinza. Quando quase toda a energia é absorvida, tem-se a cor preta e quando quase toda a energia é refletida tem-se o branco.
A quantidade de luz refletida depende da:

- composição, direção e geometria da fonte de luz;

[2] A aparência metálica de um objeto, por exemplo, é obtida a partir desta interação.
[3] Na realidade, a função de colorização é a projeção da função de iluminação sobre a tela virtual.

- orientação da superfície do objeto em relação à fonte de luz;
- propriedades da superfície do objeto.

### 12.1.1 Modelo de Bouknight

A forma de reflexão da luz pode ser caracterizada como especular ou difusa. Um objeto que reflete luz preferencialmente em uma direção é chamado de objeto refletivo, enquanto aquele que reflete a luz iqualmente em todas as direções é dito difuso (fig 12.3).

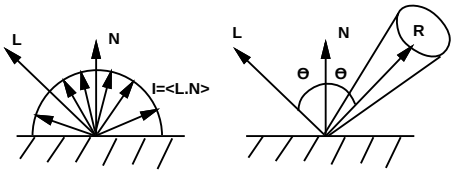



Figura 12.3: Reflexão Difusa e Especular.

A lei de Lambert estabelece que dada uma fonte pontual de luz e um difusor perfeito, a intensidade da luz refletida é proporcional ao cosseno do ângulo entre a normal à superfície e a direção de incidência da luz (fig. 12.4):

$$I = I_l k_d cos(\theta); \qquad 0 \leq \theta \leq \pi/2; \qquad 0 \leq k_d \leq 1.$$

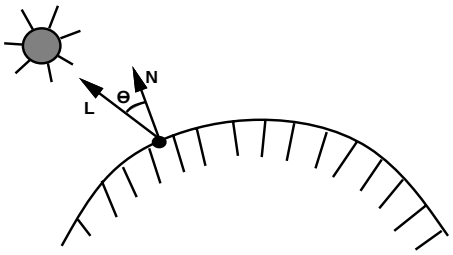



Figura 12.4: Lei de Lambert.

A variação da intensidade com o cosseno do ângulo entre a normal e a fonte de luz vem do fato de que a energia incidente por unidade de tempo e área em uma área infinitesimal $A$ é calculada baseado no fluxo de energia através de uma superfície de área $A'$, perpendicular à direção de incidência, em 1s (fig. 12.5). Assim $A' = Acos(\theta)$.

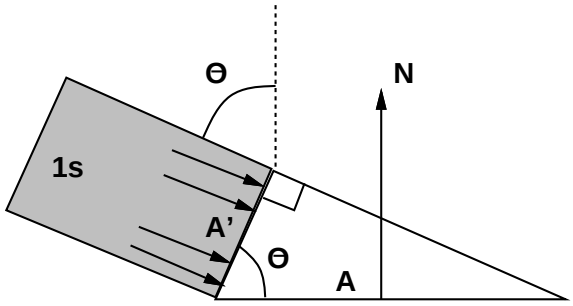



Figura 12.5: Variação com o Cosseno.

Na equação, $I_l$ é a intensidade da luz incidente e $k_d$ é um coeficiente de atenuação que depende do material.

A hipótese de uma fonte de luz pontual implica em que pontos que não recebem luz diretamente da fonte aparecem pretos. Em cenas reais, os objetos recebem luz de forma indireta, a partir de reflexões no ambiente. Para modelar de forma bastante simplificada esta contribuição indireta, pode-se adicionar um termo difuso constante à equação de iluminação:

$$I = I_a k_a + I_l k_d cos(\theta); \qquad 0 \leq k_a \leq 1.$$

O termo $I_a$ representa a intensidade da luz ambiente e é o responsável pela iluminação de objetos visíveis ao observador mas invisíveis a partir da fonte de luz. Este modelo de iluminação foi criado por Bouknight e é aplicado individualmente a cada componente de cor $(r, g, b)$. Tem-se então três constantes $(k_d = (k_{dr}, k_{dg}, k_{db}))$, que definem a cor do objeto. É comum, no entanto, escrever-se $k_d(O_{dr}, O_{dg}, O_{db})$, permitindo que $k_d$ escale a componente difusa refletida sem alterar a cor do objeto.

O problema do modelo de Bouknight é que ele despreza completamente a componente refletida de forma especular. A intensidade da luz refletida especularmente depende do ângulo de incidência, do comprimento de onda, do tipo de material e obedece à equação de Fresnel, que estabelece que para superfícies polidas perfeitas, o ângulo de incidência é igual ao ângulo de reflexão.

### 12.1.2 Modelo de Phong

Phong estendeu, de forma inteiramente empírica, o modelo de Bouknight para simular a reflexão especular. A componente especular é modulada por uma função que possui um máximo unitário e que decai rapidamente. O termo introduzido foi (fig. 12.6):

$$I_e = I_l k_e cos^n(\alpha); \qquad 0 \leq k_e \leq 1,$$

onde $\alpha$ é o ângulo entre o raio refletido e a direção de observação. A escolha da função cosseno deve-se ao fato dela aproximar, adequadamente, a distribuição espacial do realce percebido na iluminação de superfícies metálicas. Quanto maior for o $n$ menor a área do realce. Para superfícies altamente metálicas um valor típico seria 70.

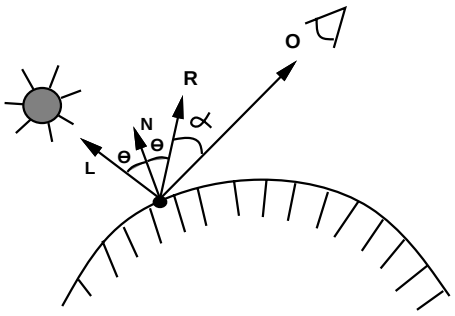



Figura 12.6: Modelo de Phong.

Quando existem $m$ fontes de luz na cena a equação resultante é:

$$I = I_a k_a + \sum_{j=1}^{m} I_l(k_d cos(\theta_j) + k_e cos^n(\alpha_j)) =$$

$$I_a k_a + \sum_{j=1}^{m} I_l(k_d < N, L_j > + k_e < R_j, O >^n).$$

Este modelo de iluminação é chamado de modelo local porque não leva em conta a troca de energia entre os diversos objetos em cena (a intensidade luminosa em um ponto depende apenas das fontes diretas de luz). Isto significa que as regiões da cena que estão na sombra não podem ser determinadas a partir do modelo de iluminação, obrigando que sejam executados algoritmos para determinação explícita das regiões sombrias.

## 12.2 Cálculo do Vetor de Reflexão

Para implementar um modelo de iluminação é necessário o cálculo do vetor de reflexão. Este cálculo é bastante simples e a partir da figura 12.7, vê-se que o vetor de reflexão é dado por:

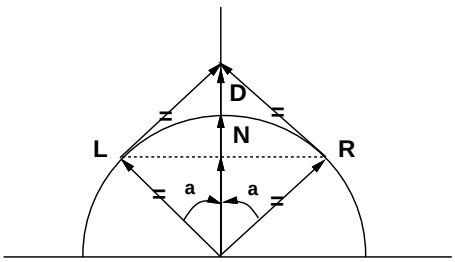
$$R = 2 < L, N > N - L,$$

onde $N$ é o vetor normal à superfície no ponto e $L$ é o vetor que indica a direção da fonte de luz.

## 12.3 Exercícios

**12.1** *Que tipo de fenômeno é modelado pela equação de Fresnell?*

**12.2** *Descreva o modelo de iluminação de Phong. Explique cada termo da equação de iluminação associada.*




$$|L| = |R| = |N| = 1$$
$$D = 2 < L, N >$$
$$cos(a) =< L, N >=< R, N >$$

Figura 12.7: Cálculo do Vetor de Reflexão.

**12.3** *Discuta como estender o modelo de Phong para incluir translucidez e refração (levar em conta o raio transmitido e o refletido).*

**12.4** *É muito comum na literatura o uso do vetor* $H = \frac{L+O}{|L+O|}$*, ao invés do vetor* $R$*, no cálculo da reflexão especular no modelo de iluminação de Phong, conforme pode ser visto na figura 12.8. Desta forma, o termo especular é controlado por* $cos(\beta)^n = < N, H >^n$*.*

*a) discuta geometricamente o problema.*

*b) enumere uma vantagem de se usar* $H$ *ao invés de* $R$*.*

*c) qual o efeito que essa mudança traz para a reflexão especular?*

*d) prove que se* $O$ *(observador) está no mesmo plano de* $L, N$ *e* $R$*, então* $\alpha = 2\beta$*.*

**12.5** *Analise os seguintes métodos de aproximação da integral de iluminação:*

*a) Método Direto*

*b) Traçado de Raios*

*c) Radiosidade*

**12.6** *Considere um observador localizado no ponto* $A = (10, 12, 3)$ *do espaço. Seja* $P = (1, 12, 3)$ *um ponto de uma esfera de centro* $C = (0, 10, 0)$ *e raio* $\sqrt{14}$*. Determine a direção de uma fonte de luz de modo que o realce especular por ela produzido seja máximo e esteja localizado no ponto* $P$*.*

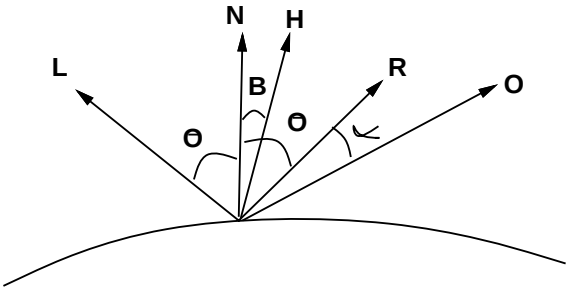



Figura 12.8: Uso do Vetor H.

**12.7** *Para criar o efeito de luz direcional, tipo luz de estúdio fotográfico onde um facho de luz é apontado para um certo objeto, Warn fez uma pequena alteração no modelo de iluminação de Phong. A idéia dele foi imaginar que a luz provém de uma fonte pontual e é refletida em um anteparo (que reflete apenas luz especular e tem $k_e = 1$), antes de se propagar pelo ambiente, conforme pode ser visto na figura 12.9. Com isto a intensidade da fonte de luz imaginária sobre o anteparo varia com a direção, sendo máxima quando $\gamma = 0$ e diminuindo à medida que $\gamma$ cresce.*

*a) Escreva a nova equação de iluminação usando um raciocínio similar ao usado por Phong na criação do termo especular do seu modelo de iluminação.*

*b) Adicione um teste para confinar a luz a um cone com vértice na fonte imaginária e formando um ângulo $\beta$ com $L'$.*

*c) Sugira um método para "retirar"o excesso de luz de uma certa área de uma superfície, usando um facho de luz "escura"(ou anti-luz). Para que isto serve?*

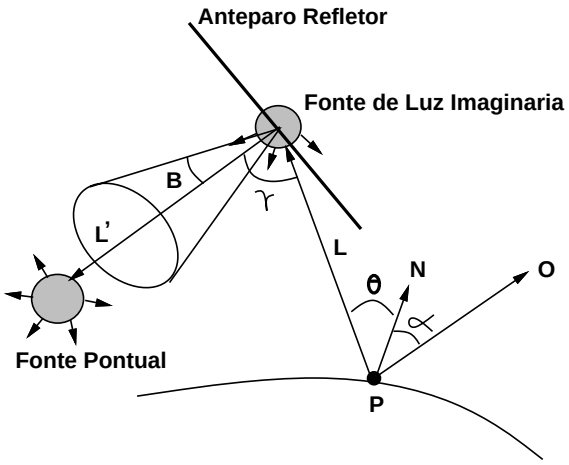



Figura 12.9: Facho de Luz direcional.

# Capítulo 13

# Colorização

No capítulo 12 foram estudados os modelos computacionais utilizados no cálculo da função de iluminação $I$ no espaço da cena. Chama-se de função de colorização, ou função de intensidade de cor $I_c$, a função definida no espaço da tela virtual e dada por

$$I_c(x, y) = I(x', y', z'),$$

onde $(x', y', z')$ é o único ponto visível do ambiente correspondente ao ponto $(x, y)$ da tela virtual[1].

O cálculo da função de colorização é que permite a geração efetiva dos valores de intensidade dos pixels da imagem. O domínio da função $I_c$ é a tela virtual, que consiste em um retângulo no qual são projetados os objetos geométricos do espaço da cena.

Como o objetivo é gerar uma imagem matricial, define-se na tela virtual um reticulado uniforme de pontos. Cada retângulo neste reticulado é chamado de pixel virtual ou, simplesmente, pixel. Cada pixel virtual corresponde a um pixel na imagem digital, sendo a unidade básica da tela virtual. Um subconjunto da tela virtual (um conjunto finito de pixels) será chamado de região digital. Deve-se calcular a função $I_c$ em cada pixel virtual de acordo com o modelo de iluminação utilizado. O cálculo da função de colorização é feito em quatro etapas:

- visibilidade;
- rasterização;
- amostragem;
- e o cálculo propriamente dito.

A visibilidade determina a região da tela virtual onde a função de colorização deve ser calculada. Essa região é formada pela projeção na tela virtual de todas as superfícies visíveis dos objetos em cena.

A descrição dos objetos em cena é dada geometricamente em uma forma vetorial (pontos, polígonos, equações paramétricas, etc.). Esses objetos definem superfícies que compõem a partição da tela virtual em regiões visíveis, nas quais a função de colorização deve ser

[1] Se a projeção da cena sobre a tela virtual for ortográfica, tem-se: $x = x', y = y'$.

calculada. O processo de rasterização consiste em determinar a região digital correspondente a essas regiões. Esse processo depende diretamente do esquema de representação utilizado para descrever os objetos em cena.

A região digital da tela virtual determinada no processo de rasterização é o domínio da função colorização. Desse modo, deve-se calcular o valor da função $I_c$ em cada pixel dessa região. Esse processo de amostragem, além de ser importante, é uma das etapas mais delicadas no cálculo da função $I_c$, pois não deve haver perda de informação no processo. Esse fenômeno de perda de informação é bastante comum em todo o processo onde há uma amostragem de um sinal contínuo. Existem diversos métodos para fazer uma amostragem efetiva e eficiente da função de colorização.

Após a escolha de um método de amostragem, é feito o cálculo da cor de cada pixel na tela virtual, através do cálculo da intensidade luminosa na região correspondente do espaço. Note-se que, em alguns modelos computacionais, o cálculo da função de colorização é feito juntamente com o cálculo da função de iluminação, por exemplo, no método de traçado de raios. De um ponto de vista conceitual, no entanto, é sempre útil pensar no processo em duas etapas: cálculo da função de iluminação e, em seguida, o cálculo da função de colorização na região digital da tela virtual.

Dependendo da representação dos objetos em cena, podem-se empregar técnicas de interpolação, de modo a evitar o cálculo da função de iluminação para todos os pixels. Se os objetos são representados por uma B-rep poligonal, então, a região visível, na tela virtual, é formada por um conjunto de polígonos. Existem três métodos clássicos para o cálculo da função de colorização em regiões poligonais: *flat* ou constante, Gouraud e Phong. Esses métodos foram introduzidos associados a um modelo local de iluminação.

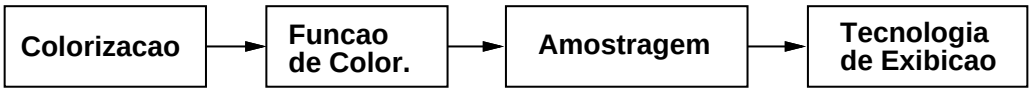



Figura 13.1: Modelo Conceitual.

## 13.1 Colorização Constante

A função de colorização determina a cor dos pixels na tela virtual. Considerando-se que os objetos em cena utilizam uma representação B-rep poligonal, significa que a normal em cada face é dada por um único vetor, definido pelo plano da face. Se for considerado também que a fonte de luz está no infinito, significa que cada face possui uma cor constante. Estas condições foram assumidas por Bouknight na definição de um método de colorização conhecido como *flat shading*. Embora simples, este método produz imagens facetadas onde não se tem, adequadamente, a sensação de curvatura nos objetos, pois nota-se, nitidamente, a aproximação poligonal utilizada.

## 13.2 Colorização de Gouraud

Levando em conta que o modelo B-rep poligonal é apenas uma aproximação do objeto em cena, Gouraud utilizou o conceito de interpolação poligonal para calcular o valor da função de colorização $I_c$ nos pontos interiores de cada polígono. Neste método, calcula-se uma normal em cada vértice da B-rep poligonal. Esta normal aproxima a normal à superfície original e depende da curvatura da superfície no ponto. Se a superfície estiver definida implicitamente, a normal é dada pelo gradiente calculado no vértice. Caso contrário, o cálculo da normal é efetuado a partir da média das normais de todas as faces que compartilham o vértice em questão, conforme pode ser visto na figura 13.2.

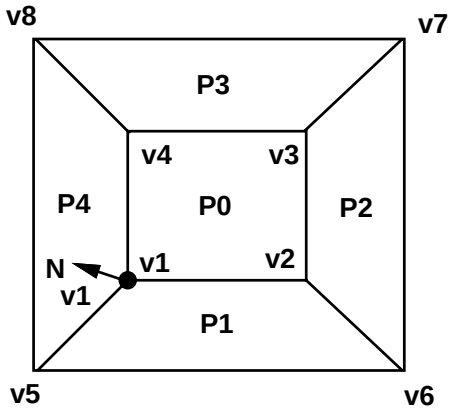



Figura 13.2: Cálculo da Normal em um Vértice.

$$N_{v1} = (a_0 + a_1 + a_4)i + (b_0 + b_1 + b_4)j + (c_0 + c_1 + c_4)k \quad \text{ou então}$$
$$N_{v1} = (\overline{v_1v_2} \times \overline{v_1v_4} + \overline{v_1v_5} \times \overline{v_1v_2} + \overline{v_1v_4} \times \overline{v_1v_5}),$$

de maneira que a normal aponte para o exterior do objeto.

Com estas normais, calculam-se as intesidades nos vértices das faces que passam a fazer parte do banco de dados que armazena a B-rep. Durante a rasterização dos polígonos correspondentes às projeções das faces sobre a tela virtual, as intensidades são interpoladas ao longo de cada aresta e ao longo dos segmentos de varredura.

A técnica de interpolação poligonal é feita da seguinte forma:
Sejam $P$ um polígono plano, fechado, com vértices $v_1, v_2, ..., v_n \in \Re^n$; $F : \Re^n \to \Re$ uma função; $p$ um ponto de $P$; e $r$ uma reta contendo $p$, contida no plano de $P$. Se a reta $r$ intercepta $P$ nas arestas $\overline{v_{i-1}v_i}$ e $\overline{v_{j-1}v_j}$, então, define-se:

$$F(p) = \text{ lerp } [\text{ lerp } (\overline{v_{i-1}, v_i}),\ \text{lerp } (\overline{v_{j-1}, v_j})],$$

onde lerp $(a, b)$ indica a interpolação linear no segmento $\overline{ab}$ (fig. 13.3).

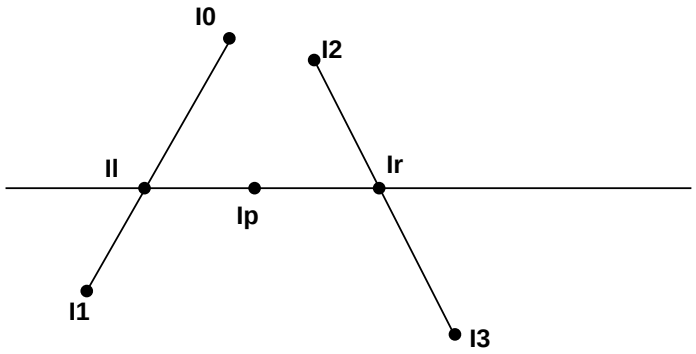



Figura 13.3: Interpolação Poligonal.

$$\begin{aligned} I_l &= \frac{y - y_0}{y_1 - y_0}(I_1 - I_0) + I_0, \\ I_r &= \frac{y - y_2}{y_3 - y_2}(I_3 - I_2) + I_2, \\ I_p &= \frac{x - x_l}{x_r - x_l}(I_r - I_l) + I_l. \end{aligned} \tag{13.1}$$

Se $P$ é um triângulo e $\lambda_1, \lambda_2, \lambda_3$ são as coordenadas baricêntricas do ponto $p$, ou seja,

$$p = \sum_{i=1}^{3} \lambda_i v_i, \qquad \lambda_i > 0, \qquad \sum_{i=1}^{3} \lambda_i = 1,$$

então: $F(p) = \sum_{i=1}^{3} \lambda_i F(v_i)$. Segue-se do resultado acima que a interpolação poligonal no triângulo não depende da reta $r$. Se, no entanto, $P$ não é um triângulo, então, o valor $F(p)$ calculado pelo método da interpolação poligonal não é único, dependendo, pois, da escolha da reta $r$.

Este método de interpolação de intensidades produz, na maioria dos casos, resultados bastante superiores aos obtidos com a colorização constante. Porém a silhueta[2] dos objetos ainda é percebida como estando linearizada. Isto se deve ao chamado Match Banding, um efeito resultante da maior sensibilidade do sistema visual humano à taxa de variação (a derivada) da intensidade. O método de Gouraud tem várias deficiências, dentre elas, a mais importante é desprezar a variação da normal no interior do polígono. Por essa razão, pontos de reflexão especular intensa (*highlights*), que ocorrem em pontos interiores de uma face, desaparecem.

[2]Definida pelo conjunto de arestas que só pertencem a um único polígono.

## 13.3 Colorização de Phong

Para atenuar o problema da colorização de Gouraud, Phong propôs um método de interpolação que, ao invés de interpolar as intensidades, interpola diretamente as normais (ao longo das arestas e ao longo dos segmentos de varredura). Os resultados obtidos são excelentes, porém paga-se um alto preço computacional. A função de iluminação deve ser avaliada no ponto da cena tri-dimensional que corresponde a um certo pixel. Pior ainda, como as transformações de visualização não preservam ângulos, esta avaliação deve ser levada a cabo no espaço da cena, ou em qualquer espaço isométrico a ele, por exemplo o espaço da câmara. Isto acarreta a necessidade de inverter-se a transformação de visualização de forma a mapear o ponto de volta ao espaço da cena.

Os métodos baseados em interpolação são aplicados no sistema de coordenadas normalizadas, após a transformação perspectiva, que comprime o eixo $z$ de forma não linear. Desta forma, a interpolação linear (ou poligonal) produz um incremento constante entre uma linha de varredura e outra, mas, devido ao encurtamento da transformação perspectiva, este incremento, ao longo do eixo $z$, aumenta, à medida em que se aproxima do centro de projeção, como se ve na figura 13.4. Assim, se

$$y_s = \frac{y_1 + y_2}{2}, \qquad \text{então } I_s = \frac{I_1 + I_2}{2}, \qquad \text{mas } z_s \neq \frac{z_1 + z_2}{2}.$$

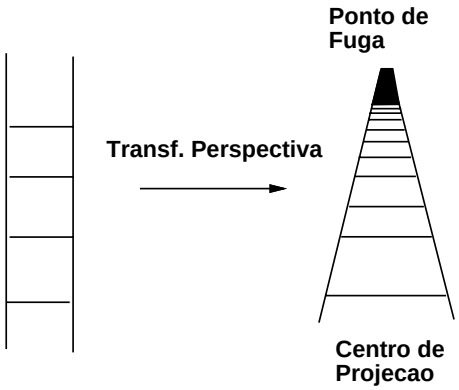



Figura 13.4: Encurtamento da Transf. Perspectiva.

Além disso, tanto o método de Gouraud como o de Phong utilizam uma interpolação poligonal. Se o polígono não for um triângulo, o valor obtido depende da posição do polígono no espaço da cena, ou seja, da orientação do polígono em relação à linha da tela. Esse fato pode criar efeitos indesejáveis em uma animação.

## 13.4 Colorização por Traçado de Raios

Um método que faz o cálculo da função de iluminação de forma conjugada com o cálculo da função de colorização é o método de traçado de raios (*ray tracing*). Esta classe de algoritmo surgiu em 1970, a partir do trabalho de Roth, no contexto da determinação da visibilidade (consulte-se o cap. 11). Nesta época, devido as limitações do *hardware* disponível, o método não foi muito utilizado até 1980, quando Whitted propôs a sua utilização para fazer a integração da equação de iluminação. Em linhas gerais, o método traça raios a partir do observador até cada pixel[3]. Estes raios, ao se chocarem com as superfícies dos objetos, são desviados de acordo com a lei da reflexão e da refração.

Tipicamente, a implementação emprega recursão. A partir do segundo nível, as contribuições das intensidades luminosas, provenientes dos raios secundários, no ponto original (da primeira colisão) são atenuadas. O modelo de iluminação utilizado continua sendo um modelo local, como o modelo de Phong, porém passa a existir uma interação global, uma vez que são considerados (alguns) raios de luz que chegam ao objeto de forma indireta, a partir de multi-reflexões, conforme pode ser visto na figura 13.5.

A grande deficiência do método de traçado de raios é ignorar completamente a interação luminosa difusa. Isto cria imagens altamente refletivas que evidenciam, claramente, o método que as gerou.

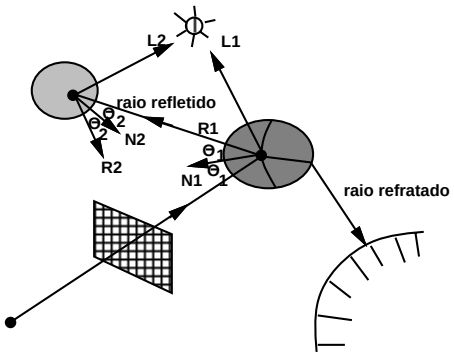



Figura 13.5: Traçado de Raios.

## 13.5 Integração da Equação de Iluminação

Uma outra forma de integrar a equação de iluminação é através do método de radiosidade criado por Goral em 1982. Este é um método de relaxação e é bastante eficaz para calcular a interação difusa entre os objetos em cena, a partir da discretização do ambiente em elementos onde a intensidade luminosa pode ser considerada constante. É calculado, então,

[3]Isto porque só interessam os raios que chegam ao olho. Daí o traçado no sentido inverso.

um valor de intensidade para cada elemento, considerando-se a troca de energia entre todos os elementos e assumindo-se um ambiente fechado.

A combinação do método de traçado de raios com o método de radiosidade produz as melhores imagens já geradas até hoje. Note-se que isto apenas melhora o cálculo da função de iluminação.

Uma outra abordagem possível para melhorar a qualidade das imagens é melhorar a própria equação de iluminação. Uma equação de iluminação exata, chamada *"The Rendering Equation"* foi proposta por Jim Kajya. Trata-se de uma equação integral (a função de iluminação está dentro do integrando). As técnicas para a resolução de equações integrais são estudadas em cálculo numérico e, basicamente, existem duas abordagens possíveis: Monte Carlo e Elementos Finitos. A primeira técnica é empregada no método de traçado de raios distribuído e a segunda no método de radiosidade.

## 13.6 Exercícios

**13.1** *Prove que o resultado da interpolação de dados associados aos vértices de um polígono, ao longo das suas arestas e das linhas da imagem, frente ao método de rasterização incremental, independe de orientação no caso de triângulos. Mostre que isso não é verdade no caso de polígonos com mais de três lados. Qual a conseqüência deste fato numa animação?*

**13.2** *Apresente um caso no qual o uso de flat shading produz um resultado melhor do que o método de Phong ou Gouraud.*

**13.3** *Qual o efeito da compressão não linear, ao longo do eixo z, após a transformação perspectiva, na interpolação de quantidades que não sejam distâncias, por exemplo, intensidade luminosa? Como atenuar este problema?*

**13.4** *É possível levar em conta, de alguma forma, o realce no interior de polígonos usando a interpolação de Gouraud?*

**13.5** *Discuta a relação entre a resolução de aproximações poligonais de uma superfície e o fenômeno de Match banding causado pelo cálculo da iluminação com a interpolação de Gouraud.*

**13.6** *Considere um esquema para o cálculo de iluminação usando um mapa de cor de 24 bits associado a uma imagem de 12 bits por pixel, de tal modo que mudanças na iluminação impliquem apenas em mudança no mapa de cor.*

*a) Descreva o método.*

*b) Qual a natureza da informação armazenada na imagem?*

*c) Descreva o cálculo da iluminação.*

*d) Quais as limitações do método?*

**13.7** *Verifique que o cálculo de normais em um vértice, utilizando a equação dos planos dos polígonos que compartilham o vértice, ou o produto vetorial das arestas que incidem no vértice, dão resultados diferentes, tanto em direção como em magnitude. Explique o motivo.*
*Sugestão: utilize um tronco de pirâmide.*

# Capítulo 14

# Mapeamentos

A interação da luz com a superfície de um objeto define o que se costuma chamar da textura do objeto. Se, por algum processo, esta interação for manipulada, a percepção da textura é modificada. Isto permite a criação de efeitos interessantes. Os objetivos do mapeamento são:

- alterar informações estruturais;
- modelar a micro geometria do objeto;
- alterar a função de iluminação.

Os parâmetros da função de iluminação (normal, coeficiente de reflexão difusa, coeficiente de reflexão especular, etc.) podem ser modulados por uma função de textura. Os problemas que se colocam de forma natural são:

- Quais os parâmetros a serem modulados?
- Qual a função de mapeamento?
- Como resolver os problemas de amostragem e reconstrução?
- Como gerar a textura?

De acordo com o parâmetro modulado têm-se tipos de mapeamentos diferentes a saber:

- modulação do coeficiente de reflexão difusa (modulação da "cor" do objeto) → mapeamento de textura;
- modulação do coeficiente de reflexão especular → mapeamento de ambiente (*environment mapping*);
- modulação da normal → mapeamento de rugosidade (*Bump Mapping*);
- modulação do coeficiente de transparência → mapeamento de transparência;
- modulação do coeficiente de refração → mapeamento de refração.

## 14.1 Mapeamento de Textura

O mapeamento de textura corresponde à colagem de um decalque sobre a superfície de um objeto. É claro que o decalque deve ser esticado de forma a se adaptar ao objeto.

O espaço da textura (o decalque) é um subconjunto do $\Re^n$ e o mapeamento (responsável pelo esticamento do decalque) é uma função que associa um ponto do $\Re^n$ a um ponto sobre a superfície de um objeto. Este ponto é mapeado em um pixel pela transformação de visualização (fig. 14.1). O conjunto $A$ da figura 14.1 é dado por: $A = \{p/T(p) = P\}$. Uma forma de atribuir-se a cor apropriada a um pixel, é filtrando-se a imagem inversa do pixel (que pode ser determinada a partir de $T^{-1}.T_{vis}^{-1}$).

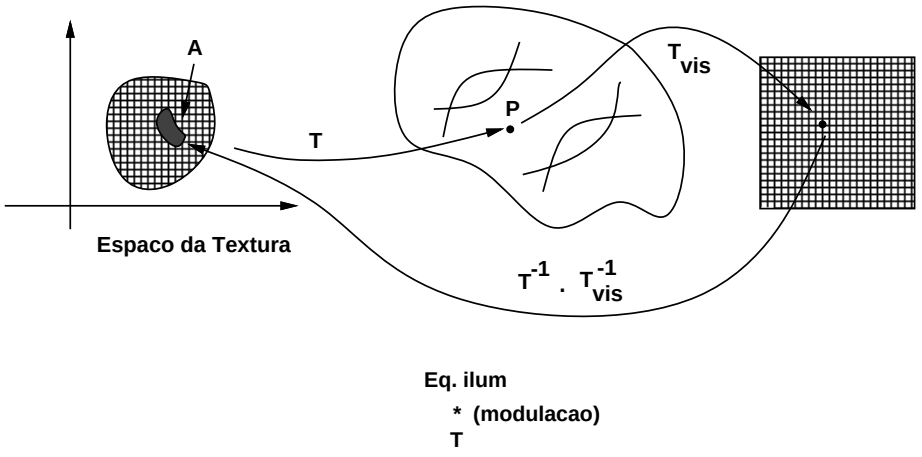



Figura 14.1: Mapeamento.

Pela dificuldade de encontrar-se uma função de mapeamento adequada, Jim Blinn sugeriu um mapeamento em duas etapas: em primeiro lugar mapeia-se a textura sobre uma superfície intermediária simples (cilindro, cubo ou esfera) que envolve o objeto. Posteriormente mapeia-se esta superfície sobre a superfície do objeto, conforme indicado na figura 14.2.

Em um mapeamento tri-dimensional costuma ser difícil gerar a textura. No entanto, a função de mapeamento é a indentidade. O resultado obtido em uma animação é muito bom (imagine-se o corte de uma madeira. O que aconteceria com os seus veios se o mapeamento não fosse 3D?).

## 14.2 Mapeamento de Rugosidade

A rugosidade de uma superfície é função do padrão da luz refletida. O mapeamento de rugosidade, criado por Jim Blinn, perturba a normal à superfície no ponto onde se está

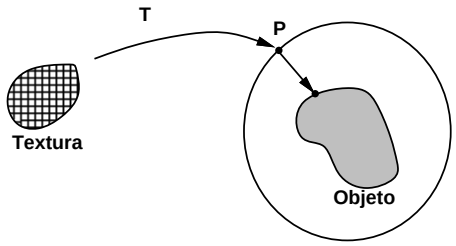



Figura 14.2: Superfície Intermediária para Mapeamento.

calculando a intensidade luminosa. Isto é equivalente a ter-se uma superfície ondulada (fig. 14.3).

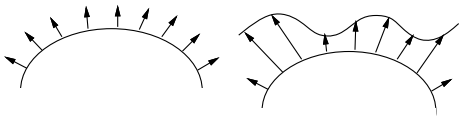


Figura 14.3: Mapeamento de Rugosidade.

## 14.3 Mapeamento de Ambiente

O mapeamento de ambiente pode ser empregado para simular o método de *ray tracing*. A idéia é modular o coeficiente de reflexão especular e difusa a partir da imagem do ambiente. A dificuldade deste tipo de mapeamento é gerar o mapa. A função de mapeamento pode empregar, por exemplo, uma superfície intermediária. Costuma-se empregar um cubo ou uma esfera.

Caso utilize-se uma esfera como superfície intermediária, um centro de projeção é escolhido, à partir do qual o ambiente a ser refletido é mapeado sobre a superfície de uma esfera que circunda os objetos da cena. O ambiente mapeado pode ser tratado como como uma textura bi-dimensional. Em cada ponto do objeto, o mapa de textura pode ser endereçado pelas coordenadas polares do vetor obtido refletindo $O$ em torno de $N$ (fig. 14.4).

Vai-se considerar um sistema de coordenadas com origem no centro da esfera, e a notação corrente de cartografia, onde o eixo $z$ aponta para o polo norte; a longitude, que corresponde aos meridianos, é um ângulo $\lambda \in [-180^\circ, 180^\circ)$ medido no sentido anti-horário à partir do

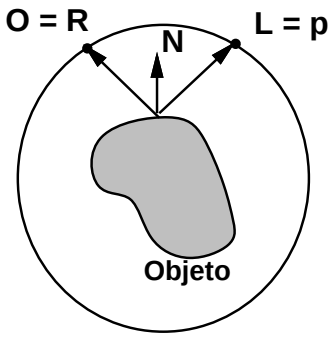



Figura 14.4: Mapeamento de Ambiente.

eixo $x$; a latidude, que corresponde aos paralelos, é um ângulo $\phi \in [-90°, 90°]$ medido à partir do eixo $z$, com $0°$ correspondendo à linha do equador. A geração do mapa de textura pode ser feito através de um tipo qualquer de projeção empregado em cartografia. O mais simples é uma projeção cilíndrica retangular onde $x = \lambda$ e $y = \phi$.

A forma alternativa utiliza seis projeções sobre os lados de um cubo alinhado com o sistema de coordenadas. Assim, a maior coordenada do vetor de reflexão normalizado permite endereçar o lado apropriado do cubo.

# Referências Bibliográficas


[1] J. D. Foley, A. Van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics - Principles and Practice.* Addison Wesley, 1989.

[2] Newman. *Principles of Interactive Computer Graphics.* McGraw-Hill, 1979.

[3] Alan Watt. *Fundamentals of Three-dimensional Computer Graphics.* Addison Wesley, London, 1988.

[4] Gerald Farin. *Curves and Surfaces for Computer Aided Geometric Design.* Academic Press, 1990.

[5] Martti Mäntylä. *Introduction to Solid Modeling.* Computer Science Press, 1988.

[6] Christoph M. Hoffman. *Geometric & Solid Modeling.* Morgan Kaufmann Publishers, Inc., San Mateo, California, 1989.

[7] Richard H. Bartels, John C. Beatty, and Brian A. Barsky. *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling.* Morgan Kaufmann Publishers, Inc., San Mateo, California, 1987.

[8] Andrew Glassner, editor. *An Introduction to Ray Tracing.* Academic Press, San Diego, California, 1989.

[9] R. A. Hall. *Illumination and Color in Computer Generated Imagery.* Springer-Verlag, New York, 1989.

[10] Rafael C. Gonzalez and Paul Wintz. *Digital Image Processing.* Addison Wesley, 1987.

[11] George Wolberg. *Digital Image Warping.* IEEE Computer Society Press, 1990.

[12] S. Harrington. *Computer Graphics, A Programming Approach.* McGraw-Hill, 1983.

[13] Andrew Glassner, editor. *Graphics Gems I.* Academic Press, San Diego, California, 1990.

[14] James Arvo, editor. *Graphics Gems II.* Academic Press, San Diego, California, 1991.

[15] David Kirk, editor. *Graphics Gems III.* Academic Press, San Diego, California, 1992.

[16] Paul Heckbert, editor. *Graphics Gems IV.* Academic Press, Boston, 1994.

[17] Alan Paeth, editor. *Graphics Gems V.* Academic Press, Boston, 1995.

[18] Hanan Samet. *Applications of Spatial Data Structure.* Addison Wesley, 1990.

[19] Theo Pavlidis. *Graphics and Image Processing.* Computer Science Press, 1982.

[20] David F. Rogers. *Procedural Elements for Computer Graphics.* McGraw-Hill, 1985.