

VOL. 3

*Reuniões em
Matemática Aplicada e
Computação Científica*

**X Congresso Nacional de
Matemática Aplicada e Computacional**

Gerald Jean F. Banon

MINICURSO

Gramado RS - 21 a 25 de Setembro de 1987

REUNIÕES EM MATEMÁTICA APLICADA E COMPUTAÇÃO CIENTÍFICA

A coleção Reuniões em Matemática Aplicada e Computação Científica da Sociedade Brasileira de Matemática Aplicada e Computacional - SBMAC - destina-se à divulgação de trabalhos selecionados, apresentados em encontros nas diversas áreas de ciências matemáticas apoiados por esta Sociedade.

São seus objetivos incentivar o aprimoramento dos trabalhos que serão publicados posteriormente em forma definitiva e promover a rápida disseminação de projetos de desenvolvimento e pesquisas em execução, contribuindo para maior intercâmbio científico e inovação tecnológica.

Coordenador Responsável:

Julio Cesar Ruiz Claeysen

SBMAC

Rua Lauro Muller, 455

22290 - Rio de Janeiro - RJ

BASES DA COMPUTAÇÃO GRÁFICA

Gerald Jean F. Banon

Ministério da Ciência e Tecnologia - MCT

Instituto de Pesquisas Espaciais - INPE

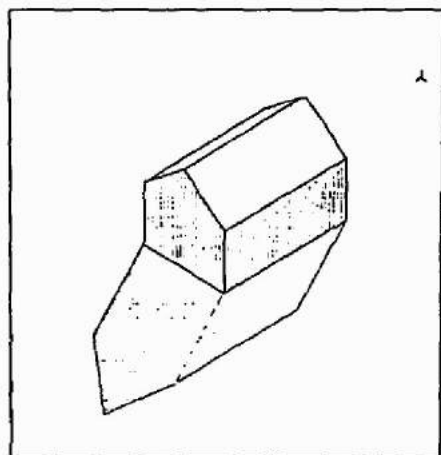
X Congresso Nacional de Matemática
Aplicada e Computacional

Gramado 1987

À Gabriela, Lise e Martha

ã Michelle

ã René com eterna saudade



PREFÁCIO

O texto apresentado aqui foi redigido para servir de apoio a um minicurso intensivo (10 a 12 horas) de introdução à computação gráfica, a ser dado por ocasião do X Congresso Nacional de Matemática Aplicada e Computacional (CNMAC - 21 a 25 de setembro de 1987, Gramado - RS).

O esquema desse curso foi montado ao longo destes últimos anos. Inspirou-se em um curso dado em março de 1985 por Charles H. Aspel (Professor Adjunto de Ciência da Computação na Universidade da Cidade de Nova York e pesquisador no Instituto de Pesquisas de Sistemas da IBM na mesma cidade) no "Latin American Systems Research Institute" (LASRI) da IBM Brasil no Rio de Janeiro.

O material ilustrando os aspectos da computação gráfica, que ocupa um lugar importante neste texto, foi criado nos primeiros meses de 1986 por ocasião da preparação do curso intitulado "Bases da Computação Gráfica", dado pelo autor, em abril de 1986 no LASRI.

Finalmente, este curso foi dado, na sua forma atual, no Instituto de Pesquisas Espaciais (INPE), em São José dos Campos, para o pessoal do Departamento de Processamento de Imagens (DPI).

O livro-texto deste mini-curso é o do Foley e Van Dam intitulado "Fundamentals of Interactive Computer Graphics", Addison-Wesley, 1984. Uma grande parte da matéria tratada no presente texto, apresenta-se como um apanhado geral desse excelente livro. No entanto, algumas seções são praticamente originais (seções 1.5, 1.6, 2.2, 3.2, 5.1 e 8.2) e outras mais desenvolvidas (seções do capítulo 4).

As seções 2.2 e 3.2 trazem um material matemático sobre vetores e sistemas de coordenadas que o autor julgou importante inserir, para entender bem, em particular, o problema da orientação do plano e do espaço e suas conseqüências.

Todas as ilustrações apresentadas no texto comportam duas partes. A primeira parte, chamada de "transparência" (porque é mostrada como tal durante o curso) apresenta, em APL, de forma concisa, o algoritmo que gerou na segunda parte a "figura", ou seja, o resultado da saída da gráfica.

O autor é muito grato a Charles H. Aspel que o iniciou na computação gráfica, ao antigo Centro Científico da IBM em Brasília que lhe deu as condições de montar as ilustrações deste curso, a Almir L. de Almeida que permitiu que sobre a orientação do autor fosse desenvolvido por Marcelo L. Braunstein, estagiário do LASRI, uma versão do GKS funcionando de modo interativo debaixo do APL, à Comissão Organizadora do X Congresso Nacional de Matemática Aplicada e Computacional, e, em particular, a Luiz Alberto Vieira Dias, através de quem foi feito o convite para ministrar um minicurso em Computação Gráfica, o que encorajou o autor escrever este texto e permitiu sua publicação.

Finalmente, o autor deseja agradecer ao INPE e em particular a Ricardo Cartaxo Modesto de Souza pelo apoio recebido na preparação deste curso, a Maria Masae, Angela Blanco Gomes e Janete da Cunha pelo trabalho datilográfico e a Silvio Pereira Coimbra, Joaquim Godoi Filho, Carlos Roberto dos Santos, Lúcia Eutímia Vianna e Ana Lúcia Castro pelos desenhos apresentados.

São José dos Campos, agosto de 1987,

G. B.

SUMÁRIO

	<u>Pág.</u>
1 - INTRODUÇÃO	01
1.1 - Computação Gráfica e Tratamento de Imagens	01
1.2 - Breve Histórico da Computação Gráfica	02
1.3 - Equipamentos Gráficos de Saída	04
1.4 - Equipamentos Gráficos de Entrada	13
1.5 - Alguns Aspectos do GKS	16
1.6 - Alguns Aspectos do APL	21
2 - ASPECTOS GEOMÉTRICOS DA COMPUTAÇÃO GRÁFICA EM 2D	27
2.1 - Exemplos de Desenhos em 2D	27
2.2 - Transformações geométricas em 2D	31
2.3 - Determinação de Uma Transformação de Normalização	50
3 - ASPECTOS GEOMÉTRICOS DA COMPUTAÇÃO GRÁFICA EM 3D	57
3.1 - Transformações Geométricas em 3D	57
3.2 - Equações de Uma Reta e de Um Plano	60
3.3 - Visão em 2D de Objetos em 3D	69
4 - ESTRUTURAS DE DADOS PARA POLÍGONOS E MALHAS DE POLÍGONOS	85
4.1 - Introdução	85
4.2 - Representação por Polígonos Explícitos	86
4.3 - Representação por Polígonos Definidos como Apontadores numa Lista de Vértices	86
4.4 - Representação por Lados Explícitos	89
4.5 - Exemplos de Aplicações	90
5 - ELIMINAÇÃO DAS LINHAS ESCONDIDAS	103
5.1 - Visão em 2D de Poliedros Convexos	103
5.2 - Cerceamento de Segmentos	113

	<u>Pág.</u>
6 - ELIMINAÇÃO DOS POLÍGONOS ESCONDIDOS	131
6.1 - Conversão Vetorial/Raster	131
6.2 - Teste das Extensões	137
6.3 - Algoritmo da Ordenação por Profundidade	140
6.4 - Algoritmo da Linha Exploratória para Superfícies Escondidas ..	141
6.5 - Algoritmo do z-buffer	142
6.6 - Algoritmo de Subdivisão de Área	144
6.7 - Eficiência dos Algoritmos e Problema do "aliasing"	151
7 - SUPERFÍCIES CURVAS	155
7.1 - Curvas Cúbicas Paramétricas	155
7.2 - Superfícies Cúbicas Paramétricas	169
8 - SOMBREAMENTO E COR	177
8.1 - Modelos de Sombreamento	177
8.2 - Determinação da Zona de Sombra	181
8.3 - Lançamento de Raios	187
8.4 - Tom de Cinza e Cor	192
ANEXO	203
REFERÊNCIAS	209
ÍNDICE	211

1- INTRODUÇÃO

1.1- COMPUTAÇÃO GRÁFICA E TRATAMENTO DE IMAGENS

A computação gráfica ("Computer graphics" em Inglês, "Infographie" em Francês) é parte da Ciência da computação e é a área de estudo de alguns aspectos da comunicação entre o homem e o computador. O aspecto principal abordado pela computação gráfica é o da comunicação visual no sentido "máquina-homem" através da síntese de imagens em dispositivos de saída apropriados.

Em vez de apresentar um resultado numérico na forma de uma tabela com uma ou duas entradas, esse resultado é apresentado na forma de um gráfico em duas ou três dimensões (2D ou 3D) facilitando assim sua interpretação pelo homem. A síntese de gráficos foi uma das primeiras aplicações da computação gráfica na época em que a computação significava essencialmente o cálculo científico.

Com a diversificação das aplicações, a computação gráfica tomou mais importância. Novas técnicas foram introduzidas para a síntese de desenhos industriais, de plantas (e.g. edifícios, processos de fabricação), a síntese de imagens com realismo visual e até mesmo de seqüências de imagens para a realização de animações.

No processo de síntese, as imagens são criadas a partir do modelo de um objeto ou até de uma cena constituída por vários objetos. Em muitas situações o modelo é criado interativamente através do controle manual de dispositivos de entrada que permitem a comunicação no sentido "homem-máquina". Essa interatividade é o segundo aspecto abordado pela computação gráfica.

Hoje em dia, a computação gráfica tem um papel importante em inúmeras áreas, por exemplo: criação de gráficos em negócios, ciência e economia, realização de mapa em cartografia, elaboração de desenhos de novos produtos na área industrial (o CAD/CAM - Computer Aided Design / Computer Aided Manufacturing), visualização de simulação e controle de

processo nas áreas industrial, de produção de energia e de transporte, e criação de animação para simuladores de vôo ou realização de "clips" publicitários.

Mas a computação gráfica não é a única área que trabalha com imagens. O processamento digital de imagens também trabalha com imagens mas não tanto do ponto de vista de síntese, quanto de análise. Os objetos de estudo são as imagens digitalizadas obtidas do mundo real, por exemplo: imagens da superfície da terra obtidas por satélites. O objetivo do processamento de imagens é melhorar a qualidade visual das imagens (e.g. modificação do contraste), comprimir os dados numéricos representando as imagens, filtrar os ruídos e extrair temas ou objetos.

Durante muito tempo a computação gráfica e o processamento de imagens caminharam isoladamente. Hoje em dia começam a surgir aplicações mais sofisticadas que envolvem técnicas de ambas as áreas. Por exemplo, o problema da reconstrução de um objeto em três dimensões (e.g. uma artéria) a partir de um conjunto de suas projeções, é resolvido usando tanto técnicas de análise de imagens (e.g. transformadas de Fourier, reconhecimento de formas) quanto de síntese (e.g. projeção em perspectiva, sombreamento).

1.2- BREVE HISTÓRICO DA COMPUTAÇÃO GRÁFICA

A computação gráfica nasceu junto com as primeiras máquinas eletrônicas de computação. Já em 1950 no MIT foi acoplado um tubo de raios catódicos a um computador. A introdução da caneta luminosa ("light pen") alguns anos depois permitiu identificar elementos pictoriais no tubo de raios catódicos e assim ter um meio de interagir diretamente com a imagem. Na mesma época surgiram as primeiras plotadoras controladas por computadores digitais. Foi no início dos anos 60 que começou o uso de tubos de raios catódicos a cores.

Depois de um pouco mais de dez anos de progresso na área de desenvolvimento de equipamentos de entrada/saída apareceu o primeiro

sistema de comunicação gráfica: o "SKETCHPAD". Este sistema [Sutherland, 1963] já tinha muitas das características dos sistemas de hoje, por exemplo: uso de dados estruturados para representar hierarquias de elementos pictoriais e de técnicas interativas para desenhar ou trabalhar com "menus" diretamente na tela do monitor.

No meio dos anos 60 foram comercializados os primeiros monitores vetoriais (cf. seção 1.3) oferecendo condições para o surgimento dos primeiros pacotes CAD/CAM na área de construção automobilística. No final dos anos 60 apareceram os monitores vetoriais com tubo de memória.

Com o progresso tecnológico na fabricação de memórias surgiram nos meados dos anos 70 os primeiros monitores a varredura (cf. seção 1.3) trazendo novas possibilidades para criar e visualizar verdadeiras imagens.

Os anos 70 foram os anos dos primeiros pacotes gráficos e das primeiras conferências do SIGGRAPH ("Special Interest Group on Graphics"). Em 1977 foi proposta por um comitê essencialmente norte americano do ACM SIGGRAPH (ACM: "Association for Computing Machinery") a primeira padronização gráfica com o "Core Graphics System" conhecido como o CORE. O objetivo era de propor para as aplicações em 2D ou 3D um padrão comportando um conjunto de funções gráficas que não dependessem na sua utilização dos equipamentos gráficos envolvidos na aplicação, e permitindo assim uma verdadeira portabilidade dos aplicativos. Este objetivo é similar ao do FORTRAN na área das linguagens de programação. Mas ao contrário do que aconteceu com o FORTRAN que surgiu muito cedo e foi muito bem aceito, a definição de um padrão na área de computação gráfica mostrou-se e mostra-se ainda uma tarefa muito mais delicada. A dificuldade é que para ser amplamente aceito, um padrão não pode depender de convenções e práticas particulares. No fim dos anos 70, inspirado em parte pelo CORE, foi formulado, principalmente pelos europeus, um outro padrão gráfico chamado de GKS ("Graphic Kernel System") que deveria adaptar-se melhor a grande diversidade dos equipamentos gráficos de um lado e das aplicações potenciais do outro, através da introdução da noção de estação de trabalho (virtual). O GKS foi adotado pela ISO ("International

Standards Organization") como "Draft International Standard" e a ANSI ("American National Standard Institute") iniciou seu processo de adoção. Apesar da introdução do GKS representar um avanço importante, observa-se que ele sofre críticas na medida em que todas as possibilidades gráficas oferecidas por alguns novos equipamentos não podem ser bem aproveitadas. A proposta de um padrão GKS para aplicações gráficas em 3D deveria contribuir para melhorar sua aceitação.

A partir dos anos 80, com a chegada dos micros e seus aperfeiçoamentos constantes, as aplicações da computação gráfica deixaram de ser reservadas aos especialistas.

1.3- EQUIPAMENTOS GRÁFICOS DE SAÍDA

Os equipamentos gráficos de saída têm por função a produção ou exibição de textos, desenhos e/ou imagens. Estes são produzidos tomando como referência uma grade. Geralmente essa grade é regular, retangular ou quadrada, mas ela pode ser hexagonal ou de outro tipo. Uma grade pode ser vista como um conjunto de polígonos adjacentes. Os vértices dos polígonos são também chamados de pontos da grade.

Os equipamentos de saída são classificados em duas classes importantes.

Nos equipamentos do tipo vetorial, na posição dos pontos da grade pode-se, exibir um ponto físico ou um caracter alfanumérico, ou, o que é importante, exibir um segmento de reta (física) entre certos pares de pontos (ver figura 1.1). Daí o qualificativo de "vetorial".

Nos equipamentos do tipo a varredura, na posição dos pontos da grade deve-se exibir um padrão escolhido numa coleção de padrões possíveis (e.g. um ponto físico, um caracter alfanumérico, uma matriz binária de pontos). Mas não se pode exibir um segmento de reta entre dois pontos da grade. A outra grande diferença com o tipo vetorial é que todos os pontos da grade devem receber um padrão. Por essa razão, sempre

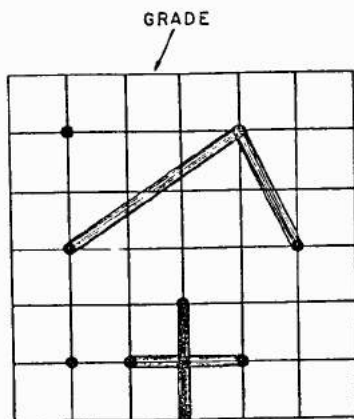


Figura 1.1 Exemplo de saída num equipamento do tipo vetorial. A saída é formada de pontos e segmentos de reta (ligando pontos da grade).

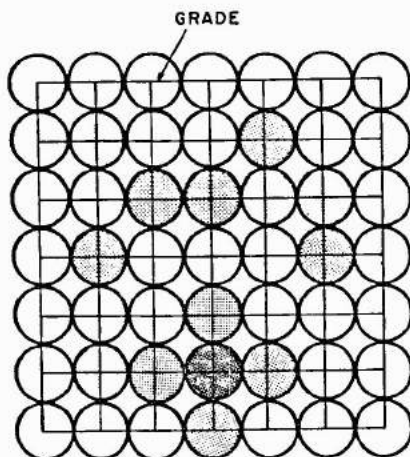


Figura 1.2 Exemplo de saída num equipamento do tipo "a varredura". A saída é formada de quatro tipo de pontos de brilho diferente (incluindo o "branco").

consta o padrão "branco" dentro da coleção de padrões (ver figura 1.2). Geralmente os pontos da grade são alinhados horizontalmente e os padrões são exibidos linha por linha. Daí o qualificativo de "a varredura".

As características importantes de um equipamento de saída são o tamanho da grade e o número dos seus pontos. A grade sendo geralmente regular, define-se a resolução do equipamento de saída com o número de pontos da grade por unidade de comprimento ou de superfície. A outra característica importante é o tempo necessário para efetuar uma saída. Nos equipamentos a varredura, este tempo é geralmente constante e é o tempo necessário para varrer todos os pontos da grade. Nos equipamentos vetoriais este tempo depende do próprio conteúdo da saída. Pode-se definir então um tempo médio para visitar um certo número de pontos da grade escolhidos aleatoriamente.

Em seguida são descritos alguns equipamentos de saída, procurando, classificando-os em tipo vetorial ou "a varredura".

As impressoras têm por função principal a impressão em papel de textos compostos de caracteres alfanuméricos. Estes caracteres formam os padrões que são impressos de acordo com uma grade retangular. Os padrões são pré-escolhidos ou programáveis (usa-se então padrões em forma de matrizes de pontos). A impressão é feita linha por linha o que leva a classificar as impressoras como equipamentos do tipo a varredura. Apesar da sua resolução baixa, as impressoras podem ser usadas para representar fluxogramas, gráficos ou até mesmo imagens. Neste caso, a escala de ton de cinza dos pontos da imagem é obtida, segundo a técnica de "meios-tons" (cf. seção 8.4), pela escolha dos padrões mais apropriados dentro dos disponíveis.

As plotadoras têm por função principal a produção em papel de desenhos ou imagens. Quando usam canetas móveis, as plotadoras são do tipo vetorial e são bem adaptadas para traçar desenhos a partir de segmentos de reta. As plotadoras eletrostáticas ou com jatos de tinta ("ink-jet") são do tipo "a varredura" e são bem adaptadas para produzir imagens digitais em preto e branco ou coloridas (cf. seção 8.4). Dentro

dessa classe de plotadoras tem-se também aquelas que usam a técnica de xerografia: um raio laser controlado por um micro-processador local cria uma distribuição de carga num tambor de selênio.

Como as plotadoras, os monitores com tubo de raios catódicos exibem desenhos ou imagens, mas não os gravam de maneira permanente, o que permite efetuar mudanças rápidas. No tubo de raios catódicos (ver figura 1.3) as tensões aplicadas nas placas horizontais e verticais controlam a direção do feixe de elétrons e assim a posição do ponto brilhante na tela do tubo. A tensão aplicada na grade controla o brilho do ponto. Por sua vez essas tensões são controladas pelo computador ou pelo micro-processador local, através de conversores numérico/analógico. Nos monitores com tubo sem memória a permanência do desenho ou da imagem na tela é obtida através de um processo de reconstituição por repetição (e.g. 30 vezes por segundo) da seqüência de controle que gerou o desenho ou a imagem a primeira vez. O processo de reconstituição implica na existência de uma memória dita de restauração que é lida periodicamente.

Nos monitores vetoriais ("vector display") esta memória contém as instruções de saída com as respectivas coordenadas dos pontos, linhas ou caracteres (ver figura 1.4). Estas instruções são geralmente produzidas pelo computador central e interpretadas pelo micro-processador local. A última instrução armazenada na memória de restauração é um desvio incondicional para a primeira linha de instrução garantindo assim a repetição da seqüência de geração do desenho. De acordo com as tecnologias empregadas, de 500 a 5000 deslocamentos do feixe de elétron podem ser refetuados em média, em menos de 1/30 segundo.

Nos monitores a varredura ("raster display") a memória de restauração contém os valores numéricos de controle do tom de cinza ou da cor de cada ponto da imagem (ver figura 1.5). Assim, nos monitores comportando 1024 linhas de 1024 pontos, a memória de restauração tem 1024 x 1024 palavras. A memória é lida em seqüência periodicamente (e.g. 30 vezes por segundo).

Os monitores vetoriais são bem adaptados para exibir li-

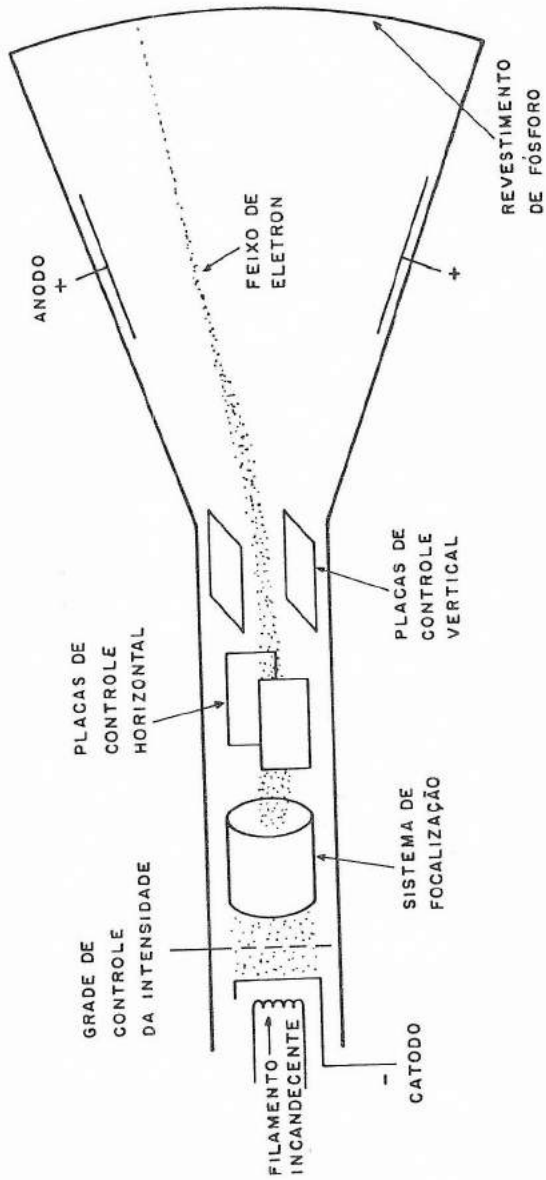


Figura 1.3 Tubo de raios catódicos.

MEMÓRIA DE RESTAURAÇÃO

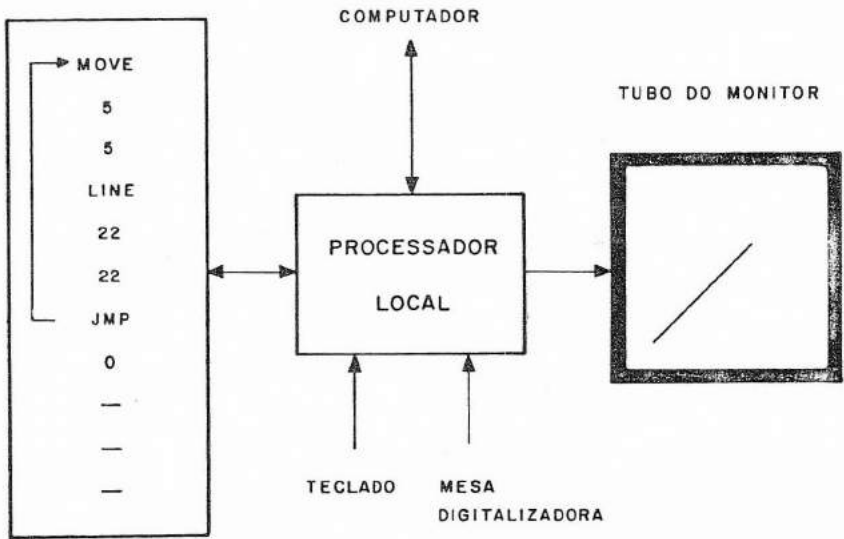


Figura 1.4 Monitor vetorial. A memória de restauração contém as instruções para a exibição de uma diagonal.

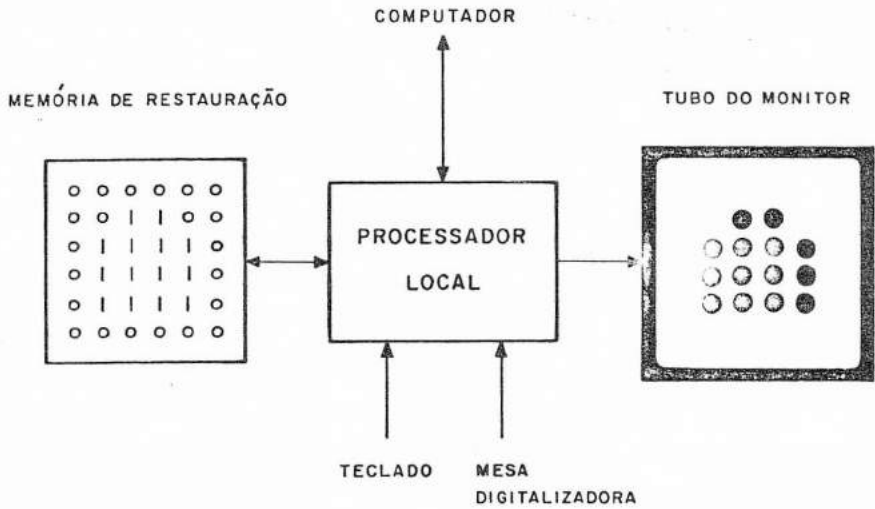


Figura 1.5 Monitor a varredura. A memória de restauração contém os valores de brilho dos pontos da imagem.

nhas retas e assim representar desenhos geométricos. Eles têm altas resoluções (grade de até 4096 x 4096 pontos) e permitem visualizar facilmente os efeitos de transformações geométricas (e.g. rotação, escalonamento; cf. capítulos 2 e 3). Por outro lado, eles têm uma capacidade muito limitada para preencher área, oferecem poucas cores e a qualidade do desenho (ausência de cintilação) depende de sua complexidade. Equipado com um tubo com memória, o monitor vetorial não precisa mais utilizar a memória de restauração para manter o desenho na tela, evitando assim o problema da cintilação. Em contrapartida, há uma perda de adaptabilidade do desenho a modificações já que o tempo para apagar a tela não é desprezível.

Os monitores a varredura são bem adaptados para exibir pontos coloridos dentro de uma ampla palheta de cores (cf. seção 8.4) e assim preencher áreas e finalmente representar imagens coloridas. As transformações radiométricas (e.g. mudança do contraste) são facilmente implementáveis se o monitor dispõe de uma tabela de cores ("look-up table"). Por outro lado, sua resolução é limitada (1024 x 1024 é considerado como um máximo) o que dificulta a exibição das linhas retas inclinadas (com orientação diferente de 0° , 45° , 90° , ... em relação a horizontal).

Os monitores com painel de plasma apresenta somente dois padrões: um ponto físico branco e um preto. Pelo seu princípio de funcionamento, sua tela é uma memória binária que dispensa a tradicional memória de restauração dos monitores a varredura. O painel é constituído de três placas de vidro superpostas. As duas placas externas têm, do seu lado interno, linhas de condutores horizontais para uma, e verticais para outra, formando uma grade. A placa do meio é constituída de células (buracos) centradas nos pontos da grade (ver figura 1.6). As células contêm em baixa pressão um gás neon. Quando a diferença de potencial entre uma linha vertical e uma horizontal ultrapassa um certo limiar superior, a célula correspondente dispara, i.é., cria-se dentro dela uma coluna ionizada positivamente chamada de plasma que emite uma luz. Quando a diferença de potencial abaixa, mas ficando acima de um certo limiar inferior, a célula continua disparada (daí o efeito de memória). Abaixo desse limiar inferior a diferença de potencial é insuficiente para manter o plasma, e,

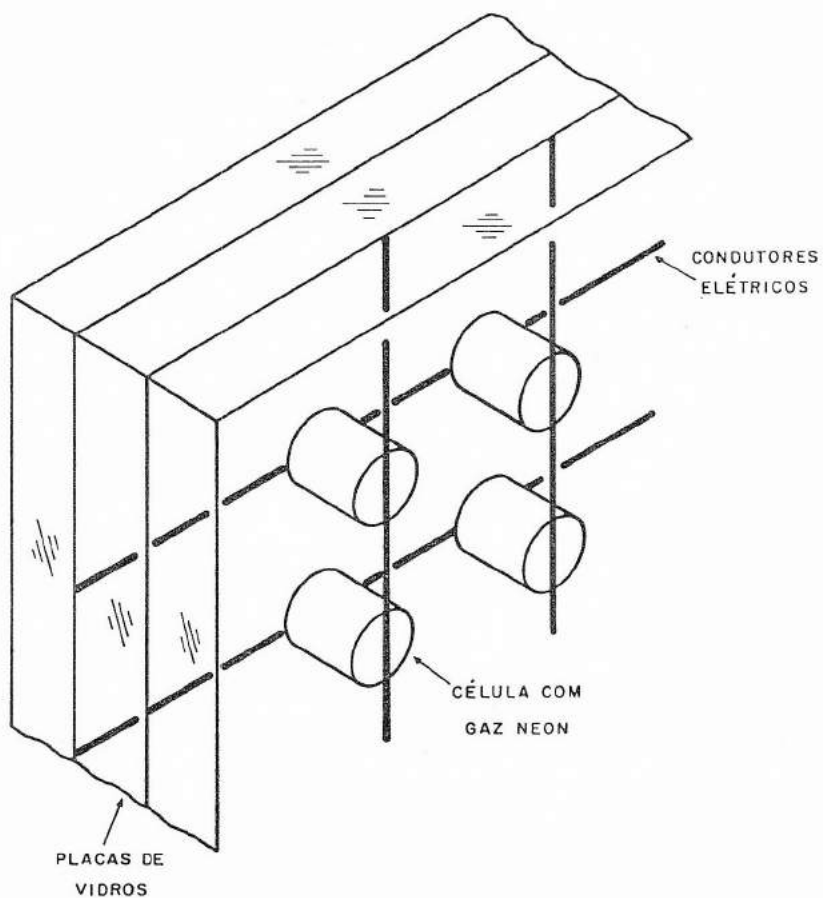


Figura 1.6 Panel de plasma.

assim, a emissão de luz.

Os dispositivos a estado sólido baseados em tecnologias como a dos LED ("Light-Emitting Diodes") ou dos LCD ("Liquid-Crystal Display") já usados em relógios digitais ou máquinas de calcular, poderão ser os monitores do futuro.

Os dispositivos LASER podem ser usados para exibir figuras, seu raio sendo desviado, por exemplo, através de um espelho móvel, cuja posição é controlada por um micro-processador local.

1.4- EQUIPAMENTOS GRÁFICOS DE ENTRADA

Em computação gráfica os equipamentos de entrada usualmente usados não são tão prestigiosos como os de saída. Eles têm como função principal a aquisição de informações como textos para o controle do processo de síntese de imagens. Os dispositivos de entrada armazenam essas informações de forma codificada em registros, que são lidos pelo computador sob o efeito de uma interrupção.

Os dispositivos de entrada sendo dos mais variados, os conceitos de dispositivos lógicos de entrada ou de função de entrada são introduzidos. Estes conceitos são tão valiosos quanto o de arquivos lógicos para os sistemas operacionais, e são extensivamente usados nos padrões gráficos como o GKS (cf. seção 1.2) para garantir a independência entre o aplicativo e os equipamentos disponíveis. Entre os dispositivos lógicos ou funções de entrada tem-se:

- a- o localizador ("locator") que fornece uma posição,
- b- o avaliador ("valuator") que fornece um número real,
- c- a função "pick" que identifica uma entidade dentro de um desenho,
- d- a função "string" que fornece uma cadeia de caracteres,
- e- a função "escolha" ("choice") que seleciona uma alternativa dentro de uma coleção reduzida de alternativas.

Em seguida, são descritos alguns equipamentos de entrada, procurando, classificando-os segundo a função principal deles.

As mesas digitalizadoras ("tablet") são por construção localizadores 2D. A posição do ponto de interesse na mesa é indicada através de uma caneta ou de um cursor e adquirida pressionando a caneta ou um botão no cursor (esses últimos dispositivos realizam a função "escolha").

Os "mouses" e "trackballs" são localizadores por construção. Neles a posição de um rolamento é transmitida ao computador através de dois potenciômetros a noventa graus. Quando em contato com uma superfície plana, os movimentos de "mouse" são captados pelo rolamento e geralmente reproduzidos por um cursor de tela no monitor. O "trackball" é um "mouse" de cabeça para baixo seu rolamento (maior que do "mouse") é acionado pela palma da mão.

Os "joysticks" são também localizadores, mas geralmente não são usados como tal. A posição relativa da sua alavanca em relação a sua posição central de repouso é interpretada pelo computador como o vetor velocidade do cursor de tela do monitor.

Os "touch panels" são localizadores, mas ao contrário dos "mouses", "trackballs" ou "joysticks" que podem fornecer indiretamente a posição de um ponto de interesse na tela do monitor, o "touch panel" dá, por construção, diretamente a posição do ponto de interesse apontando-o com o dedo na tela do monitor. Isto é possível através de uma série de diodos eletroluminescentes, e de diodos ou transistores fotosensíveis, montados no quadro da tela e criando assim uma "grade" sensível à presença de um dedo nas proximidades da tela. Outras tecnologias baseadas em medida do tempo de propagação de ultra-som numa placa de vidro ou de ten são elétrica em dispositivos resistivos ou capacitivos, oferecem muito mais resoluções.

Os dispositivos sônicos são por construção localizadores. Eles são baseados na medida do tempo de propagação do som emitido pelo

disparo de uma faísca elétrica na ponta de uma caneta. Esta tecnologia pode ser aplicada para localizadores 2D ou 3D.

Os "light pens" são por construção feitos para assumir a função "pick", quando usados junto com os monitores vetoriais. No momento em que uma entidade do desenho aparecendo na tela do monitor encontra-se com a ponta da caneta, esta capta um pico de variação luminosa e uma interrupção ocorre ao nível do processador local. Então, uma leitura do registro de endereço permite identificar qual entidade do desenho estava sendo exibida. Quando usado junto com um monitor a varredura, os "light pens" servem de localizadores. Dessa vez o registro de endereço contém as coordenadas do ponto da imagem que estava sendo exibido.

Os potenciômetros são por construção avaliadores. A posição do eixo do potenciômetro é transformado numa tensão. Quando é necessário, essa tensão é numerizada através de um conversor analógico/digital.

Os teclados alfanuméricos realizam por construção a função "string". Eles produzem um código correspondente à tecla pressionada. O código pode depender da posição de uma outra tecla (e.g. seleção das letras maiúsculas ou minúsculas).

As chaves ou teclas programáveis realizam por construção a função "escolha". Ao contrário das teclas do teclado alfanumérico, as teclas programáveis não são associadas a nenhum caráter ou símbolo definido, e seu número pode ser qualquer um. Geralmente as teclas programáveis são usadas para entrar comandos ou escolher opções num "menu" visualizado na tela do monitor. Neste caso, a visualização do "menu" dá o significado atual de cada tecla.

Numa instalação de computação gráfica raramente comporta todas as variedades de equipamentos de entrada descritos acima. A configuração mais comum comporta um teclado alfanumérico, teclas programáveis e às vezes um "mouse" ou mesa digitalizadora. Na prática, as aplicações gráficas podem necessitar de dispositivos ou funções de entrada não diretamente disponíveis como por exemplo, a função "pick" do "light pen" ou

de avaliadores como são os potenciômetros. Neste caso, a solução é a simulação dos dispositivos lógicos de entrada faltando, a partir dos dispositivos físicos de entrada disponíveis. Qualquer equipamento de entrada pode, em princípio, servir na simulação de qualquer dispositivo lógico. No entanto, a eficiência da simulação depende do equipamento escolhido.

Assim, é muito comum realizar a função "pick" com um localizador físico como a mesa digitalizadora ou o "mouse". Um localizador, por sua vez, pode ser simulado através de quatro teclas programáveis, correspondente a quatro tipos de incrementos de posição.

Um avaliador pode ser simulado através de um localizador 1D definindo uma posição ao longo de uma escala linear.

A função "escolha", isto é, a simulação de teclas programáveis, pode ser realizada através de um localizador. Neste caso, o reconhecimento de um certo movimento (como o de vai e vem) pode significar uma certa escolha (e.g. "apagar"). Combinado com a simulação de um "pick", o localizador físico pode assim cumprir duas funções ao mesmo tempo (e.g. apagar a entidade do desenho em cima da qual é executado um movimento de vai e vem). No entanto, o mais comum é simular uma função "escolha" com um localizador simulando a função "pick" para identificar uma opção dentro de um "menu" visualizado na tela do monitor.

1.5- ALGUNS ASPECTOS DO GKS

Somente os aspectos do GKS necessários para entender o uso dos comandos feitos nas ilustrações dos próximos capítulos são apresentados em seguida.

O objetivo do GKS é assegurar a portabilidade de um aplicativo, de uma instalação gráfica para outra.

No GKS foi introduzido o conceito de "estação de trabalho" para representar uma unidade lógica comportando um dispositivo de saída e eventualmente um ou mais dispositivos de entrada. O GKS pode ser imple

mentado por níveis de complexidade crescente. No nível mais simples considerado aqui, a estação de trabalho é limitada a um dispositivo de saída. A informação gráfica gerada pelo GKS é transferida às estações de trabalho e construída a partir de unidades pictoriais chamadas de primitivas de saída. As primitivas de saída consideradas aqui são:

- a- "polyline": linha poligonal,
- b- "polymarker": conjunto de marcas (símbolos) idênticas,
- c- "fill area": área poligonal.

A cada primitivas de saída está associado um ou mais atributos. Por exemplo, entre os atributos da primitiva "fill-area" tem-se o tipo de preenchimento (FILL AREA INTERIOR STYLE) que pode assumir valores como "HOLLOW" (ausência de preenchimento - só o contorno é traçado), "SOLID" (preenchimento denso), "HATCH" (preenchimento com hachuras), e a cor (COLOUR INDEX).

As primitivas de saída são definidas no aplicativo num sistema de coordenadas chamado de sistema de coordenadas universais ("World Coordinates" - WC). Na hora de efetuar a saída, o espaço das coordenadas universais é mapeado no espaço do dispositivo de saída referenciada pelo chamado sistema de coordenadas de dispositivo normalizadas ("Normalized device coordinate" - NDC) ou sistema de coordenadas de visualização. A passagem das coordenadas universais para as coordenadas de visualização é feita através de uma transformação geométrica chamada de transformação de normalização ("normalization transformation"). Essa transformação é determinada a partir da especificação no espaço das coordenadas universais de uma área quadrada ou retangular chamada de janela ("window") e no espaço das coordenadas de visualização de uma área quadrada ou retangular chamada de campo de visualização ("viewport"), sabendo que a primeira área deve ser mapeada exatamente na segunda (cf. seção 2.3). Por convenção, o campo de visualização tem que ser contido dentro da área quadrada $[0,1] \times [0,1]$, (ver figura 1.7). Cada janela, campo de visualização e transformação de normalização é identificado no GKS por um número inteiro. Finalmente o cerceamento ("clipping") nos limites do campo de visualização pode ser ativado e desativado (cf. seção 5.2).

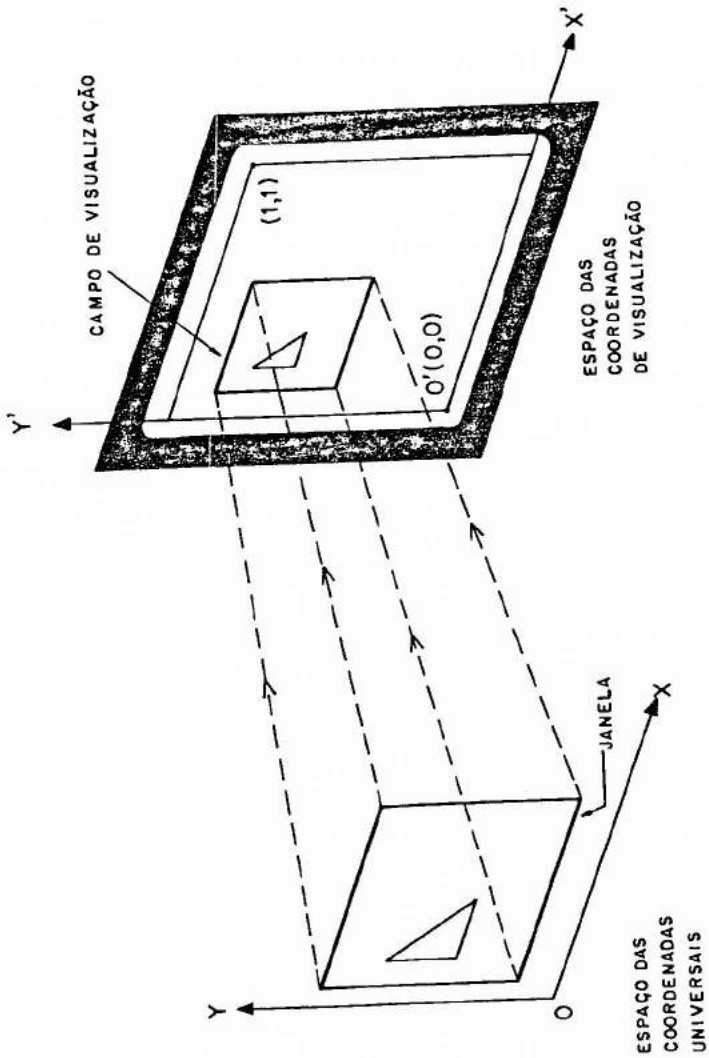


Figura 1.7 Mapeamento da janela no espaço das coordenadas universais para o campo de visualização no espaço das coordenadas de visualização.

As funções GKS diretamente ligadas aos aspectos abordados acima, são descritas em seguida (entre parênteses aparece o nome adotado na implementação do GKS usada nas ilustrações dos próximos capítulos).

a- Função de controle

CLEAR WORKSTATION (GCLRWK)

parâmetro: identificador da estação de trabalho (1, 2, ...)

efeito : limpa a tela do monitor.

b- Funções de saída

POLYLINE (GPL)

parâmetros: número de pontos (2, 3, ...)

pontos (WC)

(todas as abscissas em primeiro, todas as ordenadas a seguir)

efeito : traça uma linha poligonal começando no primeiro ponto e terminando no último.

atributo : LINETYPE

POLYMAKER (GPM)

parâmetros: os mesmos que GPL

efeito : traça uma marca em cada ponto.

FILL AREA (GFA)

parâmetros: os mesmos que GPL

efeito : preenche uma área poligonal

atributos : AREA INTERIOR STYLE e COLOUR INDEX

observação: O contorno do polígono é traçado somente quando o tipo de preenchimento é "HOLLOW", neste caso a cor do contorno é dado pelo COLOUR INDEX.

c- Funções de controle dos atributos de saída

SET LINETYPE (GSLN)

parâmetro : tipo da linha (1, 2, ...)

efeito : atualiza o tipo da linha em LINETYPE

observação: 1 para "solid line"
 2 para "dashed line"
 3 para "dotted line"
"default" : 1

SET FILL AREA INTERIOR STYLE (GSFAIS)

parâmetro : tipo de preenchimento (HOLLOW, SOLID, PATTERN, HATCH)
efeito : atualiza o tipo de preenchimento em AREA INTERIOR STYLE.
"default" : HOLLOW

SET FILL AREA COLOUR INDEX (GSFACI)

parâmetro : cor (0, 1, ...)
efeito : atualiza a cor em COLOUR INDEX
observação: 1 para preto
 8 para branco
"default" : 1

d- Funções de transformações e cerceamento

SET WINDOW (GSW)

parâmetros: número da transformação (1, 2, ...)
 janela (WC)
efeito : atualiza a transformação de normalização especificada.
observação: a janela é definida por x_{min} , $x_{m\bar{a}x}$, y_{min} , $y_{m\bar{a}x}$ nesta ordem.
"default" : 0, 1, 0, 1

SET VIEWPORT (GSVW)

parâmetros: número da transformação (1, ...)
 campo de visualização (NDC)
efeito : atualiza a transformação de normalização especificada
observação: o campo é definido por x_{min} , $x_{m\bar{a}x}$, y_{min} , $y_{m\bar{a}x}$ nesta ordem.
"default" : 0, 1, 0, 1

SELECT NORMALIZATION TRANSFORMATION (GSELNT)

parâmetro : número da transformação (1, ...)
efeito : atualiza o número da transformação de normalização usada.

"default" : 0 (corresponde à transformação de identidade).

SET CLIPPING INDICATOR (GSCLIN)

parâmetro : valor do indicador de cerceamento (CLIP, NOCLIP)

efeito : atualiza o indicador de cerceamento CLIPPING INDICATOR.

"default" : CLIP

Para ter uma descrição completa do GKS consultar [GKS, 1984]. Uma boa introdução ao GKS é dada em [Hopgood, 1983].

1.6- ALGUNS ASPECTOS DO APL

Somente os aspectos do APL necessários para entender os algoritmos que gerarão as figuras dos próximos capítulos, são apresentados a seguir.

O APL ("A Programming Language") é uma linguagem de programação para manipular dados estruturados (e.g. escalar, vetor, matriz, vetor de vetor) contidos em variáveis através de funções e operadores.

O uso do APL aqui deve ser visto como um meio de representar, de uma maneira concisa, alguns exemplos de computação gráfica.

Em APL os dados são do tipo numérico (e.g. 6, 1.5), binário (e.g. 0, 1) ou alfanumérico (e.g. 'A'). Esses dados podem ser estruturados na forma de escalar que não tem dimensão (e.g. 6), na forma de vetor simples que tem uma dimensão (e.g. 1 2 3), na forma encaixado de vetor de vetores (e.g. (1 2 3) (4 5 6)), na forma de matriz que tem duas dimensões (e.g. se A é uma matriz de dimensão 2 3, 2 é o número de linhas e 3 de colunas).

Os dados podem ser guardados em variáveis (e.g. A ← 1 2 3 para guardar o vetor simples 1 2 3 na variável A). Os dados são manipulados por funções. As funções podem ser monádica, isto é, com um só argumento (à direita) ou diádica, isto é, com dois argumentos (um à esquerda

e o outro à direita).

+ - x ÷ *

Na sua forma diádica as funções + - x ÷ * representam, respectivamente, a adição, subtração, multiplicação, divisão e exponenciação (e.g. 4 - 2). Na sua forma morádica as funções - e ÷ fornecem, respectivamente, o oposto e o inverso (e.g. -1 produz $\bar{1}$; ÷ 2 produz 0.5). O domínio do uso de muitas funções estende-se de escalário para estruturas mais complexas (e.g. 1 + 1 2 3 produz 2 3 4; 1 2 3 + 4 5 6 produz 5 7 9).

Γ L

A forma diádica de Γ e L fornecem, respectivamente, o máximo e o mínimo (e.g. 2 Γ 1 produz 2).

~

A forma diádica de ~ elimina os elementos de um vetor que pertencem a um outro vetor (e.g. 1 2 3 4 ~ 3 1 produz 2 4). Na sua forma morádica ~ realiza a complementação lógica (e.g. ~ 1 0 1 produz 0 1 0).

< >

As desigualdades < > são funções diádicas e produzem dados binários (e.g. 3 < 4 1 5 produz 1 0 1).

^ v

0 "e" ^ e "ou" v lógicos são funções diádicas (e.g. 0 0 1 1 ^ 0 1 0 1 produz 0 0 0 1).

f²

A forma diádica de f² permite criar uma matriz (e.g. 4 4 f² 0 produz a matriz 4 por 4 cheio de zero), sua forma monádica fornece a dimensão de um dado (e.g. f² 6 4 5 produz 3). A forma monádica de z₁ fornece a seqüência dos primeiros inteiros (e.g. z₁ 4 produz 1 2 3 4).

A forma monádica de \uparrow fornece o primeiro elemento de um vetor (e.g. $\uparrow 6$ 4 5 produz 6; $\uparrow(1\ 2\ 3)$ (4 5 6) produz 1 2 3), sua forma diádica fornece os primeiros elementos (e.g. $2\uparrow 6$ 4 5 produz 6 4).

A função \downarrow é diádica e fornece um vetor sem os seus primeiros ou últimos elementos (e.g. $1\downarrow 6$ 4 5 produz 4 5; $\bar{1}\downarrow 6$ 4 5 produz 6 4).

, $[1]$, $[1.1]$, $[.1]$

Na sua forma monádica a função \downarrow transforma uma matriz em um vetor obtido por concatenação das linhas da matriz (e.g. se 1 2 3 e 4 5 6 são as duas linhas de uma matriz A então $\downarrow A$ produz o vetor 1 2 3 4 5 6).

Na sua forma diádica a função \downarrow concatena dois vetores ou duas matrizes (e.g. 1 2 3 , 4 5 produz 1 2 3 4 5). A função diádica $\downarrow [1]$ concatena ao longo das linhas de uma matriz um vetor (e.g. se A é uma matriz de dimensão 2 3 então $\downarrow A$, $[1]$ 4 5 6 produz uma matriz de dimensão 3 3, o vetor 4 5 6 sendo concatenado por baixo de A). Na sua forma diádica a função $\downarrow [1.1]$ concatena dois vetores e produz uma matriz com duas colunas (e.g. 1 2 3 , $[1.1]$ 4 5 6 produz uma matriz de dimensão 3 2). Na sua forma monádica a função $\downarrow [1.1]$ transforma um vetor em uma matriz coluna (e.g. $\downarrow [1.1]$ 6 2 5 produz uma matriz de dimensão 3 1). Na sua forma diádica a função $\downarrow [.1]$ concatena dois vetores e produz uma matriz com duas linhas (e.g. 1 2 3 , $[.1]$ 4 5 6 produz uma matriz de dimensão 2 3).

$\bowtie \phi$

Na sua forma monádica a função \bowtie transpõe uma matriz (e.g. se A é de dimensão 2 3, $\bowtie A$ é de dimensão 3 2). Na sua forma diádica ϕ provoca uma rotação entre os elementos de um vetor ou as linhas de uma matriz (e.g. $2\phi 1$ 2 3 4 5 produz 3 4 5 1 2).

$\supset \subset$

Na sua forma monádica a função \supset transforma um vetor em uma matriz (e.g. $\supset (1\ 2\ 3) (4\ 5\ 6)$ produz uma matriz de dimensão 2 3). Na sua forma diádica a função \supset seleciona um vetor dentro de um vetor de vetor (e.g. $2 \supset (1\ 2\ 3) (4\ 5)$ produz 4 5). A função monádica \subset superpõe uma estrutura de escalar a um dado qualquer (e.g. $\subset 1\ 2\ 3$ produz um escalar (isto é, um dado sem dimensão) contendo o vetor 1 2 3).

Φ

A função monádica Φ executa uma linha APL dada na forma de uma cadeia alfanumérica (e.g. $\Phi '1 + 2'$ produz 3).

Em APL os operadores têm como argumentos uma variável ou uma ou duas funções e produzem uma nova função.

$\cdot\cdot\ /$

Aplicada a uma função, o operador $\cdot\cdot$ (cada) manda que a função seja aplicada a cada elemento dos dados (e.g. $\cdot\cdot (1\ 2\ 3) (4\ 5)$ produz o vetor 3 2). O operador diádico \cdot realiza o produto interno (e.g. se A e B são dois vetores, $A \cdot B$ produz o produto escalar de A por B; se A e B são duas matrizes, $A \cdot B$ produz o produto matricial de A por B; se A é um vetor e B uma matriz, $A \cdot B$ é equivalente ao produto matricial de A visto como uma matriz linha por B, o resultado sendo na forma de um vetor). O operador monádico $/$ realiza a função duplicação quando aplicada a um número (e.g. $3 / 4$ produz 4 4 4; $1\ 0\ 1 / 6\ 4\ 5$ produz 6 5). Quando aplicada a uma função diádica, o operador $/$ realiza a função integração (e.g. $+ / 1\ 2\ 3$ produz 6; $\int / 3\ 6\ 5$ produz 6). Quando a função resultante é diádica a integração é repetida acima de um horizonte limitado (e.g. $3 + / 1\ 2\ 3\ 4\ 5$ produz 6 9 12).

Em APL, as prioridades sintáticas são de direita para esquerda, isto é, a linha APL tem que se interpretada da direita para a esquerda (e.g. $1 + 2 \times 3$ é idêntico a $1 + (2 \times 3)$). Para mudar a ordem de

execução, os parênteses são usados (e.g. $(1 + 2) \times 3$ não é idêntico a $1 + 2 \times 3$. A primeira expressão produz 9, a segunda 7).

Em APL, não precisa programar explicitamente um "loop" para repetir uma operação. Por exemplo, para programar o produto matricial dos dois vetores 1 2 e 3 4 cada um por uma matriz A, basta criar um vetor contendo os dois vetores 1 2 e 3 4, isto é, escrevendo (1 2) (3 4), superpor uma estrutura de escalar a matriz A, isto é, escrevendo $\subset A$ e aplicar o operador cada (``) ao produto matricial (+ . x) escrevendo (1 2) (3 4) + . x`` $\subset A$. Esta última expressão é equivalente a (1 2) (3 4) + . x``A A, onde A A representa um vetor de duas matrizes.

Para ter uma descrição completa do APL (APL2) consultar [APL2, 1985].

Nas "transparências" apresentadas nos próximos capítulos, para reconhecer as linhas APL entrada pelo usuário das linhas de resultado, as primeiras começam na coluna 7 e as segundas na coluna 1.

2- ASPECTOS GEOMÉTRICOS DA COMPUTAÇÃO GRÁFICA EM 2D

2.1- EXEMPLOS DE DESENHOS EM 2D

A "transparência" = 1 = e as figuras 1 e 2 ilustram como são obtidos desenhos elementares em 2D (duas dimensões). Para desenhar a casa da figura 1, a variável APL CASA2 é criada contendo um vetor APL de vetores de coordenadas representando a seqüência de pontos a serem interligados por segmentos de reta. Cada ponto é definido pelas suas coordenadas cartesianas (abscissa e ordenada) num sistema de coordenadas ortogonal (cf. seção 2.2). O desenho da casa reduz-se a uma linha poligonal fechada, isto é, a última parte da seqüência tem que ser idêntico ao primeiro. Para desenhar a casa, a função de saída GPL que traça uma linha poligonal (cf. seção 1.5) é usada. No entanto, antes de se usar GPL tem que transformar o vetor de vetores CASA2 em um vetor simples verificando a sintaxe de formação de argumento da função GPL. Para isto foi criada a função APL de conversão CV que tem como argumento um vetor de vetores como CASA2 e produz um vetor simples no formato exigido por GPL. Assim, a linha GPL CV CASA2 produz o desenho da casa da figura 1.

Em vez de representar a seqüência de pontos na forma de um vetor de vetores, pode-se usar uma estrutura diferente, por exemplo a de matriz de duas colunas onde cada linha representa as coordenadas de um ponto. As variáveis BR, IM e LP da "transparência" = 1 = tem essa estrutura, e representam três linhas poligonais de 483, 24 e 19 pontos cada que correspondem respectivamente ao contorno do Brasil, da Ilha de Marajó e do Lago do Pato. Antes de desenhar os mapas no dispositivo de saída através da função GPL, tem que se certificar se o desenho cabe na tela. Para isto é necessário definir uma janela no espaço dos pontos, isto é, no espaço das coordenadas universais. Os limites da janela são obtidos aqui, calculando o máximo e o mínimo das coordenadas e deixando uma margem de ± 50 em volta do mapa. Para comunicar ao pacote gráfico os limites da janela (identificada pelo número 2), a função GSW é usada. Dentro do pacote gráfico, os limites dessa janela servem para calcular (cf. seção 2.3) os parâmetros da transformação de normalização (identificada pelo mesmo número 2). A seleção da transformação é feita pela fun-

a DESENHO DE LINHAS POLIGONAIS = 1 =

CASA2+(1 1)(3 1)(3 3)(2 4)(1 3)(1 1)

CASA2

1 1 3 1 3 3 2 4 1 3 1 1

CV CASA2

6 1 3 3 2 1 1 1 1 3 4 3 1

GPL CV CASA2

(produz figura 1)

p^{BB} IM LE

483 2 24 2 19 2

50+r/,BB,[1]IM,[1]LE

780

50+L/,BB,[1]IM,[1]LE

44

GSW 2 44 780 44 780

GSELNT 2

GCLRWK 1

GPL 483, BB

GPL (pIM), IM

GPL (pLE), LE

(produz figura 2)

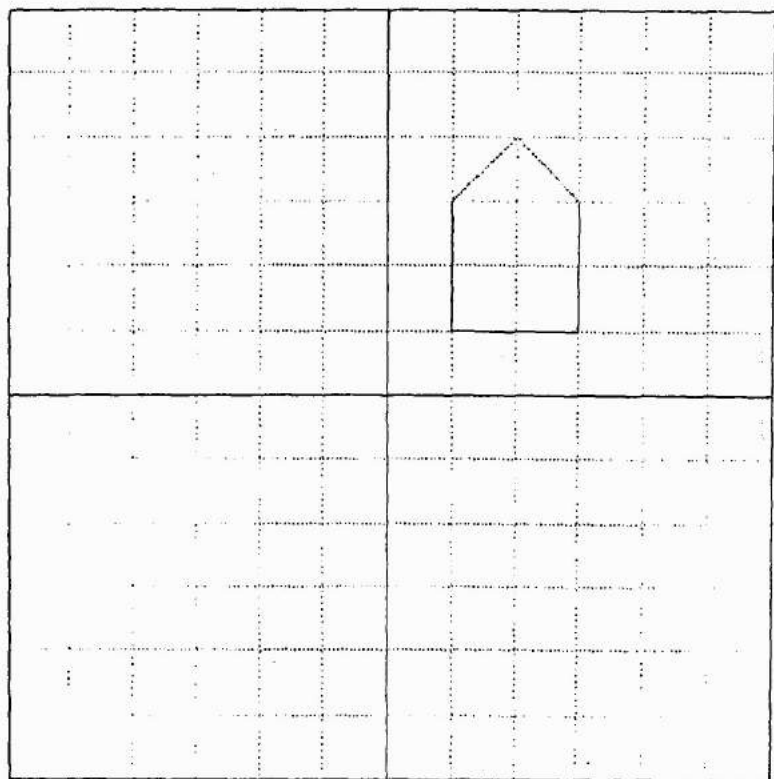


Fig. 1 - A casa na sua posição, tamanho e orientação original.



Fig. 2 - Os contornos do Brasil, Ilha de Marajó e Lagoa dos Patos.

ção GSELNT. Finalmente as matrizes são transformadas em vetores no formato exigido por GPL. O resultado das três chamadas à função GPL é mostrado na figura 2 (As variáveis BR, IM e LP foram comunicadas por Sr Luiz Cláudio Tamanini, da IBM Brasil).

2.2- TRANSFORMAÇÕES GEOMÉTRICAS EM 2D

As transformações geométricas ocupam um papel muito importante em computação gráfica. Cada vez que a saída de uma primitiva é efetuada, uma transformação geométrica é necessária para mapear o espaço das coordenadas universais no espaço das coordenadas de visualização (cf. seção 2.1). Geralmente a determinação e o uso dessa transformação é feita dentro do pacote gráfico e fica transparente ao usuário. Em aplicações gráficas complexas, os recursos do pacote gráfico podem ser insuficientes. Então certas transformações devem ser feitas explicitamente dentro do aplicativo.

Antes de introduzir as definições sobre as transformações geométricas, algumas definições sobre vetores e coordenadas são lembradas.

Dois pontos A e B do plano definem um segmento $[AB]$ (ou $[BA]$) e dois segmentos orientados ou vetores geométricos denotados \vec{AB} e \vec{BA} . A classe da equivalência de todos os vetores equipolentes a um vetor \vec{AB} chama-se também de vetor e é denotado geralmente por uma só letra. Escrivendo $\vec{v} = \vec{AB}$ significa que \vec{AB} é um representante da classe \vec{v} . O módulo de \vec{v} é o comprimento de $[AB]$ e denotado $\|\vec{v}\|$.

Os vetores geométricos do plano formam um espaço vetorial em \mathbb{R} (os reais) ou melhor, um espaço vetorial euclidiano (já que foi definida a noção de módulo).

Sejam O, X e Y três pontos do plano, os vetores \vec{OX} e \vec{OY} , nessa ordem formam um sistema de vetores denotado $\{\vec{OX}, \vec{OY}\}$ ou $\{0, \vec{x}, \vec{y}\}$ onde $\vec{x} = \vec{OX}$ e $\vec{y} = \vec{OY}$, ou simplesmente S se não houver possível confusão. Se O, X e Y não são colineares então $\{0, \vec{x}, \vec{y}\}$ é um sistema de coordena

das do plano. As semi-retas suportes de \vec{x} e \vec{y} são os eixos x e y do sistema de coordenadas. Num sistema de coordenadas $S = \{\vec{OX}, \vec{OY}\}$ pode-se associar, de uma maneira única, um par (x, y) de \mathbb{R}^2 a um ponto M ou a um vetor \vec{v} do plano escrevendo $\vec{v} = \vec{OM} = x \vec{OX} + y \vec{OY}$; x e y são as coordenadas cartesianas (x é abscissa e y ordenada) do ponto no sistema S ; (x, y) é o vetor de coordenadas do ponto M ou vetor \vec{v} no sistema S . Para simplificar a fraseologia escreve-se $M(x, y)$ em S ou $\vec{v}(x, y)$ em S , ou até omitindo S quando não houver possível confusão. Por exemplo, em S tem-se $X(1,0)$ e $Y(0,1)$. A expressão de "vetor" de coordenadas justifica-se pelo fato que \mathbb{R}^2 pode ser visto como um espaço vetorial em \mathbb{R} . No espaço \mathbb{R}^2 o sistema de vetores $N = \{(1,0), (0,1)\}$ chama-se de base natural.

Se as direções de dois vetores geométricos \vec{v}_1 e \vec{v}_2 são perpendiculares entre si, os vetores são dito ortogonais, denotado $\vec{v}_1 \perp \vec{v}_2$, e o sistema $\{\vec{v}_1, \vec{v}_2\}$ dito ortogonal. Se a mais $\|\vec{v}_1\| = \|\vec{v}_2\| = 1$, o sistema é dito ortonormal.

Seja $\vec{v}(x, y)$ em S ortonormal, então $\|\vec{v}\| = (x^2 + y^2)^{1/2}$ (isto é, $\|\vec{v}\|$ é dado pela expressão em N da norma euclidiana do vetor de coordenadas).

Sejam $M_1(x_1, y_1)$ e $M_2(x_2, y_2)$ em $\{0, \vec{x}, \vec{y}\}$ ortonormal, então a distância entre os pontos M_1 e M_2 , isto é, o módulo de $\vec{M}_1\vec{M}_2$ é dado por $\|\vec{M}_1\vec{M}_2\| = \|\vec{OM}_2 - \vec{OM}_1\| = ((x_2 - x_1)^2 + (y_2 - y_1)^2)^{1/2}$.

O produto escalar de \vec{v}_1 e \vec{v}_2 , denotado por $\langle \vec{v}_1, \vec{v}_2 \rangle$ é dado pela expressão $\frac{1}{2} (\|\vec{v}_1 + \vec{v}_2\|^2 - \|\vec{v}_1\|^2 - \|\vec{v}_2\|^2)$. Se $\vec{v}_1(x_1, y_1)$ e $\vec{v}_2(x_2, y_2)$ em S ortonormal então $\langle \vec{v}_1, \vec{v}_2 \rangle = x_1x_2 + y_1y_2$ (isto é, $\langle \vec{v}_1, \vec{v}_2 \rangle$ é dado pela expressão em N do produto escalar dos dois vetores de coordenadas). Observa-se que $\vec{v}_1 \perp \vec{v}_2$ é equivalente a $\langle \vec{v}_1, \vec{v}_2 \rangle = 0$.

É muito prático usar a seguinte representação matricial. A um vetor $\vec{v}(x, y)$ em S associa-se a matriz linha chamada de matriz de coordenadas do vetor \vec{v} em S , denotada $[\vec{v}]_S$ ou simplesmente $[\vec{v}]$, se não houver confusão possível quanto ao sistema de coordenada usada, ou ainda

$[x \ y]$. Da mesma maneira, a $\{\vec{v}_1, \vec{v}_2\}$ com $\vec{v}_1(x_1, y_1)$ e $\vec{v}_2(x_2, y_2)$ em S associa-se a matriz 2×2 denotada $\begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \end{bmatrix}_S$ ou $\begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \end{bmatrix}$ ou $\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix}$. Dependendo do contexto pode ser mais interessante associar a \vec{v} uma matriz coluna denotada também por $\begin{bmatrix} \vec{v} \end{bmatrix}$ ou $\begin{bmatrix} x \\ y \end{bmatrix}$ e ao sistema $\{\vec{v}_1, \vec{v}_2\}$ a matriz 2×2 de notada $\begin{bmatrix} \vec{v}_1 & \vec{v}_2 \end{bmatrix}$ ou $\begin{bmatrix} x_1 & x_2 \\ y_1 & y_2 \end{bmatrix}$. Esta representação matricial estende-se da mesma maneira a pontos ou vetores no espaço.

Seja $S' = \{0', \vec{x}, \vec{y}, \vec{z}\}$ um sistema de coordenadas no espaço. A um ponto $M(x, y, 1)$ do plano $z = 1$ pode-se associar de uma maneira única os pontos da reta $0'M$ (menos a origem $0'$). Os pontos M' dessa reta tem por vetor de coordenadas $(x', y', z') = t(x, y, 1)$ com $t \in \mathbb{R}$ mas $t \neq 0$ (cf. equação paramétrica da reta na seção 3.2). Seja $0(0, 0, 1)$, 0 pertence ao plano $z = 1$, por definição o triplete de coordenadas homogêneas de um ponto $M(x, y)$ em $\{0, \vec{x}, \vec{y}\}$ é o vetor de coordenadas cartesianas em S' de um dos pontos M' , isto é, (tx, ty, t) ($t \neq 0$). As coordenadas homogêneas de M dependem da constante arbitrária t .

A noção de coordenadas homogêneas vai ser muito importante para determinar o transformado de um ponto nas transformações geométricas usadas em computação gráfica.

Uma transformação geométrica \mathcal{T} no plano é uma aplicação que a um ponto M ou vetor \vec{v} faz corresponder o único ponto $M' = \mathcal{T}(M)$ ou o vetor $\vec{v}' = \mathcal{T}(\vec{v})$. As transformações mais usadas em computação gráfica são as transformações projetivas que conservam a propriedade de linha reta.

As transformações projetivas \mathcal{P} são as transformações geométricas para cada uma das quais é possível (uma vez escolhido um sistema de coordenadas $S = \{0, \vec{x}, \vec{y}\}$) associar uma aplicação que a matriz de coordenadas homogêneas $X = [u \ v \ t]$ de um ponto M do plano faz corresponder a matriz de coordenadas $X' = [u' \ v' \ t']$ do ponto transformado $M' = \mathcal{P}(M)$ dada por:

$$X' = XA + X_0, \quad (2.1)$$

onde A é uma matriz 3×3 não singular, chamada de matriz de transformação, e X_0 uma matriz linha arbitrária.

Dentro da classe das transformações projetivas, a classe das transformações afinas tem um papel especial em computação gráfica. As transformações dessa classe conservam a propriedade de paralelismo e entre elas são as translações, as homotetias e as rotações.

As transformações afinas são as transformações projetivas nas quais $X_0 = [0 \ 0 \ 0]$ e A é da seguinte forma:

$$A = \begin{bmatrix} a & b & 0 \\ c & d & 0 \\ e & f & 1 \end{bmatrix}, \quad (2.2)$$

onde $a, b, c, d, e, f \in \mathbb{R}$ e $a d - b c \neq 0$.

Nas transformações afinas tem-se $t' = t$. Assim é comum fazer $t = 1$ e escrever o vetor de coordenadas homogêneas $(x, y, 1)$ para um ponto de vetor de coordenadas cartesianas (x, y) .

Sejam \mathcal{F}_1 e \mathcal{F}_2 duas transformações afins e A_1 e A_2 suas respectivas matrizes, seja X a matriz de coordenadas homogêneas do ponto M , a partir de (2.1), a matriz de coordenadas X' do ponto $M' = \mathcal{F}_2(\mathcal{F}_1(M))$ é dada por: $X' = (XA_1)A_2$, (2.3)

por associatividade do produto matricial:

$$X' = X (A_1 A_2). \quad (2.4)$$

Pode-se verificar que $A_1 A_2$ é uma nova matriz de transformação afina. Em outros termos, a composição (ou o produto) de duas transformações afins é uma nova transformação afina. O produto matricial não é geralmente comutativo ($A_1 A_2 \neq A_2 A_1$). A ordem nas quais são feitas as transformações afins é significativa.

As translações de vetor $\vec{v}(\alpha, \beta)$ $\alpha, \beta \in \mathbb{R}$ são as trans -

formações afins nas quais:

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \alpha & \beta & 1 \end{bmatrix}. \quad (2.5)$$

Verifica-se que o transformado do ponto da matriz de coordenadas $[x \ y \ 1]$ é o ponto de matriz de coordenadas:

$$[x + \alpha \ y + \beta \ 1] = [x \ y \ 1] A.$$

Observa-se que em termo da matriz de coordenadas cartesianas tem-se:

$$[x + \alpha \ y + \beta] = [x \ y] + [\alpha \ \beta].$$

Essa maneira aparentemente mais simples de representar uma translação não é geralmente usada porque ela não caracteriza uma transformação afina. No caso de combinar uma translação com uma outra transformação afina não seria tão fácil expressar a transformação resultante.

As translações modificam a posição de uma figura no plano (conservando sua orientação).

Os escalonamentos em S de fatores α e $\beta \in \mathbb{R}$ são as transformações afins nas quais:

$$A = \begin{bmatrix} \alpha & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.6)$$

Verifica-se que o transformado do ponto de matriz de coordenadas $[x \ y \ 1]$ é o ponto de matriz de coordenadas:

$$[\alpha x \ \beta y \ 1] = [x \ y \ 1] A.$$

Uma homotetia de centro O e de razão $k \in \mathbb{R}$ é um escalonamento de $S = \{O, \vec{x}, \vec{y}\}$ de fatores $\alpha = \beta = k$.

As homotetias conservam as proporções. Os escalonamentos modificam essencialmente o tamanho das figuras.

As rotações de centro 0 e de ângulo $\theta \in [0, 2\pi]$ são as transformações afins nas quais:

$$A = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.7)$$

Verifica-se que o transformado do ponto de matriz de coordenadas $[r \cos \phi \ r \sin \phi \ 1]$ é o ponto de matriz de coordenadas:

$$[r \cos(\phi + \theta) \ r \sin(\phi + \theta) \ 1] = [r \cos \phi \ r \sin \phi \ 1] A.$$

Aqui, (r, ϕ) é o par de coordenadas polares do ponto a ser transformado. Na rotação, o ponto transformado tem $(r, \phi + \theta)$ como par de coordenadas polares. As rotações fazem girar as figuras.

Uma matriz M é ortogonal se $M^{-1} = M^T$ (M^T é a matriz transposta de M). Se M é ortogonal então M^T também é e $\det M = \pm 1$.

Quando a submatriz $M = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ da matriz A (cf. expressão (2.2)) é ortogonal, a transformação geométrica correspondente é dita métrica. As rotações e as translações são um caso particular de transformações métricas, para essas transformações $\det M = 1$. As transformações métricas conservam os ângulos e os comprimentos.

Seja $S = \{0, \vec{x}, \vec{y}\}$ ortonormal, como já foi visto a todo sistema $\{\vec{v}_1, \vec{v}_2\}$ pode-se associar de uma única maneira a matriz $M = \begin{bmatrix} \vec{v}_1 & \vec{v}_2 \end{bmatrix}_S$ e vice-versa. Dizer que M é ortogonal é equivalente a dizer que $\{\vec{v}_1, \vec{v}_2\}$ é ortonormal, pois se M é ortogonal $M^T M = I$ (a matriz identidade), isto é, tem-se:

$$\begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \end{bmatrix} \begin{bmatrix} \vec{v}_1 & \vec{v}_2 \end{bmatrix} = I.$$

Usando a definição de norma e produto escalar, tem-se:

$$\begin{bmatrix} \|\vec{v}_1\| & \langle \vec{v}_1, \vec{v}_2 \rangle \\ \langle \vec{v}_1, \vec{v}_2 \rangle & \|\vec{v}_2\| \end{bmatrix} = I.$$

Daí $\vec{v}_1 \perp \vec{v}_2$ e $\|\vec{v}_1\| = \|\vec{v}_2\| = 1$.

Da ortogonalidade de M deduz-se também que

$$\begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \end{bmatrix}_S^M = \begin{bmatrix} \vec{x} \\ \vec{y} \end{bmatrix}_S \quad (2.8)$$

já que a matriz do lado direito da igualdade é a matriz identidade. A expressão (2.8) indica que na transformação métrica, caracterizada pela matriz ortogonal M, os vetores \vec{v}_1 e \vec{v}_2 de matrizes coordenadas (no sistema S) dadas pelas submatrizes colunas de M, são levados respectivamente para \vec{x} e \vec{y} . Essa propriedade é muito importante e será usada em sessão subsequente para determinar as matrizes de rotação.

Sejam $\{\vec{v}_1, \vec{v}_2\}$ e $\{\vec{v}'_1, \vec{v}'_2\}$ ortonormais, então existe uma transformação métrica \mathcal{M} tal que $\vec{v}'_1 = \mathcal{M}(\vec{v}_1)$ e $\vec{v}'_2 = \mathcal{M}(\vec{v}_2)$, isto é, existe uma matriz ortogonal M tal que para qualquer $S = \{0, \vec{x}, \vec{y}\}$ tem-se:

$$\begin{bmatrix} \vec{v}'_1 \\ \vec{v}'_2 \end{bmatrix} = \begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \end{bmatrix} M.$$

Os dois sistemas acima tem a mesma orientação se a transformação \mathcal{M} acima é uma rotação ou o que é equivalente se:

$$\det [\vec{v}'_1 \vec{v}'_2]_S = \det [\vec{v}_1 \vec{v}_2]_S \quad (2.9)$$

(lembrando que se M é uma matriz de rotação $\det M = 1$ e que o determinante do produto de duas matrizes é o produto dos determinantes).

Observando que $\det [\vec{x} \vec{y}]_S = 1$, de (2.9) tem-se que um sistema ortonormal $\{\vec{v}'_1, \vec{v}'_2\}$ tem a mesma orientação que o sistema de coor

denadas S se

$$\det [\vec{v}_1 \ \vec{v}_2]_S = 1.$$

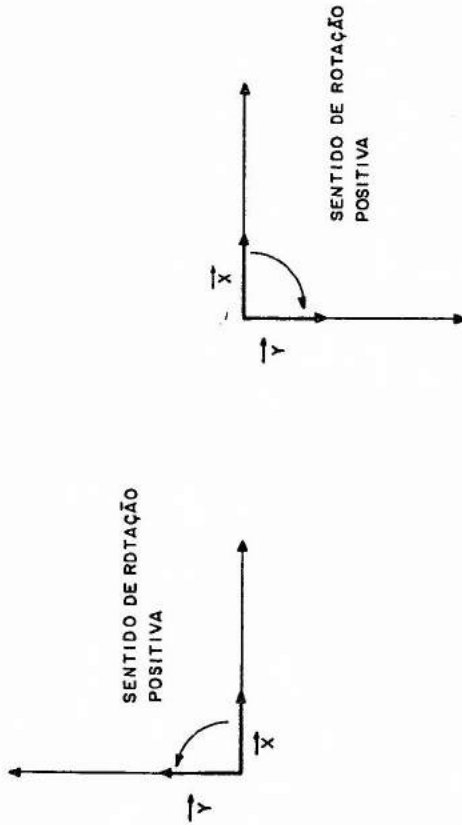
No caso de um sistema de coordenadas $\{\vec{v}_1, \vec{v}_2\}$ qualquer, a condição é simplesmente:

$$\det [\vec{v}_1 \ \vec{v}_2]_S > 0. \quad (2.10)$$

Na rotação \mathcal{R}^+ de ângulo $\pi/2$, $\mathcal{R}^+(\vec{x}) = \vec{y}$, se \vec{v}_1, \vec{v}_2 é ortonormal e tem a mesma orientação que S, então $\mathcal{R}^+(\vec{v}_1) = \vec{v}_2$. Assim o sentido de rotação positiva no plano é definido como o sentido de rotação que leva \vec{x} para \vec{y} , ou \vec{v}_1 para \vec{v}_2 se (\vec{v}_1, \vec{v}_2) tem a mesma orientação que S. O plano é dito orientado desde que nele seja definido, através de um sistema S de coordenadas, o sentido de rotação positiva. O sentido de orientação positiva pode ser horário ou anti-horário. Um sistema S de coordenadas é orientado no sentido horário (respectivamente anti-horário) se ele induz um sentido de rotação positiva horário (respectivamente anti-horário). Aqui, como é geralmente o caso em computação gráfica, o sistema de coordenadas usado é orientado no sentido anti-horário (ver figura 2.1 a).

A "transparência" = 2 = mostra o transformado do ponto (1, 1, 1) por uma translação de vetor (4, 3), um escalonamento de fatores 2 e 3 e uma rotação de 30 graus. Para realizar estas transformações de translação, escalonamento e rotação foram criadas, respectivamente, as funções APL T2, S2 e R2. O argumento das funções T2 e S2 é um vetor APL cujos elementos são os parâmetros α e β da transformação, o argumento da função R2 é um escalário representando o ângulo θ (em graus). Essas funções produzem a matriz de transformação 3 x 3 correspondente. A função (+ . x) realiza o equivalente ao produto matricial (cf. seção 1.6).

A "transparência" = 3 = dá um exemplo de translação de uma figura. Aqui, a figura é uma linha poligonal representando uma casa. Em vez de transformar cada ponto da casa (teoricamente em número infinito), transforma-se os vértices da casa e usa-se uma só vez a função GPL



a- Sistema de coordenadas orientado no sentido anti-horário.
b- Sistema de coordenadas orientado no sentido horário.

Figura 2.1 As duas orientações do plano.

A COORDENADAS HOMOGENEAS
A REPRESENTAÇÃO MATRICIAL DE
A TRANSFORMAÇÕES 2D

= 2 =

T2 4 3

1 0 0
0 1 0
4 3 1

1 1 1+.xT2 4 3

5 4 1

S2 2 3

2 0 0
0 3 0
0 0 1

1 1 1+.xS2 2 3

2 3 1

R2 30

0.8660254038 0.5 0
-0.5 0.8660254038 0
0 0 1

1 1 1+.xR2 30

0.3660254038 1.366025404 1

» COORDENADAS HOMOGENEAS = 3 =
» TRANSLAÇÃO DE UMA LINHA POLIGONAL

CASA2

1 1 3 1 3 3 2 4 1 3 1 1

CH CASA2

1 1 1 3 1 1 3 3 1 2 4 1 1 3 1 1 1 1

HC CH CASA2

1 1 3 1 3 3 2 4 1 3 1 1

$A+HC(CH\ CASA2)+.x^{T2}-2\ 2$

GPL CV A

(produz figura 3)

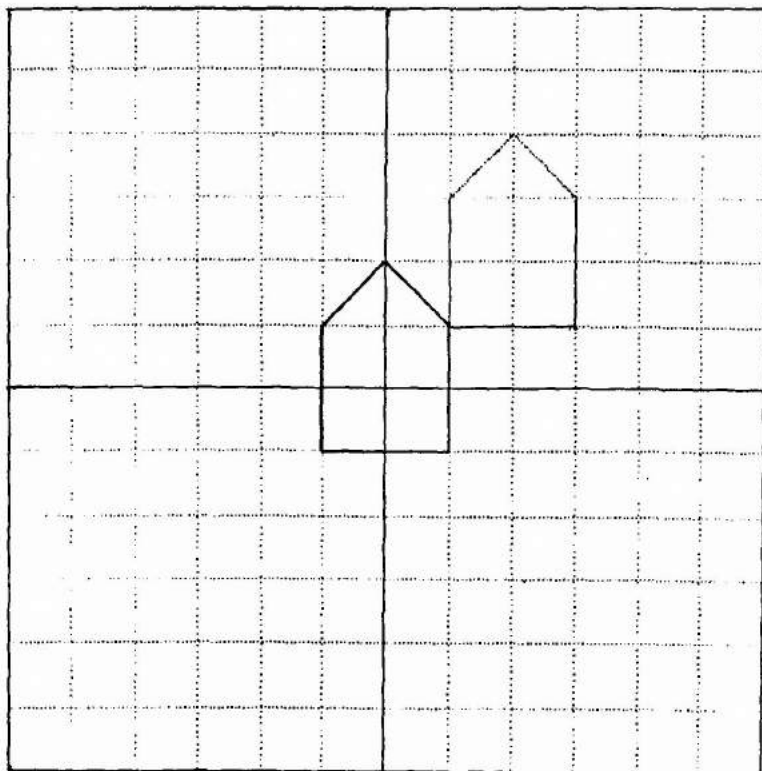


Fig. 3 - A casa (da Figura 1) centrada na origem por uma translação $(-2,-2)$.

para desenhar a casa na sua nova posição. Para aplicar o mecanismo matricial descrito acima foi criada a função CH para converter as coordenadas cartesianas em coordenadas homogêneas e sua inversa HC. Para transformar cada ponto aparecendo no vetor APL CH CASA2, basta superpor uma estrutura de escalário à matriz de translação (usando \llcorner) e aplicar o operador cada (**) ao produto matricial (+ . x) (cf. seção 1.6).

O resultado da translação de vetor (-2, -2) do vetor APL CASA2 é um novo vetor APL chamado aqui de A. Traçando a linha poligonal correspondente a seqüência de ponto em A obtêm-se a figura 3 onde a casa passa a ser centralizada no ponto de origem (0, 0).

A "transparência" = 4 = mostra como foram obtidas as figuras 4, 5 e 6. Usando o vetor APL A da "transparência" = 3 = são efetuadas sucessivamente uma homotetia de razão 2, uma rotação de 45 graus e uma translação de vetor (2, 2). Observa-se que o resultado final das quatro transformações é uma ampliação por um fator 2 da casa em volta do ponto (2, 2) assim que uma rotação de 45 graus em volta desse mesmo ponto.

Esse exemplo mostra que para efetuar homotetias ou rotações em volta de um ponto (α , β) que não seja o ponto origem (0, 0), basta começar por uma translação ($-\alpha$, $-\beta$) (levando assim, em particular, o ponto (α , β) para (0, 0)), efetuar as homotetias e/ou rotações (observa-se que a ordem em que são feitas essas transformações não importa) e terminar por uma translação (α , β) levando assim, em particular, o ponto (0, 0) na sua posição original (α , β)).

Quando há mais de uma transformação geométrica a ser efetuada das duas expressões (2.3) e (2.4) acima, é a expressão (2.4) que é implementada, pois geralmente a transformação de uma figura envolve as transformações de mais de um ponto. Isto é, é mais eficiente calcular primeiro a matriz da transformação composta (e.g. $A_1 A_2$) e aplicar essa transformação a cada ponto a ser transformado.

A "transparência" = 5 = ilustra a eficiência deste proce-

a TRANSFORMAÇÕES AFINAS DE UMA LINHA = 4 =
a POLIGONAL

$$A \leftarrow HC(CH A) + \cdot x'' \in S2 \ 2 \ 2$$

GPL CV A (produz figura 4)

$$A \leftarrow HC(CH A) + \cdot x'' \in R2 \ 45$$

GPL CV A (produz figura 5)

$$A \leftarrow HC(CH A) + \cdot x'' \in T2 \ 2 \ 2$$

GPL CV A (produz figura 6)

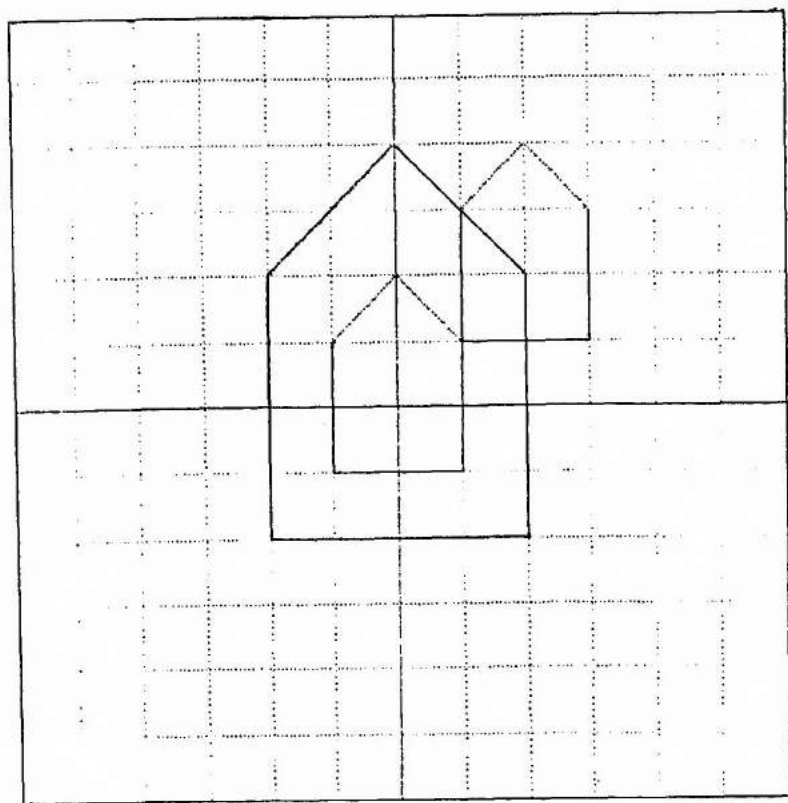


Fig. 4 - A casa (da Figura 3) ampliada por uma homotetia de razão 2.

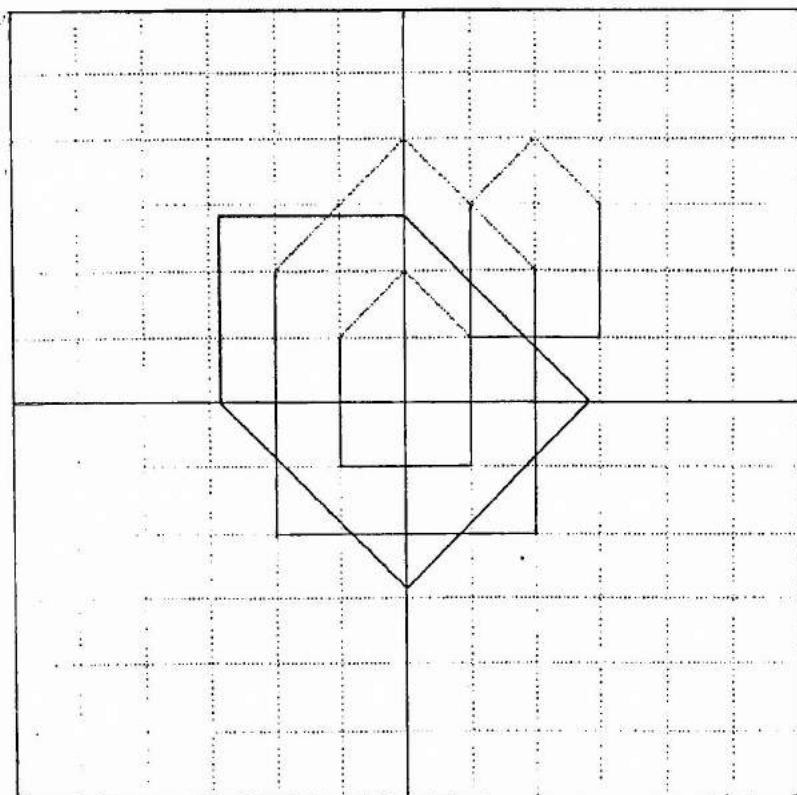


Fig. 5 - A casa (da Figura 4) girada por uma rotação de 45 graus.

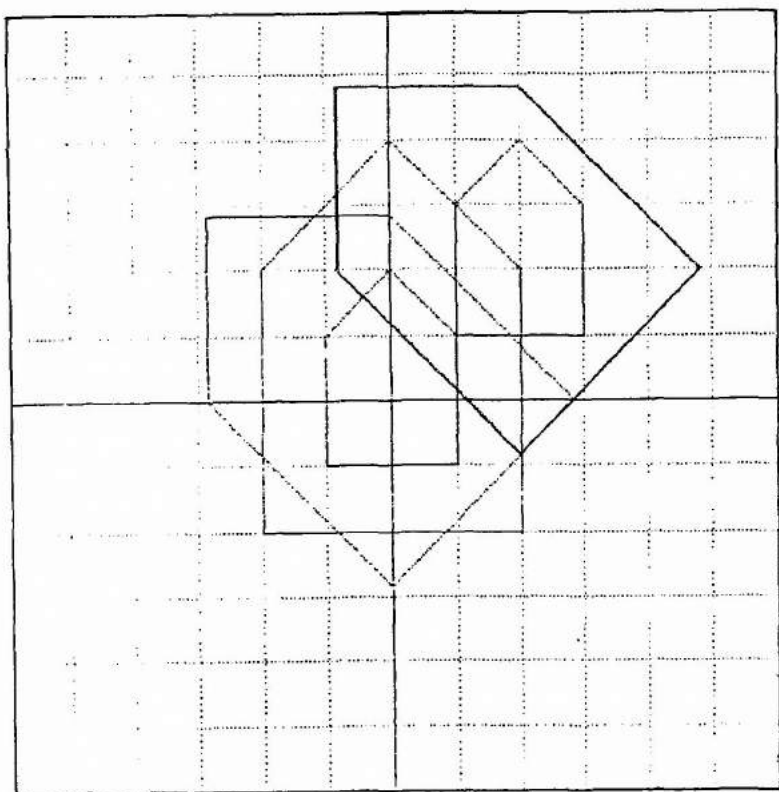


Fig. 6 - A casa (da Figura 5) centrada no ponto original (2,2) por uma translação (2,2).

a COMPOSIÇÃO DE TRANSFORMAÇÕES 2D = 5 =
M ← (T2-2 2) + .x(S2 2 2) + .x(R2 45) + .xT2 2 2
A + HC(CH CASA2) + .x'' ← M
GPL CV CASA2
GPL CV A (produz figura 7)

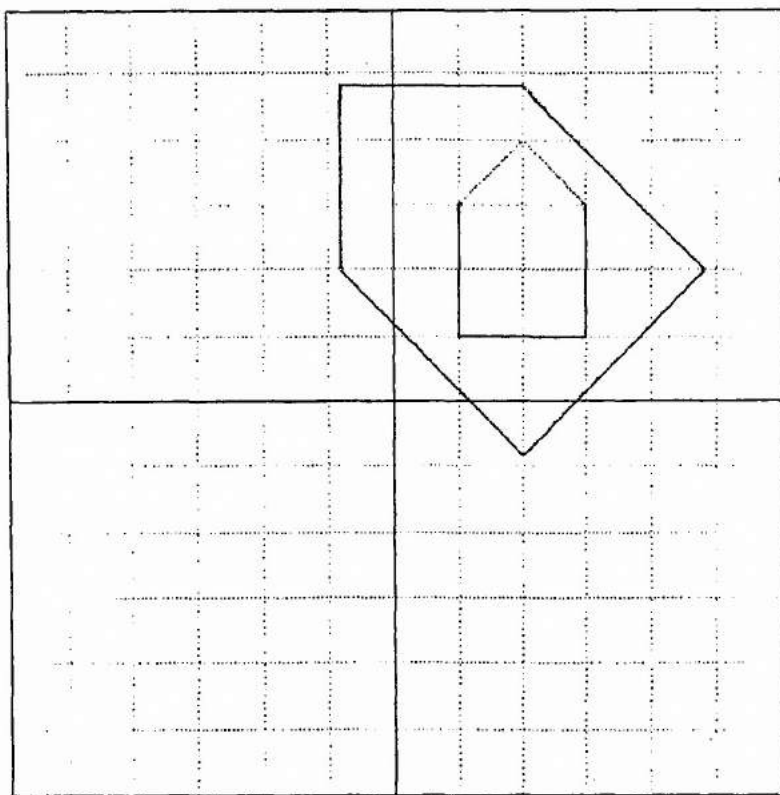


Fig. 7 - A casa (da Figura 1) ampliada e girada por uma transformação afina produto de duas translações, uma homotetia e uma rotação

dimento. O resultado obtido pela sucessão das transformações apresentadas nas "transparências" = 3 = e = 4 = pode ser obtido com eficiência calculando primeiro a matriz produto M e aplicando-a a cada vértice da casa. A figura 7 mostra que chega-se assim ao mesmo resultado da figura 6.

2.3- DETERMINAÇÃO DE UMA TRANSFORMAÇÃO DE NORMALIZAÇÃO

O exemplo dado em seguida ilustra o problema da determinação de uma transformação de normalização, isto é, a transformação representando o mapeamento de uma janela definida no espaço das coordenadas universais referenciado por $\{0, \vec{x}, \vec{y}\}$, num campo de visualização definido no espaço das coordenadas de visualização referenciado por $\{0', \vec{x}', \vec{y}'\}$. (cf. seção 1.5, e ver figura 1.7).

A janela escolhida é representada na figura 8. Ela enquadra a casa, seus vértices são definidos através dos vetores APL A, B, C e D da "transparência" = 6 =. O campo de visualização é suposto ocupar toda a área permitida, isto é, é o quadrado unitário delimitado pelas retas $x' = 0$, $x' = 1$, $y' = 0$ e $y' = 1$.

No mapeamento, decide-se que os pontos A, B e C do espaço das coordenadas universais devem corresponder aos pontos (0, 0), (1, 0) e (0, 1) do espaço das coordenadas de visualização. Uma transformação geométrica pode tipicamente representar esse mapeamento, pois as coordenadas (de visualização) do ponto imagem no mapeamento podem ser vistas como as coordenadas (universais) do ponto transformada numa transformação geométrica atuando no espaço das coordenadas universais. Essa transformação dita transformação de normalização, é obtida aqui como a composição de uma translação levando o ponto A para o ponto origem (0, 0), seguida de uma rotação levando $\vec{AB}/\|\vec{AB}\|$ para \vec{x} (e $\vec{AD}/\|\vec{AD}\|$ para \vec{y}), e uma homotetia transformando o módulo de \vec{AB} (e \vec{AD}) em módulo unitário.

Usando a propriedade expressa em (2.8) acima, a respeito de uma matriz ortogonal M, a matriz de rotação é

$$\begin{bmatrix} M & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{onde}$$

» VISUALIZAÇÃO EM 2D
» ESPECIFICAÇÃO DE UMA JANELA

= 6 =

A+1 -1

B+5 1

C+3 5

D+1 3

GPL CV CASA2

GPL CV A B C D A

(produz figura 8)

DST A B

4.472135955

U+(B-A)÷DST A B

DST(0 0)U

1

U

0.894427191 0.4472135955

V+(D-A)÷DST A D

V

0.4472135955 0.894427191

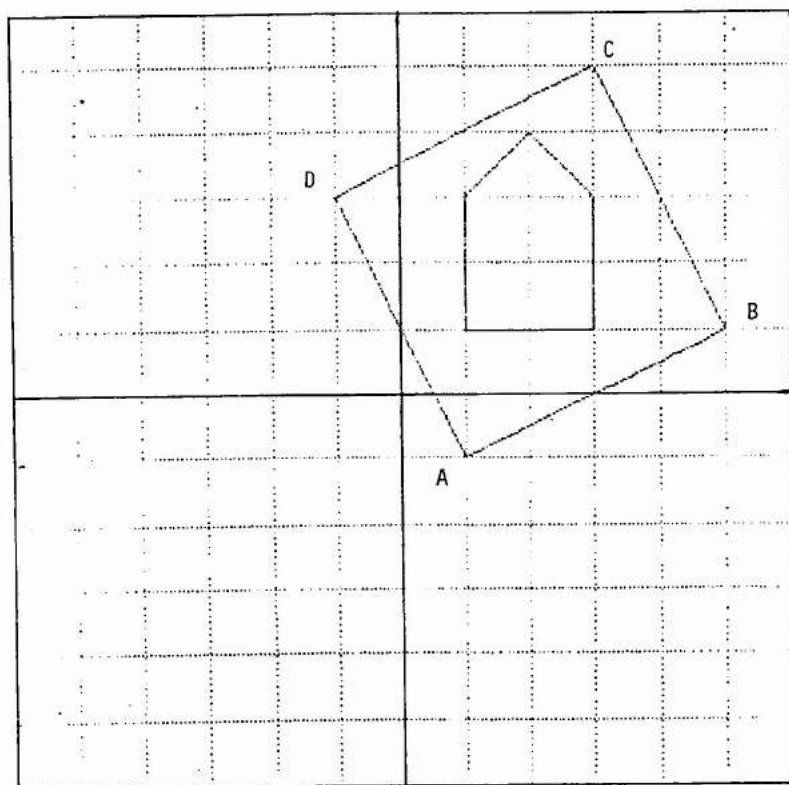


Fig. 8 - A casa e a janela no espaço das coordenadas universais.

$$M = \left[\begin{array}{cc} \vec{AB} / \|\vec{AB}\| & \vec{AD} / \|\vec{AD}\| \end{array} \right].$$

Para determinar as matrizes de rotação e homotetia, usa-se a função APL DST que calcula o módulo de um vetor, o melhor, a distância (euclidiano) entre suas extremidades. A função DST tem como argumento um vetor APL de dois vetores de coordenadas representando dois pontos e produz um escalário representando a distância entre esses dois pontos. A "transparência" = 7 = mostra o cálculo da matriz de rotação ROT e da matriz de transformação de normalização M. A figura 9 mostra a casa dentro do campo de visualização.

a VISUALIZAÇÃO EM 2D = 7 =
a TRANSFORMAÇÃO DAS COORDENADAS UNIVERSAIS
a EM COORDENADAS DE VISUALIZAÇÃO

U, [1.1]V

0.894427191 -0.4472135955
0.4472135955 0.894427191

ROT=((U, [1.1]V), 0), [1]0 0 1

ROT

0.894427191 -0.4472135955 0
0.4472135955 0.894427191 0
0 0 1

M=(T2-A)+.xROT+.xS2 2/÷DST A B

M

0.2 -0.1 0
0.1 0.2 0
-0.1 0.3 1

GSW 3 0 1 0 1

GSELNT 3

GFL CV HC(CH CASA2)+.x"=M (produz figura 9)

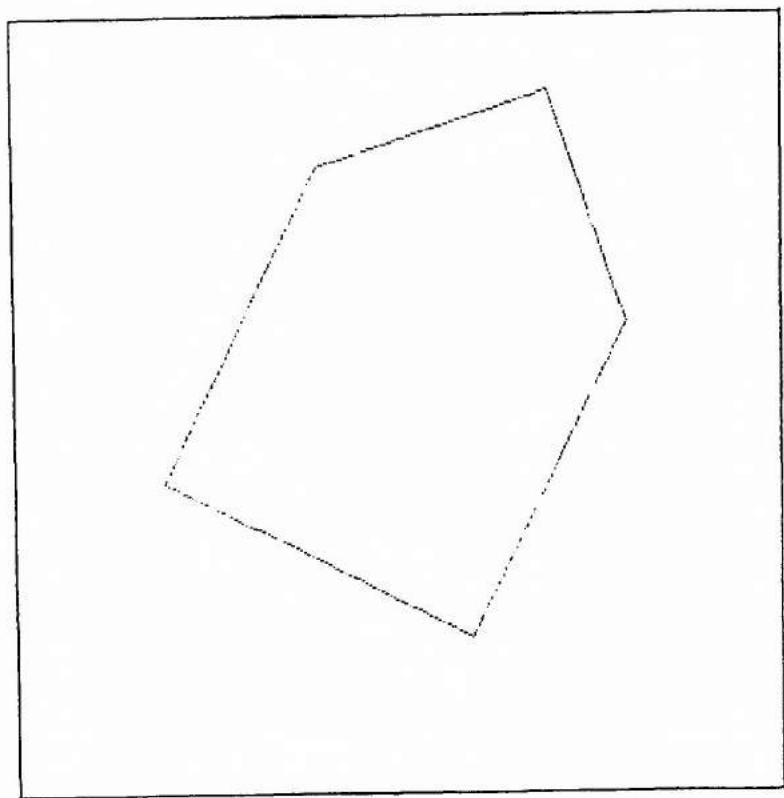


Fig. 9 - A casa (da Figura 8) no campo de visualização (espaço das coordenadas de visualização).

3- ASPECTOS GEOMÉTRICOS DA COMPUTAÇÃO GRÁFICA EM 3D

3.1- TRANSFORMAÇÕES GEOMÉTRICAS EM 3D

Muitas das definições da seção 2.2 relativas ao plano podem ser estendidas para o espaço.

Para as transformações geométricas consideradas aqui o quadrupeto de coordenadas homogêneas de um ponto de vetor de coordenadas cartesianas (x, y, z) é simplesmente $(x, y, z, 1)$.

Como no caso do plano, as transformações geométricas no espaço apresentadas a seguir, são definidas em termo de uma correspondência entre matrizes linhas de coordenadas homogêneas, desde que seja previamente definido um sistema de coordenadas cartesianas $(O, \vec{x}, \vec{y}, \vec{z})$ ortogonal.

As translações de vetor $\vec{v}(\alpha, \beta, \gamma)$ são as transformações afins nas quais:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \alpha & \beta & \gamma & 1 \end{bmatrix} . \quad (3.1)$$

Verifica-se que o transformado do ponto da matriz de coordenadas $[x \ y \ z \ 1]$ é o ponto da matriz de coordenadas:

$$[x + \alpha \ y + \beta \ z + \gamma \ 1] = [x \ y \ z \ 1] A.$$

Os escalonamentos em S de fatores $\alpha, \beta, \gamma \in \mathbb{R}$ são as transformações afins nas quais

$$A = \begin{bmatrix} \alpha & 0 & 0 & 0 \\ 0 & \beta & 0 & 0 \\ 0 & 0 & \gamma & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.2)$$

Verifica-se que o transformado do ponto da matriz de coordenadas $[x \ y \ z \ 1]$ é o ponto da matriz de coordenadas:

$$[\alpha x \ \beta y \ \gamma z \ 1] = [x \ y \ z \ 1] A.$$

As rotações de eixo z e ângulo $\theta \in [0, 2\pi]$ são as transformações afins nas quais

$$A = \begin{bmatrix} \cos \theta & \sin \theta & 0 & 0 \\ -\sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (3.3)$$

Verifica-se que o transformado do ponto da matriz de coordenadas $[r \cos \phi \ r \sin \phi \ z \ 1]$ é o ponto da matriz de coordenadas:

$$\begin{aligned} & [r \cos(\phi + \theta) \ r \sin(\phi + \theta) \ z \ 1] = \\ & [r \cos \theta \ r \sin \theta \ z \ 1] A. \end{aligned}$$

Aqui (r, ϕ, z) é o tripleto de coordenadas cilíndricas do ponto a ser transformado. Na rotação, o ponto transformado tem $(r, \phi + \theta, z)$ como tripleto de coordenadas cilíndricas.

Da mesma maneira, as rotações do eixo x de ângulo $\theta \in [0, 2\pi]$ são definidas a partir da matriz:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & \sin \theta & 0 \\ 0 & -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (3.4)$$

e as rotações de eixo z de ângulo $\theta \in [0, 2\pi]$ a partir da matriz:

$$A = \begin{bmatrix} \cos \theta & 0 & -\sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (3.5)$$

Combinando as três rotações elementares acima, obtêm-se as rotações em volta da origem O, que são as transformadas afins nas quais:

$$A = \begin{bmatrix} & & 0 \\ & M & 0 \\ & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (3.6)$$

onde a submatriz M 3×3 é ortogonal e $\det M = 1$. Como no caso 2D (cf. seção 2.2), escrevendo $M = [\vec{v}_1 \ \vec{v}_2 \ \vec{v}_3]_S$ tem-se $\vec{v}_1 \perp \vec{v}_2$, $\vec{v}_2 \perp \vec{v}_3$, $\vec{v}_3 \perp \vec{v}_1$ e $\|\vec{v}_1\| = \|\vec{v}_2\| = \|\vec{v}_3\| = 1$.

No caso geral, numa transformação métrica caracterizada por uma matriz M ortogonal e $\det M = \pm 1$, os vetores \vec{v}_1 , \vec{v}_2 e \vec{v}_3 de matrizes de coordenadas em S dadas pelas submatrizes colunas de M , são levados respectivamente para \vec{x} , \vec{y} e \vec{z} :

$$\begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \\ \vec{v}_3 \end{bmatrix}_S = M \begin{bmatrix} \vec{x} \\ \vec{y} \\ \vec{z} \end{bmatrix}_S. \quad (3.7)$$

De (3.7), pode-se concluir que um sistema ortonormal $(\vec{v}_1, \vec{v}_2, \vec{v}_3)$ tem a mesma orientação que o sistema de coordenadas S se:

$$\det [\vec{v}_1 \ \vec{v}_2 \ \vec{v}_3]_S = 1.$$

No caso de um sistema de coordenadas $(\vec{v}_1, \vec{v}_2, \vec{v}_3)$ qualquer, a condição é simplesmente:

$$\det [\vec{v}_1 \ \vec{v}_2 \ \vec{v}_3]_S > 0. \quad (3.8)$$

Seja \mathcal{R}_x^+ (resp. \mathcal{R}_y^+ , \mathcal{R}_z^+) a rotação do eixo x (resp. y , z) de ângulo $\pi/2$, então tem-se $\mathcal{R}_x^+(y) = z$, $\mathcal{R}_y^+(z) = x$ e $\mathcal{R}_z^+(x) = y$. Assim, o sentido de rotação positiva do eixo x (resp. y , z) é definido como o sentido que leva o eixo y (resp. z , x) para z (resp. x , y). O espaço é dito orientado desde que nele seja definido, através de um sistema S de coordenadas ortonormal, o sentido de rotação positiva de

um eixo qualquer para um observador situado nesse eixo e olhando para a origem 0. Para o observador, esse sentido pode ser horário ou anti-horário. Um sistema S de coordenadas ortonormal é orientado no sentido horário ou é da mão esquerda (resp. anti-horário ou é da mão direita) se ele induz para um tal observador um sentido de rotação positiva horário (resp. anti-horário). Aqui, o sistema de coordenadas universais usado é da mão direita e o sistema de coordenadas de visualização é da mão esquerda (ver figura 3.1).

A "transparência" = 8 = mostra o transformado do ponto (1, 1, 1, 1) por uma translação de vetor (4, 3, 5) e um escalonamento de fatores 2, 4 e 3. Para realizar estas transformações foram criadas as funções T3 e S3. Essas funções tem por argumento um vetor APL cujos elementos são os parâmetros α , β e γ da transformação, e produzem a matriz de transformação 4 x 4 correspondente.

A "transparência" = 9 = mostra três funções RX, RY e RZ. Essas funções tem por argumento um escalário alfanumérico representando o nome de uma variável APL, e produzem as matrizes de rotação de eixo x, y e z respectivamente. Essas matrizes são na forma simbólica (isto é, seus elementos são caracteres alfanuméricos). A segunda parte da "transparência" mostra como obter a matriz numérica representando a rotação de eixo x de 30 graus e o transformado do ponto (1, 1, 1, 1) nesta rotação.

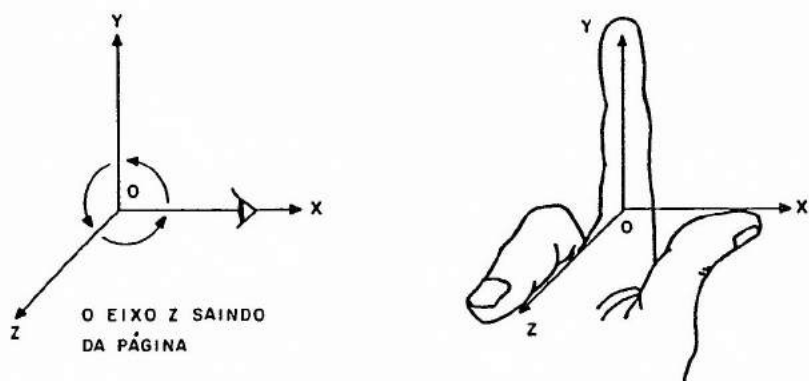
3.2- EQUAÇÕES DE UMA RETA DE UM PLANO

Esta seção tem por objetivo lembrar algumas definições e resultados da geometria analítica.

No plano ou no espaço uma reta de direção \vec{D} passando pelo ponto M_0 é o conjunto de pontos M tal que:

$$\vec{M_0M} = t \vec{D} \quad \text{com} \quad t \in \mathbb{R} \quad (3.8)$$

quando t percorre \mathbb{R} , M percorre a reta. Seja O um ponto do plano ou do espaço, então a expressão (3.8) é equivalente a (escrevendo $\vec{M_0M} = \vec{OM} - \vec{OM_0}$):



a- Sistema de coordenadas da mão direita (usado para as coordenadas universais).



b- Sistema de coordenadas da mão esquerda (usado para as coordenadas de visualização).

Figura 3.1 As duas orientações do espaço.

a REPRESENTAÇÃO MATRICIAL DE
a TRANSFORMAÇÕES 3D
a TRANSLAÇÃO e "SCALING"

= 8 =

T3 4 3 5
1 0 0 0
0 1 0 0
0 0 1 0
4 3 5 1

1 1 1 1+.xT3 4 3 5
5 4 6 1

S3 2 4 3
2 0 0 0
0 4 0 0
0 0 3 0
0 0 0 1

1 1 1 1+.xS3 2 4 3
2 4 3 1

A REPRESENTAÇÃO MATRICIAL DE
A ROTACIONES EM 3D

= 9 =

RX 'A'
1 0 0 0
0 COS A SEN A 0
0 -SEN A COS A 0
0 0 0 1

RY 'A'
COS A 0 -SEN A 0
0 1 0 0
SEN A 0 COS A 0
0 0 0 1

RZ 'A'
COS A SEN A 0 0
-SEN A COS A 0 0
0 0 1 0
0 0 0 1

A+30

A''RX 'A'
1 0 0 0
0 0.8660254038 0.5 0
0 -0.5 0.8660254038 0
0 0 0 1

1 1 1 1+.x A''RX 'A'
1 0.3660254038 1.366025404 1

$$\vec{OM} = t \vec{D} + \vec{OM}_0 \quad (3.9)$$

Sejam $M(x, y, z)$, $M_0(x_0, y_0, z_0)$ e $\vec{D}(d_x, d_y, d_z)$ em $S = \{0, \vec{x}, \vec{y}, \vec{z}\}$ no espaço. Então a equação paramétrica da reta de direção \vec{D} passando por M_0 é dada partir de (3.9) por

$$(x, y, z) = t(d_x, d_y, d_z) + (x_0, y_0, z_0), \quad (3.10)$$

ou ainda, escrevendo (3.10) na forma de três equações:

$$\begin{aligned} x &= d_x t + x_0 \\ y &= d_y t + y_0 \\ z &= d_z t + z_0 \end{aligned} \quad (3.11)$$

Seja $M_1(x_1, y_1, z_1)$ em S um segundo ponto da reta ($\neq M_0$). Fazendo $\vec{D} = \vec{M}_0\vec{M}_1 = \vec{OM}_1 - \vec{OM}_0$, então $(d_x, d_y, d_z) = (x_1 - x_0, y_1 - y_0, z_1 - z_0)$ e as equações (3.11) podem escrever-se:

$$\begin{aligned} x &= (x_1 - x_0) t + x_0 \\ y &= (y_1 - y_0) t + y_0 \\ z &= (z_1 - z_0) t + z_0 \end{aligned} \quad (3.12)$$

Observa-se que o segmento $[M_0M_1]$ corresponde aos valores de t no intervalo $[0, 1]$.

Sejam M_0 um ponto e \vec{N} um vetor do plano ou do espaço. Seja \mathcal{C} o conjunto dos pontos M tal que $\langle \vec{M}_0\vec{M}, \vec{N} \rangle = 0$ (isto é, $\vec{M}_0\vec{M} \perp \vec{N}$), então \vec{N} é dito ser uma normal a \mathcal{C} (ver figura 3.2). Observa-se que $M_0 \in \mathcal{C}$. Seja O um ponto do plano ou do espaço, então:

$$\langle \vec{M}_0\vec{M}, \vec{N} \rangle = \langle \vec{OM}, \vec{N} \rangle - \langle \vec{OM}_0, \vec{N} \rangle,$$

e $M \in \mathcal{C}$ implica:

$$\langle \vec{OM}, \vec{N} \rangle - \langle \vec{OM}_0, \vec{N} \rangle = 0 \quad (3.13)$$

Seja P um ponto do plano ou do espaço e H sua projeção em

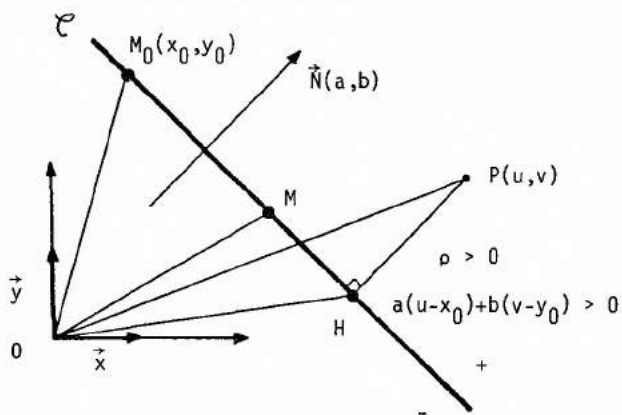


Figura 3.2 \mathcal{C} conjunto dos pontos contendo M_0 e tendo \vec{N} como normal;
 ρ distância algébrica de P para \mathcal{C}
(aqui representação no plano).

\mathcal{C} paralelamente a \vec{N} . Seja ρ a distância algébrica de P para \mathcal{C} (ver figura 3.2), isto é:

$$\vec{HP} = \rho \vec{N} / \|\vec{N}\| . \quad (3.14)$$

Por outro lado:

$$\langle \vec{HP}, \vec{N} \rangle = \langle \vec{OP}, \vec{N} \rangle - \langle \vec{OH}, \vec{N} \rangle .$$

O ponto H sendo um ponto de \mathcal{C} (3.13) implica:

$$\langle \vec{HP}, \vec{N} \rangle = \langle \vec{OP}, \vec{N} \rangle - \langle \vec{OM}_0, \vec{N} \rangle . \quad (3.15)$$

Usando (3.14) tem-se também:

$$\langle \vec{HP}, \vec{N} \rangle = \langle \rho \vec{N} / \|\vec{N}\|, \vec{N} \rangle = \rho \|\vec{N}\| . \quad (3.16)$$

Juntando (3.15) e (3.16) chega-se a uma expressão para ρ :

$$\rho = (\langle \vec{OP}, \vec{N} \rangle - \langle \vec{OM}_0, \vec{N} \rangle) / \|\vec{N}\| . \quad (3.17)$$

Considera-se agora o caso do plano com seu sistema de coordenadas S . Sejam $M(x, y)$, $M_0(x_0, y_0)$ e $\vec{N}(a, b)$ em S , então a expressão (3.13) pode-se escrever:

$$a(x - x_0) + b(y - y_0) = 0.$$

Denotando por $c = -(ax_0 + by_0)$, (3.18)

tem-se $ax + by + c = 0$.

Reconhece-se a equação de uma reta no plano. Em outros termos, no plano \mathcal{C} é uma reta.

Sejam $M(x, y)$, $M_1(x_1, y_1)$ e $M_2(x_2, y_2) \in \mathcal{C}$, M_1 e M_2 distintos, então:

$$\begin{aligned} ax + by + c &= 0 \\ ax_1 + by_1 + c &= 0 \\ ax_2 + by_2 + c &= 0, \end{aligned}$$

essas três equações formam um sistema de equações homogêneas nas variáveis a , b e c . Para ter uma solução outra que a solução trivial ($a = b = c = 0$) tem que ter:

$$\det \begin{bmatrix} x & y & 1 \\ x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \end{bmatrix} = 0.$$

Desenvolvendo o determinante acima em termo dos cofatores da primeira linha, tem-se:

$$x \det \begin{bmatrix} y_1 & 1 \\ y_2 & 1 \end{bmatrix} - y \det \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \end{bmatrix} + \det \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} = 0$$

ou ainda:

$$x \det \begin{bmatrix} 1 & y_1 \\ 1 & y_2 \end{bmatrix} + y \det \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \end{bmatrix} - \det \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} = 0.$$

Identificando com a equação da reta tem-se:

$\forall k \in \mathbb{R}$, $k \neq 0$, $a = k \alpha$, $b = k \beta$ e $c = k \gamma$ onde

$$\alpha = \det \begin{bmatrix} 1 & y_1 \\ 1 & y_2 \end{bmatrix}, \quad \beta = \det \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \end{bmatrix} \quad \text{e} \quad \gamma = - \det \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix}. \quad (3.15)$$

Dai um algoritmo para calcular os parâmetros da equação de uma reta passando por dois pontos M_1 e M_2 , ou calcular o vetor de coordenadas de uma normal a essa reta.

Seja $\vec{N}(\alpha, \beta)$ em S , α e β dados por (3.19), então o sistema $(\vec{N}, \overrightarrow{M_1 M_2})$ tem a mesma orientação que S , pois $\det [\vec{N}, \overrightarrow{M_1 M_2}]_S > 0$. Verificando:

$$[\vec{N}, \overrightarrow{M_1 M_2}]_S = \begin{bmatrix} \alpha & x_2 - x_1 \\ \beta & y_2 - y_1 \end{bmatrix} = \begin{bmatrix} y_2 - y_1 & x_2 - x_1 \\ x_1 - x_2 & y_2 - y_1 \end{bmatrix},$$

assim:

$$\det \begin{bmatrix} \vec{N} & \overrightarrow{M_1 M_2} \end{bmatrix}_S = (x_2 - x_1)^2 + (y_2 - y_1)^2 > 0.$$

Seja $P(u, v)$ em S , então (3.17) implica que a distância algébrica ρ de $P(u, v)$ a reta de normal $\vec{N}(a, b)$ passando por $M_0(x_0, y_0)$ é:

$$(3.18) \quad \rho = (a(u - x_0) + b(v - y_0)) / (a^2 + b^2)^{1/2}. \text{ Usando tem-se:}$$

$$\rho = (au + bv + c) / (a^2 + b^2)^{1/2}.$$

Se $\rho > 0$, isto é, se $a(u - x_0) + b(v - y_0) > 0$ ou $au + bv + c > 0$, então (3.14) implica que \vec{MP} tem o mesmo sentido que $\vec{N}(a, b)$, e que $P(u, v)$ é situado no semi-plano delimitado pela reta e apontado por N . Esse semi-plano é dito positivo, ou outro é dito negativo (ver figura 3.2).

Considera-se agora o caso do espaço com seu sistema de coordenadas S . Sejam $M(x, y, z)$, $M(x_0, y_0, z_0)$ e $\vec{N}(a, b, c)$ em S , então (3.13) pode-se escrever: $a(x - x_0) + b(y - y_0) + c(z - z_0) = 0$. Denotando por $d = -(ax_0 + by_0 + cz_0)$, (3.20) tem-se $ax + by + cz + d = 0$.

Reconhece-se a equação de um plano no espaço. Em outros termos, no espaço \mathcal{E} é um plano.

Sejam $M(x, y, z)$, $M_1(x_1, y_1, z_1)$, $M_2(x_2, y_2, z_2)$ e $M_3(x_3, y_3, z_3) \in \mathcal{E}$, M_1, M_2 e M_3 não colineares, então pelo mesmo raciocínio feito no caso do plano tem-se:

$\forall k \in \mathbb{R}, k \neq 0, a = k\alpha, b = k\beta, c = k\gamma$ e $d = k\delta$ onde

$$\alpha = \det \begin{bmatrix} 1 & y_1 & z_1 \\ 1 & y_2 & z_2 \\ 1 & y_3 & z_3 \end{bmatrix}, \quad \beta = \det \begin{bmatrix} x_1 & 1 & z_1 \\ x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \end{bmatrix}$$

$$\gamma = \det \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix} \text{ e } \quad \delta = - \det \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix} \quad (3.21).$$

Dá um algoritmo para calcular o vetor de coordenadas de uma normal a um plano passando por três pontos.

Seja $\vec{N}(\alpha, \beta, \gamma)$ em S , α, β e γ dados por (3.21), então \vec{N} é o produto vetorial de $\vec{M_1 M_2}$ por $\vec{M_2 M_3}$ (ou $\vec{M_2 M_3}$, por $\vec{M_3 M_1}$, ou $\vec{M_3 M_1}$ por $\vec{M_1 M_2}$) e denota-se $\vec{M_1 M_2} \times \vec{M_2 M_3}$. Por construção $\vec{M_1 M_2} \times \vec{M_2 M_3}$ é normal ao plano contendo os pontos M_1, M_2 e M_3 . Pode-se verificar que o sistema $\{\vec{N}, \vec{M_1 M_2}, \vec{M_2 M_3}\}$ tem a mesma orientação que S .

3.3- VISÃO EM 2D DE OBJETOS EM 3D

Como já foi visto (cf. seção 2.3), visualizar uma cena em 2D consiste em escolher uma janela no espaço das coordenadas universais e um campo de visualização no espaço das coordenadas de visualização, em seguida em determinar a transformação de normalização correspondente (isto é, uma matriz 3×3) e finalmente em aplicar esta ao modelo da cena.

A visualização de uma cena em 3D decompõe-se também em três passos. No entanto, a novidade é a introdução da noção de projeção para resolver a passagem de uma cena descrita em 3D para sua representação em forma de uma imagem em 2D.

No primeiro passo escolhe-se um volume de visualização (no lugar da janela) no espaço das coordenadas universais. Na prática, esse volume é especificado indiretamente a partir da escolha de um plano de projeção no espaço das coordenadas universais, de uma janela no plano de projeção, de um tipo de projeção, e eventualmente de dois planos de cerceamento. No primeiro passo escolhe-se também um campo de visualização no espaço das coordenadas de visualização.

No segundo passo determina-se, a partir dos dados do primeiro passo, a transformação de normalização (cf: seção 2.3). Essa transformação não é mais afina (como no caso 2D), mas projetiva (cf. seção 2.2), e caracterizada por uma matriz 4×4 . A transformação de normalização é obtida considerando que ela deve transformar o volume de visualização no cubo unitário $[0, 1]^3$.

O terceiro e último passo consiste, como no caso em 2D em aplicar esta transformação ao modelo da cena.

As projeções geralmente consideradas em computação gráfica são paralelas ou em perspectiva, dependendo se o centro de projeção situa-se no infinito ou não. No caso da projeção paralela, no lugar de um centro de projeção define-se uma direção de projeção (ver figura 3.3).

Aqui é apresentado apenas o caso das projeções paralelas. Esse tipo de projeção produz imagens de cena 3D com menos realismo, mas por outro lado a propriedade de paralelismo é conservada, o que é interessante no caso por exemplo do desenho industrial.

A projeção paralela pode ser ortogonal ou ôblica. No primeiro caso a direção de projeção é normal ao plano de projeção, no segundo caso ela não é. Em seguida é mostrado como obter a matriz de transformação correspondente a uma projeção paralela ôblica (o caso ortogonal sendo visto como um caso particular de caso ôblico).

Para definir uma visão em 2D de uma cena em 3D numa projeção paralela, precisa-se especificar (ver figura 3.4):

1- No espaço das coordenadas universais referenciado por $S = \{0, \vec{x}, \vec{y}, \vec{z}\}$:

- um plano de projeção definido a partir de um ponto R de referência pelo qual ele deve passar e um vetor \vec{N} indicando a direção da normal.

- uma janela no plano de projeção definido a partir de dois pontos (u_{\min}, v_{\min}) e (u_{\max}, v_{\max}) expressos num sistema de coordenadas $\{R, \vec{u}, \vec{v}\}$ ortonormal e situado no plano. O vetor \vec{v} (indicando a verti

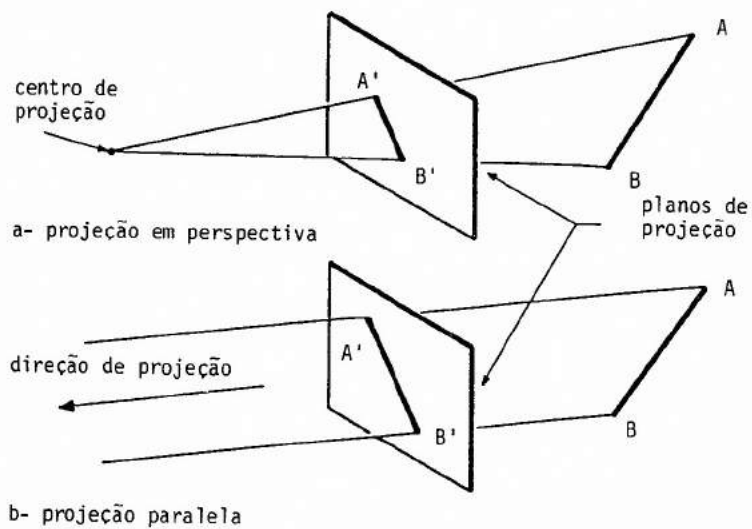


Figura 3.3 Projeções em perspectiva e paralela de um segmento $[AB]$.

cal da janela) é obtida a partir da especificação de um vetor \vec{V} . Mais precisamente, \vec{v} apresenta-se com um fator de proporcionalidade positivo, como a projeção paralelamente a \vec{N} no plano de projeção do vetor \vec{V} . O vetor \vec{u} é escolhido de tal maneira que o sistema $\{\vec{u}, \vec{v}, \vec{N}\}$ seja da mão esquerda.

- uma direção de projeção dada pelo vetor \vec{D} .
- dois planos de cerceamento (um dianteiro, o outro traseiro) paralelos ao plano de projeção e situados, respectivamente, a uma distância d e t desse plano.

2- No espaço das coordenadas de visualização referenciado por $S = \{0', \vec{x}', \vec{y}', \vec{z}'\}$:

- um campo de visualização. Para simplificar, considera-se aqui que o campo de visualização ocupa todo o quadrado delimitado pelas retas $x' = 0$, $x' = 1$, $y' = 0$ e $y' = 1$ no plano $z' = 0$.

Pela descrição acima, o volume de visualização apresenta-se como um paralelepípedo com quatro faces paralelas à direção de projeção e duas outras opostas formadas pelos dois planos de cerceamento.

O problema agora é achar a transformação de normalização, caracterizada por uma matriz 4×4 , representando o mapeamento do volume de visualização acima no cubo unitário delimitado pelos planos $x' = 0$, $x' = 1$, $y' = 0$, $y' = 1$, $z' = 0$ e $z' = 1$. Nesse mapeamento decide-se que o ponto $0'$ deve corresponder à projeção, paralelamente a \vec{D} no plano de cerceamento dianteiro, do vértice (u_{\min}, v_{\min}) da janela, que x' , y' e z' devem corresponder a \vec{u} , \vec{v} e $\vec{D} / \|\vec{D}\|$ respectivamente, e que o ponto $(1, 1, 1)$ em S' deve corresponder à projeção, paralelamente a \vec{D} no plano de cerceamento traseiro, do vértice (u_{\max}, v_{\max}) (ver figura 3.4).

A transformação da normalização representando esse mapeamento, é definida no espaço das coordenadas universais, e obtida como a composição de seis transformações elementares:

1- Uma translação, de matriz T , que leva o ponto R para o ponto origem O , isto é, de vetor $-\vec{OR}$

2- Uma rotação, de matriz R_3 , em volta da origem O tal que \vec{N} seja levado para o eixo $-z$, \vec{u} para \vec{x} (e conseqüentemente \vec{v} para \vec{y}). Escrevendo R_3 na forma (3.6) onde $M = \begin{bmatrix} \vec{v}_1 & \vec{v}_2 & \vec{v}_3 \end{bmatrix}_S$ e usando a propriedade (3.7), então deduz-se que $M = \begin{bmatrix} \vec{u} & \vec{v} & \vec{n} \end{bmatrix}$ onde $\vec{n} = -\vec{N} / \|\vec{N}\|$.

Por construção \vec{u} é ortogonal ao plano paralelo a \vec{N} e \vec{V} , e o sistema $\{\vec{u}, \vec{N}, \vec{V}\}$ tem a mesma orientação que S , daí $\vec{u} = \vec{N} \times \vec{V} / \|\vec{N} \times \vec{V}\|$. Por construção também, $\{\vec{u}, \vec{v}, \vec{n}\}$ forma um sistema ortonormal, daí $\vec{v} = \vec{n} \times \vec{u}$. Isto implica que:

$$\begin{aligned} \vec{v}_3 &= -\vec{N} / \|\vec{N}\| \\ \vec{v}_1 &= \vec{N} \times \vec{V} / \|\vec{N} \times \vec{V}\| \\ \vec{v}_2 &= \vec{v}_3 \times \vec{v}_1 \end{aligned}$$

3- Um "retorno", de matriz T_{RL} , para trocar o sentido de orientação do sistema de coordenadas, isto é, aqui para passar do sistema da mão direita (das coordenadas universais) para o sistema de mão esquerda (das coordenadas de visualização). Isto é obtido trocando o sinal da coordenada z :

$$T_{RL} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

4- Uma "tesourada" de eixo z , de matriz SH_z , do volume de visualização já transformado pelos passos 2 e 3 acima de maneira que suas quatro faces situadas entre as duas faces opostas ortogonais a z , sejam ortogonais a um dos eixos x ou y . Uma "tesourada" de eixo z é dada pela matriz:

$$SH_z = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ a & b & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Seja $\vec{D}' = [\alpha \ \beta \ \gamma \ 1] = [\vec{D}] R_3 T_{RL}$, os parâmetros a e b tem que ser tais que:

$$\text{isto é: } \begin{bmatrix} [\vec{D}'] \cdot SH_z \\ \alpha + a \ \gamma \ \beta + b \ \gamma \ \gamma \ 1 \end{bmatrix} = \begin{bmatrix} 0 \ 0 \ \gamma \ 1 \\ 0 \ 0 \ \gamma \ 1 \end{bmatrix} .$$

Assim a e b são: $a = -\alpha / \gamma$ e $b = -\beta / \gamma$.

No caso de uma projeção ortogonal $\alpha = \beta = 0$, isto é, $a = b = 0$ e SH_z passa a ser a matriz identidade.

5- Uma translação, de matriz T_{par} , do volume de visualização de maneira que o vértice (u_{min}, v_{min}, d) seja levado para a origem 0. O vetor dessa translação é $(-u_{min}, -v_{min}, -d)$.

6- Um escalonamento em S' , de matriz S_{par} , de maneira a transformar o volume de visualização num cubo unitário. Os fatores desse escalonamento são $1 / (u_{max} - u_{min})$, $1 / (v_{max} - v_{min})$ e $1 / (t-d)$.

Finalmente a matriz de transformação de normalização N_{par} é dada por:

$$N_{par} = T R_3 T_{RL} SH_z T_{par} S_{par} .$$

O volume de visualização foi levado ao cubo unitário para facilitar o cerceamento (cf. seção 5.2) e também a operação de projeção do conteúdo desse volume para o plano de projeção. De fato agora a projeção é simplesmente dada pela matriz de projeção ortogonal de direção z no plano $z = 0$:

$$M_z = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} .$$

Assim, a visão 2D é obtida a partir dos pontos do plano $z = 0$ transformados dos pontos da cena em 3D por $N_{par} \cdot M_z$.

Para ilustrar a técnica acima, foi criada a função APL

NPAR que produz a matriz de normalização N_{par} . Essa função tem por argumento o seguinte vetor APL:

VRP VPN VUP DOP UMIN VMIN F UMAX VMAX B,

onde VRP, VPN, VUP e DOP são os vetores de coordenadas de R , \vec{N} , \vec{V} e \vec{D} , respectivamente, e UMIN, VMIN, F, UMAX, VMAX e B são os escalários u_{min} , v_{min} , d , u_{max} , v_{max} , t . A função NPAR usa internamente as funções T3, RRR, TRL, SHPAR, TPAR e SPAR que produzem respectivamente as matrizes T , R_3 , TRL, SH_z , T_{par} e S_{par} . Por sua vez, algumas dessas funções usam internamente as funções RR, O3 (produz o produto vetorial de dois vetores geométricos do espaço), e N3 (cf. anexo).

O exemplo escolhido é o da visualização de uma casa. O modelo de dados para representar a casa é constituído por dois vetores APL CASAF e CASAT. Cada elemento desses dois vetores representa um vértice respectivamente da parte da frente e de trás da casa.

A "transparência" = 10 = mostra as coordenadas desses pontos. Observa-se em particular que a parte da frente e a de trás são contidos em dois planos paralelos ao plano $z = 0$.

A "transparência" = 11 = mostra as condições escolhidas para obter a visão em 2D da casa:

- R (VRP) é o vértice da casa mais afastada da origem 0.
- \vec{N} (VPN) forma com os eixos x , y e z ângulos iguais.
- \vec{D} (DOP) é igual a \vec{N} .
- \vec{V} (VUP) é paralelo ao eixo y .

A escolha feita para \vec{N} e \vec{D} corresponde a uma projeção isométrica. Essa projeção não conserva os ângulos, mas permite a medida com a mesma escala dos segmentos paralelos a um dos eixos x , y ou z .

- u_{min} , v_{min} , d (F), u_{max} , v_{max} , t (B) são escolhidos de maneira que a casa coubesse inteiramente (mas também sem desproporção) dentro do volume de visualização.

A "transparência" = 11 = mostra também a matriz de trans-

A DESENHO EM 3D = 10 =
A DESCRIÇÃO DO OBJETO

CASAF+(0 0 54)(16 0 54)(16 10 54)

CASAF+CASAF,(8 16 54)(0 10 54)(0 0 54)

CASAT+(0 0 30)(16 0 30)(16 10 30)

CASAT+CASAT,(8 16 30)(0 10 30)(0 0 30)

>CASAF

0 0 54
16 0 54
16 10 54
8 16 54
0 10 54
0 0 54

>CASAT

0 0 30
16 0 30
16 10 30
8 16 30
0 10 30
0 0 30

a DESENHO EM 3D = 11 =
a COMPUTAÇÃO DA TRANSFORMAÇÃO DE NORMALIZAÇÃO
a PARA UMA PROJEÇÃO PARALELA

VRP←16 10 54

VPN←-1 -1 -1

VUP←0 1 0

DOP←-1 -1 -1

FMIN←-30 -30 -1

FMAX←30 30 40

M←NFAR VRP VPN VUP DOP,FMIN,FMAX

M

1.17851130E-2	-0.00680413817	-0.0140817138 0
1.30297007E-17	0.01360827635	-0.0140817138 0
-1.17851130E-2	-0.00680413817	-0.0140817138 0
9.47834294E-1	0.8402069087	1.150927355 1

formação de normalização N_{par} chamada aqui de M e produzida pela função APL NPAR.

A "transparência" = 12 = mostra a transformação de cada vértice da casa. Assim são obtidos os vetores APL AF e AT cujos elementos representam os vértices transformados no espaço das coordenadas universais ou, o que é equivalente as novas coordenadas no espaço das coordenadas de visualização.

Já que na projeção paralela de direção z no plano $z = 0$ os vértices conservam suas abscissas e ordenadas e que a coordenada z não tem mais utilidade no processo de saída, o último passo antes da saída é a seleção das duas primeiras coordenadas de cada vértice. Isto é feito usando a função APL C2.

A saída da casa é obtida traçando as duas linhas poligonais representando a parte da frente e a de trás da casa como mostrado na figura 10, e em seguida traçando as cinco linhas poligonais (reduzi - das a um segmento) ligando os vértices correspondentes à parte da frente e de trás da casa (ver figura 11).

Para ilustrar a composição de transformações em 3D, na "transparência" = 13 = a matriz MT, produto de uma rotação de eixo z de - 90 graus com a matriz de transformação de normalização M da "transparência" = 11 = é gerada. Usando o mesmo algoritmo que na "transparência" = 12 = obtêm-se a saída da casa girada de lado (ver figura 12).

```
0 DESENHO EM 3D                               = 12 =
0 TRANSFORMAÇÃO DAS COORDENADAS UNIVERSAIS
0 EM COORDENADAS DE VISUALIZAÇÃO
```

```
AF+HC(CH CASAF)+.x''=M
```

```
AT+HC(CH CASAT)+.x''=M
```

```
>AF
```

```
0.3114381917 0.4727834473 0.3905148049
0.5           0.3639172365 0.1652073827
0.5           0.5           0.0243902439
0.4057190958 0.6360827635 0.05255367167
0.3114381917 0.6088662108 0.249697666
0.3114381917 0.4727834473 0.3905148049
```

```
>C2 AF
```

```
0.3114381917 0.4727834473
0.5           0.3639172365
0.5           0.5
0.4057190958 0.6360827635
0.3114381917 0.6088662108
0.3114381917 0.4727834473
```

```
GSW 3 0 1 0 1
```

```
GSELNT 3
```

```
GPL''CV''C2''AF AT (produz figura 10)
```

```
GPL''CV''=[1](C2 AF),[.1]C2 AT (produz figura 11)
```

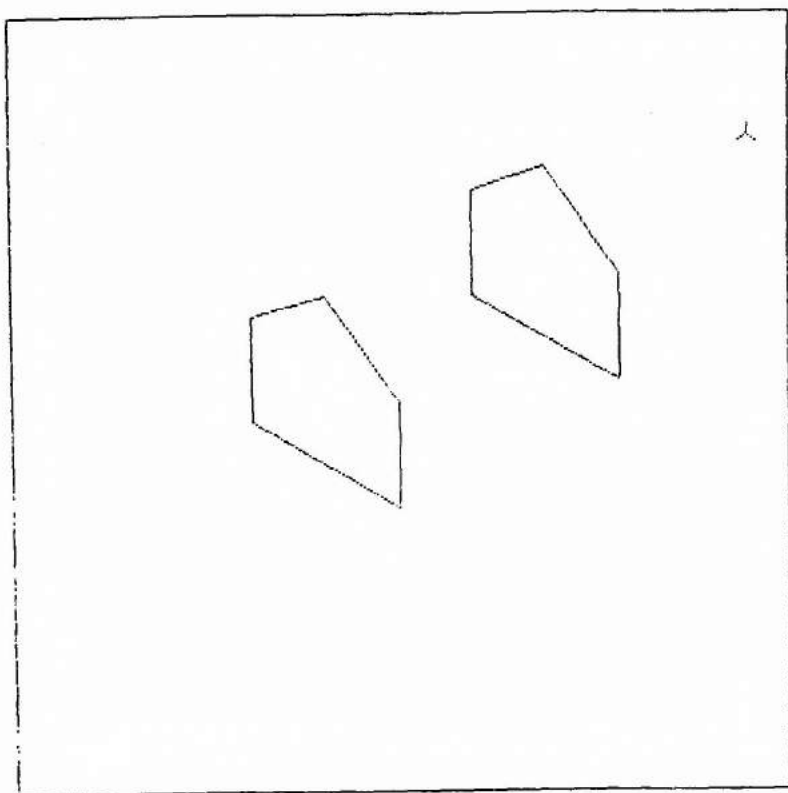



Fig. 10 - As partes da frente e de trás da casa numa projeção paralela isométrica.

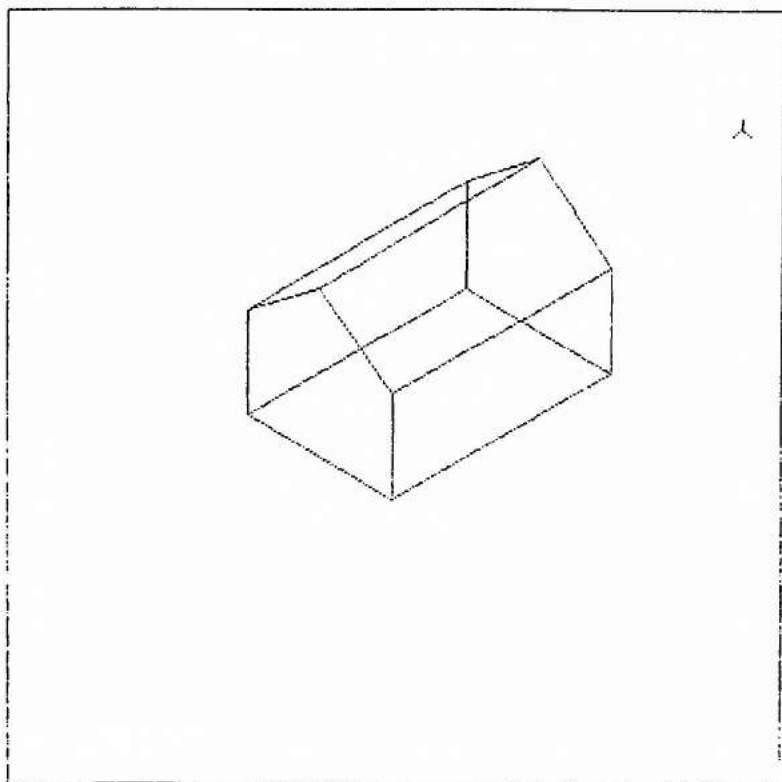


Fig. 11 - A casa com todas suas arestas numa projeção paralela isométrica.

A COMPOSIÇÕES DE TRANSFORMAÇÕES 3D = 13 =
A ROTACÃO EM VOLTA DO EIXO Z

A+90

A'RZ 'A'

1.743934249E-16	-1.000000000E0	0 0
1.000000000E0	1.743934249E-16	0 0
0.000000000E0	0.000000000E0	1 0
0.000000000E0	0.000000000E0	0 1

MT+(A'RZ 'A')+ .xM

BF+HC(CH CASAF)+.x"=MT

BT+HC(CH CASAT)+.x"=MT

GPL"CV" C2"BF BT

GPL"CV"=[1](C2 BF),[.1]C2 BT (produz figura 12)

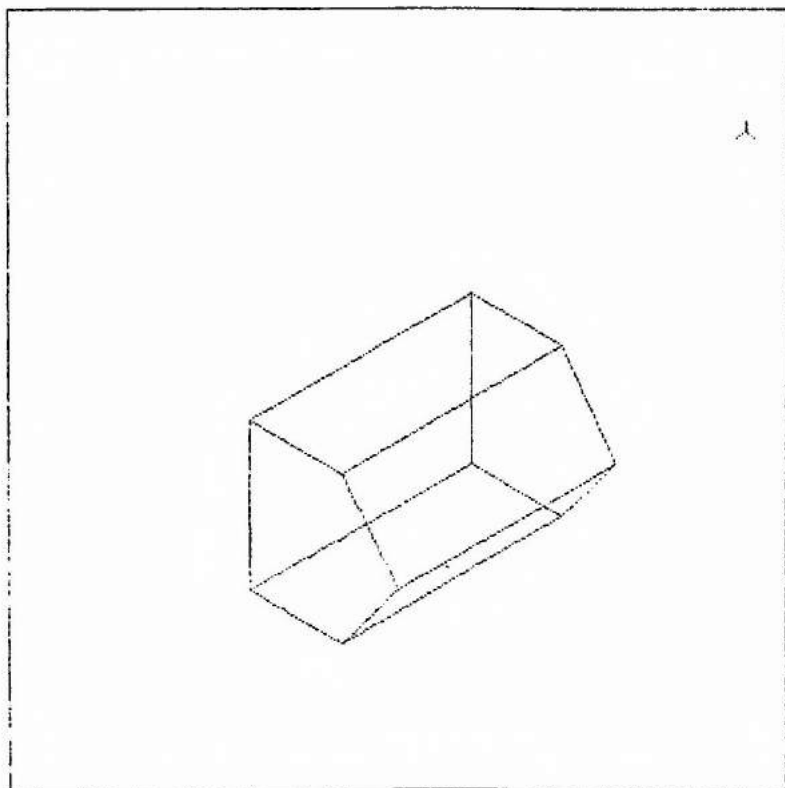


Fig. 12 - A casa (Figura 11) girada por uma rotação de eixo z de - 90 graus.

4- ESTRUTURA DE DADOS PARA POLÍGONOS E MALHAS DE POLÍGONOS

4.1- INTRODUÇÃO

No exemplo da casa em 3D do capítulo anterior, o traçado era obtido a partir de um conjunto de vértices armazenado em dois vetores (CASAF e CASAT) correspondendo às duas faces da casa (a parte da frente e a de trás). Com essa estrutura de dados era fácil traçar essas duas faces (ver figura 10), mas já era muito mais difícil completar o desenho. Para isto precisou de um algoritmo específico. Como pode-se ver na figura 11, na realidade a casa é constituída não de dois, mas de sete polígonos. Esses sete polígonos não são isolados, mas tem vértices e lados em comum. Eles formam uma malha de polígonos.

Os componentes de uma malha de polígonos são os vértices, os lados e os polígonos. Os polígonos são formados de vértices e lados, e os lados ligam dois vértices.

Uma malha de polígonos pode ser representada de várias maneiras. Dependendo da aplicação, procura-se satisfazer em parte os seguintes critérios:

- Facilidade de identificar os lados incidentes em um mesmo vértice.
- Facilidade de identificar os polígonos que partilham um mesmo lado.
- Facilidade de identificar os dois vértices de um mesmo lado.
- Facilidade de identificar todos os lados de um mesmo polígono.
- Facilidade de definir a malha.
- Facilidade de desenhar a malha.
- Eficiência de execução.
- Economia de espaço.

De uma maneira geral, quanto mais completa é a representação da malha, mais eficiência tem-se na resolução do problema gráfico, o preço a pagar é um aumento do espaço em memória para armazenar a malha e muitas vezes a necessidade de verificar a coerência da representação. A seguir são descritos três tipos de representação de malha de polígonos do caso 2D. Os detalhes específicos do caso 3D são dados na seção 5.1.

4.2- REPRESENTAÇÃO POR POLÍGONOS EXPLÍCITOS

Esta representação é a mais simples. Ela consiste em representar cada polígono na forma de uma lista de vértices, e cada vértice na forma de um vetor de coordenadas. Os vértices são ordenados de maneira que o polígono seja percorrido no sentido de orientação do sistema de coordenadas. Isto é arbitrário, mas é importante (cf. seção 4.5) adotar uma convenção uma vez por toda.

Na "transparência" = 14 =, um exemplo em 2D de uma casa composta de duas partes: o corpo da casa e o telhado, é dado. Cada uma dessas partes é representada por um polígono. Esses dois polígonos formam uma malha já que eles tem um lado em comum.

Conforme a representação por polígonos explícitos, os dados são armazenados na forma de dois vetores APL CORPO e TELHADO. Cada elemento desses vetores é um outro vetor APL de dois elementos representando a abscissa e ordenado do vértice. Observa-se que o lado (1, 3) - (3, 3) é comum ao corpo da casa e ao telhado.

Para representar um só polígono, ou polígonos isolados, essa representação economiza espaço. No entanto, para malha de polígonos tem redundância de dados já que para os vértices partilhados por dois ou mais polígonos suas coordenadas são repetidas. Essa representação também não permite identificar as relações entre vértices, lados e polígonos, a não ser, achar os vértices de um mesmo polígono. Finalmente, na hora de traçar o desenho, os lados em comum são plotados duas ou mais vezes.

4.3- REPRESENTAÇÃO POR POLÍGONOS DEFINIDOS COMO APONTADORES NUMA LISTA DE VÉRTICES

Nesta representação, as coordenadas dos vértices são armazenadas uma só vez na forma de uma lista.

Continuando o exemplo da casa e seguindo essa representação, foi criado na "transparência" = 15 = o vetor APL CASA2VT cujos ele-

^ REPRESENTAÇÃO DE MALHAS DE POLIGONOS = 14 =
^ POLIGONOS EXPLICITOS
^ EXEMPLO 2D

,[1.1]CORPO

1 1
3 1
3 3
1 3

,[1.1]TELHADO

3 3
2 4
1 3

A REPRESENTAÇÃO DE MALHAS DE POLIGONOS = 15 =
A POLIGONOS DEFINIDOS COMO APONTADORES
A NUMA LISTA DE VERTICES
A EXEMPLO 2D

```
      ,[1.1]CASA2PLVT  
1 2 3 5  
3 4 5
```

```
      (15),,[1.1]CASA2VT  
1 1 1  
2 3 1  
3 3 3  
4 2 4  
5 1 3
```


mentos são os vetores das coordenadas dos vértices. Dessa vez a ordem dos vértices pode ser qualquer.

Nessa representação, um polígono é definido como um vetor de apontadores para a lista de vértices. Os apontadores são ordenados de maneira que o polígono seja percorrido no sentido de orientação do sistema de coordenadas.

No exemplo da casa, dois polígonos são definidos na "transparência" = 15 = num só vetor APL CASA2PLVT. O primeiro elemento de CASA2PLVT corresponde ao corpo da casa e o segundo ao telhado. Observa-se que os dois polígonos tem os vértices 3 e 5 em comum.

Um exemplo de uso dessa representação é dado na "transparência" = 20 = da seção 4.5.

Já que as coordenadas dos vértices são armazenadas uma só vez, tem-se uma economia de espaço e as coordenadas de um vértice podem ser facilmente trocadas. No entanto, não se sabe explicitamente quais são os polígonos que partilham um mesmo lado, e os lados comuns a dois ou mais polígonos são traçados mais de uma vez. Esses dois problemas são eliminados na representação seguinte.

4.4- REPRESENTAÇÃO POR LADOS EXPLÍCITOS

Nesta representação tem-se também a lista de vértices da seção anterior. Continuando o exemplo da casa, isto significa que o vetor APL CASA2VT continua a ser usada. Mas dessa vez um polígono é representado como um vetor de apontadores para uma lista de lados na qual cada lado aparece uma só vez. Os apontadores de lados são ordenados de maneira que o polígono seja percorrido no sentido de orientação do sistema de coordenadas. Por sua vez, cada lado na lista de lados aponta para seus dois vértices na lista de vértices, e também para um ou mais polígonos nos quais o lado pertence. Adota-se também a convenção que o apontador de lado é positivo se ele aponta para um lado cujos vértices são ordenados no sentido de orientação do sistema de coordenadas, senão o aponta

tador é negativo. Um exemplo de uso da identificação explícita dos polígonos é dado na seção 5.1.

No exemplo da casa, os dois polígonos são definidos, na "transparência" = 16 = , num sô vetor APL CASA2PLLD. O primeiro elemento de CASA2PLLD corresponde ao corpo da casa, e o segundo ao telhado. A lista de lados é armazenada no vetor APL CASA2LD. Cada elemento de CASA2LD é um vetor correspondente a um determinado lado, os dois primeiros elementos apontam para os dois v̄rtices do lado, os elementos seguintes apontam para os polígonos contendo o lado. Por exemplo, o telhado (o segundo polígonos) é composto dos lados 3, 4 e - 6. Os v̄rtices desse polígonos são dados na primeira (por ser apontador positivo) coluna dos vetores 3 e 4 de CASA2LD, ou seja v̄rtices 3 e 4, e na segunda (por ser apontador negativo) coluna do vetor 6, ou seja v̄rtice 5.

Nesta representação, a malha de polígonos é desenhada traçando os lados ao invés dos polígonos, evitando assim traçar mais de uma vez os lados comuns a dois ou mais polígonos. A "transparência" = 17 = mostra como é feito esse tipo de saída. Para isto foi criada a função APL LDVT de dois argumentos. O da esquerda é a lista dos v̄rtices e o da direita é a descrição de um lado como feita na lista de lados. A função produz um vetor de dois vetores de coordenados representando os v̄rtices do lado. No exemplo da casa são assim obtidos os seis lados que são em seguida traçados usando a função GPL (ver figura 13).

4.5- EXEMPLOS DE APLICAÇÕES

Um polígonos pode ser representado por seus v̄rtices ou seus lados. Em seguida, dois exemplos de uso dessas representações aplicados ao preenchimento de polígonos são dados.

Em computação gráfica, preencher um polígonos significa associar aos pontos interiores um nível de cinza ou uma cor prefixada. Para isto tem que reconhecer se um ponto qualquer é exterior ou interior a um polígonos. No caso dos polígonos convexos, esse problema é resolvido desde que se sabe reconhecer a posição de um ponto em relação a uma reta

REPRESENTAÇÃO DE MALHAS DE POLIGONOS = 16 =
LADOS EXPLICITOS
EXEMPLO 2D

(12),,[1.1]CASA2PLLD
1 1 2 6 -5
2 3 4 -6

(16),,[1.1]CASA2LD
1 1 2 1
2 2 3 1
3 3 4 2
4 4 5 2
5 1 5 1
6 3 5 1 2

(15),,[1.1]CASA2VT
1 1 1
2 3 1
3 3 3
4 2 4
5 1 3

A DESENHO EM 2D USANDO A REPRESENTAÇÃO = 17 =
A POR LADOS EXPLICITOS

```
CASA2VT LDVT 2>CASA2LD  
3 1 3 3
```

```
=(<CASA2VT)LDVT"CASA2LD  
1 1 3 1  
3 1 3 3  
3 3 2 4  
2 4 1 3  
1 1 1 3  
3 3 1 3
```

- GPL"CV"(<CASA2VT)LDVT"CASA2LD (produz figura 13)

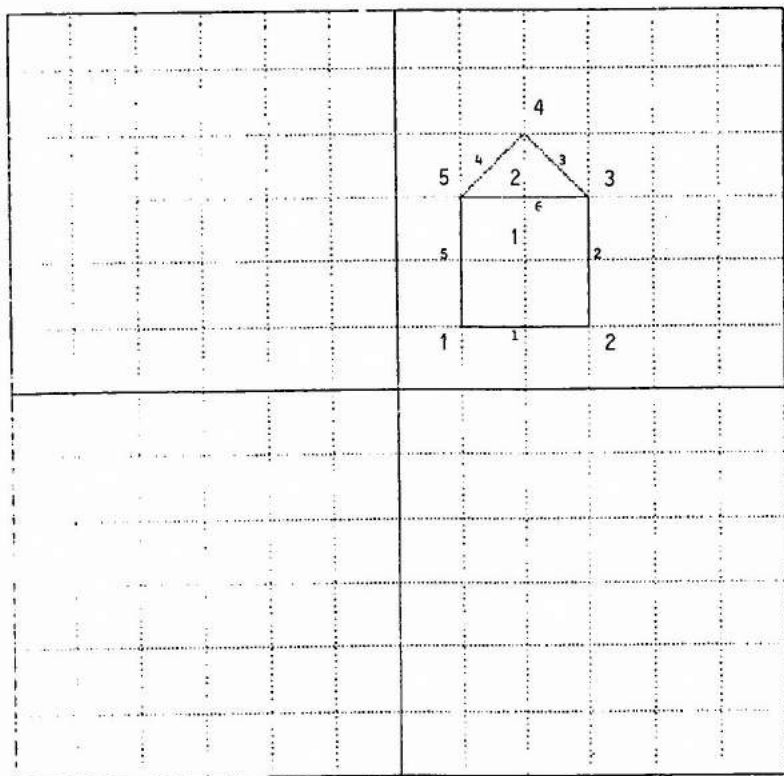


Fig. 13 - A casa desenhada usando a representação por lados explícitos (as arestas são traçadas só uma vez).

orientada.

Seja M_1 e M_2 dois pontos e α , β e γ os parâmetros da equação da reta passando por M_1 e M_2 dados por (3.19). Seja $N(\alpha, \beta)$, já foi visto também na seção 3.2 que $(\vec{N}, \overrightarrow{M_1 M_2})$ tem o mesmo sentido que o sistema de coordenadas S . Assim, se S é orientado no sentido anti-horário \vec{N} aponta para o lado direito da reta quando percorrido de M_1 para M_2 , isto é, o semi-plano positivo é do lado direito. Assim, se $\alpha u + \beta v + \gamma$ é positivo, então $P(u, v)$ é situado à direita da reta, senão P é situado à esquerda.

A "transparência" = 18 mostra o detalhe do cálculo dos parâmetros α , β e γ da equação da reta contendo o segundo lado da casa, isto é, passando pelos pontos (3, 1) e (3, 3) nesta ordem. Esses parâmetros valem 2, 0 e -6. Foi criada a função APL N2 para calcular esses parâmetros conhecendo um par de pontos. As duas últimas linhas da "transparência" mostram que a expressão $\alpha u + \beta v + \gamma$ é negativa para o ponto (2, 2) situado à esquerda da reta e positiva para o ponto (4, 2) situado à direita, o que bate com a teoria já que S é orientado no sentido anti-horário.

Um polígono é convexo se ele é totalmente contido num dos dois semi-planos delimitados pela reta contendo qualquer lado. O semi-plano contendo o polígono é dito interior e o outro exterior.

Seja M_1 e M_2 dois vértices de um lado de um polígono, encontrados nesta ordem quando ele é percorrido no sentido anti-horário. Se o sentido de orientação do sistema de coordenadas S é também anti-horário, e o polígono convexo, então $\vec{N}(\alpha, \beta)$ (α e β dados por (3.19)) aponta para o semi-plano exterior para qualquer lado dos polígonos. Trocando o sentido anti-horário por horário, os vértices acima são encontrados na ordem inversa, e conseqüentemente $\vec{N}(\alpha, \beta)$ continua apontando para o semi-plano exterior. Assim, tem-se a seguinte regra. Se o polígono é percorrido no sentido de orientação do sistema de coordenadas, $\vec{N}(\alpha, \beta)$ aponta para o semi-plano exterior para qualquer lado. Em outros termos, os semi-planos exteriores são positivos e os interiores negativos. Nessa

A NORMAL DE UMA RETA = 18 =
A POSIÇÃO RELATIVA ENTRE UM PONTO E UMA RETA

R←CASA2VT LDVT 2→CASA2LD

R

3 1 3 3

A←B←C←→R

A[,1]←1

B[,2]←1

A B C

1 1 3 1 3 1

1 3 3 1 3 3

DET"A B C

2 0 6

1 1 -1×DET"A B C

2 0 -6

N2 R

2 0 -6

2 2 1+ .xN2 R

-2

4 2 1+ .xN2 R

2

condição, se $P(u, v)$ é um ponto interior ao polígono, ele pertence a todos os semi-planos interiores e $\alpha + \beta v + \gamma$ é negativo para qualquer lado do polígono.

Para extrair o modelo do polígono em termo dos seus lados, usa-se a representação por lados explícitos da seção anterior (que garante a condição acima). Foi criada a função APL PLLDVT para produzir esse modelo. O argumento da esquerda dessa função é o vetor APL contendo a lista dos vértices e a lista dos lados, o argumento da direita é a descrição do polígono em termo de apontadores na lista de lados. A função produz um vetor APL cujos elementos são os lados representados na forma de vetores de dois vetores de coordenadas representando os vértices do lado.

A "transparência" = 19 = mostra o modelo do primeiro polígono da casa (representando o corpo da casa) obtido usando a função PLLDVT. Em seguida, a transparência mostra como é obtida a posição dos pontos (2, 2) e (0, 0) em relação ao polígono. Para o primeiro ponto $\alpha + \beta v + \gamma$ é sempre negativo, isto é, o ponto é interior. Para o segundo ponto, $\alpha + \beta v + \gamma$ não é sempre negativo, isto é, o ponto é exterior.

O algoritmo descrito acima pode servir para o preenchimento de um polígono convexo. No entanto, na prática usa-se outros algoritmos mais eficientes, como aqueles apresentados na seção 6.1. Na seção 6.6, esse algoritmo será usado para saber se um polígono é contido num outro convexo.

Para o preenchimento da área poligonal, o padrão GKS tem uma função de saída dedicada, chamada GFA (cf. seção 1.5). A "transparência" = 20 = mostra como usar essa função para desenhar a casa. Para extrair o modelo do polígono em termo de seus vértices, usa-se a representação por apontadores numa lista de vértices da seção 4.3. Foi criada a função APL PLVT para produzir esse modelo. O argumento da esquerda é a lista de vértices, o da direita é a descrição do polígono em termo de apontadores na lista de vértices. Na primeira chamada a GFA, o primeiro

a POSIÇÃO RELATIVA ENTRE UM PONTO E = 19 =
a UM POLIGONO CONVEXO

P=CASA2VT CASA2LD PLLDVT 1=CASA2PLLD

▷P

1 1 3 1
3 1 3 3
3 3 1 3
1 3 1 1

(c2 2 1)+.x"N2"P

72 72 72 72

λ/0)(c2 2 1)+.x"N2"P

1 a PONTO INTERIOR

(c0 0 1)+.x"N2"P

2 76 76 2

λ/0)(c0 0 1)+.x"N2"P

0 a PONTO EXTERIOR

A DESENHO EM 2D

= 20 =

A PREENCHIMENTO DE AREAS

P=CASA2VT PLVT 1=CASA2PLVT

P

1 1 3 1 3 3 1 3

GFA CV P

(produz figura 14)

P=CASA2VT PLVT 2=CASA2PLVT

P

3 3 2 4 1 3

GSFAIS 'SOLID'

GFA CV P

(produz figura 15)

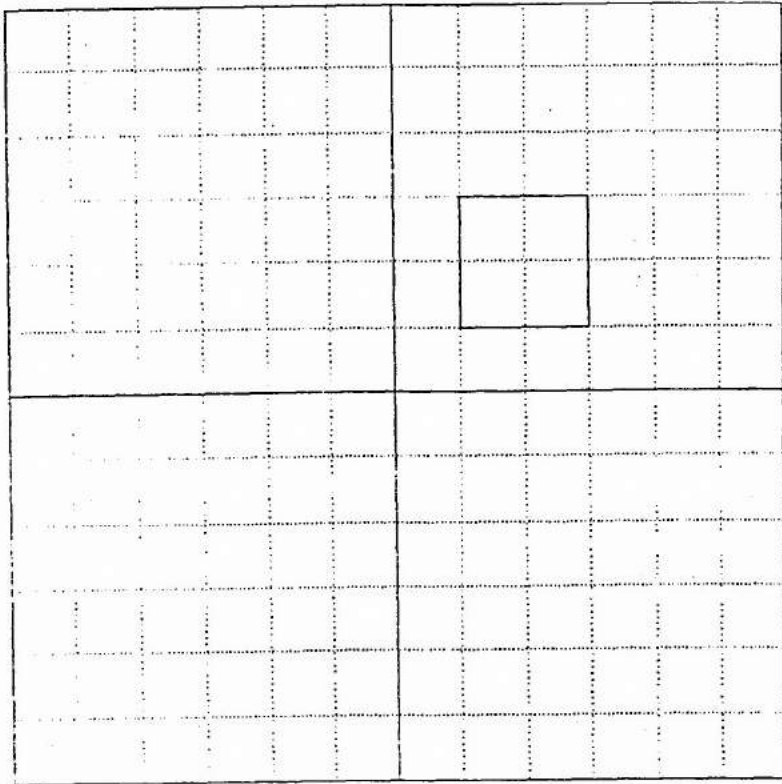


Fig. 14 - O "corpo" da casa traçado usando o preenchimento de área (tipo "HOLLOW").

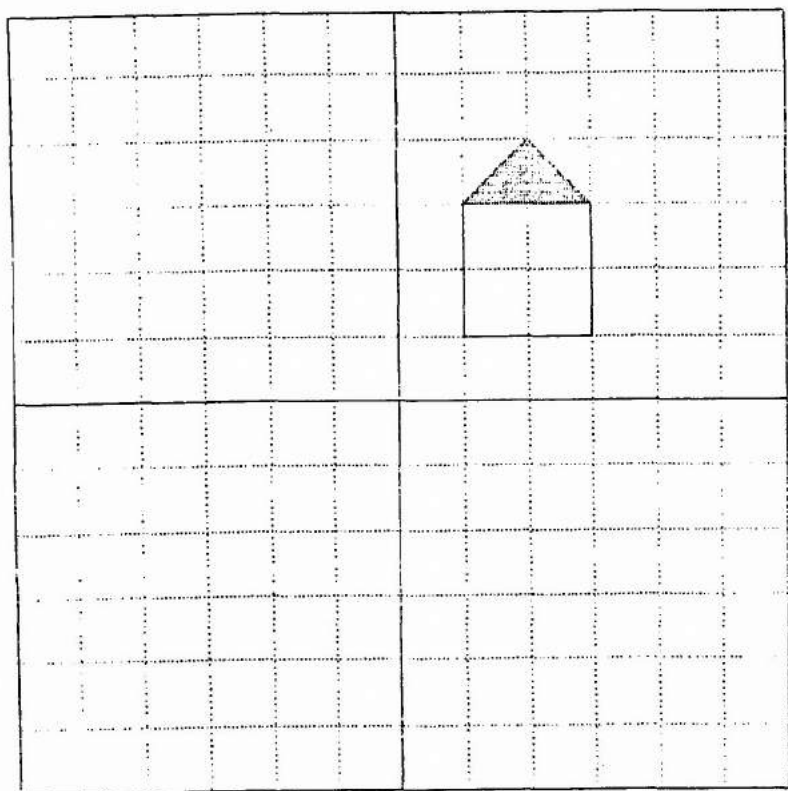


Fig. 15 - O "telhado" da casa traçado usando o preenchimento de área (tipo "SOLID").

polígono (o corpo da casa) é traçado. O valor do atributo "tipo de preenchimento" usado é o valor de "default" ou seja "HOLLOW", isto é, ausência de preenchimento, então somente o contorno de polígono aparece (ver figura 14).

Antes da segunda chamada a GFA para a saída do segundo polígono (o telhado) o valor do "tipo de preenchimento" é trocado por "SOLID", isto é, preenchimento denso. O tom de cinza do preenchimento depende então do valor do atributo "cor". O valor usado aqui é o valor de "default" ou seja 1, isto é, o preto (ver figura 15).

5- ELIMINAÇÃO DAS LINHAS ESCONDIDAS

5.1- VISÃO EM 2D DE POLIEDROS CONVEXOS

Na saída gráfica, um segmento de reta pode ser, por vários motivos, totalmente ou parcialmente visível, ou até mesmo invisível. Nesta seção e na próxima, são examinadas duas situações em que um segmento deve ser cerceado ou até mesmo escondido.

Os poliedros são uma extensão em 3D dos polígonos. Um poliedro é definido a partir de um número finito de pontos chamados de vértices, interligados por segmentos chamados de arestas. De um vértice saem pelo menos três arestas, e pode-se atingir qualquer vértice seguindo arestas consecutivas. Um plano, contendo duas arestas consecutivas, contém um polígono cujos vértices são os vértices do poliedro. Esse polígono forma uma face do poliedro.

Um poliedro é convexo se ele é totalmente contido num dos dois semi-espacos delimitados pelo plano contendo qualquer face. O semi-espaco contendo o poliedro é dito interior e o outro exterior. As faces de um poliedro convexo são convexas.

A casa em 3D da figura 11 é um poliedro convexo.

Na visão em 2D de um poliedro convexo considerado cheio e opaco, suas arestas são visíveis ou não. O método usado em computação gráfica para eliminar as arestas traseiras de um poliedro convexo, e o método de Robert, baseado na propriedade que se pelo menos uma das duas faces formando a aresta é visível, então a aresta é visível, se não a aresta é traseira e pode ser eliminada.

O problema é então reconhecer se uma face é visível ou não. A solução desse problema é dada em seguida. Considerando o poliedro definido no espaço das coordenadas de visualização, referenciado por $S' = \{O', \bar{x}', \bar{y}', \bar{z}'\}$, uma face é visível se uma normal ao plano contendo a face e apontando por dentro do poliedro, aponta no sentido positivo do eixo z' (isto é, aponta no sentido oposto ao observador).

Sob a condição de que o poliedro \bar{e} convexo, e usando as representações do capítulo 4 para malhas de polígonos, é possível garantir que uma normal a qualquer face do poliedro aponta para seu semi-espaço interior.

Nas representações de polígonos no plano do capítulo 4, era feita referência ao sentido de orientação do sistema de coordenadas como regra para ordenar as vértices. No caso do espaço os vértices de uma face são ordenados de maneira que a face seja percorrida no sentido do sistema $\{\vec{u}, \vec{v}\}$ onde \vec{u} e \vec{v} são paralelos ao plano da face e que $\{\vec{u}, \vec{v}, \vec{w}\}$ tenha a orientação do sistema de coordenadas com \vec{w} apontando para o semi-espaço exterior.

Como visto na seção 3.2, o vetor $\overrightarrow{M_1 M_2} \times \overrightarrow{M_2 M_3}$ é um candidato a normal ao plano contendo os pontos M_1 , M_2 e M_3 , e coincide com $\vec{N}(\alpha, \beta, \gamma)$ (α, β, γ dados por (3.2)) e seu sentido é tal que o sistema $\{\vec{N}, \overrightarrow{M_1 M_2}, \overrightarrow{M_2 M_3}\}$ tenha a mesma orientação que o sistema de coordenadas usado para representar os pontos M_1 , M_2 e M_3 .

Supondo que M_1 , M_2 e M_3 sejam três vértices consecutivos qualquer, de uma mesma face de um poliedro convexo na ordem definida acima, então $\vec{N} = \overrightarrow{M_1 M_2} \times \overrightarrow{M_2 M_3}$ é uma normal a face apontando para o semi-espaço exterior.

Assim, $\vec{N}(\alpha, \beta, \gamma)$ no sistema de coordenadas de visualização S' (já suposto com orientação contrária ao do sistema de coordenadas universais no qual o poliedro é definido) aponta para o semi-espaço interior. Finalmente se a coordenada γ é positiva, a face é visível, se não ela é escondida.

Voltando ao exemplo da casa em 3D do capítulo 3, adotando a representação por polígonos definidos como apontadores numa lista de vértices da seção 4.3 (com as adaptações assinaladas acima) (ver "transparência" = 21=), e conservando as mesmas condições de projeção, determina-se a coordenada γ para cada face da casa. Seguindo o algoritmo dado na seção 3.2 (expressão (3.21)), foi criada a função APL N3 para calcu -

a REPRESENTAÇÃO DE MALHAS DE POLIGONOS = 21 =
a POLIGONOS DEFINIDOS COMO AFONDADORES
a NUMA LISTA DE VERTICES
a EXEMPLO 3D

), [1.1]CASA3PLVT

1 2 3 4 5
10 9 8 7 6
6 7 2 1
7 8 3 2
8 9 4 3
9 10 5 4
1 5 10 6

(110), [1.1]CASA3VT

1 0 0 54
2 16 0 54
3 16 10 54
4 8 16 54
5 0 10 54
6 0 0 30
7 16 0 30
8 16 10 30
9 8 16 30
10 0 10 30

lar os parâmetros α , β , γ e δ da equação de um plano contendo os pontos M_1 , M_2 e M_3 . O argumento de N3 é um vetor de três vetores de coordenadas representando os pontos M_1 , M_2 e M_3 . A função produz o vetor $(\alpha, \beta, \gamma, \delta)$. Como mostrado na "transparência" = 23 = o primeiro passo é transformar a lista de vértices usando a matriz de normalização M da "transparência" = 11 =. Em seguida, a "transparência" = 23 = mostra o algoritmo para o cálculo de γ para a face 5 e sua generalização para todas as faces da casa. Esse algoritmo serve para criar na "transparência" = 24 = o vetor APL booleano FV das faces visíveis baseando-se no sinal de cada coordenada γ .

Para determinar as arestas visíveis, usa-se a representação da casa por lados explícitos da seção 4.4 (ver "transparência" =22=). Extraído da lista de lados (aqui as arestas) os polígonos contendo um dado lado e efetuando a operação lógica "ou" (\vee) entre os valores booleanos associados a esses polígonos para cada aresta, cria-se, na "transparência" = 24 = o vetor APL booleano AV das arestas visíveis da casa. Em seguida, as arestas visíveis são traçadas (ver figura 16).

Quando numa visão em 2D de uma cena em 3D tem-se um objeto opaco na frente de um outro, as arestas do objeto de trás podem ser parcialmente visíveis. O cálculo explícito das extremidades, da(s) parte(s) visível(eis) de um segmento é possível, mas é uma operação custosa. Na prática prefere-se traçar primeiro o objeto de trás e em seguida o objeto da frente, uma vez as posições relativas dos objetos em relação ao observador conhecidas.

Na última parte da "transparência" = 24 = esta técnica é ilustrada. Cria-se por translação uma segunda casa que nas condições de projeção encontra-se na frente da primeira. Supondo a casa de trás já desenhada (ver figura 16), preenche-se o interior dos polígonos visíveis da casa da frente usando a função de saída GFA com o valor "SOLID" para o "tipo de preenchimento" e o valor 8 (isto é, branco) para a "cor". Dessa maneira, uma parte da casa de trás é apagada (ver figura 17). Para visualizar a casa da frente basta agora desenhar as suas arestas visíveis (ver figura 18).

A REPRESENTAÇÃO DE MALHAS DE POLIGONOS = 22 =
A LADOS EXPLICITOS
A EXEMPLO 3D

(17),,[1.1]CASA3PLLD

1	1	2	3	4	-5
2	-6	10	-9	-8	-7
3	-1	11	6	-12	
4	-2	12	7	-13	
5	-3	13	8	-14	
6	-4	14	9	-15	
7	5	15	-10	-1	

(115),,[1.1]CASA3LD

1	1	2	1	3
2	2	3	1	4
3	3	4	1	5
4	4	5	1	6
5	1	5	1	7
6	6	7	2	3
7	7	8	2	4
8	8	9	2	5
9	9	10	2	6
10	6	10	2	7
11	1	6	3	7
12	2	7	3	4
13	3	8	4	5
14	4	9	5	6
15	5	10	6	7

A CASA3VT

FACES ESCONDIDAS = 23 =

CASA3VTM+HC(CH CASA3VT)+.X"CM

5=CASA3PLVT

B 9 4 3

3+5=CASA3PLVT

B 9 4

CASA3VTM PLVT 3+5=CASA3PLVT

0.7828427125 0.6632993162 0.3623513771
0.6885618083 0.7993820797 0.3905148049
0.4057190958 0.6360827635 0.05255367167

[1.1]N3 CASA3VTM PLVT 3+5=CASA3PLVT

-0.04139161646 # A
-0.03982910151 # B
0.05388602512 # C
0.03929606569 # D

3=N3 CASA3VTM PLVT 3+5=CASA3PLVT

0.05388602512 # C

[1.1]3=N3(CASA3VTM)PLVT"3+CASA3PLVT

0.02566001196
-0.01539600718
-0.06158402871
0.03849001795
0.05388602512
0.007698003589
-0.03849001795

n DESENHO EM 3D = 24 =
n ARESTAS ESCONDIDAS

FV←0(3>"N3"(←CASA3VTH)PLVT"3↑"CASA3PLVT

FV

1 0 0 1 1 1 0 n FACES VISIVEIS (1)

AV←v/"(2↑"CASA3LD)▷""←c←FV

AV

1 1 1 1 1 0 1 1 1 0 0 1 1 1 1 n ARESTAS

pAV

15

pAV/CASA3LD

12 n SD' 12 ARESTAS VISIVEIS

GPL"CV"C2"(←CASA3VTH)LDVT"AV/CASA3LD (fig. 16)

MT←(T3 18 0 12)+.xM

C←HC(CH CASA3VT)+.x"←c←MT n OUTRA CASA

GSFACI 8

GSFAIS 'SOLID'

GFA"CV"C2"(←C)PLVT"AV/CASA3PLVT (figura 17)

GPL"CV"C2"(←C)LDVT"AV/CASA3LD (figura 18)

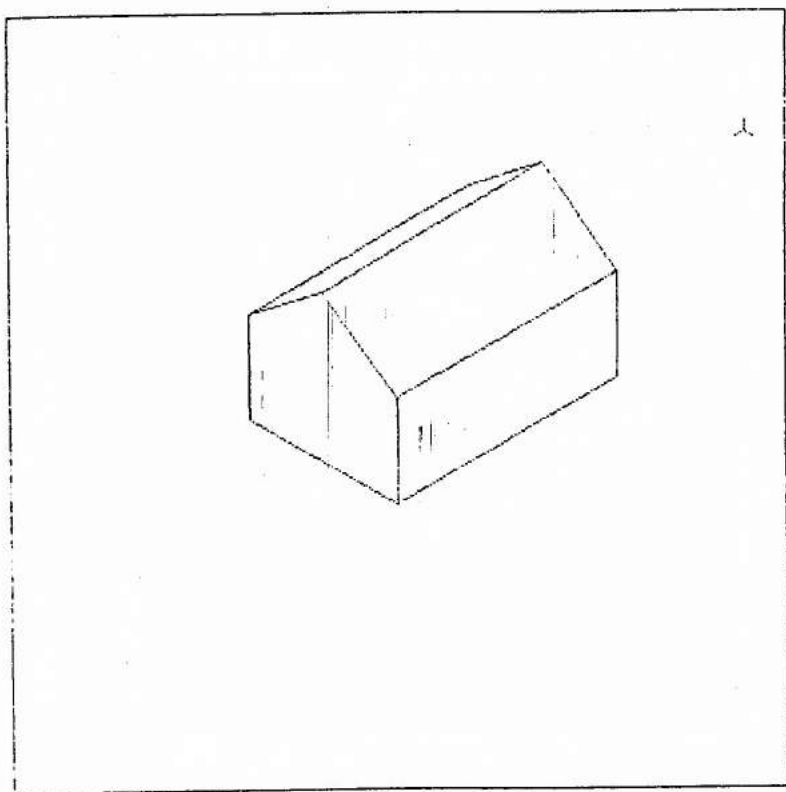


Fig. 16 - A casa com a eliminação das arestas traseiras.

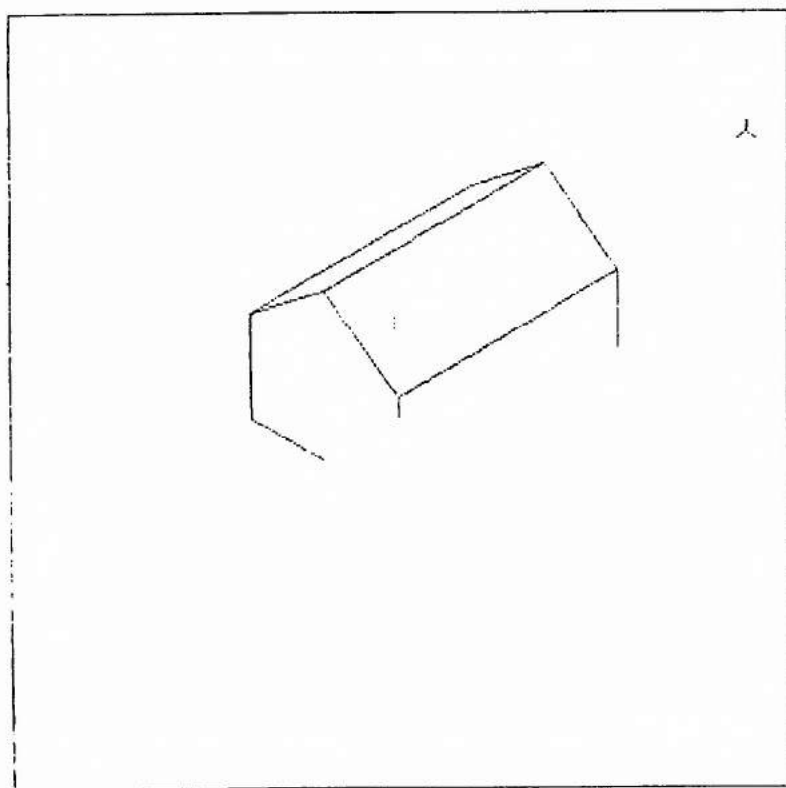


Fig. 17 - A casa (de trás) com eliminação parcial ou total das arestas escondidas pela casa da frente (cujas arestas não foram ainda traçadas).

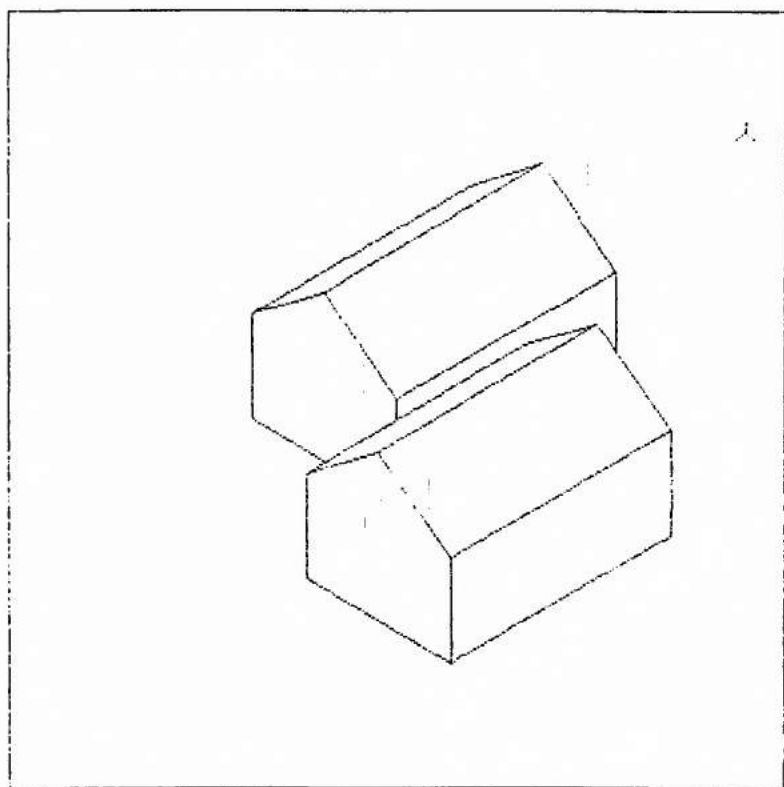


Fig. 18 - As casas de trás e da frente com a eliminação das arestas escondidas).

5.2- CERCEAMENTO DE SEGMENTOS

Na visão da cena em 20 ou 30, somente a parte da cena interior à janela ou ao volume de visualização é visível. O processo que consiste em eliminar as partes invisíveis e conservar as visíveis chama-se de cerceamento ("clipping"). Para evidenciar o efeito do cerceamento é dado em seguida um exemplo.

Na "transparência" = 25 = um campo de visualização menor que o quadrado unitário é definido e identificado (com o número 4) usando o comando GSVW. O comando GSELNT seleciona efetivamente a transformação de normalização correspondente. O resultado da saída gráfica da casa em 30 aparece na figura 19. Como a casa é totalmente contida no volume de visualização, nenhuma parte dela precisa ser cerceada. Agora escolheu-se um volume de visualização menor que a casa, essa vai aparecer maior dentro do campo de visualização e até passar dos limites desse campo. Isto é feito na "transparência" = 25 = redefinindo os vetores APL PMIN e PMAX. O resultado é mostrado na figura 20, onde pode ser visto o efeito do cerceamento que eliminou as partes invisíveis da casa. Sem o cerceamento, isto é, desativando o processo de cerceamento com o comando GSCLIN, o resultado é mostrado na figura 21.

No processo de cerceamento acima, usou-se as capacidades de cerceamento em 20 implementadas debaixo do GKS. Essa capacidade era suficiente já que a casa continuou dentro dos limites dos dois planos de cerceamento dianteiro e traseiro (cf. seção 3.3). Se não tivesse sido o caso, teria sido necessário executar o cerceamento explicitamente (isto é, acima do GKS). Geralmente o cerceamento é efetuado no espaço das coordenadas de visualização, porque neste a representação do volume de visualização é mais simples. Esse volume é o cubo unitário delimitado pelos planos $x' = 0$, $x' = 1$, $y' = 0$, $y' = 1$, $z' = 0$ e $z' = 1$ no caso de uma projeção paralela.

Em seguida é estudado o problema do cerceamento em 20. O cerceamento será efetuado no espaço das coordenadas universais e com uma janela com seus lados paralelos aos eixos (esta situação é similar àque-

* JANELA E CAMPO DE VISUALIZAÇÃO = 25 *
* CERBEAMENTO (CLIPPING)

GSVW 4 .2 .8 .2 .8

GSELNT 4

GPL"CV"C2"(cCASA3VTM)LDVT"CASA3LD (fig. 19)

PMIN=-14 -14 -1

PMAX=14 14 40

MM=NPAR VRP VFN VUF DOP,PMIN,PMAX

CASA3VTMM=HC(CH CASA3VT)+.x"eMM

GCLRWK 1

GPL"CV"C2"(cCASA3VTMM)LDVT"CASA3LD (fig. 20)

GCLRWK 1

GSCLIN 'NOCLIP'

GPL"CV"C2"(cCASA3VTMM)LDVT"CASA3LD (fig. 21)

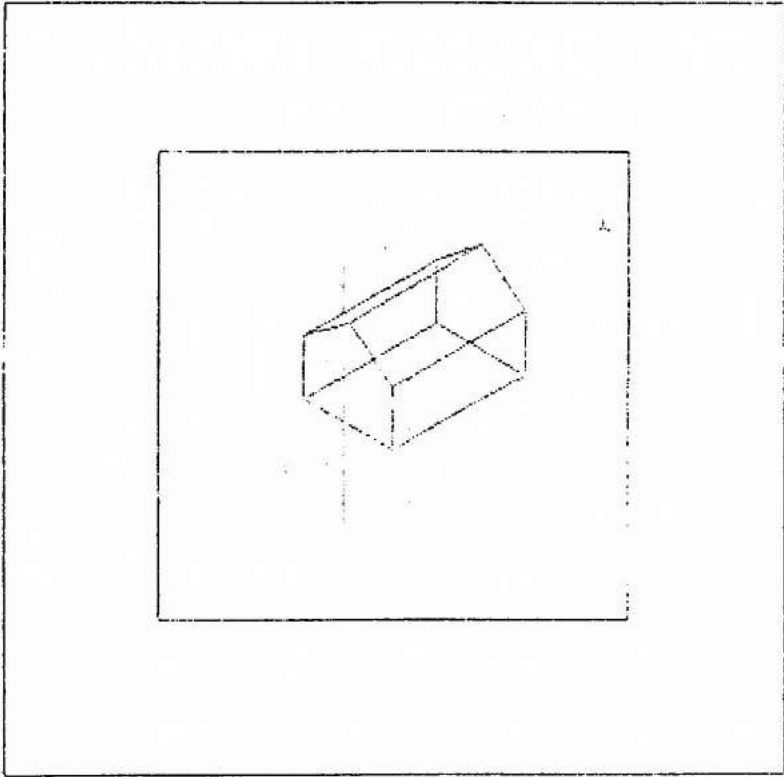


Fig. 19 - A casa num campo de visualização menor que o quadrado unitário.

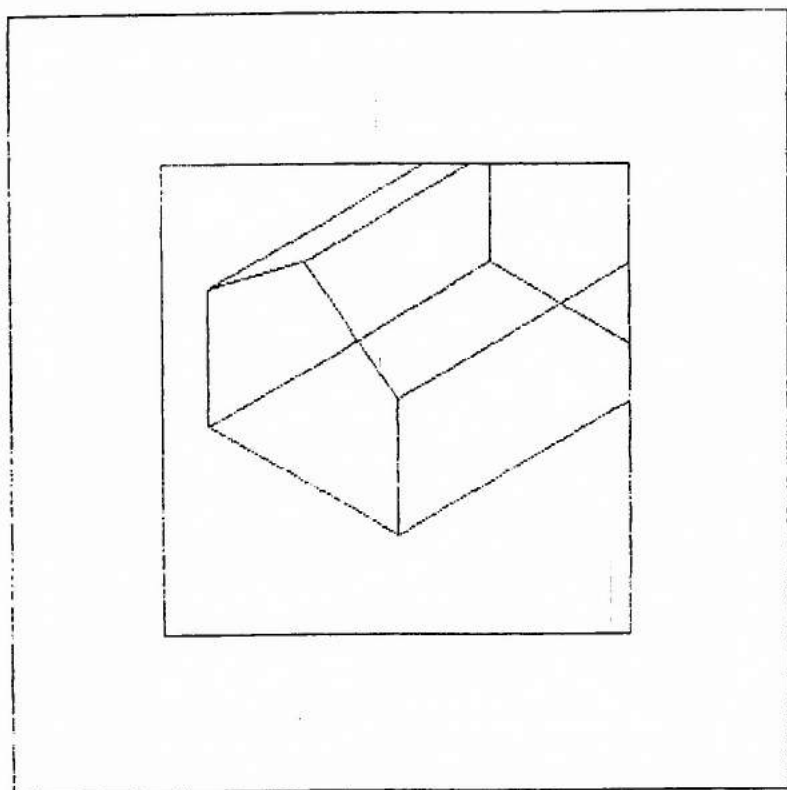


Fig. 20 - A casa dentro de um volume de visualização menor do que ela (com a função "cerceamento" ativada).

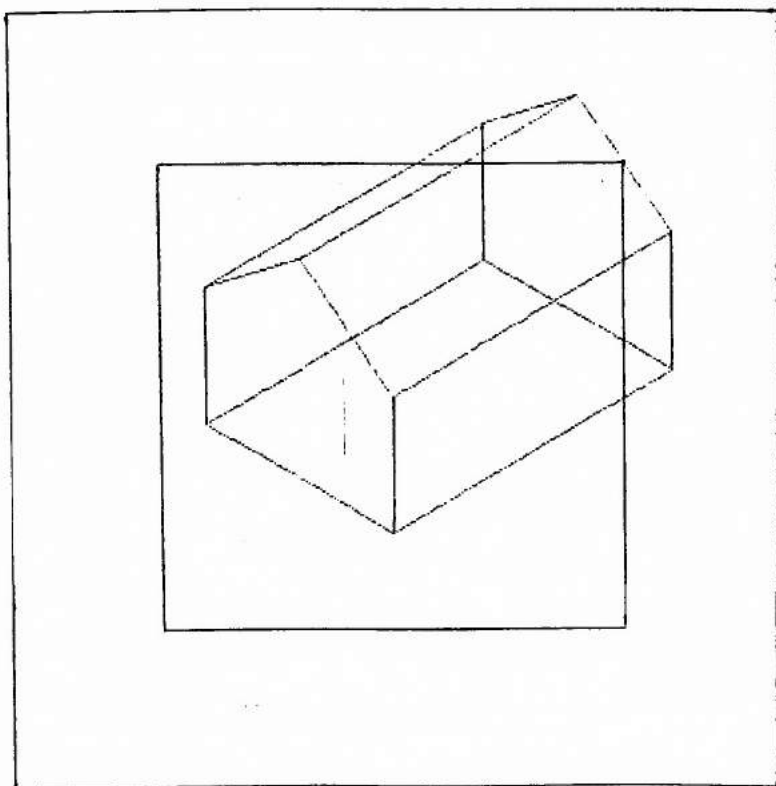


Fig. 21 - A casa dentro de um volume de visualização menor do que ela (com a função "cerceamento" desativada).

la de um cerceamento em 2D no espaço de visualização).

Na "transparência" = 26 = são definidos uma janela e oito segmentos visualizados na figura 22. Os segmentos são armazenados no vetor APL R. Depois de executar o processo de cerceamento obtêm-se um vetor APL A contendo seis segmentos visualizados na figura 23. Desses seis um sô não mudou porque estava totalmente contido na janela. Os outros ficaram mais curtos, tendo pelo menos uma nova extremidade situada no contorno da janela. Observa-se que o problema do cerceamento é a determinação dessas novas extremidades.

O modelo paramétrico de uma reta contendo os pontos $M_0(x_0, y_0)$ e $M_1(x_1, y_1)$ (cf. seção 3.2) é:

$$\begin{aligned} x &= (x_1 - x_0) t + x_0 \\ y &= (y_1 - y_0) t + y_0, \text{ com } t \in \mathbb{R}. \end{aligned}$$

Acrescentando a condição $t \in [0, 1]$ o modelo acima passa a ser o modelo do segmento $[M_0 M_1]$.

Supõe-se que $[M_0 M_1]$ não é totalmente contido na janela. Denota-se por (x_0^s, y_0^s) e (x_1^s, y_1^s) as extremidades do segmento $[M_0 M_1]$ a ser cerceado, e (x_0^j, y_0^j) e (x_1^j, y_1^j) as extremidades de um lado da janela, então tem-se:

$$\begin{bmatrix} 1 & 0 & x_0^s - x_1^s & 0 \\ 0 & 1 & y_0^s - y_1^s & 0 \\ 1 & 0 & 0 & x_0^j - x_1^j \\ 0 & 1 & 0 & y_0^j - y_1^j \end{bmatrix} \begin{bmatrix} x \\ y \\ t^s \\ t^j \end{bmatrix} = \begin{bmatrix} x_0^s \\ y_0^s \\ x_0^j \\ y_0^j \end{bmatrix}.$$

Se a matriz 4×4 acima não é singular (isto é, o segmento e o lado da janela não são paralelos), e se a resolução do sistema acima der t^s e $t^j \in [0, 1]$, então (x, y) é uma nova extremidade. Repetindo o cálculo acima para os outros lados da janela, descobre-se as possíveis novas extremidades. Se o segmento não tem novas extremidades, ele

R CERCEAMENTO (CLIPPING) = 26 =
R ALGORITMO DE COHEN-SUTHERLAND

JANELA
3 -3 3 -3 R YMAX YMIN XMAX XMIN

▷R
-6 2 2 5
-4 0 4 5
-5 -2 -1 1
-5 -4 5 5
-1 -2 1 0
1 -1 4 3
1 -5 5 4
4 -4 5 2

GPL"CV"R (produz figura 22)

▷R
8
▷A←JANELA CERCEAMENTO R

6

GCLRWK 1
GPL"CV"A (produz figura 23)

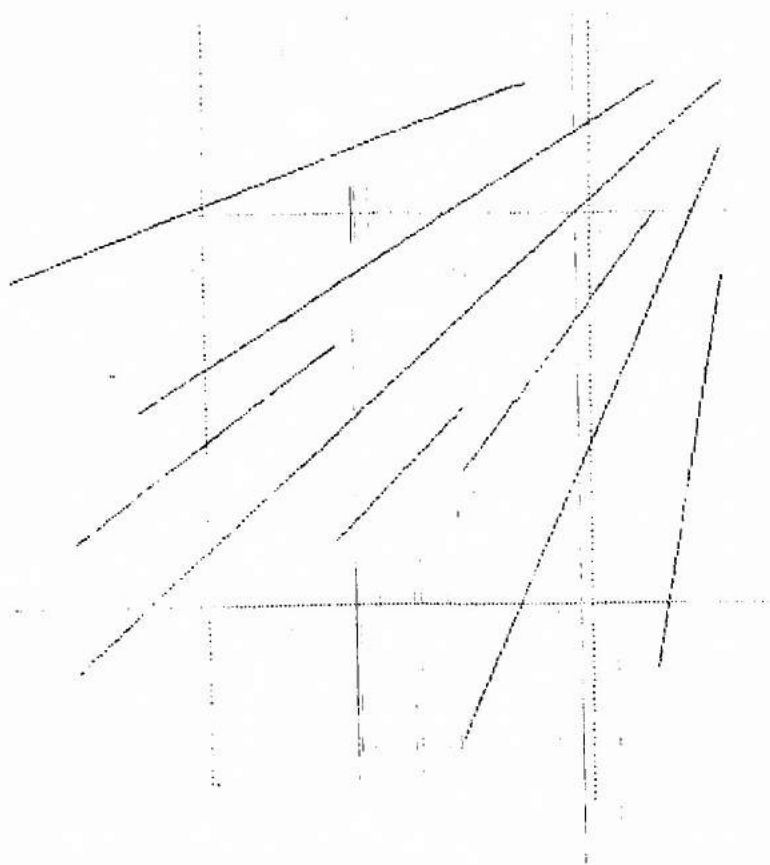


Fig. 22 - A janela (delimitando 9 regiões) e os 8 segmentos (sem cerceamento).

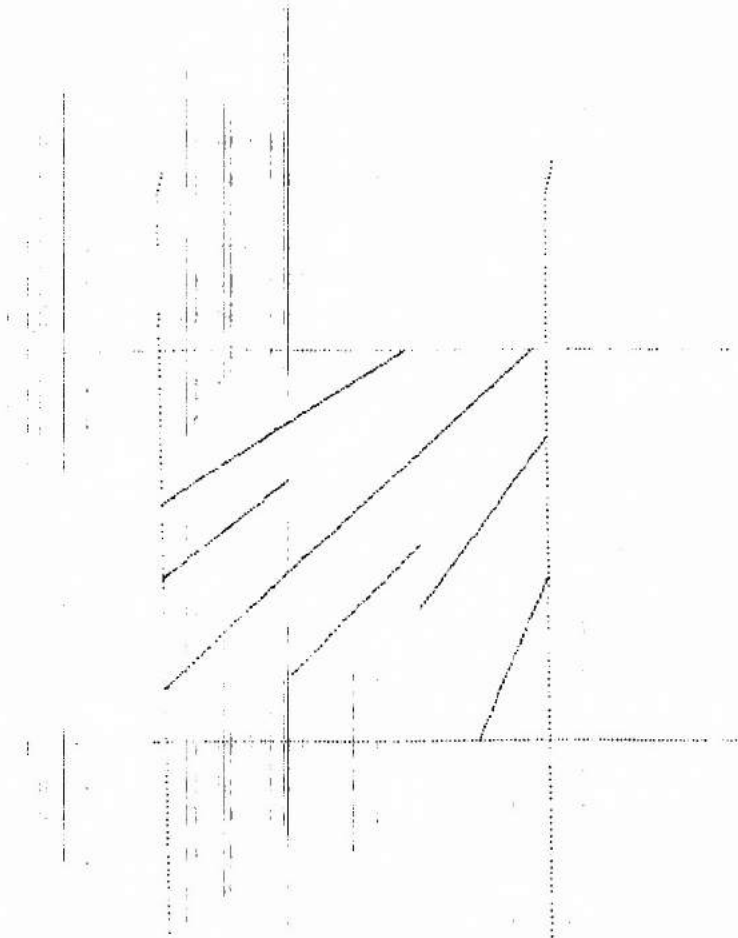


Fig. 23 - A janela e os 8 segmentos cerceados.

é rejeitado. Se ele tem só uma nova extremidade rejeita-se sua extremidade original não contida na janela.

Esse algoritmo é teoricamente perfeito, mas ele não é eficiente. Para ser mais eficiente, um algoritmo de cerceamento deve evitar, se possível, o cálculo das interseções que não caíam nos lados da janela. Deve-se também aproveitar o fato que os lados da janela são paralelos aos eixos do sistema de coordenadas para simplificar o cálculo das interseções.

Em seguida, uma implementação do algoritmo de Cohen-Sutherland é descrito. Esse algoritmo é geralmente muito mais eficiente que o algoritmo descrito acima.

Para sua implementação em APL, as funções RJ, AC e DV foram criadas (cf. anexo).

A função RJ tem como argumento o vetor APL contendo os segmentos (na forma de um vetor contendo suas extremidades (na forma de um vetor de coordenadas)), ela produz um vetor APL similar sem os segmentos totalmente acima, abaixo, à direita ou à esquerda da janela.

A função AC, tem como argumento também um vetor de segmentos. Ela produz um vetor reduzido apenas aos segmentos totalmente dentro da janela (lados incluídos).

A função DV tem como argumento de esquerda um inteiro, valendo 1, 2, 3 ou 4, e de direita um vetor de segmentos. Ela produz um vetor de segmentos cerceados ao longo da reta contendo o lado superior (resp. inferior, direito, esquerdo) da janela quando o argumento de esquerda for 1 (resp. 2, 3, 4).

Essas funções usam extensivamente (RJ e AC usando exclusivamente) a informação sobre qual das nove regiões do plano, delimitadas pelas quatro retas contendo os lados da janela, pertence uma determinada extremidade. Essa informação é fácil de se obter, e é codificada (fun

ção CODE) sobre quatro bits. O 1º (resp. 2º, 3º, 4º) bit vale 1 se o ponto é na região acima (resp. abaixo, direita, esquerda) da janela.

! Somente a função DV pode eventualmente precisar do cálculo de interseção que é simples na medida que uma das suas coordenadas já é conhecida.

O algoritmo de Cohen-Sutherland é implementado como mostrado nas "transparências" = 27 = e = 28 =. Cada passo do cerceamento dos segmentos no vetor R (ver "transparência" = 26 =) é visualizado nas figuras 24 - 27. A visualização é feita em quatro quadrantes (campos de visualização). O quadrante acima à esquerda (resp. acima à direita, abaixo à esquerda, abaixo à direita) é usado selecionando com a função GSELNT a transformação de normalização 11 (resp. 12, 13, 14).

O algoritmo pode se dividir em quatro etapas similares. As primeiras três etapas começam com uma chamada a função RJ, para rejeitar os segmentos totalmente acima, abaixo, à direita ou à esquerda da janela. Em seguida nas quatro etapas tem-se uma chamada a função AC para selecionar os segmentos que podem ser aceitos, esses são acumulados no vetor APL Z. Finalmente, em seguida, nas quatro etapas tem-se uma chamada a função DV, com o argumento direito igual a 1 (resp. 2, 3, 4), para o cerceamento ao longo da reta contendo o lado superior (resp. inferior, direito, esquerdo) da janela, na etapa 1 (resp. 2, 3, 4). Os últimos segmentos cerceados na etapa 4 são também acumulados no vetor APL Z que contém assim o resultado do cerceamento dos segmentos inicialmente em R.

No caso de um cerceamento em 3D relativamente a um cubo de arestas paralelas aos eixos, o algoritmo de Cohen-Sutherland estende-se sem dificuldades. As nove regiões passam a ser quinze e codificadas em seis bits em vez de quatro, e as quatro etapas passam a ser seis, uma etapa para cada face do cubo.

• CERCEAMENTO (CLIPPING) = 27 =
• ALGORITMO DE COHEN-SUTHERLAND (DETALHE)

• ETAPA 1

GSELNT 11

GPL"CV" A+R J R

GSELNT 12

GPL"CV" Z+AC A

GSELNT 13

GPL"CV" A+1 DV A~Z

(produz figura 24)

• ETAPA 2

GSELNT 11

GPL"CV" A+R J A

GSELNT 12

GPL"CV" Z+Z, AC A

GSELNT 13

GPL"CV" A+2 DV A~Z

(produz figura 25)

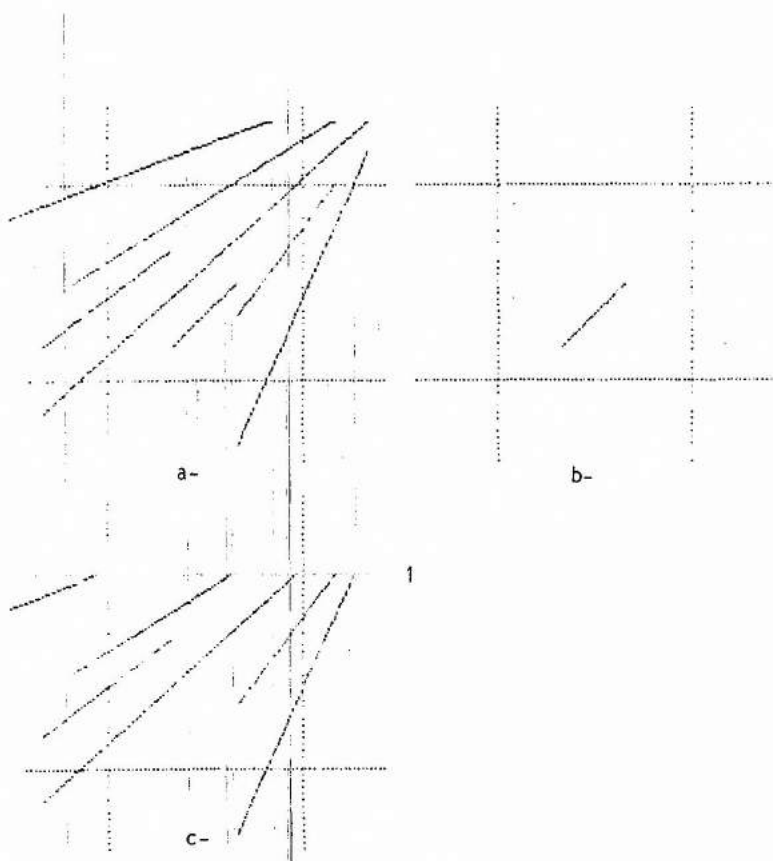


Fig. 24 - A etapa 1 do cerceamento:
a- rejeição (um segmento rejeitado)
b- aceitação (um segmento aceito)
c- cerceamento relativo ao lado superior
(5 interseções calculadas).

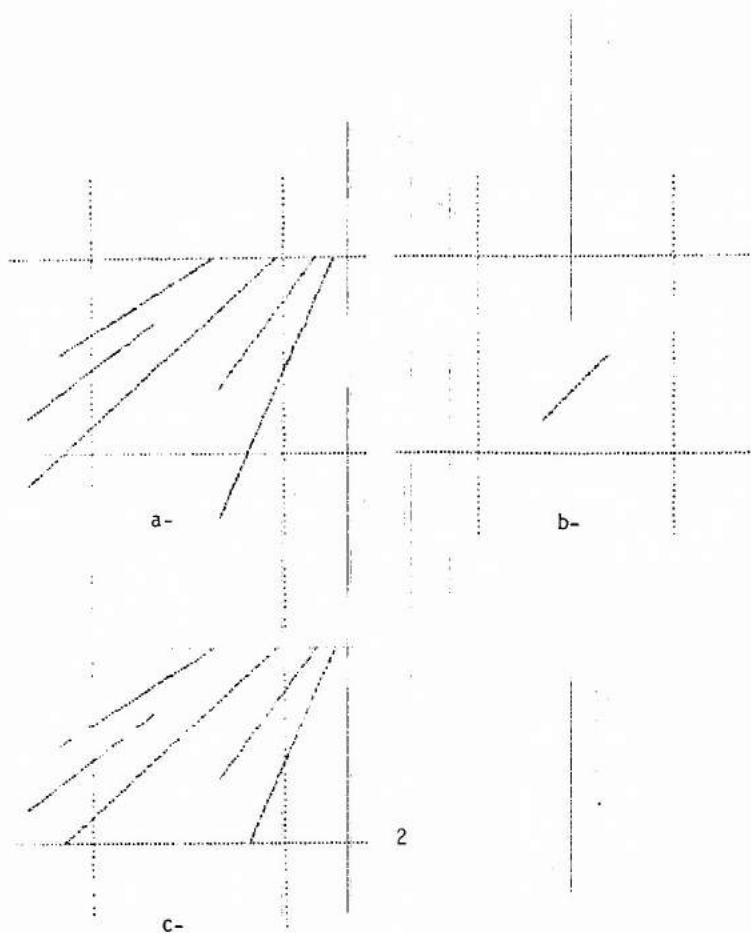


Fig. 25 - A etapa 2 do cerceamento:
a- rejeição (um segmento rejeitado)
b- aceitação (zero novo segmento aceito)
c- cerceamento relativo ao lado inferior
(2 interseções calculadas)

CERCEAMENTO (CLIPPING) = 28 =
ALGORITMO DE COHEN-SUTHERLAND (DETALHE)

ETAPA 3

GSELNT 11

GPL"CV"A+RJ A

GSELNT 12

GPL"CV"Z+Z,AC A

GSELNT 13

GPL"CV"A+3 DV A~Z

(produz figura 26)

ETAPA 4

GSELNT 12

GPL"CV"Z+Z,AC A

GSELNT 13

GPL"CV"A+4 DV A~Z

GSELNT 14

GPL"CV"Z+Z,A

(produz figura 27)

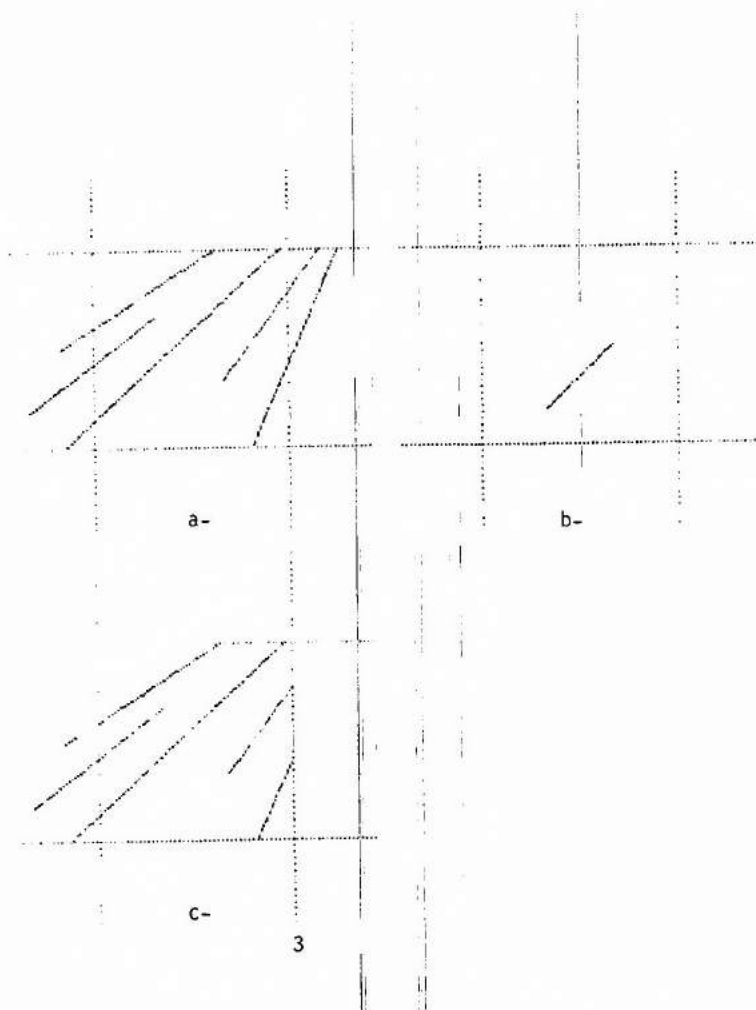


Fig. 26 - A etapa 3 do cerceamento:
a- rejeição (zero segmento rejeitado)
b- aceitação (zero novo segmento aceito)
c- cerceamento relativo ao lado direito
(2 interseções calculadas)

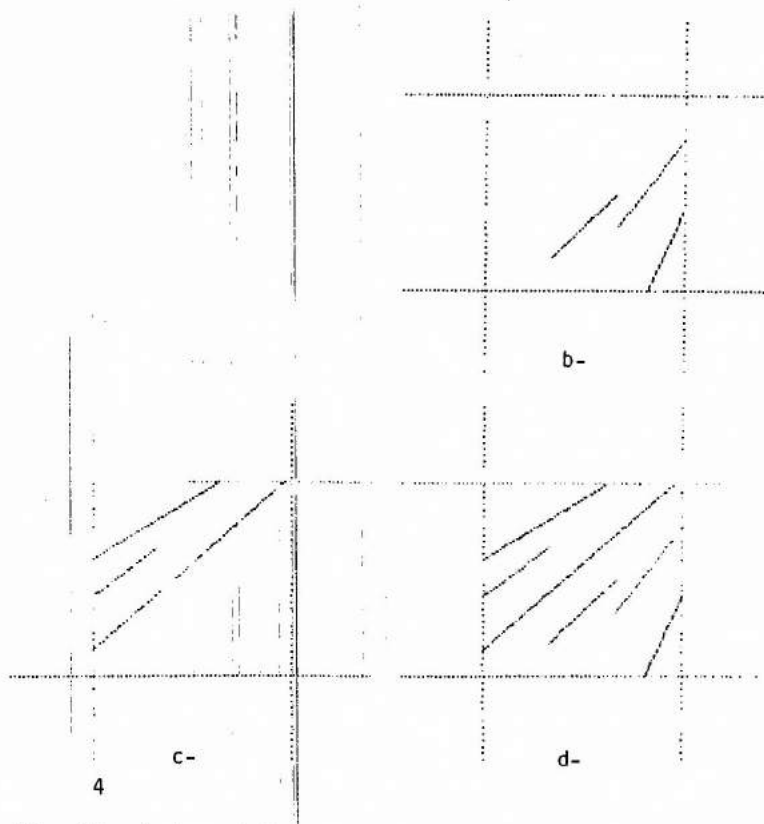


Fig. 27 - A etapa 4 do cerceamento:
b- aceitação (dois novos segmentos aceitos)
c- cerceamento relativo ao lado esquerdo
(3 interseções calculadas)
d- concatenação (b,c) (resultado final do cerceamento)

6 - ELIMINAÇÃO DOS POLÍGONOS ESCONDIDOS

6.1 - CONVERSÃO VETORIAL/RASTER

No capítulo anterior foram apresentados alguns aspectos do problema das linhas escondidas. Em particular, na seção 5.1, a respeito do método de Robert, foi visto que a eliminação das arestas escondidas de um poliedro passava pela determinação das fases escondidas. A técnica em pregada nessa determinação era baseada explicitamente no modelo do poliedro, ou seja, a técnica desenvolveu-se no espaço do objeto. Nesse capítulo são apresentadas outras técnicas mais gerais e desenvolvendo-se principalmente no espaço da imagem, i.é., o conjunto dos pontos formando a imagem digital.

Seja $S = \{0, \vec{x}, \vec{y}\}$ o sistema de coordenadas no plano da imagem digital (construído acima de uma grade quadrada) tal que o eixo x especificasse o número da coluna do ponto da imagem, y sua linha, e o ponto $(1,1)$ seja o primeiro ponto abaixo à esquerda. Este sistema, chama-se de sistema de coordenadas do dispositivo. Dependendo do contexto pode-se acrescentar em S uma terceira dimensão expressa por meio de um eixo z tendo exatamente o mesmo papel que o eixo z' do sistema de coordenadas de visualização.

Para cada um dos pontos das imagens (i.é., pontos de grade), procura-se qual é o polígono visível numa dada projecção. Apesar de uma imagem poder ter até quatro milhões de pontos (grade máxima de um monitor a varredura), essa abordagem pode ser bastante eficiente.

Porque o problema da passagem do espaço do objeto para o espaço da imagem é central no assunto dos polígonos escondidos, começa-se aqui com a apresentação de uma técnica de conversão vetorial/raster que serve para o preenchimento de polígonos.

Para converter a representação de um polígono na forma de um lista de lados para a representação na forma de um conjunto de pontos interiores (repartidas acima de uma grade quadra), usa-se o algoritmo da

linha exploratória ("scan-line").

No algoritmo da linha exploratória o polígono já é suposto cerceado [Sutherland, 1974a],[Weiler, 1977], e expresso no sistema de coordenada do dispositivo. Procura-se então os pontos interiores linha por linha.

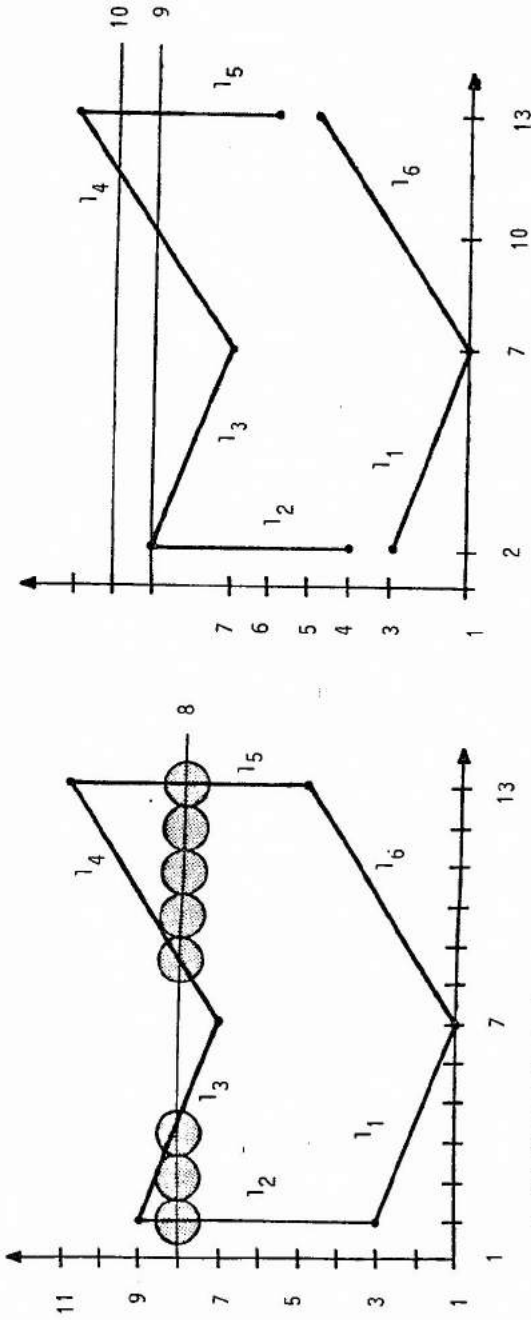
Na seção 4.5 foi introduzido um algoritmo para determinar se um ponto é interior a um polígono convexo. Mesmo se um tal algoritmo não precisa ser precedido pelo processo de cerceamento, sua eficiência é muito baixa para ser usado aqui.

Usando a idéia de coerência espacial, i.é. notando-se que se um ponto é interior a um polígono, todos seus vizinhos, ao longo de uma reta (passando por esse ponto), são também interiores até o momento de encontrar os lados, pode-se melhorar muito a eficiência de um algoritmo para achar pontos interiores. O que se precisa é reconhecer as interseções da reta com os lados. Para simplificar a busca das interseções usam-se retas passando pelas linhas da imagem. Essas retas são chamadas de linhas exploratórias (suas equações são $y = \text{número da linha da imagem}$).

Observa-se que nem precisa conhecer um ponto interior, pois conhecendo a "primeira" interseção da linha exploratória com os lados do polígono quando percorre-se a linha de ponta a ponta, todos os pontos na linha a partir daí são interiores até encontrar a segunda interseção. Assim os pontos interiores ao longo de uma linha exploratória são simplesmente os pontos situados entre pares de interseções ordenadas em x crescente (ver figura 6.1a).

Repetindo o algoritmo para todas as linhas exploratórias que interceptam o polígono, tem-se todos seus pontos interiores. Preencher o polígono significa associar um dado tom de cinza ou cor, para todos seus pontos interiores, usa-se a expressão de preencher seus pontos interiores.

Assim, para cada linha exploratória que intercepta o polí



a- Polígono original
(preenchimento ao longo da linha
exploratória 8).

b- Polígono encurtado.

Figura 6.1 Conversão vetorial/raster (preenchimento de polígono).

gono, o algoritmo consiste em:

- Calcular as interseções da linha exploratória com os lados;
- Classificar as interseções por ordem de x crescente;
- Preencher os pontos da linha entre pares de interseções.

Um problema pode aparecer se a linha exploratória intercepta um vértice do polígono. Neste caso, pelo algoritmo tem-se duas interseções confundidas. Isto é correto se o vértice é um extremo local (como no caso dos vértices (7,1), (2,9), (7,7) e (13,11) do polígono da figura 6.1a), mas é errado se o vértice não é um extremo local (como no caso dos vértices (2,3) e (13,5)) pois neste caso o número certo é de uma interseção entre a linha exploratória e o polígono. Observa-se que um vértice não é extremo local se as duas outras extremidades dos lados incidentes não são do mesmo lado da linha exploratória.

Uma solução para este problema é identificar os vértices que são contidos numa linha exploratória e que não são extremos locais e em seguida, para cada um desses vértices encurtar um dos dois lados incidentes de maneira que a sua nova extremidade termine na linha exploratória vizinha (ver figura 6.1b).

Finalmente, observa-se que os lados paralelos às linhas exploratórias não precisam ser consideradas no algoritmo.

O algoritmo é organizado de maneira a evitar o cálculo de interseções inúteis entre a linha exploratória e as retas contendo os lados do polígono. O algoritmo usa outra vez a idéia de coerências espacial, aplicando-a ao cálculo das interseções de uma linha para a outra. Se x_i e x_{i+1} são as abscissas das interseções do lado com uma linha exploratória e sua vizinha imediatamente superior tem-se

$$x_{i+1} = x_i + 1/m,$$

onde m é a inclinação do lado. Em termos dos dois vetores de coordenadas (x_0, y_0) e (x_1, y_1) das extremidades do lado, m é dado por $(y_1 - y_0)/(x_1 - x_0)$, i.é. $1/m = (x_1 - x_0)/(y_1 - y_0)$.

Para o funcionamento do algoritmo da linha exploratória cria-se uma lista de lado ET e uma lista de lados ativos AET. Para representar o polígono não se usa aqui a representação por lados explícitos da seção 4.4, mas uma estrutura de dados diferentes e específicos ao algoritmo.

A lista de lados ET pode ser vista como um vetor APL cujo número de elementos é igual ao número de linhas na imagem. Seus elementos são vetores vazios ou contendo vetores descritivos dos lados do polígono com a seguinte forma:

$$(y_{\max}, x, 1/m)$$

onde y_{\max} é a maior ordenada das extremidades do lado, e x é a abscissa da extremidade do lado com menor ordenada.

A posição de cada um desses vetores na lista é dada por y_{\min} , i.é. a menor ordenada das extremidades do lado (ver figura 6.2, l_1 e l_6 tem $y_{\min} = 1$, l_2 tem $y_{\min} = 4...$).

A lista dos lados ativos AET é um vetor APL contendo os vetores descritivos dos lados que tem uma interseção com a atual linha exploratória. Nesses vetores descritivos, x representa a abscissa da interseção do lado com a atual linha exploratória (ver figura 6.3).

O detalhe do algoritmo da linha exploratória é dado em seguida.

- Criar a lista de lado ET.
- Dar a y o valor da posição do primeiro vetor de ET não vazio.
- Repetir até que ET e AET estejam vazios:
 1. Transferir os vetores descritivos na posição y em ET para AET.
 2. Classificar em AET os vetores descritivos por ordem de x crescente.
 3. Preencher os pontos da linha y da imagem entre os pares de abscissas dados em AET.

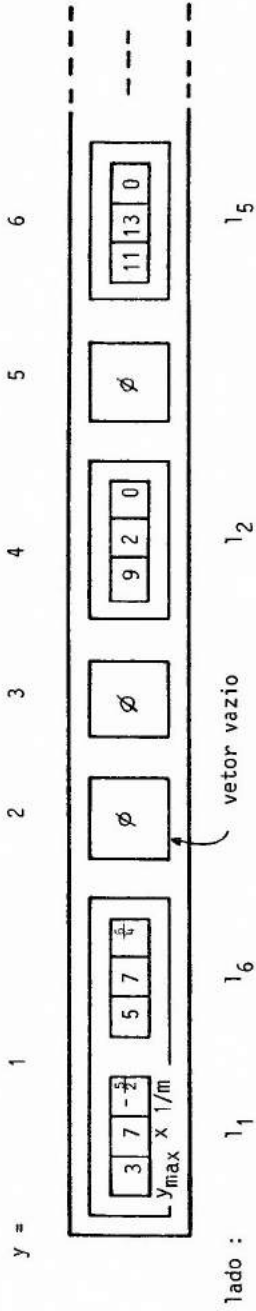


Figura 6.2 Lista de lados ET (para o polígono da figura 6.1b).

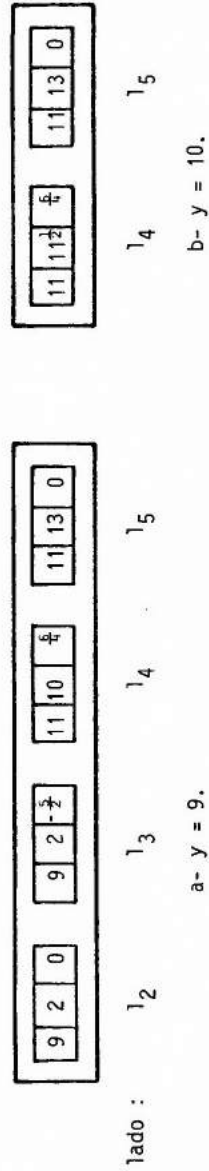


Figura 6.3 Lista de lados ativos AET para dois valores de y depois do passo 2 (usando os dados em ET da figura 6.2).

4. Tirar de AET os vetores descritivos para os quais $y = y_{\max}$.
5. Fazer $x = x + 1/m$ para os vetores descritivos restantes.
6. Incrementar y de 1.

6.2 - TESTE DAS EXTENSÕES

No caso das projeções paralelas (cf. seção 3.3), a eliminação de superfícies escondidas é geralmente feita diretamente no espaço das coordenadas do dispositivo considerando os valores relativos z , i.é., a profundidade dos pontos dessas superfícies que se projetam no mesmo ponto da imagem. No entanto é interessante, para simplificar, detectar as situações nas quais se pode evitar a comparação das profundidades.

No exemplo da seção 5.1, a cena 3D era composta de duas casas. Na projeção considerada uma casa escondia parcialmente a outra e por isso foi usado um algoritmo para a eliminação das porções de arestas escondidas. Numa outra projeção essas duas casas poderiam muito bem não se superpor. Neste caso, aplicar o algoritmo é perda de tempo.

Um método simples para detectar se dois objetos são disjuntos é o teste das extensões. Um exemplo de extensão da projeção da casa em 3D da seção 3.3. é dado na figura 28. A extensão de um objeto é um retângulo (em 2D) ou um paralelepípedo (em 3D) cujos lados ou arestas são paralelos aos eixos do sistema de coordenadas e cujas extremidades da sua diagonal principal coincidem com os pontos de menor e maior vetor de coordenadas que podem ser construídas a partir dos vetores de coordenadas dos pontos do objeto.

A "transparência" = 29 = mostra como calcular a extensão da projeção da casa. Essa extensão é dada na forma do vetor APL EXT contendo os quatro vetores de coordenadas de seus vértices consecutivos.

No teste das extensões, se duas extensões são disjuntas então os dois objetos são disjuntos. Evidentemente, se as duas extensões interceptam-se, nada se pode concluir sobre os objetos.

EXTENSÃO DE UM OBJETO (EXTENT) = 29 =

>C2 CASA3VTM

0.3114381917	0.4727834473
0.5	0.3639172365
0.5	0.5
0.4057190958	0.6360827635
0.3114381917	0.6088662108
0.5942809042	0.6360827635
0.7828427125	0.5272165527
0.7828427125	0.6632993162
0.6885618083	0.7993820797
0.5942809042	0.772165527

□←MIN←↑L/C2 CASA3VTM

0.3114381917	0.3639172365
--------------	--------------

□←MAX←↑Γ/C2 CASA3VTM

0.7828427125	0.7993820797
--------------	--------------

EXT←,0 1φ(MIN[1],MAX[1])°. ,MIN[2],MAX[2]

>EXT

0.3114381917	0.3639172365
0.3114381917	0.7993820797
0.7828427125	0.7993820797
0.7828427125	0.3639172365

GPL"CV"C2"(←CASA3VTM)LDVT"CASA3LD

GFA CV EXT

(produz figura 28

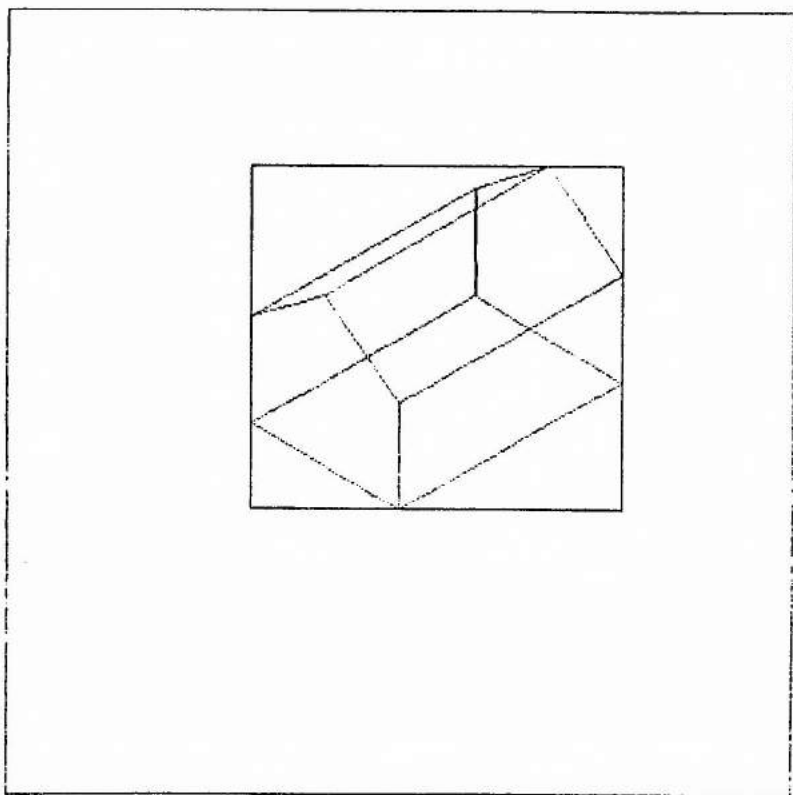


Fig. 28 - A extensão da projeção da casa.

6.3 - ALGORITMO DA ORDENAÇÃO POR PROFUNDIDADE

A idéia empregada na visualização das duas casas da seção 5.1 é de traçar primeiro as arestas da casa de trás e em seguida as da casa da frente. Isto é a idéia do algoritmo da ordenação por profundidade desenvolvido por Newell, Newell e Sancho [Newell, 1972]. Este algoritmo atua tanto no espaço do objeto quanto no espaço da imagem.

As três etapas do algoritmo são dadas a seguir (considerando que os polígonos são expressos no sistema de coordenadas do dispositivo).

- Ordenar os polígonos em função das suas maiores coordenadas z (z_{\max}).
- Resolver as ambiguidades que podem surgir quando há superposição das extensões ao longo do eixo z .
- Executar as conversões vetorial/raster (cf. seção 6.1) em ordem de z_{\max} decrescente (de trás para frente).

A resolução das ambiguidades da segunda etapa consiste em aplicar vários testes, incluindo o teste das extensões da seção anterior, para decidir se a ordem dos polígonos deve ser ou não mudada. No entanto, quando um polígono é dividido pelo plano de um outro, o problema só pode ser resolvido cerceando-o e incluindo suas partes na lista dos polígonos (cf. detalhe em [Newell, 1972]).

6.4 - ALGORITMO DA LINHA EXPLORATÓRIA PARA SUPERFÍCIES ESCONDIDAS

Este algoritmo desenvolvido por Wylie, Rommey, Evans e Erdahl [Wylie, 1967], Bouknight [Bouknight, 1970a] e [Bouknight, 1970b], e Watkins [Watkins, 1970] atua no espaço da imagem criando uma imagem linha por linha. Ele é uma extensão do algoritmo de conversão vetorial/raster de polígonos dado na seção 6.1.

Em vez de considerar um só polígono, tem-se agora um con

junto de polígonos. Na lista de lados ET são introduzidos, na posição certa (cf. seção 6.1) os vetores descritivos de todos os lados (não horizontais) desses polígonos. Nos vetores descritivos acrescenta-se mais um elemento apontando pelo polígono de origem:

$(y_{\max}, x, 1/m, \text{identificador do polígono})$

Além de ET e AET, cria-se a lista dos polígonos PT na forma de um vetor APL de vetores descritivos dos polígonos com três elementos:

$((a,b,c,d), \text{tom de cinza ou cor, "flag"})$

onde (a,b,c,d) são os parâmetros da equação do plano contendo o polígono, no sistema de coordenadas do dispositivo (cf. seção 3.2); "flag" é uma variável binária tomando os valores "exterior" ou "interior" (com o valor inicial "exterior").

O algoritmo é o da linha exploratória onde o passo 3 é modificado da seguinte maneira:

3. Processar os lados em ET de esquerda para direita atualizando os "flags" em PT (i.é., trocar o valor do "flag" de um polígono quando se cruza um dos seus lados). Entre dois pares de interseção:

3.1. Se só um "flag" é "interior" preencher os pontos usando o tom de cinza ou a cor do polígono.

3.2. Se mais de um "flag" é "interior", calcular para cada polígono correspondente a coordenada z do ponto do polígono de abscissa x e de ordenada y :

$$z = -(ax + by + d)/c$$

(x é dado no vetor descritivo do último lado encontrado, y é o número da linha exploratória), e preencher os pontos usando o tom de cinza ou a cor do polígono com z mínimo).

A vantagem desse algoritmo sobre o da seção anterior é que são convertidos os polígonos visíveis.

Quando a cena 3D é constituída somente por objetos sólidos, i.é., que não há interpenetração entre os polígonos, e quando tem mais de dois polígonos "interior", observa-se que não precisa calcular de novo as profundidades z na hora de sair de um polígono escondido. Por outro lado, quando há interpenetração de polígonos, o polígono que penetra o outro, tem que ser dividido em dois antes de usar o algoritmo descrito acima.

A figura 6.4 mostra a projeção de dois triângulos no espaço das coordenadas do dispositivo. As linhas escondidas são indicadas em pontilhados, e a linha exploratória na posição α . Seguindo o algoritmo, quando se percorre a linha exploratória ($y = \alpha$), entrando no polígono 1, seu "flag" passa a ser "interior" e preenche-se então os pontos com o tom de cinza ou a cor do polígono 1, entrando no polígono 2, seu "flag" passa a ser "interior". Como mais de um "flag" é "interior" calcula-se a profundidade z de cada ponto que se projeta na interseção entre a linha exploratória e o atual lado. No exemplo o polígono 2 tem z menor e por isso é visível, e se preenchem então os pontos com o tom de cinza ou a cor do polígono 2. Saindo do polígono 1, seu "flag" passa a ser "exterior", então somente um "flag" é "interior" e se preenchem os pontos de acordo com o tom de cinza ou a cor do polígono 2 (i.é. nada muda) até sair do polígono 2.

6.5 - ALGORITMO DO Z-BUFFER

O algoritmo do "z-buffer" é o mais simples dos algoritmos para resolver o problema das superfícies escondidas. Ele atua no espaço da imagem e usa também o algoritmo da linha exploratória da seção 6.1. Por outro lado, ele não precisa que sejam ordenados os polígonos. Em compensação ele necessita, além da memória de restauração, de uma outra memória do mesmo tamanho chamada de "z-buffer".

O algoritmo consiste nos seguintes passos.

- Iniciar o "z-buffer" com o maior valor de z possível, e a memória de restauração com o valor de tom de cinza ou da cor correspondente ao fundo.

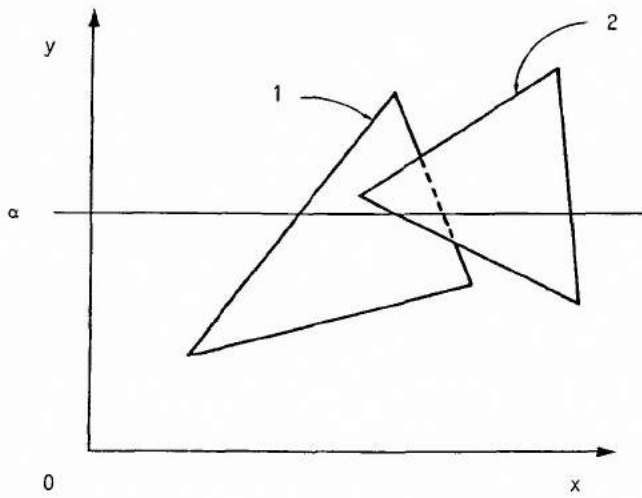


Figura 6.4 A linha exploratória e a projeção de dois triângulos.

- Para cada polígono e cada um dos seus pontos interiores (x,y) (cf. o algoritmo da linha exploratória para a determinação dos pontos interiores), calcular a coordenada z (a profundidade):

$$z = (ax + by + d)/c$$

onde a, b, c e d são os parâmetros da equação do plano contendo o polígono, expresso no sistema de coordenadas do dispositivo.

Se z é inferior ao valor contido no "z-buffer" na posição (x,y), carregar nessa posição z no "z-buffer" e o valor do tom de cinza ou da cor na memória de restauração. Se z é superior a este valor não modificar nada no "z-buffer" e a memória de restauração.

Quando z é menor que o valor contido no "z-buffer", isto significa que o ponto do atual polígono é mais próximo do observador, do que o ponto cujo valor de tom de cinza ou da cor encontra-se na memória de restauração, e que o "z-buffer" e a memória de restauração devem ser atualizados.

Observa-se que a cada passo da execução do algoritmo, o "z-buffer" contém os menores valores de profundidade encontrados para cada ponto da imagem.

Aplicando a idéia de coerência espacial, e conhecendo o valor da profundidade z, de um ponto interior de um polígono de abscissa x e de ordenada y, sua profundidade z_{i+1} no ponto vizinho de abscissa x+1 e da mesma ordenada y (i.é., projetando-se na mesma linha exploratória) é dada por:

$$z_{i+1} = -(a(x+1) + by + d)/c = z_i - a/c.$$

Assim o cálculo da profundidade reduz-se a uma subtração (já que a/c é constante e pode ser pré-calculado).

6.6 - ALGORITMO DE SUBDIVISÃO DE ÁREA

A idéia desse algoritmo é detectar situações nas quais o problema de achar qual é o polígono visível fica mais simples. Para isto

uma área do espaço das coordenadas do dispositivo é examinada, se é fácil achar o polígono visível, então a área é preenchida senão ela é subdividida e cada uma das suas partes forma uma nova área a ser examinada. Quanto menor é a área examinada mais simples devem ser as conclusões ao seu respeito. O algoritmo atua no espaço da imagem.

Observa-se que em relação a uma dada área de interesse, os polígonos podem ser envolventes (i.é., conter a área de interesse), cortadas (i.é. interceptar a área de interesse), contidos (i.é., ser dentro da área de interesse) ou disjuntos (i.é. ser fora da área de interesse). Essas quatro situações são ilustradas na figura 29 por quatro polígonos PL1, PL2, PL3, PL4 e uma área de interesse AREA quadrada (cujo contorno aparece em pontilhado). Esses são especificados na forma de listas de vértices na "transparência" = 30 = . A "transparência" = 31 = mostra como reconhecer o tipo de relação entre os polígonos convexos PL1, PL2, PL3 e PL4 e a área de interesse AREA. Para saber se um polígono convexo é envolvente basta verificar que os quatro vértices da área de interesse são interiores usando o algoritmo da seção 4.5. Para saber se um polígono convexo é disjunto da área de interesse basta estudar a posição dos quatro vértices da área de interesse, relativamente às retas contendo os lados do polígono.

Observa-se que um polígono convexo é disjunto se existe pelo menos um semi-plano exterior contendo os quatro vértices da área de interesse. Nas condições definidas na seção 4.5, os semi-planos exteriores são positivos, i.é., se o polígono é disjunto existe um lado para o qual $\alpha u + \beta v + \gamma$ (α, β, γ dados por 3.19) fica positivo quando calculado em qualquer vértice (u, v) da área de interesse.

Na "transparência" = 31 = a função $(2, /c^{**})$ permite transformar a lista de vértices numa lista de lados, cada lado sendo na forma de um vetor APL de dois vetores de coordenadas representando seus vértices. Se todos os elementos de A são 1, o polígono é envolvente, se pelo menos todos os elementos de uma coluna de A são 0, o polígono é disjunto. Assim duas expressões booleanas são calculadas para reconhecer o tipo de relação entre o polígono e a área de interesse.

▣ SUPERFICIES ESCONDIDAS = 30 =
▣ ALGORITMO DE SUBDIVISÃO DE AREA
▣ EXEMPLO

AREA+(2 2)(2 2)(2 2)(2 2)

PL1+(3 3)(3 4)(4 3)(1 5)(5 3)(3 3)

PL2+(5 5)(1 5)(4 4)(5 5)

PL3+(1 1)(1 1)(1 1)(1 1)

PL4+(3 0)(1 1)(3 2)(3 0)

GSLN 3

GFA CV AREA

GSLN 1

GPL CV PL1 PL2 PL3 PL4 (produz figura 29)

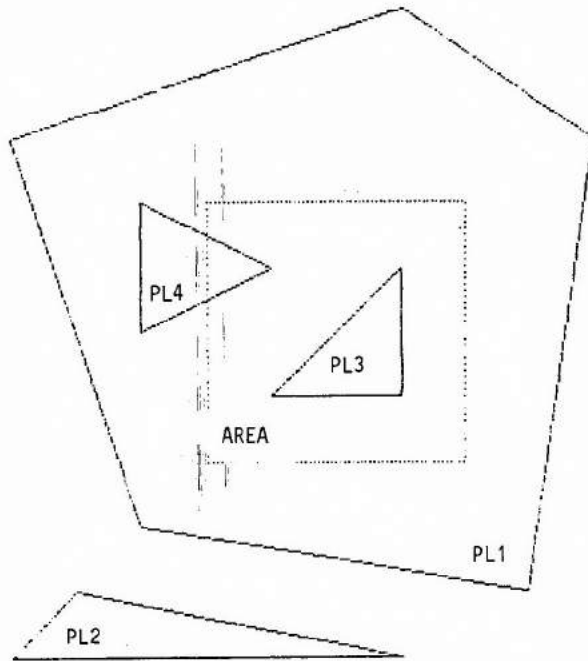


Fig. 29 - Os quatro tipos de relação entre a área de interesse (quadrada) e um polígono convexo.

PL1: polígono envolvente

PL2: polígono disjunto

PL3: polígono contido

PL4: polígono cortado

```
n SUPERFICIES ESCONDIDAS = 31 =
n ALGORITMO DE SUBDIVISÃO DE AREA
n TESTE DA NORMAL (POLIGONO CONVEXO)

      0) > (c"CH AREA)+.x""cN2"2,/c"PL2
1 0 0
1 0 1
1 0 1
1 0 0

      A(0) > (c"CH AREA)+.x""cN2"2,/c"PL1
      ( ^/,A)(v/^f~A)
1 0      n PL1 E' ENVOLVENTE

      A+0) > (c"CH AREA)+.x""cN2"2,/c"PL2
      ( ^/,A)(v/^f~A)
0 1      n PL2 E' DISJUNTO

      A+0) > (c"CH AREA)+.x""cN2"2,/c"PL3
      ( ^/,A)(v/^f~A)
0 0      n PL3 NÃO E' ENVOLVENTE NEM DISJUNTO

      A+0) > (c"CH AREA)+.x""cN2"2,/c"PL4
      ( ^/,A)(v/^f~A)
0 0      n PL4 NÃO E' ENVOLVENTE NEM DISJUNTO
```

Baseado nesse tipo de relação o processo de subdivisão da área prossegue ou não. Quando ele termina, a área é preenchida, isto acontece nas quatro situações seguintes.

- 1 - Todos os polígonos são disjuntos então a área é preenchida com o tom de cinza ou a cor do fundo.
- 2 - Tem somente um polígono contido ou cortado, então a área é preenchida com o tom de cinza ou a cor do fundo, e em seguida o polígono ou parte do polígono é preenchido usando o algoritmo da linha exploratória.
- 3 - Tem somente um polígono envolvente, então a área é preenchida com o tom de cinza ou a cor do polígono.
- 4 - Tem mais de um polígono cortado, contido ou envolvente, e o da frente é envolvente então a área é preenchida com o tom de cinza ou a cor do polígono. Para determinar se um polígono envolvente está na frente, a coordenada z dos pontos dos planos contendo os polígonos e projetando-se nos vértices da área de interesse é calculada. O polígono envolvente está na frente dos outros, se suas quatro coordenadas z são menores que em qualquer outro polígono. Se não é verdade ele pode ou não ser na frente. Neste caso prefere-se subdividir a área do que levantar a dúvida.

Quando a subdivisão não é mais possível porque chega-se ao nível de um ponto elementar da imagem e que nenhuma das quatro situações acima ocorre, então calcula-se a profundidade z relativa a todos os polígonos relevantes e o ponto é preenchido com o tom de cinza ou a cor do polígono com z menor.

O algoritmo de subdivisão de área proposto inicialmente por Warnock [Warnock, 1969] subdividia uma área quadrada em quatro sub-áreas quadradas. A figura 6.5 mostra um exemplo de subdivisão seguindo o algoritmo acima. Os números indicam qual é a situação que levou a interromper o processo de subdivisão. As áreas sem números correspondem às áreas que deveriam ser ainda subdivididas.

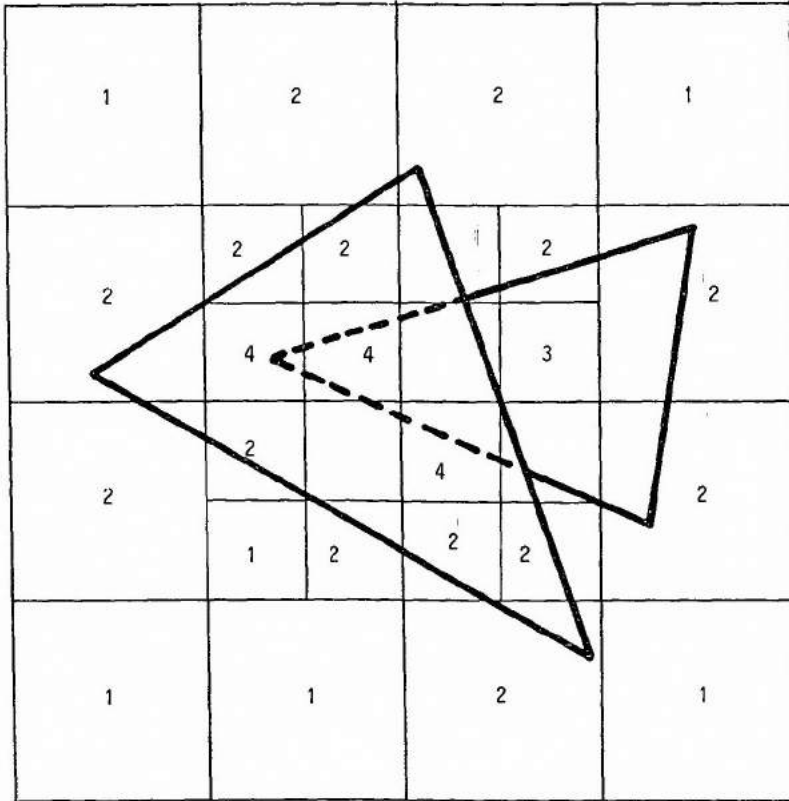


Figura 6.5 Exemplo de aplicação do algoritmo de subdivisão de área.

Uma outra estratégia desenvolvida por Weiler e Atherton [Weiler, 1977], consiste em subdividir a imagem seguindo o contorno dos polígonos.

6.7 - EFICIÊNCIA DOS ALGORITMOS E O PROBLEMA DE "ALIASING"

Sutherland, Sproull e Schumacker [Sutherland, 1974b] fizeram um estudo comparativo dos quatro algoritmos descritos nas seções 6.3 - 6.6. A eficiência relativa em termo de tempo de execução é dada na tabela 6.1.

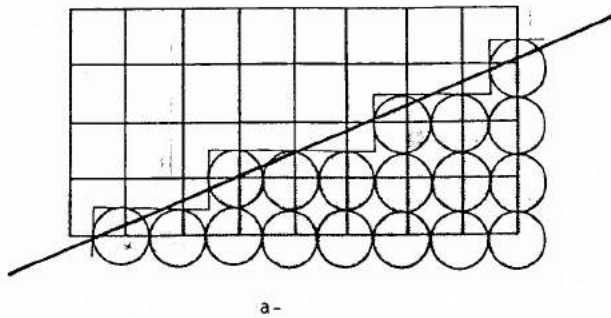
Observa-se que o algoritmo do "z-buffer" tem uma eficiência constante com o número de polígonos na cena. Isto é devido ao fato que no experimento realizado, quando o número de polígonos aumentava, suas áreas diminuam na mesma proporção (o que é a situação mais correta).

A representação de uma cena através de uma imagem digitalizada conduz a uma degradação da informação contida na cena, os detalhes finos são perdidos. A criação de uma imagem digitalizada é obtida por meio de um processo chamado de amostragem na teoria do sinal.

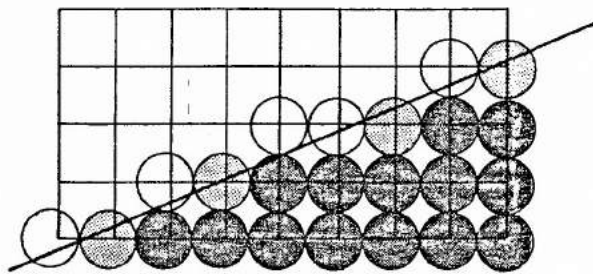
A amostragem de uma cena introduz distorção dependendo da informação contida na cena, essas distorções podem provocar efeitos visuais secundários chamados "aliasing". O processo de amostragem comum consiste em preencher um ponto da imagem com o tom de cinza ou a cor do ponto geométrico correspondente da cena. Lembrando que um ponto da imagem não é um ponto geométrico mas na realidade uma área geralmente circular ou quadrada e que somente o centro dessa área está em correspondência com o ponto da cena, pode-se explicar o aparecimento desses efeitos secundários. Como exemplo desses efeitos tem-se o caso do contato retilíneo entre duas regiões. Se essas duas regiões têm tons de cinza ou cores diferentes o contato deveria aparecer como uma linha reta, na realidade, dependendo da inclinação do contato, ele pode aparecer como uma linha poligonal em forma de pequenos degraus (ver figura 6.6a).

Algoritmo	Número de polígonos na cena		
	100	2500	60000
ordenação por profundidade	1	10	507
linha exploratória	5	21	100
"z-buffer"	54	54	54
subdivisão de área	11	64	307

Tabela 6.1 Eficiência relativa entre quatro algoritmos de eliminação de polígonos.



a-



b-

Figura 6.6 Digitalização do contato entre duas regiões. Amostragem comum (a), e amostragem com técnica de "anti-aliasing" (b).

Por outro lado, se é possível definir um modelo espectral para a cena, a teoria da amostragem indica que a frequência espacial de amostragem (frequência de Nyquist) tem que ser o dobro da maior frequência espacial contida na cena (teorema de Shannon). Se essa condição não é satisfeita, efeitos secundários podem aparecer provocando por exemplo os famosos "moirés" quando a cena contém uma frequência espacial próxima da frequência de amostragem (esse efeito sendo similar à de um batimento).

Para atenuar esses efeitos secundários, usam-se técnicas ditas de "anti-aliasing". Nessas técnicas procura-se associar a um ponto da imagem, não um ponto geométrico da cena, mas a área correspondente, e de um modo ou de outro integrar todas as informações contidas nesta área para determinar o tom de cinza ou a cor mais conveniente para o ponto da imagem.

Por exemplo, no caso do contato entre duas regiões, procura-se determinar qual é a superfície ocupada por cada face dentro do "ponto" elementar da imagem (ver figura 6.6b).

Pitteway e Watkinson [Pitteway, 1980] desenvolveram uma extensão do algoritmo de conversão vetorial/raster para linhas de Bresenham [Bresenham, 1965] para estimar eficientemente a superfície ocupada por um polígono dentro de um ponto elementar da imagem.

7 - SUPERFÍCIES CURVAS

7.1 - CURVAS CÚBICAS PARAMÉTRICAS

As malhas de polígonos vistas no capítulo 4 é uma das maneiras de representar formas em 3D. A outra é através das superfícies cúbicas paramétricas.

Com as malhas de polígonos, as formas são obtidas pela justaposição de um grande número de pequenos polígonos. Nesse capítulo as superfícies curvas são obtidas pela justaposição (em número relativamente menor) de porções de superfícies cúbicas chamadas aqui de retalhos bicúbicos paramétricos. Os limites desses retalhos são porções de curvas cúbicas paramétricas que são estudados adiante.

A representação paramétrica de curvas em 3D é mais conveniente. Em particular com essa representação não há problema para trabalhar com curvas cuja tangente pode ser paralela aos eixos do sistema de coordenadas.

Na seção 3.2 foi definida a representação paramétrica de uma reta, na qual as coordenadas x , y e z de um ponto M da reta dependia linearmente de um parâmetro t . No caso de uma curva cúbica paramétrica, as coordenadas x , y e z de um ponto M na curva são polinômios de ordem 3 (daí o adjetivo cúbico) no parâmetro t . Por convenção o ponto M da curva percorre a porção ou segmento de interesse da curva quando t percorre o intervalo $[0,1]$. As equações paramétricas de uma porção de curva cúbica são:

$$\left. \begin{aligned} x &= a_x t^3 + b_x t^2 + c_x t + d_x \\ y &= a_y t^3 + b_y t^2 + c_y t + d_y \\ z &= a_z t^3 + b_z t^2 + c_z t + d_z \end{aligned} \right\} \left. \begin{array}{l} \text{no} \\ \text{plano} \end{array} \right\} \text{no} \\ & \hspace{10em} \left. \hspace{10em} \right\} \text{espaço}$$

para $t \in [0,1]$.

Usando a representação matricial, e chamando de M_t o pon

to da curva para um certo valor t do parâmetro, as equações acima, no caso do espaço podem se escrever:

$$[M_t] = T_t A \quad \text{para } t \in [0,1]$$

onde $[M_t] = [x \ y \ z]$

$$T_t = [t^3 \ t^2 \ t \ 1]$$

$$\text{e } A = \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix}.$$

O modelo cúbico é usado porque é o modelo mais simples (polinomial da ordem menor) que garante que na junção de dois segmentos de curvas cúbicas as tangentes possam ter a mesma direção (i.e., a primeira derivada continua) e que as extremidades dos segmentos passassem por pontos pré-determinados. As formas de Hermite e de Bezier são as formas usadas para especificar as equações das porções de curvas quando as propriedades acima devem ser garantidas.

O modelo cúbico é também o modelo mais simples que garante que na junção de dois segmentos de curva, a primeira e a segunda derivada sejam contínuas e que a curva passasse "mais ou menos" por um conjunto de pontos. A forma B-spline é então a forma usada para especificar as equações das porções de curvas.

- A forma de Hermite

O vetor \vec{V}_t tangente a curva no ponto M_t verifica a equação:

$$[\vec{V}_t] = [3t^2 \ 2t \ 1 \ 0]A \quad \text{para } t \in [0,1].$$

A forma de Hermite permite determinar A uma vez fixadas as

extremidades da porção de curva M_0 e M_1 , e suas tangentes em M_0 e M_1 ou seja \vec{V}_0 e \vec{V}_1 . Tem-se então o seguinte sistema de equação

$$\begin{bmatrix} M_0 \\ M_1 \\ \vec{V}_0 \\ \vec{V}_1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} A,$$

do qual se pode determinar A, já que a matriz 4x4 pré-multiplicando A tem uma inversa:

$$M_h = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

$$\text{Assim } A = M_h G_h$$

onde G_h é a matriz de geometria de Hermite:

$$G_h = \begin{bmatrix} M_0 \\ M_1 \\ \vec{V}_0 \\ \vec{V}_1 \end{bmatrix},$$

M_h chama-se matriz de Hermite.

Finalmente:

$$[M_t] = T_t M_h G_h \text{ para } t \in [0,1],$$

i.é., fixado a matriz da geometria de Hermite G_h , pode se calcular o vetor de coordenadas de um ponto qualquer M_t na porção de curva entre M_0 e M_1 , $t \in [0,1]$.

A forma de Bezier

A forma de Bezier permite determinar A uma vez fixado qua

tro pontos: as duas extremidades M_0 e M_1 , chamadas agora de P_1 e P_4 e mais dois pontos P_2 e P_3 controlando as tangentes \vec{V}_0 e \vec{V}_1 da seguinte maneira:

$$\vec{V}_0 = k(\vec{OP}_2 - \vec{OP}_1) \text{ e } \vec{V}_1 = k(\vec{OP}_4 - \vec{OP}_3) \text{ com } k > 0$$

i.ê., a relação entre as matrizes de geometria de Hermite G_h e de Bezier

$$G_b = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix}, \text{ e:}$$

$$G_h = M_{hb} G_b \text{ onde}$$

$$M_{hb} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -k & k & 0 & 0 \\ 0 & 0 & -k & k \end{bmatrix}.$$

Assim

$$[M_t] = T_t M_h M_{hb} G_b \text{ para } t \in [0,1]$$

Seja $M_b = M_h M_{hb}$, sua expressão é

$$M_b = \begin{bmatrix} 2-k & k & -k & -2+k \\ -3+2k & -2k & k & 3-k \\ -k & k & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

Em termo de M_b tem-se para $t \in [0,1]$

$$[M_t] = T_t M_b G_b \tag{7.1}$$

A matriz M_b depende do parâmetro k . Uma vez fixada a matriz de geometria de Bezier G_b , a curva cúbica depende ainda do parâmetro k . Esse parâmetro é escolhido de maneira a ser o maior número real que garante que qualquer que seja os pontos P_1, P_2, P_3 e P_4 , a curva cúbica fica no interior

do envoltório convexo desses quatro pontos. Para garantir que um ponto P seja dentro do envoltório convexo, os quatro coeficientes α_i em

$$\vec{OP} = \alpha_1 \vec{OP}_1 + \alpha_2 \vec{OP}_2 + \alpha_3 \vec{OP}_3 + \alpha_4 \vec{OP}_4$$

ou $[P] = [\alpha_1 \ \alpha_2 \ \alpha_3 \ \alpha_4] \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix}$

tem que ser tais que:

$$\alpha_i \in [0,1] \quad i = 1, 4 \quad \text{e} \quad \sum_{i=1}^4 \alpha_i = 1. \quad (7.2)$$

Assim para que a curva cúbica seja dentro do envoltório convexo os quatro elementos da matriz linha $T_t M_b$ devem verificar as condições (7.2). Observa-se que o coeficiente de \vec{OP}_4 é

$$t^2((k-2)t + 3 - k),$$

e que para ser positivo para qualquer $t \in [0,1]$ k deve ser inferior ou igual a 3. Para $k = 3$ tem-se:

$$\vec{OP} = (1-t)^3 \vec{OP}_1 + 3t(1-t)^2 \vec{OP}_2 + 3t^2(1-t) \vec{OP}_3 + t^3 \vec{OP}_4,$$

e os coeficientes verificam (7.2). Assim $k = 3$ é a solução e a matriz M_b correspondente é a matriz de Bezier:

$$M_b = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}. \quad (7.3)$$

Finalmente os pontos da curva cúbica especificada por meio da matriz de geometria de Bezier são dados por (7.1) onde M_b tem as expressões (7.3) com $t \in [0,1]$.

A "transparência" = 32 = mostra a matriz de Bezier denotada da MB e quatro pontos P1, P2, P3 e P4. A matriz de geometria denotada GB

a CURVAS CUBICAS PARAMETRICAS
a FORMA DE BEZIER

= 32 =

```
      MB
T1  3  -3  1
   3  -6  3  0
T3  3  0  0
   1  0  0  0
```

P1←T4 T3

P2←T4 3

P3←5 T3

P4←T1 T3

GB←P1 P2 P3 P4

T←(T1+141)÷40

TT←T*"

CV1←TT+.x"

GFL CV CV1

GFM CV P1 P2 P3 P4

GSLN 2

GFA CV ENVOLTORIO P1 P2 P3 P4 (produz figura 30)

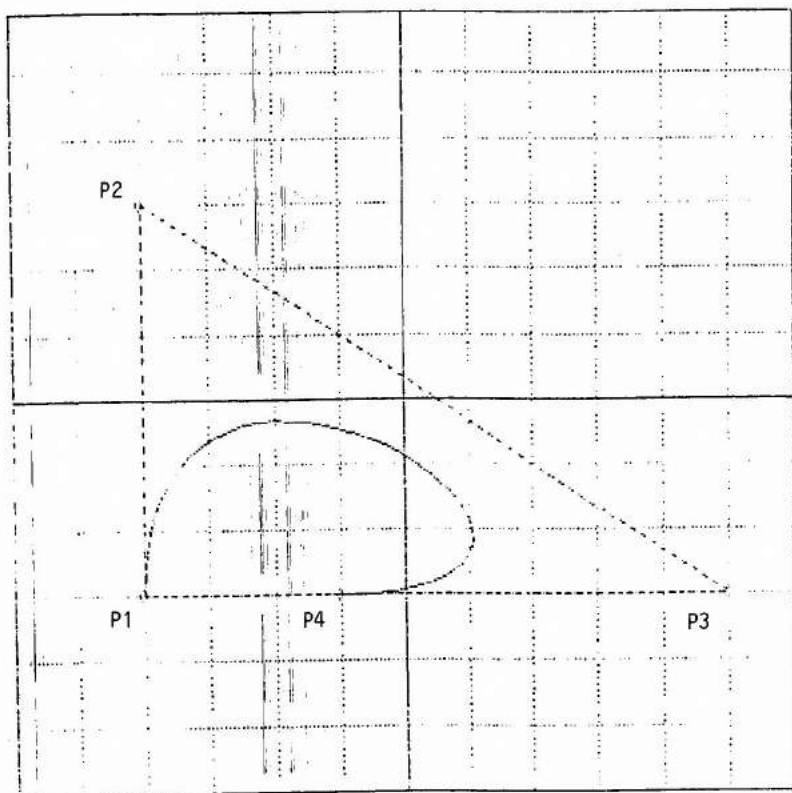


Fig. 30 - A porção de curva cúbica de matriz de geometria de Bezier:

$$\begin{bmatrix} P1 \\ P2 \\ P3 \\ P4 \end{bmatrix},$$

e o envoltório convexo dos pontos de controle.

é construída a partir desses quatro pontos. O vetor APL T contém uma sequência crescente de 41, números reais igualmente distribuídos entre 0 e 1. Os 41 elementos do vetor APL TT são os vetores $(t^3, t^2, t, 1)$ avaliados para os valores de t em T.

Aplicando a equação da curva cúbica (7.3), o vetor APL CV1 é criado contendo os vetores de coordenadas de 41 pontos correspondentes aos valores do parâmetro em T. Em seguida a curva cúbica é traçada (ver figura 30) assim como os quatro pontos P₁, P₂, P₃ e P₄ e o envoltório convexo deles. Verifica-se que as extremidades da curva são os pontos P₁ e P₄, que os vetores $\vec{P}_1\vec{P}_2$ e $\vec{P}_3\vec{P}_4$ indicam a direção das tangentes nos pontos P₁ e P₄, e que a curva é totalmente contida no envoltório convexo dos pontos P₁, P₂, P₃ e P₄.

A "transparência" = 33 = segue os mesmos passos que os da "transparência" anterior só que o papel dos pontos P₃ e P₄ é invertido na matriz de geometria de Bezier GB, i.é., P₃ passa a ser a segunda extremidade e P₄ o ponto de controle da tangente. A curva cúbica obtida aparece na figura 31. O envoltório convexo dos pontos P₁, P₂, P₃ e P₄ continua o mesmo, e a curva, totalmente, no seu interior. A função APL ENVOLTÓRIO tem por argumento de entrada um vetor APL de vetores de coordenadas representando os pontos em qualquer ordem, e produz um vetor APL representando esses mesmos pontos mas na ordem em que eles se encontram percorrendo o envoltório convexo deles.

A figura 32 mostra duas curvas de Bezier tendo uma extremidade em comum P₄. As duas matrizes de geometria de Bezier são:

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} \quad \text{e} \quad \begin{bmatrix} P_4 \\ P_5 \\ P_6 \\ P_7 \end{bmatrix}.$$

Os pontos P₃ e P₅ foram escolhidos de maneira que P₃, P₄ e P₅ sejam alinhados garantindo assim a continuidade da primeira derivada no ponto P₄.

a CURVAS CUBICAS PARAMETRICAS
a FORMA DE BEZIER

= 33 =

P1←-4 -3

P2←-4 3

P3←5 -3

P4←-1 -3

T←(-1+14i)+40

TT←T*³c3 2 1 0

GB←P1 P2 P4 P3

CV2←TT+.x³cMB+.xGB

GPL CV CV2

GPM CV P1 P2 P4 P3

GSLN 2

GFA CV ENVOLTORIO P1 P2 P4 P3 (produz figura 31)

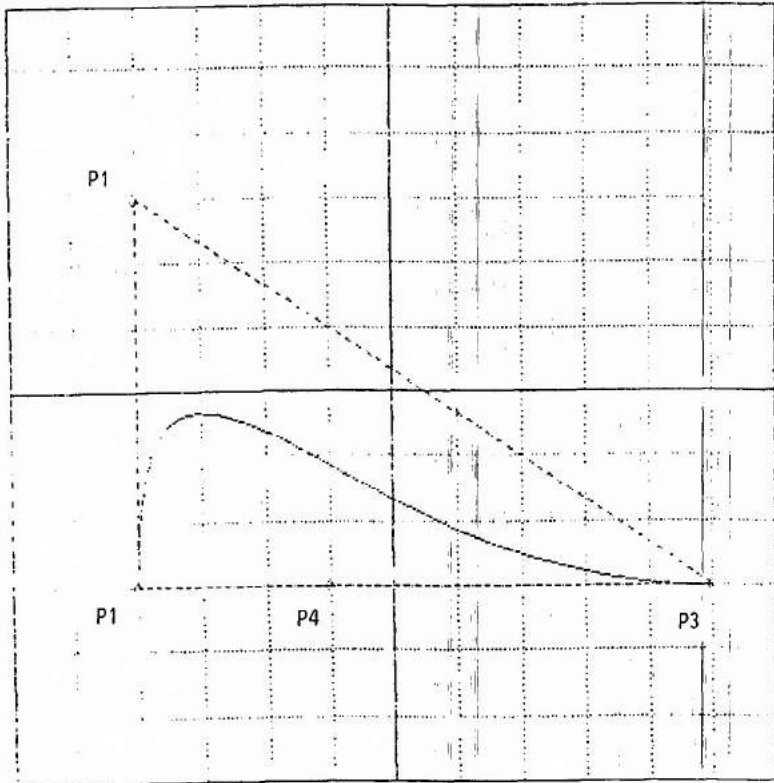


Fig. 31 - A porção de curva cúbica de matriz de geometria de Bezier:

$$\begin{bmatrix} P1 \\ P2 \\ P4 \\ P3 \end{bmatrix},$$

e o envoltório convexo dos pontos de controle.

A "transparência" = 34 = mostra os sete pontos P_1, \dots, P_7 . As duas matrizes de geometria são armazenadas no vetor APL A. Concatenando as duas porções de curvas cúbicas obtêm-se a sequência de pontos CV3.

Na figura 32 aparecem também os dois envoltórios convexos. A figura 33 mostra a importância para visualizar o envoltório de ordenar os vetores de coordenadas dos pontos através da função ENVOLTÓRIO antes de traçar a linha poligonal. Para a primeira porção da curva, sem ordenar os pontos, obtêm-se uma linha poligonal cruzada.

- A Forma B-spline

Na forma B-spline a matriz da geometria G_s é:

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix}$$

onde P_1, P_2, P_3 e P_4 são quatro pontos, i.é. esta matriz é idêntica à matriz G_b da forma de Bezier. A matriz A de coeficiente é dada por

$$A = M_s G_s$$

onde

$$M_s = \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & 6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \quad (7.4)$$

e os pontos da curva cúbica são dados por

$$[M_t] = T_t M_s G_s \quad (7.5)$$

com $t \in [0,1]$.

Geralmente a curva não passa pelos pontos de controle P_1, P_2, P_3 e P_4 , mas tende a aproximar-se de P_2 e P_3 com uma tendência em se dirigir para P_1 e P_4 .

» CURVAS CUBICAS PARAMETRICAS = .34 =
» FORMA DE BEZIER

```
      »P1 P2 P3 P4 P5 P6 P7
-5    0
-4    2
-0.5 -2
  0    0
  1    4
  4    4
  5    2
```

T←(-1+141)÷40

TT←T*³ 2 1 0

ρA←1 0 0 1/4, /c P1 P2 P3 P4 P5 P6 P7

2

GPL CV CV3←, >(←TT)+, x³ ←c (←MB)+, x³ = A

GPM CV P1 P2 P3 P4 P5 P6 P7

GSLN 2

GFA CV ENVOLTORIO A (produz figura 32)

GCLRWK 1

GPM CV P1 P2 P3 P4 P5 P6 P7

GFA CV A (produz figura 33)

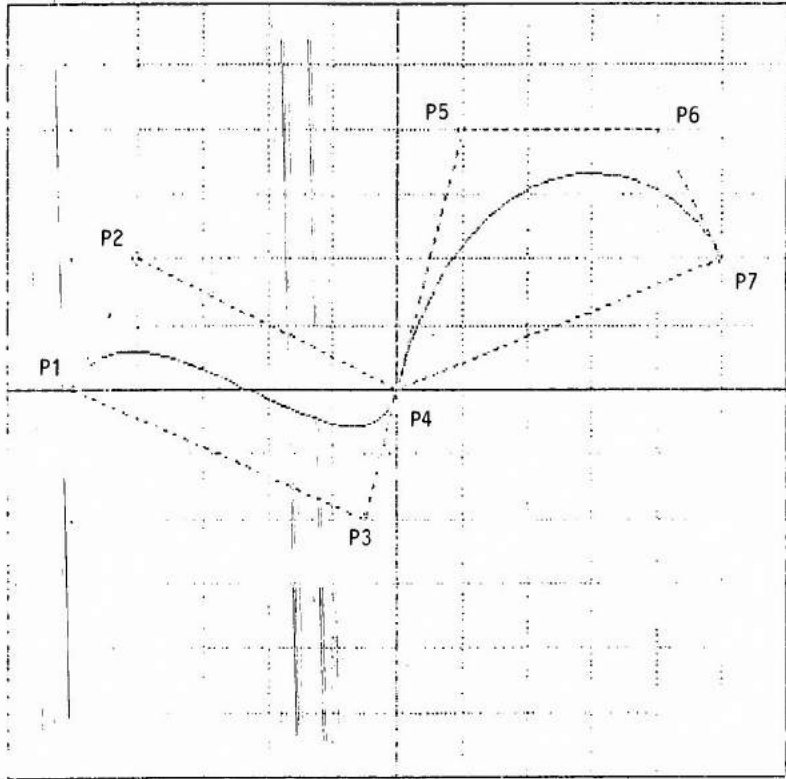


Fig. 32 - As duas porções de curva cúbica de Bezier consecutivas com os 7 pontos de controle (e os dois envoltórios convexos).

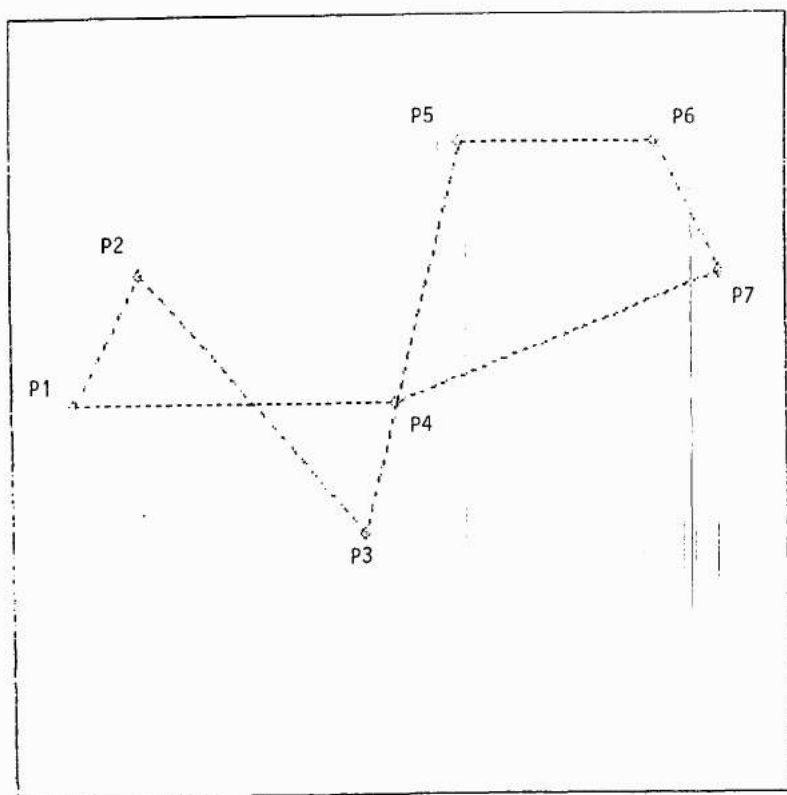


Fig. 33 - A linha poligonal cruzada P1, P2, P3, P4, P1 (somente a linha poligonal P4, P5, P6, P7, P4 forma o envoltório dos 4 pontos de controle).

Seja $\{P_1, \dots, P_n\}$ uma seqüência de pontos. A matriz M_S foi escolhida de maneira que na série de B-splines os pontos M_t^i (onde i indica a porção de B-spline obtida a partir da matriz de geometria:

$$G_S^i = \begin{bmatrix} P_{i-1} \\ P_i \\ P_{i+1} \\ P_{i+2} \end{bmatrix}, \quad 2 \leq i \leq n-2;$$

tem-se $n-3$ porções de curva) verificam as condições (para qualquer i tal que $2 \leq i \leq n-3$):

$$M_1^i = M_0^{i+1}$$

$$\vec{V}_1^i = \vec{V}_0^{i+1}$$

$$\vec{W}_1^i = \vec{W}_0^{i+1}$$

onde $[\vec{V}_S] = \left[\frac{dx}{dt} \Big|_S, \frac{dy}{dt} \Big|_S, \frac{dz}{dt} \Big|_S \right]$

$$[\vec{W}_S] = \left[\frac{d^2x}{dt^2} \Big|_S, \frac{d^2y}{dt^2} \Big|_S, \frac{d^2z}{dt^2} \Big|_S \right].$$

Em outros termos, as extremidades M_1^i e M_0^{i+1} de duas porções de B-splines consecutivas tem que coincidir, e nesse ponto a primeira e segunda derivadas da curva resultante tem que ser contínuas.

A figura 34 mostra uma B-spline controlada pelos sete pontos já usados na figura 32. A "transparência" = 35 = mostra como esta curva é obtida. Ela é composta de quatro porções cujas matrizes de geometria são armazenadas no vetor APL GS. O algoritmo para a obtenção da seqüência dos pontos CV4 e o mesmo que no caso da forma de Bezier (para a obtenção de CV3). Observa-se que a B-spline resultante é mais suave que a curva de Bezier resultante da figura 32.

7.2 - SUPERFÍCIES CÚBICAS PARAMÉTRICAS

O modelo paramétrico das curvas cúbicas é generalizado in

a CURVAS CUBICAS PARAMETRICAS
a FORMA B-SPLINE

35 =

		MS	
1	3	-3	1
3	-6	3	0
-3	0	3	0
1	4	1	0

		P1	P2	P3	P4	P5	P6	P7
1	0							
4	2							
-0.5	-2							
0	0							
1	4							
4	4							
5	2							

T←(-1+(141)÷40

TT←T×"c3 2 1 0

GS←4./c"P1 P2 P3 P4 P5 P6 P7

CV4←,=(cTT)+.x""c"(cMS+6)+.x""GS

GPL CV CV4

GPM CV P1 P2 P3 P4 P5 P6 P7 (produz figura 34)

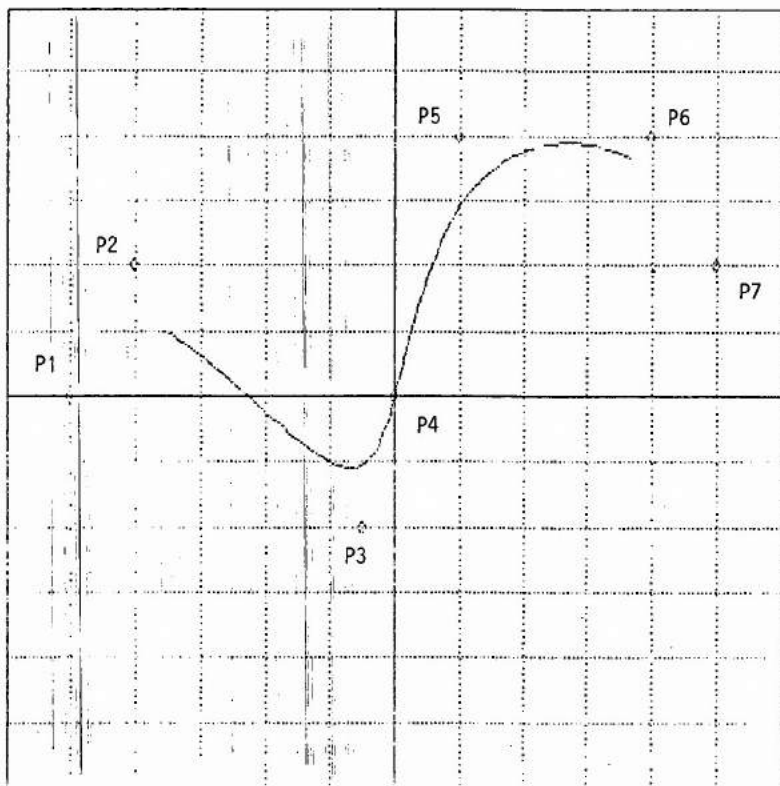


Fig. 34 - As quatro porções de curvas cúbicas B-spline consecutivas com o 7 pontos de controle.

troduzindo equações cúbicas com dois parâmetros s e t , ou melhor, equações bi-cúbicas. Por convenção o ponto M da superfície percorre a porção de superfície de interesse ou "retalho" quando (s,t) percorre o quadrado $[0,1]^2$.

As equações bi-cúbicas de um retalho são:

$$\begin{aligned} x = & a_{11}s^3t^3 + a_{12}s^3t^2 + a_{13}s^3t + a_{14}s^3 \\ & + a_{21}s^2t^3 + a_{22}s^2t^2 + a_{23}s^2t + a_{24}s^2 \\ & + a_{31}st^3 + a_{32}st^2 + a_{33}st + a_{34}s \\ & + a_{41}t^3 + a_{42}t^2 + a_{43}t + a_{44} \end{aligned}$$

ou na forma matricial

$$x = SC_x T^T \text{ onde}$$

$S = [s^3 \ s^2 \ s \ 1]$, $T = [t^3 \ t^2 \ t \ 1]$ e C_x a matriz $[a_{ij}]$ 4x4 dos coeficientes da superfície relativos a coordenada x ,

$$y = SC_y T^T$$

$$z = SC_z T^T,$$

onde C_y e C_z são as matrizes de coeficientes da superfície relativos às coordenadas y e z .

Seja $M_{s,t}$ o ponto de superfície para um certo (s,t) . Os conjuntos dos pontos $\{M_{s,0}; s \in [0,1]\}$, $\{M_{s,1}; s \in [0,1]\}$, $\{M_{0,t}; t \in [0,1]\}$ e $\{M_{1,t}; t \in [0,1]\}$ formam os limites do retalho. Para um dado valor de s $\{M_{s,t}; t \in [0,1]\}$ é uma curva cúbica da seção 7.1. Da mesma forma, para um dado valor de t , $\{M_{s,t}; s \in [0,1]\}$ é uma curva cúbica.

Na forma de Hermite a matriz C_x (o que vai ser dito vale também para C_y e C_z) é dada por:

$$C_x = M_h Q_x M_h^T$$

onde M_h é a matriz de Hermite da seção 7.1 e Q_x é:

$$Q_x = \text{abscissa de } \begin{bmatrix} M_{0,0} & M_{0,1} & \vec{V}_{0,0}^t & \vec{V}_{0,1}^t \\ M_{1,0} & M_{1,1} & \vec{V}_{1,0}^t & \vec{V}_{1,1}^t \\ \vec{V}_{0,0}^s & \vec{V}_{0,1}^s & \vec{W}_{0,0} & \vec{W}_{0,1} \\ \vec{V}_{1,0}^s & \vec{V}_{1,1}^s & \vec{W}_{1,0} & \vec{W}_{1,1} \end{bmatrix},$$

onde

$$\text{abscissa de } \vec{V}_{s,t}^s = \frac{dx}{ds} \Big|_{s,t}$$

$$\text{abscissa de } \vec{V}_{s,t}^t = \frac{dx}{dt} \Big|_{s,t}$$

$$\text{abscissa de } \vec{W}_{s,t} = \frac{dx}{dsdt} \Big|_{s,t}$$

Os pontos $M_{0,0}$, $M_{1,0}$, $M_{0,1}$ e $M_{1,1}$ na matriz acima são os cantos do retalho, os vetores \vec{V}^s e \vec{V}^t na matriz definem os planos tangentes a superfície e \vec{W} a torção da superfície nos cantos do retalho.

Na forma de Bezier a matriz C_x é dada por $C_x = M_b P_x M_b^T$ onde M_b é a matriz de Bezier (7.3) e P_x a matriz das abscissas dos dezesseis pontos de controle definindo a geometria (forma) da superfície

$$P_x = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \\ P_{41} & P_{42} & P_{43} & P_{44} \end{bmatrix},$$

(ver figura 7.1).

A continuidade na fronteira entre dois retalhos de Bezier consecutivos é obtida igualando os pontos de controle na fronteira. A continuidade das derivadas de primeira ordem na fronteira, é obtida através do alinhamento dos pontos de controle, situados na fronteira e de cada lado dela, e conservando uma razão constante entre o comprimento dos segmentos, de cada lado da fronteira (ver figura 7.2).

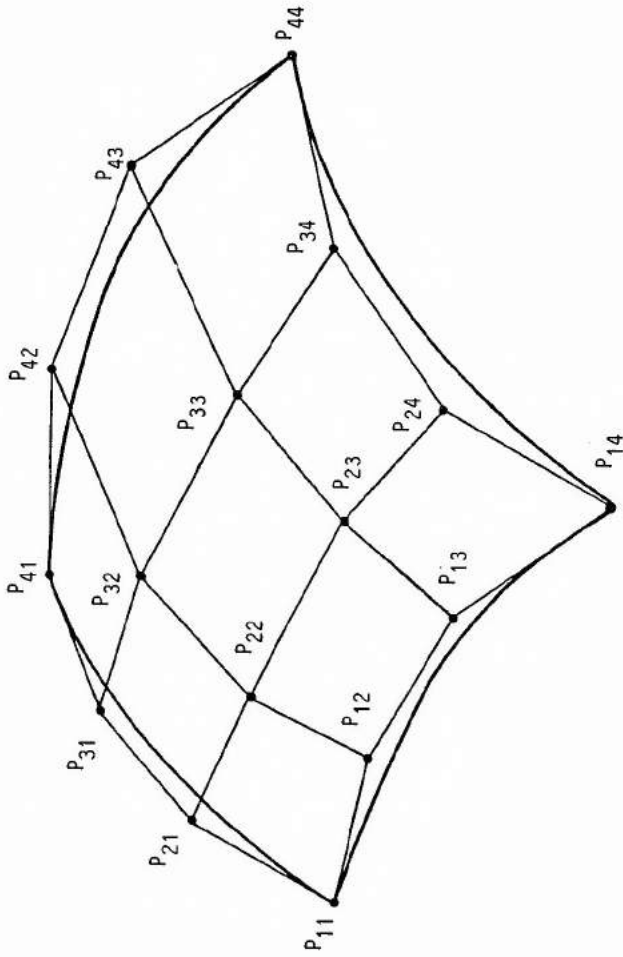


Figura 7.1 Os 16 pontos de controle definindo o retalho bicúbico de Bezier.

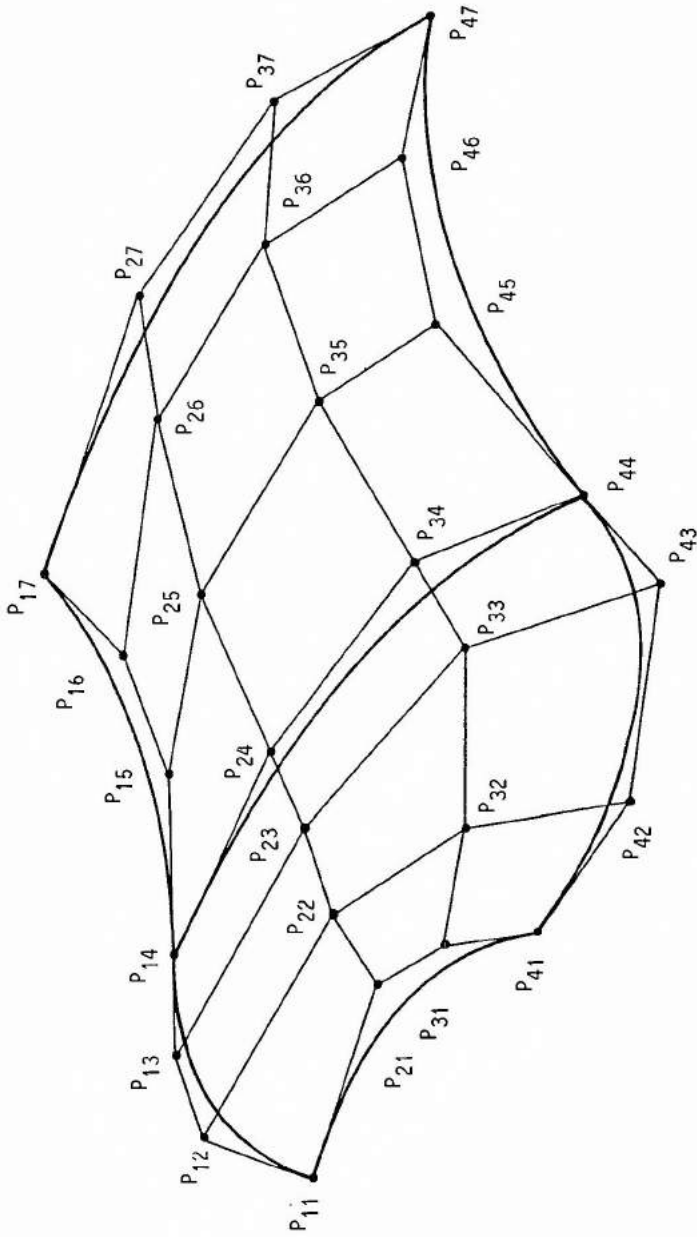


Figura 7.2 Os 26 pontos de controle definindo dois retalhos bicúbicos de Bezier consecutivos (P_{14} , P_{24} , P_{34} e P_{44} definem a fronteira entre os dois retalhos).

Na forma B-spline a matriz C_x é dada por $C_x = M_s P_x M_s^T$, onde M_s é a matriz B-spline (7.4) e P_x a mesma matriz de geometria que no caso da forma de Bezier. As propriedades de continuidade assinalada para as curvas estendem-se para as superfícies B-spline e geralmente a superfície não passa exatamente pelos pontos de controle.

Em computação gráfica, as formas de Bezier e da B-spline são bem mais adaptadas para uma definição interativa através da escolha dos pontos de controle da superfície a sintetizar usando dispositivos localizadores (cf. seção 1.4).

8 - SOBREAMENTO E COR

8.1 - MODELOS DE SOMBREAMENTO

Nos capítulos 5 e 6 foram vistas técnicas para eliminar segmentos ou polígonos escondidos. Essas técnicas são o primeiro passo para sintetizar imagens com realismo visual. Mas o segundo passo indispensável nessa direção é o sombreamento das superfícies visíveis. A aparência de uma superfície depende do tipo de fontes de luz iluminando a cena, das propriedades da superfície como a textura, a cor, a reflectância, da sua orientação e posição relativa entre o observador, a superfície e as fontes de luz, ou outras superfícies.

A aparência de uma superfície depende do tipo de luz. A luz pode ser pontual ou distribuída. Quando refletida através de uma multitude de superfícies como acontece muitas vezes no mundo real, obtém-se a chamada luz ambiente. Por sua natureza, a luz ambiente, quando refletida numa superfície, produz uma luz cuja intensidade recebida pelo observador é dada por

$$I_a k_a \quad (8.1)$$

onde I_a é a intensidade de luz ambiente, k_a o coeficiente de reflexão de luz ambiente para a superfície considerada.

O modelo de reflexão de luz ambiente é o mais simples. Considerar apenas a presença de luz ambiente é insuficiente para criar o realismo visual. Por exemplo, só com a luz ambiente não seria possível distinguir a aresta comum às duas faces visíveis de um poliedro com o mesmo k_a . Geralmente tem que considerar a presença de fontes pontuais de luz na cena.

Quando a superfície iluminada por uma fonte pontual de luz é fosca então a luz refletida é difusa, i.é., sua intensidade não depende da orientação da superfície em relação ao observador, mas somente da orientação da superfície em relação à fonte de luz.

Com esse modelo, duas superfícies de mesma natureza e paralela, recebendo raios de luzes paralelas, refletem a mesma intensidade e são vistas de forma idêntica. Para diferenciar entre essas superfícies, quando uma é afastada da outra, em relação ao observador, a intensidade recebida deve depender da distância entre a superfície e o observador. A intensidade de luz difusa recebida pelo observador é então dada por

$$I_p k_d < \vec{L}, \vec{N} > / (r+k) \quad (8.2)$$

onde I_p é a intensidade da fonte pontual de luz,
 k_d o coeficiente de difusão-reflexão da superfície,
 \vec{L} o vetor unitário de incidência da luz,
 \vec{N} o vetor unitário normal à superfície,
 r a distância entre a superfície e o observador,
 k uma constante.

($< \vec{L}, \vec{N} > = \cos \theta$; ver figura 8.1).

Quando a superfície não é fosca mas brilhante a luz refletida segue um modelo diferente. Neste caso a reflexão é dita especular e a intensidade da luz recebida pelo observador depende da orientação da superfície em relação à fonte pontual de luz e ao observador:

$$I_p k_s < \vec{R}, \vec{V} >^n / (r+k), \quad (8.3)$$

onde k_s é o coeficiente de reflexão especular,
 \vec{R} o vetor unitário na direção de reflexão,
 \vec{V} o vetor unitário na direção de observação,
 n uma constante da superfície

($< \vec{R}, \vec{V} > = \cos \theta$; ver figura 8.1).

O modelo de reflexão especular acima, o de Phong Bui-Tuong [Bui-Tuong, 1975], é uma boa aproximação mas é baseada em observação empírica e não é um modelo físico. Teoricamente k_s deveria depender de $< \vec{L}, \vec{N} >$.

No caso de um refletor perfeito (espelho) n deveria ser infinito, i.é., o observador recebe a luz refletida somente se ele está

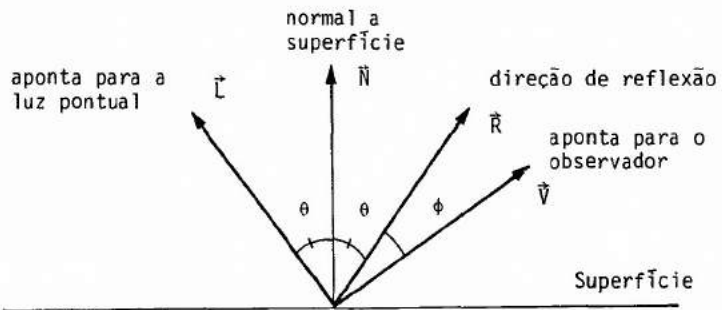


Figura 8.1 As quatro direções do modelo de sombreamento.

situado na direção de reflexão $\vec{R} = \vec{V}$. Quando n é grande (200) a reflexão especular decai muito rapidamente quando \vec{V} afasta-se de \vec{R} . Quando n é pequeno (1) a reflexão especular decai como o coseno do ângulo entre \vec{V} e \vec{R} .

Finalmente no modelo de sombreamento a intensidade da luz refletida por uma superfície (k_a , k_d , k_s , n), nas condições de orientação da figura 8.1, iluminada por uma luz ambiente de intensidade I_a e uma luz pontual de intensidade I_p , e situada a uma distância r do observador é dada por (somando 8.1), (8.2) e (8.3)):

$$I_a k_a + I_p (k_d \langle \vec{L}, \vec{N} \rangle + k_s \langle \vec{R}, \vec{V} \rangle^n) / (r+k). \quad (8.4)$$

Para sintetizar imagens coloridas, tem-se três modelos do tipo (8.4). Um para cada das três cores primárias escolhidas (cf. seção 8.3). A luz ambiente tem a cor da luz pontual iluminando a cena (caso de uma só luz pontual). Quanto aos parâmetros k_a e k_d eles dependem da cor da superfície (somente k_s não depende da cor).

No caso da malha de polígonos já considerada no capítulo 4, três tipos de algoritmos de sombreamento são apresentados rapidamente em seguida.

O algoritmo de sombreamento constante, o mais simples, preenche todos os pontos de um mesmo polígono com o mesmo tom de cinza ou a cor. Isto é válido somente se é possível considerar que a fonte de luz e o observador estão no infinito ($\langle \vec{L}, \vec{N} \rangle$ e $\langle \vec{R}, \vec{V} \rangle$ constantes) e sobretudo que a malha não é uma aproximação da superfície a modelar. Se esta última condição não é verdadeira, i.e., se a malha é realmente uma aproximação, então as facetas visíveis da malha com sombreamento diferentes são reconhecíveis. O pior é que estas diferenças são ampliadas pelo efeito de banda de Mach.

O efeito de Mach é visível ao longo de uma aresta comum a dois polígonos sombreados diferentemente. A vizinhança da aresta do polígono mais escuro aparece mais escuro e a do polígono mais claro aparece mais claro. Esse fenômeno de exageração é causado pela inibição lateral dos receptores do olho cuja resposta depende dos receptores vizinhos. Na

região fronteira, um receptor na zona obscura recebe mais inibição dos receptores vizinhos da zona clara e assim sua resposta encontra-se enfraquecida, um receptor na zona clara recebe menos inibição dos receptores vizinhos da zona obscura e assim sua resposta encontra-se fortalecida.

Se a malha é a aproximação de uma superfície curva então usa-se os algoritmos de sombreado com interpolação da intensidade ou da normal.

Os dois algoritmos consistem em calcular normais as faces visíveis. Sejam $\vec{N}_1, \dots, \vec{N}_p$ as normais relativas às p faces contendo um mesmo vértice, então calcula-se o vetor médio $\frac{1}{p}(\vec{N}_1 + \dots + \vec{N}_p)$ para esse vértice e assim com os outros vértices visíveis.

† No algoritmo de interpolação da intensidade ou algoritmo de Gouraud [Gouraud, 1971] a intensidade resultante da aplicação da fórmula (8.4) é calculada para todos os vértices visíveis usando o vetor médio acima como normal. Cada polígono é sombreado através de uma interpolação linear entre os vértices ao longo de uma aresta e então entre duas arestas ao longo da linha exploratória (cf. seção 6.1).

No algoritmo de interpolação da normal desenvolvido por Phong Bui-Tuong [Bui-Tuong, 1975], em vez de interpolar a intensidade, é a própria normal (i.é., o vetor médio acima) que é interpolado linearmente. O cálculo da intensidade de um ponto sendo efetuado a partir do resultado da interpolação da normal naquele ponto. O resultado visual desse último algoritmo é bem melhor mas ele necessita muito mais computação.

8.2 - DETERMINAÇÃO DE ZONA DE SOMBRA

Para usar corretamente o modelo (8.4) tem que verificarse o raio de luz pontual pode efetivamente atingir o ponto de interesse da superfície, senão tem-se o efeito de sombra. Esse efeito pode ser ignorado somente quando a fonte pontual de luz é situada no próprio ponto de observação, então as sombras projetam-se nas partes invisíveis.

Para saber se um ponto está na sombra pode-se usar os al

goritmos do capítulo 6 para eliminação das superfícies escondidas, pois as superfícies que são visíveis do ponto de observação e, que não o são do ponto de luz, estão na sombra.

Um outro caminho é determinar os polígonos de sombras (estes não dependem do ponto de observação), e usá-los para controlar os algoritmos de sombreamento.

Em seguida é estabelecida a matriz de transformação usada para determinar os polígonos de sombra numa iluminação com uma luz pontual no infinito.

Sejam $M(x,y,z)$ um ponto, e $M'(x',y',z')$ sua projeção num plano de equação $ax + by + cz + d = 0$ paralelamente a uma direção dada por um vetor $\vec{D}(d_x, d_y, d_z)$ (ver figura 8.2). O ponto M' está na interseção da reta de equação paramétrica (cf. seção 3.2), $t \in \mathbb{R}$:

$$\begin{aligned} x' &= td_x + x \\ y' &= td_y + y \\ z' &= td_z + z, \end{aligned} \tag{8.5}$$

e do plano de equação:

$$ax' + by' + cz' + d = 0.$$

Neste ponto de interseção o valor de t é dado por

$$atd_x + ax + btd_y + by + ctd_z + cz + d = 0,$$

seu valor é

$$t = \alpha(ax + by + cz + d), \tag{8.6}$$

onde $\alpha = -1/(ad_x + bd_y + cd_z)$ se $\vec{N}(a,b,c) \perp \vec{D}$.

Assim $M'(x',y',z')$ é dado por (levando (8.6) em (8.5)):

$$\begin{aligned} x' &= x + \alpha axd_x + \alpha byd_x + \alpha czd_x + \alpha dd_x \\ y' &= \alpha axd_y + y + \alpha byd_y + \alpha czd_y + \alpha dd_y \\ z' &= \alpha axd_z + \alpha byd_z + z + \alpha czd_z + \alpha dd_z \end{aligned}$$

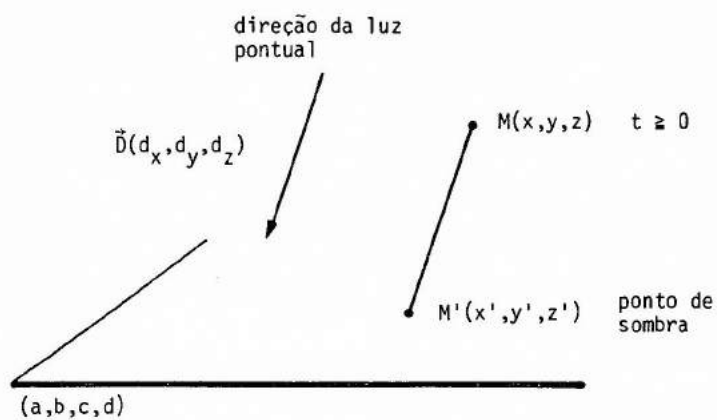


Figura 8.2 Projeção de M num plano paralelamente a direção do vetor \vec{D} .

ou ainda na forma matricial e usando as coordenadas homogêneas dos pontos M e M' (cf. seção 2.2):

$$[x' \ y' \ z' \ 1] = [x \ y \ z \ 1] \begin{bmatrix} 1+\alpha d_x & \alpha d_y & \alpha d_z & 0 \\ \alpha d_x & 1+\alpha d_y & \alpha d_z & 0 \\ \alpha d_x & \alpha d_y & 1+\alpha d_z & 0 \\ \alpha d_x & \alpha d_y & \alpha d_z & 1 \end{bmatrix}$$

i.é.:

$$[x' \ y' \ z' \ 1] = [x \ y \ z \ 1] \left(I + \alpha \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} [d_x \ d_y \ d_z \ 0] \right)$$

onde I é a matriz identidade 4x4.

Se $\vec{N}(a,b,c) \perp \vec{D}$, a matriz P de projeção no plano de coeficiente a,b,c e d paralelamente ao vetor $\vec{D}(d_x, d_y, d_z)$ é:

$$P = I + \alpha \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} [d_x \ d_y \ d_z \ 0]$$

onde $\alpha = -1/\langle \vec{N}, \vec{D} \rangle$

Assim o polígono de sombra de um polígono de vértices M_1, \dots, M_n , num plano de equação $ax + by + cz + d = 0$ paralelamente a \vec{D} é o polígono de vértices M'_1, \dots, M'_n onde

$$[M'_i] = [M_i]P, \quad i=1, \dots, n.$$

O sinal de t para um certo ponto M (dado pela expressão (8.6)) indica a posição relativa entre o ponto M, o plano e a luz. Já que $\vec{MM}' = t\vec{D}$ se \vec{D} aponta para a cena então o ponto M é do mesmo lado do plano que a luz (e assim produz um ponto de sombra) se $t \geq 0$ (ver figura 8.2).


```
n SOMBRAS = 36 =  
I←4 4p0  
(1 1nI)←1  
FACE←0 1 0 0 # PLANO Y=0  
DI←1 -1 2 0 # DIREÇÃO DE ILUMINAÇÃO  
CT←-#FACE+.xDI  
MP←I+CT×FACE+.xDI  
SCASA3VTM←HC(CH CASA3VT)+.x"←MP+.xM  
GSFAIS 'SOLID'  
GFA"CV" C2"(←SCASA3VTM)PLVT" CASA3PLVT  
FV←0<3>"N3"(←CASA3VTM)PLVT"3↑" CASA3PLVT  
GSFACI 8  
GFA"CV" C2"(←CASA3VTM)PLVT" FV/CASA3PLVT  
GSFAIS 'HOLLOW'  
GSFACI 1  
GFA"CV" C2"(←CASA3VTM)PLVT" FV/CASA3PLVT (fig. 35)
```

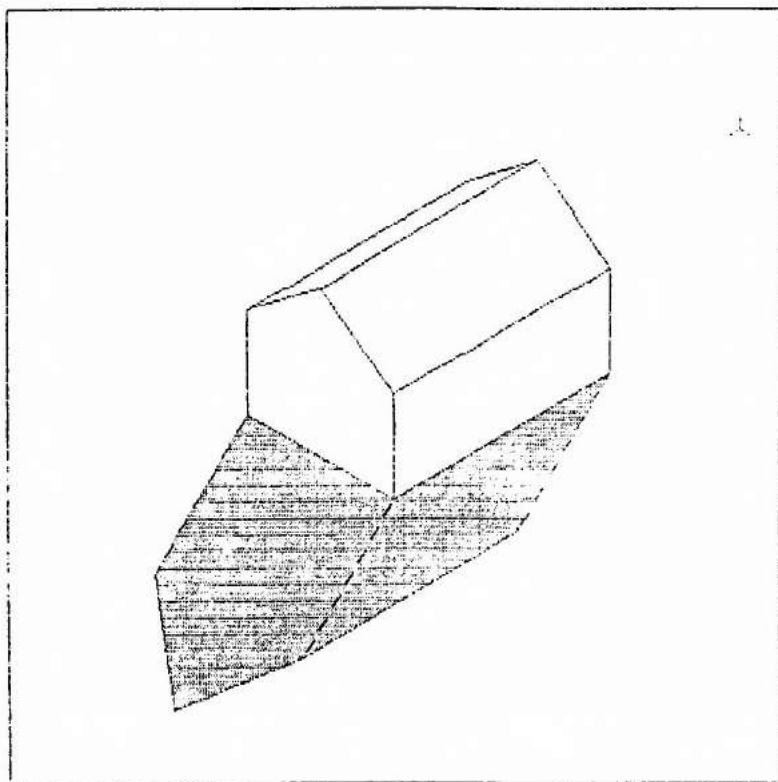


Fig. 35 - As faces visíveis da casa e sua sombra no chão.

No caso da cena com a casa em 3D (cf. seção 3.3), pode-se criar a sombra da casa no chão. Na "transparência" = 36 = o chão é definido como sendo o plano $y = 0$ (que contém a base da casa). Os parâmetros do plano são armazenados na forma de um vetor APL FACE, as coordenadas da direção de iluminação \vec{D} são armazenadas na forma de um vetor APL DI. A constante α é calculada e o resultado armazenado em CT. A matriz de projeção P é calculada e o resultado armazenado na forma de uma matriz MP. Em seguida, determina-se no espaço das coordenadas de visualização (usando a matriz M da "transparência" = 11 =), o vetor SCASA3VTM contendo os vetores de coordenadas dos vértices dos polígonos de sombra. Usando a função GFA os polígonos de sombra são preenchidos. A eliminação das sombras não visíveis (sombra debaixo da casa) é obtida seguindo o mesmo algoritmo que o da "transparência" = 24 = (cf. seção 5.1) para eliminação de linhas escondidas. O resultado final é dado na figura 35 onde aparece a casa na sua posição tradicional e a parte da sombra visível.

Para dar mais realismo visual, as faces visíveis, mas na sombra, são preenchidas com hachuras (ver figura 36). Para determinar estas zonas de sombra visíveis, usa-se aqui o algoritmo de eliminação das faces escondidas das "transparências" = 23 = e = 24 = (seção 5.1) baseado no sinal da coordenada γ do vetor normal as faces. As faces na sombra são as faces invisíveis na projeção paralela cuja direção de projeção e a normal no plano de projeção coincidem com a direção de iluminação.

A "transparência" = 37 = mostra o cálculo da matriz de transformação de normalização (cf. seção 3.3) MI, e o vetor APL CASA3VTMI contendo os vetores de coordenadas (de visualização) dos vértices da casa. A determinação das faces visíveis e das faces iluminadas é obtida a partir dos vetores APL CASA3VTM da "transparência" = 23 = (seção 5.1) e CASA3VTMI respectivamente. Daí calcula-se, finalmente o vetor booleano A das faces visíveis e na sombra (i.é., não iluminadas).

8.3 - LANÇAMENTO DE RAIOS

Nos modelos e algoritmos de sombreamento da seção 8.1 os objetos constituindo a cena eram considerados opacos. Quando os objetos

```
A SOMBRAS                                     = 37 =
      DI←1 1 2 0      A DIREÇÃO DE ILUMINAÇÃO
      MI←NFAR VRP(1↓DI)VUP(1↓DI),FMIN,FMAX
      CASA3VTMI←HC(CH CASA3VT)+.x"←MI
      FV←0(3>"N3"(←CASA3VTM)PLVT"3↑"CASA3PLVT
      FV
1 0 0 1 1 1 0
      FI←0(3>"N3"(←CASA3VTMI)PLVT"3↑"CASA3PLVT
      FI
0 1 0 0 1 1 1
      A←FV^~FI      A FACES VISIVEIS NA SOMBRA
      A
1 0 0 1 0 0 0
      GSF AIS 'HATCH'
      GFA"CV"02"(←CASA3VTM)PLVT"A/CASA3PLVT (fig. 36)
```

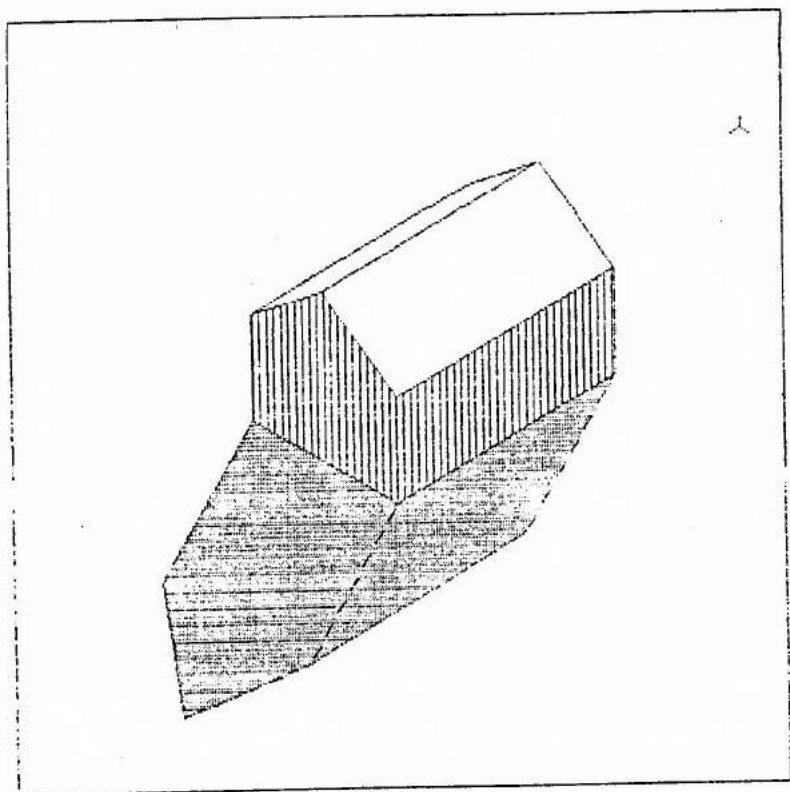


Fig. 36 - As faces visíveis da casa, sua sombra no chão e o sombreamento das faces não iluminadas.

podem ser também atravessados pela luz usa-se geralmente os algoritmos de lançamento de raio ("ray-tracing" em inglês, "lancement de rayon" em francês).

Em material translúcido a transmissão de luz é difusa, mas ainda poucos trabalhos consideraram este caso. Em material transparente a transmissão é dita especular e o efeito de refração tem que ser modelado.

Ao atravessar um contato entre dois meios distintos transparentes de índice de refração n e n' (e.g. ar $n=1$ e vidro $n'=1,5$) o raio de luz é desviado segundo a lei de Snell (para os ingleses) ou de Descartes (para os franceses):

$$n \sin \theta = n' \sin \theta'$$

onde θ e θ' são os ângulos entre o raio de luz e a normal à superfície nos meios de índice n e n' respectivamente (ver figura 8.3).

No algoritmo de lançamento de raio de Whitted [Whitted, 1980] a idéia fundamental é o acompanhamento dos raios de luz através da cena. O acompanhamento é feito ao longo dos raios que passam pelo ponto de observação e pelos quatro cantos dos pontos da imagem até a origem de cada um deles.

Um raio de luz saindo de uma superfície de contato entre dois meios distintos é a contribuição de uma luz difusa, de uma reflexão especular e de uma transmissão especular. Devido à natureza da luz difusa, somente se pode acompanhar os raios da reflexão e transmissão especular. As expressões (8.1) e (8.2) continuam a ser usadas para modelar a reflexão de luz ambiente e a luz difusa.

Os principais passos do algoritmo de lançamento de raio são dados em seguida.

- Para cada canto dos pontos da imagem:

. Construir sua árvore (de cima para baixo) criando-se um nó cada vez

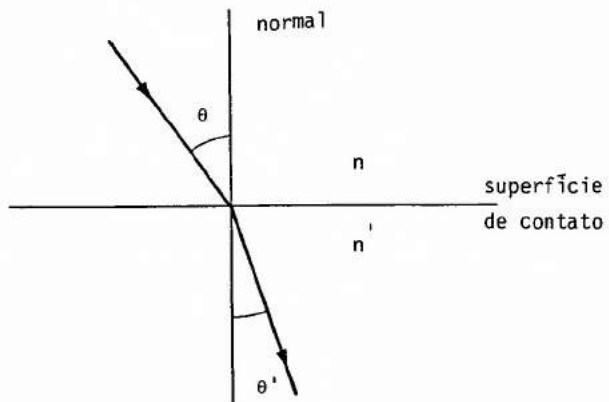


Figura 8.3 Efeito de refração no contato entre dois meios transparentes ($n < n'$).

que um raio (percorrido no sentido contrário à luz) intercepta uma superfície. A cada interseção com uma superfície, calcula-se a direção dos raios incidentes na reflexão e na transmissão especular e prossegue-se o acompanhamento até não haver mais interseção ou até chegar a um número de interseções pré-fixado (ver figura 8.4).

- Calcular sua intensidade a partir da árvore somando (de baixo para cima) as contribuições de cada raio. A cada nó as eventuais contribuições da reflexão da luz ambiente e da luz difusa são também somadas.
- Para cada ponto da imagem calcular sua intensidade média a partir da intensidade dos seus quatro cantos.

Os algoritmos de lançamento de raio necessitam geralmente de muito tempo de computação.

8.4 - TOM DE CINZA E COR

Para visualizar uma imagem digital em preto e branco, a cada valor numérico dos pontos elementares é associado um nível de intensidade ou tom de cinza. Geralmente o valor 0 é associado ao preto, 1 ao branco e tons de cinza cada vez mais claros para valores numéricos crescentes entre 0 e 1.

Dependendo da aplicação o número de tons de cinza (i.é. a resolução radiométrica) necessário pode ser de 2 (2^1) (i.é. o preto e o branco) ou até de 256 (2^8). Quando o número de tons de cinza é insuficiente pode aparecer os chamados falsos contornos na visualização de imagens representando formas suaves. A partir de 64 (2^6) tons de cinza o problema dos falsos contornos já começa a ser bem menos frequente.

Quando o equipamento de saída comporte menos tons de cinza do que seria necessário para visualizar uma imagem, por exemplo sem o problema dos falsos contornos, e se a resolução espacial o permite, então uma alternativa é a técnica dos meios-tons ("half-toning") corriqueira na reprodução de fotografias em jornais ou livros (ver fotografia 8.1). Tra

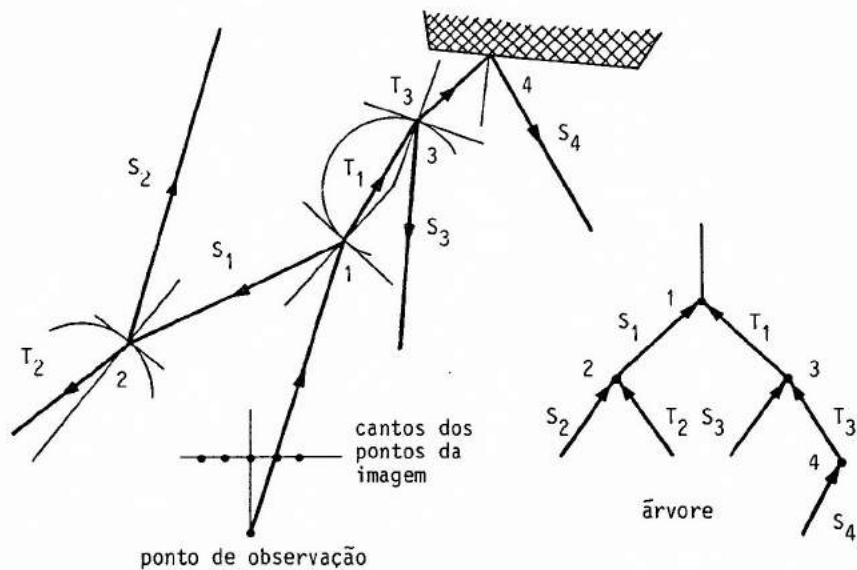
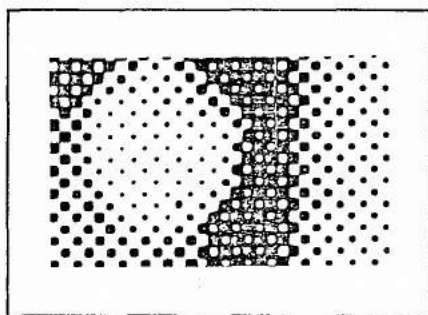


Figura 8.4 Construção da árvore de um canto acompanhando os raios de luz através da cena.



Fotografia 8.1 Amostra de uma imagem reproduzida por meios-tons (ampliação dez vezes). A superfície dos pontos pretos é proporcional ao inverso do nível de intensidade.

dicionalmente os meios-tons são obtidos pela justaposição de pequenos pontos físicos de tamanho variável. Um tom de cinza claro é obtido com pontos (preto) pequenos e um tom de cinza escuro com pontos grandes. Se a distância entre dois pontos vizinhos é pequena em relação à distância de observação então a capacidade de integração do olho permite que se crie a impressão visual de continuidade de tons de cinza.

Essa técnica dos meios-tons pode ser implementada com monitores a varredura. Se o equipamento de saída comporta somente o branco e o preto, cria-se uma escala de tons de cinza com cinco níveis de intensidade agrupando quatro pontos vizinhos para representar um ponto da imagem original (ver figura 8.5). Com padrões binários de 3x3 é possível representar dez níveis de intensidade. Na prática tem-se o compromisso entre a resolução espacial (número de pontos para representar a imagem original) e a resolução radiométrica (número de tons de cinza).

Para visualizar uma imagem digital em cores associa-se a cada valor numérico (ou sequência de valores) de um ponto elementar uma cor. Essa associação pode ser feita de muitas maneiras diferentes.

Fisicamente é a distribuição espectral de energia eletromagnética (entre 400 nm para o violeta e 700 nm para o vermelho) que dá origem a sensação de cores, mas uma infinidade de distribuições pode ser a origem de uma mesma cor.

Experiências em colorimetria mostram que as misturas aditivas (e.g. projeção de luzes coloridas na mesma área de uma tela branca), ou substrativas (mistura de tintas coloridas iluminadas por uma luz branca) de cores produzem novas cores.

A mistura aditiva de duas cores podem dar por exemplo o branco, neste caso estas cores são ditas complementares. Uma cor é pura quando sua distribuição espectral de energia eletromagnética reduz-se a um pico de energia na altura de um certo comprimento de onda. A cor branca teórica tem uma distribuição espectral uniforme no visível. O comprimento de onda dominante de uma cor que corresponde à noção subjetiva de

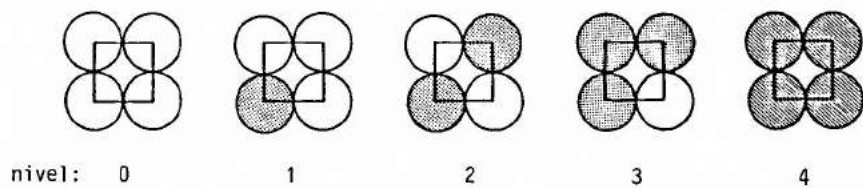


Figura 8.5 Os cinco níveis de tons de cinza com padrões binários 2x2.

matiz é o comprimento de onda da cor pura que misturada aditivamente com o branco teórico produz aquela cor. A pureza ou saturação de uma cor é a percentagem de cor pura na mistura aditiva com o branco teórico que produz aquela cor. Assim, o branco tem uma pureza de 0% e uma cor pura uma de 100%. A intensidade de uma cor que corresponde à noção de brilho, é proporcional ao total de energia dada pela área de sua distribuição espectral. A intensidade, saturação e matiz são as três grandezas que definem uma cor.

Para formar uma palheta de cores pode-se associar em proporções variadas duas ou mais cores pré-fixadas chamadas de componentes primários. Na realidade para formar a palheta de todas as cores possíveis precisaria tomar como componentes primários todas as cores puras do espectro eletromagnético. Na prática observa-se que escolhendo somente três componentes primárias, uma no vermelho, uma no verde e outra no azul, pode-se formar através de uma combinação aditiva uma palheta de cores muito completa (quanto mais ampla que as componentes primárias serão puras). É esta propriedade que é usada nos monitores coloridos cuja palheta de cor é obtida a partir da excitação de fósforo vermelho, verde e azul.

No caso das plotadoras, as imagens coloridas são obtidas a partir de uma combinação substrativa das componentes primárias: ciano, magenta, amarelo.

Para representar numericamente uma imagem colorida, a cada nível de intensidade de cada componente primária dos pontos da imagem é associado um valor numérico. Geralmente o valor 0 é associado ao nível de intensidade zero, 1 ao de intensidade máxima, e valores crescentes para intensidades crescentes. Dependendo das aplicações, o número de níveis de intensidade necessário pode ser de 2 (2^1) ou de até 32 (2^5) para cada componente primária. No primeiro caso isto corresponde a uma palheta de 8 (2^3) cores (incluindo o preto e o branco), no segundo de 32.768 (2^{15}) cores. Uma palheta de 256 cores já reproduz boas imagens coloridas (ainda com risco de falsos contornos). Neste caso 8 (2^3) níveis de intensidade são usados para cada uma de duas componentes primárias (e.g. vermelho e verde) e 4 (2^2) níveis para a terceira componente (e.g. azul) de maneira

a completar as 256 ($2^3 \cdot 2^3 \cdot 2^2 = 2^8$) cores.

A palheta de cores obtida por mistura de três componentes primárias pode ser representada pelo cubo unitário, num espaço referenciado por um sistema de coordenadas ortonormal $\{R,G,B\}$, chamado de cubo RGB ("Red", "Green", "Blue") (ver figura 8.6). Associando os níveis de intensidade das componentes primárias vermelho, verde e azul às coordenadas R, G e B respectivamente, um ponto (R,G,B) do cubo corresponde à cor obtida por combinação aditiva das componentes vermelha, verde e azul na proporção R, G, B respectivamente.

Assim o vértice $(0,0,0)$ corresponde ao preto e o vértice $(1,1,1)$ geralmente ao branco. Na diagonal principal (ligando estes dois vértices), onde as três componentes primárias encontram-se em proporções iguais, são repartidas geralmente (i.é., se as três componentes primárias são bem escolhidas) os tons de cinza do preto ao branco. Os vértices $(1,0,0)$, $(0,1,0)$ e $(0,0,1)$ correspondem às cores primárias e $(0,1,1)$, $(1,0,1)$ e $(1,1,0)$ às suas complementárias. O plano $R+G+B = 1$ contém o chamado triângulo de cores de Maxwell cujos vértices são as cores primárias e o centro é o tom de cinza médio.

O modelo RGB pode ser recodificado de maneira a exprimir a palheta de cor com o modelo YIQ usado na retransmissão comercial de sinais de TV a cores:

$$[Y \ I \ Q] = [R \ G \ B] \begin{bmatrix} 0,30 & 0,60 & 0,21 \\ 0,59 & -0,28 & -0,52 \\ 0,11 & -0,32 & 0,31 \end{bmatrix}.$$

A componente Y é a média ponderada das componentes R, G e B. Os pesos foram escolhidos de maneira a tomar em conta as propriedades do olho humano cuja sensibilidade é maior entre o verde e amarelo. Com isto o sistema visual é mais sensitivo às variações da componente Y do modelo YIQ do que nas duas outras componentes. Essa propriedade sugere que menos resolução espacial é necessária para as componentes I e Q do que para a componente Y, o que pode ser útil para certas aplicações.

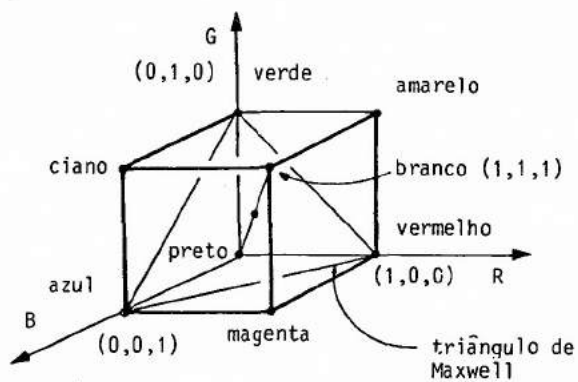


Figura 8.6 Cubo RGB.

Os dois modelos anteriores são ligados à concepção dos equipamentos. O modelo de Smith [Smith, 1978] foi desenvolvido para ajudar a escolha de cores nas aplicações. A palheta de cores é distribuída dentro de um cone com uma diretriz hexagonal situada no espaço referencial por um sistema de coordenadas H, S e V (ver figura 8.7).

O apex do cone é a origem de V, sua diretriz é um hexágono regular situado no plano $V=1$ centrado no eixo V. No hexágono são distribuídas as cores encontradas ao longo da linha poligonal $(1,0,0), (1,1,0), (0,1,0), (0,1,1), (0,0,1), (1,0,1)$ e $(1,0,0)$ do espaço RGB, i.e. nas arestas correspondentes do cubo RGB. O interior do hexágono contém as cores das faces $R=1, G=1$ e $B=1$ do cubo RGB. O hexágono de um plano $V = \alpha$ ($\alpha \in (0,1)$) contém as cores repartidas nas três faces $R = \alpha, G = \alpha$ e $B = \alpha$ de um subcubo RGB.

Uma cor no hexágono é definida em termo da coordenada angular H tomando como origem ($H=0$) o vermelho e como sentido de rotação positiva o sentido anti-horário para um observador situado no eixo V e olhando para a origem. H controla a matiz. Pode se verificar que duas cores no hexágono cuja diferença de H é de 180° são complementárias. O centro do hexágono do plano $V=1$ é branco.

A coordenada V controla o nível de intensidade da cor. As cores no plano $V=1$ tem o nível máximo.

A coordenada S controla a saturação, i.e., a proporção de cor saturada (neste modelo baseado em três cores primárias, a saturação não corresponde à noção de pureza). Num determinado plano $V=\alpha$, ao longo de uma determinada reta $H=\gamma$ ($\gamma \in [0,2\pi]$), a coordenada S vale 1 na interseção com o cone (e corresponde à cor saturada) e 0 na interseção com o eixo V (e corresponde a um tom de cinza).

No apex ($V=0$) encontra-se o preto (S e H podem ser qualquer).

Assim as três componentes H, S e V permitem escolher fácil

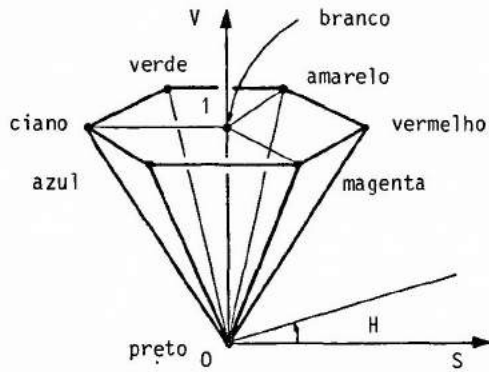


Figura 8.7 Cone hexagonal HSV.

mente uma cor na palheta RGB em termo de matiz (H para "Hue"), saturação (S para "Saturation") e intensidade (V para "Value").

APL2 DV_1

```
[0] Z←L DV_1 R;DIO;DCT;A;B;EDGE;DH
[1] A R←(¯3 1)(0 3)
[2] A XMIN←¯2
[3] A XMAX←2
[4] A YMIN←¯2
[5] A YMAX←2
[6] A JANELA←YMAX YMIN XMAX XMIN
[7] DIO←1
[8] A←¯L∘“(cJANELA)CODE”R
[9] Z←R
[10] →(A/A)/0
[11] EDGE←JANELA[L]
[12] DCT←1E¯13
[13] B←(2(L)Φ”R
[14] B←(2(L)Φ((1 1>B)+(EDGE-1 2>B)x+/+¯/B),EDGE
[15] Z←(A/R),cB A: Z←(¯3 1)(¯1.5 2)
```

APL2 ENVOLTORIO

```
[0] Z←ENVOLTORIO R;DIO;A;DH
[1] A: P1←¯4 0
[2] A: P2←¯4 5
[3] A: P3←5 0
[4] A: P4←4 0
[5] A: R←P1 P2 P3 P4 A: R←(¯4 0)(¯4 5)(5 0)(4 0)
[6] A←R∘“(+/R)÷pR A: A←(¯4.25 ¯1.25)(¯4.25 3.75)
[7] A←120+/(c-9 11)0”A
[8] DIO←1
[9] Z←R[⊕A] A: Z←(¯4 5)(5 0)(4 0)(¯4 0)
```

APL2 LDVT

```
[0] Z←L LDVT R;DH
[1] A: L←(1 1)(3 1)(3 3)(2 4)(1 3)
[2] A: CASA2LD←(1 2 1)(2 3 1)(3 4 2)(4 5 2)(1 5 1)
[3] A: DIO←1
[4] A: R←2>CASA2LD A: R←2 3 1
[5] DIO←1
[6] Z←(2↑R)∘”cL A: Z←(3 1)(3 3)
```

APL2 NPAR

```
[0] Z←NPAR R;DIO;DH
[1] a: VRF←1 1 4
[2] a: VFN←1 1 1
[3] a: VUF←2 2 1
[4] a: DOP←1 1 5
[5] a: UMIN←1
[6] a: UMAX←20
[7] a: VMIN←1
[8] a: VMAX←20
[9] a: FCP←160
[10] a: BCF←1
[11] a: R←VRF VFN VUF DOP UMIN VMIN FCP UMAX VMAX BCF
[12] DIO←1
[13] Z←(T3←R)+.x(RRR R[13])+.x(TRL)+.x(SHPAR R[14])
[14] Z←Z+.x(TPAR R[5 6 7])+.xSPAR R[4+16]
```

APL2 N3

```
[0] Z←N3 R;DIO;A;B;C;D;DH
[1] a: R←(1 0 0)(0 1 0)(0 0 1)
[2] DIO←1
[3] A←B+C+D←→R
[4] A[;1]←1
[5] B[;2]←1
[6] C[;3]←1
[7] Z←1 1 1 1←1×DET"A B C D"
```

APL2 D3

```
[0] Z←D3 R;DH
[1] a: R←(1 0 0)(0 1 0)
[2] Z←3↑N3 R,c←0 0 0 a: Z←0 0 1
```

APL2 PLLDVT

```
[0] Z←L PLLDVT R;L1;L2;DH
[1] a EXEMPLE: CASA3VT CASA3LD PLLDVT 5→CASA3PLLD
[2] L1←1→L
[3] L2←2→L
[4] Z←(R<0)φ"(2↑L2[IR])>"←c←L1
```

APL2 PLVT

```
[0] Z←L PLVT R;DH
[1] a: CASAPL←(1 2 3 4 5)(6 7 8 9 10)(1 2 7 6)(2 3 8 7)
[2] a: DIO←1
[3] a: R←2⇒CASAPL a: R←6 7 8 9 10
[4] DIO←1
[5] Z←L[R] a: Z←(0 0 30)(16 0 30)(16 10 30)(8 16 30)
```

APL2 RJ

```
[0] Z←RJ R;DH
[1] a: R←((-6 2)(2 5))((-4 0)(4 5))((-5 -2)(-1 1))
[2] a: XMIN←-3
[3] a: XMAX←3
[4] a: YMIN←-3
[5] a: YMAX←3
[6] a: JANELA←YMAX YMIN XMAX XMIN a: JANELA←3 -3 3 -3
[7] Z←(~/~/↑↑^//((cJANELA)CODE""R)/R
```

APL2 RR

```
[0] Z←RR R;DIO;RX;RY;RZ;DH
[1] a: R←(1 1 4)(2 2 3)(3 3 3)
[2] DIO←1
[3] RZ←(-↑-/R[2 1])÷DST R[1 2]
[4] RX←03(↑-/R[2 1])(↑-/R[3 1])
[5] RX←RX÷DST(0 0 0)RX
[6] RY←03 RZ RX
[7] Z←→[1]RX RY RZ
[8] Z←(Z,0),[1]0
[9] Z[4;4]←1
```

APL2 RRR

```
[0] Z←RRR R;DH
[1] a: R←VRP VPN VUP a: R←(1 1 4)(1 1 -1)(2 2 -1)
[2] DIO←1
[3] Z←RR R[1]+(c3/0),R[2 3]
```

AFL2 SHPAR

```
[0] Z+SHPAR R;DIO;AB;DOP;DH
[1] a: R+VRP VFN VUP DOP a: R+(1 1 4)(1 1 -1)(2 2 -1)
[2] DIO+1
[3] DOP+↑HC(CH R[4])+.x"e"(RRR R[13])+.xTRL
[4] AR←DOP[1 2]+DOP[3]
[5] Z+4 4p0 a: Z+4 4p0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[6] (1 1&Z)+1 a: Z+4 4p1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1
[7] Z[3;1 2]+AR
```

AFL2 SPAR

```
[0] Z+SPAR R;DIO;DH
[1] a: UMIN←1
[2] a: UMAX+20
[3] a: VMIN←1
[4] a: VMAX+20
[5] a: FCP←60
[6] a: RCP←1
[7] a: R+UMIN VMIN FCP UMAX VMAX RCP
[8] DIO+1
[9] Z+S3:(R[4]-R[1])(R[5]-R[2])(R[6]-R[3])
```

AFL2 TPAR

```
[0] Z+TPAR R;DH
[1] a: UMIN←1
[2] a: VMIN←1
[3] a: FCP←60
[4] a: R+UMIN VMIN FCP a: R←1 -1 -60
[5] Z+T3-R a: Z+4 4p1 0 0 0 0 1 0 0 0 0 1 0 1 1 60 1
```

AFL2 TRL

```
[0] Z+TRL;DH
[1] Z+4 4p0
[2] DIO+1
[3] (1 1&Z)+1
[4] Z[3;3]+1
```


REFERÊNCIAS

- APL2, 1985: "APL2 Programming Language Reference", Manual IBM nº SH 20-9227-0, 1985.
- BOUKNIGHT, 1970a: Bouknight, W.J. e K.C. Kelly, An Algorithm for Producing Half-tone Computer Graphics Presentations with Shadows and Movable Light Sources, SJCC 1970, AFIPS Press, Montvale, N.J., pp. 1-10.
- BOUKNIGHT, 1970b: Bouknight, W.J., "A Procedure for Generation of Three-dimensional Half-toned Computer Graphics Representations", Communications of th ACM, 13(9), setembro 1970, pp. 527-536.
- BRESENHAM, 1965: Bresenham, J.E., "Algorithm for Computer Control of Digital Plotter", IBM Syst. J., 4(1) 1965, pp. 25-30.
- BUI-TUONG, 1975: Bui-Tuong, Phong, "Illumination for Computer-Generated Pictures", Communications of the ACM, 18(6), junho 1975, pp. 311-317.
- FOLEY, 1984: Foley, J.D. e A. Van Dam, Fundamentals of Interactive Computer Graphics, Addison-Wesley Pub. Co., Reading, Massachusetts, 1984.
- GKS, 1984: "Draft Proposed - American National Standard Graphical Kernel System", Computer Graphics ACM SIGGRAPH, fevereiro 1984.
- GOURAUD, 1971: Gouraud, H., "Continuous Shading of Curved Surfaces", IEEE Transactions on Computers, C-20(6), junho 1971, pp. 623-628.
- HOPGOOD, 1983: Hopgood, F.R.A., et. al., "Introduction to the Graphical Kernel System GKS", Academic Press, London, 1983.
- NEWELL, 1972: Newell, M.E., R.G. Newell e T.L. Sancho, "A New Approach to the Shaded Picture Problem", Proc. ACM Nat. Conf., 1972, p. 443.
- PITTEWAY, 1980: Pitteway, M. e D. Watkinson, "Bresenham's Algorithm with Grey Scale", Communications of the ACM, 23(11), novembro 1980, pp. 625-626.
- SMITH, 1978: Smith, A.R., "Color Gamut Transform Pairs", SIGGRAPH'78 Proceedings, publicado como Computer Graphics, 12(3), agosto 1978, pp. 12-19.

- SUTHERLAND, 1963: Sutherland, I.E., SKETCHPAD: A Man-Machine Graphical Communication System, SJCC 1963, Spartan Books, Baltimore, Md. p. 329.
- SUTHERLAND, 1974a: Sutherland, I.E. e G.W. Hodgman, "Reentrant Polygon Clipping", Communications of the ACM, 17(1), janeiro 1974, pp. 32-42.
- SUTHERLAND, 1974b: Sutherland, I.E., R.F. Sproull e R.A. Schumacker, "A Characterization of Ten Hidden-Surface Algorithms", Computing Surveys, 6(1), março 1974, pp. 1-55.
- WARNOCK, 1969: Warnock, J., A Hidden-Surface Algorithm for Computer Generated Half-Tone Pictures, Univ. Utah Computer Sci. Dept., TR 4-15, 1969, NTIS AD-753 671.
- WATKINS, 1970: Watkins, G.S., A Real-Time Visible Surface Algorithm, Univ. Utah Computer Sci. Dept. UTEC-CSC-70-101, junho 1970, NTIS AD-762 004.
- WEILER, 1977: Weiler, K. e P. Atherton, "Hidden Surface Removal Using Polygon Area Sorting", SIGGRAPH'77 Proceedings, publicado como Computer Graphics, 11(2), verão 1977, p. 214.
- WHITTED, 1980: Whitted, T., "An Improved Illumination Model for Shaded Display", Communications of the ACM, 23(6), junho 1980, pp. 343-349.
- WYLIE, 1967: Wylie, C., G.W. Rommey, D.C. Evans e A.C. Erdahl, Halftone Perspective Drawings by Computer, FJCC 1967, Thompson Books, Washington, D.C., pp. 49-58.

INDICE

	<u>Pág.</u>
Abscissa (de um ponto)	32
AC	203
ACM	03
Afina (transformação)	34
"Aliasing"	151
ANSI	04
"Anti-aliasing"	154
Apontador (negativo)	90
- (positivo)	89
Aresta (de um poliedro)	103
Atributo (de uma primitiva)	17
Base (natural)	32
Bezier (forma de)	157
Bi-cúbico (retalho)	155
Branca (cor - teórica)	195
B-spline (forma)	165
CAD/CAM	01
Cartesianas (coordenadas)	32
Campo (de visualização)	17
Cerceamento	17/113
CERCEAMENTO	203
Chave	15
CODE	203
Coerência (espacial)	132
Complementares (cores)	195
Componente (primária)	197
Comprimento (de onda dominante de uma cor)	195
Contorno (falso)	192
Convexo (poliedro)	103
- (polígono)	94
Coordenadas (cartesianas)	32

	<u>Pág.</u>
- (do dispositivo)	131
- (do dispositivo normalizadas)	17
- (homogêneas para 2D)	33
- (homogêneas para 3D)	57
- (universais)	17
- (de visualização)	17
Cor (branca teórica)	195
- (comprimento de onda dominante de uma)	195
- (intensidade de uma)	197
- (pura)	195
- (pureza de uma)	197
- (saturação de uma)	197
CORE	3
Cores (complementares)	195
Cubo (RGB)	198
Diádica (função)	21
Difusa (luz)	177
- (transmissão)	190
Dispositivo (lógico de entrada)	13
- (sônico)	14
Distância (algébrica)	66
DV	203
DV_1	204
Efeito (de banda de Mach)	180
Eixo (de um sistema de coordenadas)	32
ENVOLTÓRIO	204
Equação (bi-cúbica de um retalho)	172
- (de um plano)	68
- (de uma reta)	66
- (paramétrica de uma reta)	64
Escalonamento (2D)	35
- (3D)	57
Espaço (da imagem)	131
- (do objeto)	131

	<u>Pág.</u>
- (orientado)	59
Especular (reflexão)	178
- (transmissão)	190
Evaluator	13
Extensão (de um objeto)	137
Exterior (semi-espaço)	103
- (semi-plano)	94
Face (de um poliedro)	103
Falso (contorno)	192
Função (de entrada)	13
- ("escolha")	13
- ("pick")	13
- ("string")	13
GKS	03/16
Grade (pontos da)	04
Hermite (forma de)	156
Homogêneas (coordenadas para 2D)	33
- (coordenadas para 3D)	57
Homotetia	35
Impressora	06
Intensidade (de uma cor)	197
Interior (semi-espaço)	103
- (semi-plano)	94
Interpolação (algoritmo de - da intensidade)	181
- (algoritmo de - da normal)	181
ISO	03
Isométrica (projeção)	76
Janela	17
"Joystick"	14
LDVT	204
"Light pen"	15

	<u>Pág.</u>
Linha (exploratória)	132
Localizador	13
Luz (ambiente)	177
- (difusa)	177
Mach (efeito de banda de)	180
Malha (de polígonos)	85
Mão (sistema de coordenadas da - direita)	60
- (sistema de coordenadas da - esquerda)	60
Matiz	197
Matriz (de coordenadas)	32
- (ortogonal)	36
- (de transformação)	34
Maxwell (triângulo de cores de)	198
Meios-tons	192
Memória (de restauração)	07
Mesa (digitalizadora)	09
Método (de Robert)	103
Métrica (transformação)	36
Módulo (de um vetor)	31
Moiré	154
Monódica (função)	21
Monitor (a varredura)	07
- (vetorial)	07
"Mouse"	14
Negativo (apontador)	90
- (semi-plano)	68
Normal	64
Normalização (transformação)	17
NPAR	205
N3	205
Ôblica (projecção)	70
Operador (APL)	24
Ordenada (de um ponto)	32

	<u>Pág.</u>
Orientação (de um sistema de coordenadas em 2D)	38
- (de um sistema de coordenadas em 3D)	60
- (mesma - de dois sistemas de coordenadas em 2D)	37
- (mesma - de dois sistemas de coordenadas em 3D)	59
Orientado (espaço)	59
- (plano)	38
Ortogonal (matriz)	36
- (sistema de vetores)	32
- (projeção)	70
Ortogonais (vetores)	32
Ortonormal (sistema de vetores)	32
03	205
Painel (de plasma)	11
Plano (equação de um)	68
- (orientado)	38
Plasma	11
PLLDVT	205
PLVT	206
Plotadora	06
Poliedro	103
- (aresta de um)	103
- (convexo)	103
- (face)	103
Polígono (convexo)	94
Polígonos (malha de)	85
"Polyline"	17
Ponto (abscissa de um)	32
- (coordenadas cartesianas de um)	32
- (ordenada de um)	32
Pontos (da grade)	04
Positivo (apontador)	89
- (semi-plano)	68
- (sentido de rotação - em 2D)	38
- (sentido de rotação - em 3D)	59
Potenciômetro	15

	<u>Pág.</u>
Primária (componente)	197
Primitiva	17
Produto (escalário em APL)	24
- (escalário de dois vetores)	32
- (matricial em APL)	24
- (vetorial)	69
Projeção (isométrica)	76
- (ôblica)	70
- (ortogonal)	70
Projetiva (transformação)	33
Pura (cor)	195
Pureza (de uma cor)	197
Reflexão (especular)	178
Refração	190
Reta (equação de uma)	66
- (equação paramétrica de uma)	64
Retalho (bi-cúbico)	155
- (equações bi-cúbicas de um)	172
RGB (cubo)	198
RJ	206
Robert (método de)	103
Rotação	36/58
RR, RRR	206
Saturação (de uma cor)	197
Semi-espaco (exterior)	103
- (interior)	103
Semi-plano (exterior)	94
- (interior)	94
- (negativo)	68
- (positivo)	68
Sentido (de rotação + em 2D)	38
- (de rotação + em 3D)	59
SHPAR	207
SIGGRAPH	03

	<u>Pág.</u>
Sistema (de coordenadas).....	31
- (de coordenadas do dispositivo).....	131
- (de coordenadas do dispositivo normalizadas).....	17
- (eixo de um - de coordenadas).....	31
- (de coordenadas da mão direita).....	60
- (de coordenadas da mão esquerda).....	60
- (orientação de um - de coordenadas).....	38/60
- (de coordenadas universais).....	17
- (de coordenadas de visualização).....	17
- (de vetores).....	31
- (de vetores ortogonal).....	32
- (de vetores ortonormal).....	32
Sistemas (mesma orientação de dois - de coordenadas em 2D).....	37
- (mesmo orientação de dois - de coordenadas em 3D).....	59
Sombreamento (algoritmo de - constante).....	180
SPAR	207
Tecla (programável)	15
Teclado	15
"Touch panel".....	14
"Trackball"	14
Transformação (afina)	34
- (geométrica)	33
- (matriz de)	34
- (métrica)	36
- (de normalização)	17
- (projetiva)	33
Translação (em 2D)	34
- (em 3D)	57
Transmissão (difusa)	190
- (especular)	190
Triângulo (de cores de Maxwell)	198
TPAR	207

	<u>Pág.</u>
TRL	207
Varredura (equipamento a)	04
- (monitor a)	07
Vetor (coordenadas cartesianas de um)	32
- (de coordenadas)	32
- (geométrico)	31
- (módulo de um)	31
Vetores (ortogonais)	32
- (produto escalar de dois)	32
Vetorial (equipamento)	04
- (monitor)	07
- (produto)	69
Visualização (campo)	17
- (volume de)	69
Volume (de visualização)	69
"z-buffer"	142
2D-3D	01



SOCIEDADE BRASILEIRA DE
MATEMÁTICA APLICADA E COMPUTACIONAL