Rui Manuel de Sá Pereira de Lima

"PROCESSAMENTO DE IMAGEM NA ANÁLISE DE DOCUMENTOS DE ENGENHARIA"

Faculdade de Engenharia da Universidade do Porto
1995

Processamento de Imagem na

Análise de Documentos de Engenharia

por

Rui Manuel de Sá Pereira de Lima

Licenciado em Engenharia Electrotécnica e de Computadores pela

Faculdade de Engenharia da Universidade do Porto (1991)

Tese submetida ao

Departamento de Engenharia Electrotécnica e de Computadores da

Faculdade de Engenharia da Universidade do Porto

para satisfação parcial dos requisitos do

Mestrado em Engenharia Electrotécnica e de Computadores

Perfil de Informática Industrial

Orientador: Aurélio J. de Castro Campilho (Professor Auxiliar Agregado)

1995

Agradecimentos

Ao Professor Aurélio J. de Castro Campilho, pela sua orientação no desenvolvimento deste trabalho. Os seus conhecimentos científicos, bem como, a sua permanente motivação, disponibilidade e paciência, ajudaram-me a discernir sobre os rumos a tomar na implementação prática do trabalho e na escrita desta dissertação.

Ao Professor A. Jorge Neves Padilha, pela contribuição para a minha formação científica no campo do processamento e análise de imagem.

A todos os professores e investigadores do grupo de processamento e análise de imagem do Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto, pelo apoio científico, e pela motivação fornecida ao longo da implementação prática deste trabalho.

Ao Centro de Investigação de Engenharia Biomédica do Instituto Nacional de Engenharia Biomédica, pela utilização das suas instalações.

Ao Programa CIÊNCIA da Junta Nacional de Investigação Científica e Tecnológica, pela bolsa concedida.

Aos meus pais, pelo apoio permanente.

À Cristina, pelas correcções efectuadas na fase final desta dissertação; e por TUDO.

SUMÁRIO

Nesta tese apresenta-se uma metodologia que permite analisar documentos de engenharia. Recorrendo a técnicas de processamento de imagem efectua-se uma abordagem a todas as operações incluídas na análise de documentos, desde a digitalização de documentos de engenharia e do processamento inicial das imagens resultantes, passando pela segmentação das diferentes partes do documento separando texto de desenhos de linhas, até ao reconhecimento gráfico destas. Todas as operações foram implementadas num sistema de processamento e análise de imagem, desenvolvido em ambiente Windows.

Descrevem-se processos de aproximação poligonal e aproximação por arcos de circunferência. No reconhecimento gráfico de linhas digitais, introduz-se um método de detecção de cantos utilizando curvas da familía das splines. Finalmente, aborda-se a utilização de um formato de ficheiro com o objectivo de exportar a informação vectorial de desenhos de linhas para um sistema CAD.

ÍNDICE

Capítulo 1

INTRODUÇÃO

1. INTRODUÇÃO

1.1 ÂMBITO

Todas as actividades humanas de produção ou de prestação de serviços exigem a utilização de documentos. Esses documentos são necessários para análise de requisitos e projecto de produtos e de actividades, monitorização de produção, para registos vários e transmissão de informação.

Hoje em dia, a informática é o principal meio de produção de documentos, devido às facilidades que proporciona, de edição, alteração, armazenamento, transmissão, fiabilidade, e de unicidade dos dados. No entanto, e independentemente do fim para que foi produzido, o suporte final da grande maioria dos documentos é o papel. É sob a forma de papel que muitos documentos são transmitidos, e armazenados em locais onde não há acesso à sua forma electrónica. Para além de existirem ainda muitos documentos que não foram, ou não são, produzidos electronicamente.

Existem muitas razões para a conversão de um documento impresso para um documento informático. Os dados do documento podem ser necessários para um processamento computadorizado, como num sistema CAM[1] de produção, ou para cálculos complexos. O documento pode ter que ser alterado ou combinado com outra informação, o que acontece com artigos, propostas, correspondência, ou mesmo projectos como os de engenharia. Finalmente a conversão referida, introduz as facilidades dos sistemas electrónicos no armazenamento, procura e distribuição de informação. Facilidades que são evidentes no armazenamento, consulta e distribuição de informação seleccionada das mais diversas publicações, como livros, revistas, jornais e artigos de legislação, ou ainda na conversão de desenhos para posterior consulta, processamento e armazenamento com recurso a sistemas CAD[2] existentes.

A análise de documentos corresponde ao processamento e análise de imagens de documentos digitalizados para detecção e reconhecimento de objectos nele incluídos. Estes objectos podem ser de vários tipos:

i. caracteres que podem ser reconhecidos individualmente, ou a um mais alto nível, podem ser reconhecidas as palavras e frases formadas por caracteres;

---

[1]Do inglês "Computer Aided Manufacturing"
[2]Do inglês "Computer Aided Design"

ii. linhas que podem ser representadas por primitivas gráficas básicas, ou a um mais alto nível, podem ser reconhecidas as estruturas ou símbolos gráficos complexos formados por essas linhas, dependentes mesmo do contexto em que estão inseridos;
iii. figuras de qualidade fotográfica que podem simplesmente ser detectadas e segmentadas do resto da informação, ou processadas de forma a serem classificadas de uma forma mais detalhada.

Com esta tese pretende-se ilustrar algumas metodologias úteis, para o reconhecimento de alguns dos objectos existentes num documento de engenharia. Concretamente, pretende-se efectuar o reconhecimento gráfico de desenhos de linhas dos documentos, determinando a aproximação de linhas digitais por segmentos de recta e arcos de circunferência. Também se pretende efectuar uma abordagem à aproximação de linhas digitais por curvas da família das splines. Finalmente, pretende-se obter uma descrição dos resultados obtidos sob a forma de um ficheiro compatível com um sistema CAD.

1.2 OBJECTIVOS

Para reconhecer graficamente linhas digitais, contidas em documentos de engenharia, é necessário processar a imagem do documento de forma a obter um mapa de bits que contenha unicamente, ou o mais aproximadamente possível, apenas linhas. A partir desta imagem de desenhos de linhas é possível, obter uma descrição das linhas, e efectuar a sua aproximação por segmentos de recta, por arcos de circunferência, e por splines.

De uma forma sucinta, os objectivos deste tese são:

- o estudo de métodos de reconhecimento gráfico de linhas de desenhos de engenharia;
- a concepção de um processo de análise de documentos, desde a digitalização e processamento inicial, passando pela segmentação das diferentes áreas da imagem do documento, até se obter uma imagem com desenhos de linhas para posterior reconhecimento gráfico;
- a utilização de um formato de ficheiro que permita a exportação das primitivas gráficas para um sistema CAD;
- o desenvolvimento e comparação dos diferentes métodos referidos, e o teste global do sistema.

1.3 Contribuições

Neste trabalho consideraram-se todos os passos inerentes à análise de documentos, desde a aquisição à segmentação de diferentes blocos do documento, podendo por isso, servir de base para a compreensão, de uma forma genérica, dos factores envolvidos nesta área do processamento e análise de imagem. Além disto conseguiu-se estabelecer uma metodologia de análise de documentos de engenharia, que permite segmentar blocos de texto de blocos de desenhos de linhas, podendo posteriormente, efectuar-se o reconhecimento gráfico das linhas assim obtidas.

No reconhecimento gráfico utilizaram-se dois processos de aproximação poligonal, e desenvolveu-se um método de aproximação por arcos de circunferência. Implementou-se ainda, um método de detecção de cantos recorrendo a splines, e abordou-se a interpolação de pontos recorrendo a curvas de Bézier e curvas Spline. Recorreu-se a um formato de ficheiro que permite exportar a informação obtida para um sistema CAD.

Resumindo, neste trabalho conseguiu-se estabelecer um processo, que de uma forma global permite, a partir da imagem adquirida de um documento de engenharia, segmentar os diferentes blocos da imagem, e reconhecer as primitivas gráficas dos desenhos de linhas do documento.

1.4 Estrutura da Tese

Para se ter ideia do trabalho já realizado no âmbito da análise documentos, e das metodologias aplicadas na resolução dos seus problemas específicos, condensou-se essa informação no segundo capítulo. Este é, portanto, um capítulo genérico sobre o processamento de imagem aplicado à análise de documentos.

O terceiro capítulo descreve a estrutura dum sistema de processamento e análise de imagem desenvolvido no âmbito desta tese, e descreve também, sumariamente, a sua implementação e utilização.

No quarto capítulo descrevem-se os processos utilizados na análise de documentos de engenharia, incluindo a sua digitalização, o processamento inicial das imagens, e a segmentação das diferentes partes do documento de forma a obter uma imagem apenas com desenhos de linhas, isto é, sem caracteres de texto.

O reconhecimento de primitivas gráficas de linhas digitais é descrito no quinto capítulo. Neste capítulo, descreve-se o processamento inicial para reconhecimento gráfico das imagens de desenhos de linhas, e os métodos utilizados de aproximação de linhas, nomeadamente de aproximação poligonal, aproximação por arcos de circunferência, e aproximação por splines. Termina-se com a descrição do formato de

ficheiro utilizado para exportação da informação obtida para um sistema CAD, e com o teste global de todo o processo de análise de documentos de engenharia.

No capítulo final tiram-se conclusões sobre o trabalho efectuado e apresentam-se algumas perspectivas de desenvolvimento.

Capítulo 2

ANÁLISE DE DOCUMENTOS

2. ANÁLISE DE DOCUMENTOS

2.1 INTRODUÇÃO

O conteúdo deste capítulo está relacionado com a necessidade de conhecer, tentando dar uma perspectiva genérica, o trabalho já realizado na análise de documentos, e quais as principais metodologias utilizadas, sabendo que muitos dos problemas que aparecerão no desenvolvimento da tese serão comuns a outros autores.

O que é a análise de documentos no âmbito do processamento de imagem ?

A análise de documentos [19] corresponde ao processo de conversão dum documento digitalizado da sua forma de mapa de bits para objectos atómicos que podem ser interpretados e processados por computadores.

Os objectos atómicos, manuscritos ou resultantes de impressão, podem ser texto, gráficos (do tipo financeiro, estatístico, diagrama de blocos, entre outros), desenhos (como por exemplo de engenharia ou arquitectura). Estes objectos podem ser simples esboços ou devidamente normalizados, como podem corresponder a figuras do tipo fotográfico ou similar.

Pode ainda referir-se que a maioria dos trabalhos existentes na análise de documentos podem ser classificados segundo Nadler [28] em várias classes conforme o tipo de documento que é processado, como sejam os de escritório (relatórios, cartas), os tipográficos (manuais, entre outros), os técnicos (por exemplo diagramas de circuitos, desenhos de engenharia, diagramas lógicos), mapas geográficos (cartas topográficas, mapas urbanos, de estradas, políticos), e publicações (jornais, revistas).

2.2 ARQUITECTURA GERAL DUM SISTEMA DE ANÁLISE DE DOCUMENTOS

Um modelo, apresentado por Casey e Wong [19], representa um sistema genérico de análise de documentos. Neste sistema é necessário adquirir a imagem, separar os diferentes blocos nela existentes, classificá-los e efectuar o reconhecimento dos seus objectos atómicos, para posterior armazenamento da informação extraída. As funções principais dos vários componentes ilustrados na figura 1, são, de uma forma sucinta, as seguintes:

- *Digitalização e Binarização*: a digitalização do documento permite obter uma imagem em níveis de cinzento que em seguida será binarizada para posterior

análise e processamento. A imagem obtida desta forma denomina-se *imagem documento*;

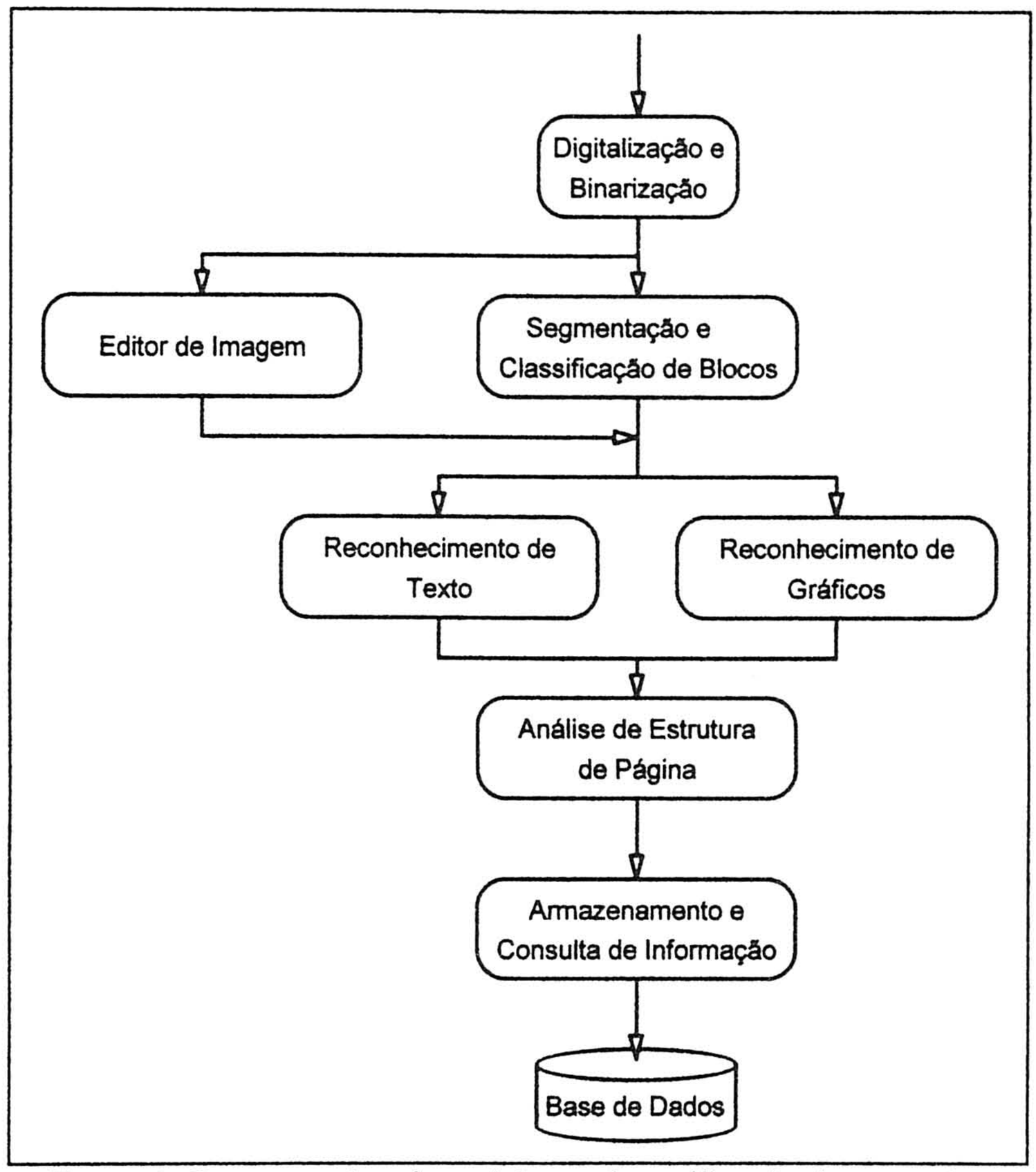


Figura 1: Componentes dum sistema de análise de documentos

- *Segmentação e Classificação de Blocos*: esta é a função de separação automática de blocos da imagem (exemplo apresentado na figura 2), e respectiva classificação como texto, gráficos ou imagens fotográficas (ou similares). Os desenhos técnicos considerados neste trabalho, serão classificados como gráficos ou desenhos de linhas;
- *Editor de Imagem*: permite ao utilizador editar a imagem, classificando os blocos acima referidos de uma forma não automática;
- *Reconhecimento de Texto*: é o processamento de blocos de texto com o reconhecimento individual de caracteres, de palavras ou mesmo de frases;
- *Reconhecimento de Gráficos*: consiste na conversão dos blocos gráficos da imagem documento em símbolos gráficos, numa forma vectorial ou com descrições gráficas de nível superior;

- *Análise de Estrutura de Página*: análise espacial das relações entre diversos blocos para classificar os componentes estruturais duma página, como sejam os cabeçalhos, sumários, parágrafos;
- *Armazenamento e Consulta de Documentos*: criação duma base de dados para armazenamento de documentos, fornecendo uma interface para consulta de informação.

necessário, também pode aplicar no seu corpo uma pomada ou uma loção à base de dietiltoluamida ou DET, que é um repelente eficaz e, em princípio, sem qualquer perigo. Mas algumas pessoas são alérgicas a esse produto e é desaconselhável utilizá-lo em crianças, porque o princípio activo pode passar a barreira da pele e infiltrar-se no sangue. De qualquer modo, não o aplique sobre um corte ou qualquer outra ferida e utilize-o com parcimónia.

Que fazer em caso de picada de insecto?

– Primeiro, lave a região atingida com água e sabão.
– Se o ferrão não sair, retire-o suavemente, recorrendo a uma pinça bem limpa (nunca o faça com as unhas, porque além de poder enterrar mais o ferrão, também corre o risco de perfurar a bolsa de veneno, que, dessa forma, se poderá espalhar).
– Desinfecte a região atingida com eosina, álcool ou outro desinfectante qualquer.

POSIÇÃO LATERAL DE SEGURANÇA (PLS)

A. Coloque o braço da vítima, que se encontra do seu lado, perpendicularmente ao corpo e dobre o outro braço sobre o ventre.

B. Dobre a coxa oposta ao braço estendido; puxe para si, ao mesmo tempo, o braço dobrado e o joelho da vítima, a fim de a deitar de lado.

C. Puxe-lhe a cabeça para trás, com a boca aberta e virada para o chão. Encoste-lhe uma almofada pequena nas costas e ponha outra debaixo da cabeça. Tape a pessoa com um cobertor ou um casaco e vigie o seu estado (temperatura, respiração e pulsação).

Para libertar as vias respiratórias da vítima, é preciso desapertar primeiro tudo o que a possa incomodar (colarinho, gravata, soutien, etc.). Depois, abra-lhe a boca, apoiando as suas mãos na testa e no queixo. Com um lenço limpo, limpe-lhe a cavidade bucal. Puxe-lhe a cabeça para trás, para facilitar a entrada do ar.

– Para acalmar a dor ou a comichão, aplique álcool, água de Colónia, vinagre ou, se nada disto der resultado, uma pomada anti-histamínica (que pode comprar na farmácia).

ATENÇÃO: consulte imediatamente o médico ou vá aos serviços de urgência mais próximos se as picadas forem muitas ou se a picada tiver sido dada na garganta ou dentro da boca, porque, nesse caso, o inchaço pode perturbar a respiração. A pessoa afectada pode chupar cubos de gelo.

Figura 2: Imagem de um documento digitalizado, onde são visíveis os diversos blocos do documento, nomeadamente: os blocos de texto em duas colunas, e sob as figuras; os blocos gráficos ou de desenhos de linhas respeitante à figura localizada no canto inferior esquerdo do documento; os blocos de imagens respeitantes às figuras localizadas na coluna da direita do documento.

Na figura 2, é representada a imagem de um documento digitalizado. Nesta imagem são visíveis os diferentes blocos de um documento, nomeadamente, blocos de texto, blocos de desenhos de linhas (blocos de gráficos) e blocos de imagens. É de notar que

os referidos blocos podem ser envolvidos por rectângulos não sobrepostos, classificando-se a estrutura de blocos do documento como rectangular.

2.3 Digitalização e Binarização

Os documentos são normalmente constituídos por informação a preto num fundo branco, sendo digitalizados utilizando um sensor que converte a intensidade de luz em níveis de cinzento. O aparelho mais utilizado para este fim é o digitalizador de documentos que associa um mecanismo de varrimento ao sensor óptico, podendo o varrimento ser mecânico em aparelhos de mesa ou manual.

A resolução de uma imagem documento é uma característica fundamental na possibilidade de extracção de objectos característicos dessa imagem. O varrimento do digitalizador permite obter pontos do documento, que serão pixels da imagem. O espaço entre pixels, determinado pela resolução de digitalização, estabelece o número de pontos de um objecto do documento que serão lidos para a imagem digitalizada, sendo portanto, uma característica determinante na qualidade da imagem formada, e consequentemente na análise de determinados símbolos ou gráficos mais complexos.

Para reconhecimento de texto Casey e Wong [19] definiram uma regra por experiência própria, que consiste em obter, em largura, duas amostras (pixels) por ponto do caracter, e pelo menos 15 pixels em altura. Isto conduz a uma resolução da imagem de 240 pontos (pixels) por polegada para documentos típicos. Considerando fontes de 12 caracteres por polegada (pitch), cada caracter terá uma largura aproximada de 0.08 polegadas, e considerando 0.2 mm por cada ponto impresso, determina-se a largura aproximada de 10.0 pontos impressos por caracter. Assim sendo, para se obterem duas amostras por ponto do caracter é necessária uma resolução mínima de 240 dpi[1] para reconhecimento de caracteres em documentos típicos, ou para gráficos de complexidade similar. É no entanto referido pelos mesmos autores que para o reconhecimento de texto de uma publicação como a *IEEE Transactions on Information Theory* com letras minúsculas de 1.8 mm de altura e pontos impressos de 0.2 mm é necessário uma resolução de 300 dpi. Para o reconhecimento de linhas em desenhos, supondo uma boa qualidade do desenho e um espaçamento adequado, não é necessária uma tão grande resolução como para o reconhecimento de caracteres e de símbolos, sendo referido pelos mesmos autores uma aplicação da empresa Toshiba para análise de diagramas de

[1] "Dots Per Inch"

circuitos em que se utilizam diferentes resoluções nomeadamente 240 dpi para análise de texto e de símbolos e 80 dpi para o seguimento de linhas.

A binarização de imagem é essencialmente um problema de classificação de pixels. O seu objectivo principal é o de classificar os pixels duma imagem em duas classes: a classe daqueles que pertencem ao objecto e a classe daqueles que pertencem ao fundo. Enquanto uma inclui pixels com nível de cinzento não superiores a um limiar de binarização, a outra inclui pixels com nível de cinzento superior a esse limiar. O método de binarização da imagem documento pode influenciar de uma forma determinante a exactidão da sua análise. A escolha de limiares de binarização incorrectos pode levar à quebra de um maior número de linhas sólidas finas, e ao aparecimento de buracos em objectos grossos cheios, ou ainda ao fecho ou abertura "errada" de linhas e símbolos.

Em determinadas imagens com grande contraste e um fundo uniforme pode-se utilizar um limiar seleccionado *a priori*, embora este seja um método muito pouco utilizado por ter desvantagens óbvias na automatização de análise de documentos diversos, devido às inúmeras diferenças que os caracterizará, tanto em relação ao documento como à imagem resultante da digitalização.

Na maior parte dos casos a imagem documento é caracterizada por uma larga banda de níveis de cinzento, com uma determinada distribuição, que varia dentro do mesmo documento e entre documentos diferentes mas do mesmo tipo, que por maioria de razão varia ainda mais entre documentos de tipos diferentes. Obviamente, a escolha de um limiar de binarização fixo não trará resultados satisfatórios para a segmentação de imagens de documentos variados, sendo necessário recorrer a métodos de selecção automática de limiares de binarização.

Em geral, as técnicas de selecção dum nível de cinzento limiar, para binarização da imagem são divididas em dois grupos, nomeadamente, binarização global e binarização local [33]. Na binarização global actua-se sobre toda a imagem com um único limiar de classificação, separando a imagem em duas classes, conforme os pixels tenham uma intensidade superior ou inferior àquele limiar. Por outro lado, na binarização local a imagem é dividida em sub-imagens determinando-se um limiar independente para cada uma delas, classificando-se assim, duas classes para toda a imagem, mas recorrendo a limiares diferentes em cada sub-imagem. Por outro lado, as técnicas de binarização podem ainda ser classificadas como pontuais ou regionais. Será pontual, se o limiar de binarização for determinado com base apenas nos níveis de cinzento de cada pixel, independentemente das características dos seus vizinhos. Será regional, se o limiar de binarização for determinado com base numa propriedade local (por exemplo, a distribuição local dos níveis de cinzento), baseada na vizinhança do pixel em análise.

Classificam-se, ainda, alguns métodos como adaptativos. Nestes, o limiar de binarização é determinado em cada pixel, em função duma propriedade local. O limiar de binarização vai-se adaptando ao longo da imagem, variando em cada pixel em função da região em está inserido.

Em [21] descreve-se um método adaptativo de binarização, em que cada pixel é classificado como pertecendo a uma de duas classes, se o seu nível de cinzento for superior, ou não, à média dos seus vizinhos, considerados numa janela MxM. Mais concretamente, o pixel central da janela MxM é classificado como pertencendo ao fundo (objecto) se o seu nível de cinzento for superior (não superior) à média dos pixels abrangidos por aquela janela. Por outro lado, em [19] refere-se um método similar ao descrito anteriormente, com a diferença de se considerar um limiar dependente não só da média dos vizinhos do pixel em análise, mas também de um operador de gradiente que deve ser sensível às orlas de caracteres, mas insensível a ruído e a fundos de meio tom[2]. Ou seja, um pixel é classificado como pertencendo ao objecto se o seu nível de cinzento não for superior à soma da média dos pixels vizinhos com um valor de gradiente, produzindo assim uma imagem com contornos mais definidos.

2.4 Segmentação e Classificação de Blocos

Fundamentalmente utilizam-se dois processos para segmentar e classificar blocos, um processo descendente[3] e um processo ascendente[4]. No primeiro, o documento é separado em grandes blocos que são analisados para se efectuar a sua classificação como blocos de texto, de gráficos ou de desenhos de linhas. O processo ascendente separa o documento em pequenos blocos, que são analisados individualmente, sendo depois agregados em blocos maiores, todos com as mesmas características.

É possível desta forma, separar várias partes da imagem, podendo ser processadas de diferentes formas. Esta abordagem permite a extracção das características de cada um desses blocos, muito diferentes entre si. Ou seja, consegue-se reconhecer o texto, extrair características gráficas dos desenhos de linhas descrevendo as suas formas, e armazenar ou processar individualmente as imagens.

---

[2]Do inglês "half-tone"
[3]Do inglês "top-down"
[4]Do inglês "bottom-up"

2.4.1 Segmentação de Blocos

Em seguida descrevem-se vários métodos de separação dos blocos duma imagem apresentados na literatura, utilizando uma das duas abordagens ascendente ou descendente, indicadas anteriormente.

Alguns destes métodos recorrem à noção de componentes ligados, considerando desta forma uma primeira separação, e depois agrupando os componentes nos blocos que melhor os caracterizem.

Outros métodos recorrem à distribuição espacial dos pixels brancos e pretos para reconhecer diferentes áreas da imagem e assim proceder à sua separação.

2.4.1.1 Etiquetagem de Componentes

A etiquetagem de componentes é um método [5] muito vulgarizado, que detecta objectos formados por pixels ligados, e efectua a sua agregação para segmentação ascendente de blocos [19, 20], recorrendo a processos de classificação referidos na próxima secção. Analisando a imagem linha a linha coloca-se uma etiqueta no primeiro pixel preto (considerando-o como objecto), e em todos os outros que lhe estiverem ligados, atribuindo-se outras etiquetas a outros pixels pretos ligados entre si, mas desligados das anteriores regiões. Nas linhas seguintes procede-se da mesma forma, com a excepção de se verificar se na linha anterior existem pixels do objecto que estejam ligados a este, dando-se neste caso a mesma etiqueta. Quando se encontram dois pixels com etiquetas diferentes, colocam-se essas etiquetas numa tabela de equivalências, prosseguindo-se apenas com uma delas. No final da imagem todos os pixels são classificados com uma determinada etiqueta, pertencendo a um objecto ligado, que pode ser composto por pixels com etiquetas diferentes, mas obrigatoriamente equivalentes (a tabela de equivalências associa pixels com etiquetas diferentes que estejam ligados). Um objecto conexo, corresponde a um conjunto de pixels conexos do objecto, com uma mesma etiqueta ou outra equivalente.

Rosenfeld e Kak [5] consideram um caminho como uma sequência de pixels pertencentes ao objecto $P_1,...,P_n$ sendo P_i vizinho de P_{i-1}, $1 \le i \le n$. Assim, um ponto P está ligado ao ponto Q se existir um caminho do ponto P ao ponto Q, totalmente contido no objecto. Note-se que um componente ligado ou região conexa, corresponde a um conjunto de pontos ligados, e pode ser 4-conexo ou 8-conexo conforme o tipo de vizinhança considerada. A figura 3 ilustra dois tipos de vizinhança, sendo 8-vizinhos todos os pontos $P_1,...,P_8$, e 4-vizinhos os pontos P_1, P_3, P_5, P_7.

Uma outra forma de efectuar a etiquetagem de componentes, também descrita por Rosenfeld e Kak [5], consiste na utilização de comprimentos de sequência[5], descrição que indica para as sequências de pixels pretos numa linha apenas a posição do primeiro ponto e o seu comprimento. Por este processo, colocam-se etiquetas diferentes em cada sequência, e em linhas seguintes verifica-se a linha anterior, repetindo-se o processo. No entanto, se existir uma sequência pertencente à linha anterior, que seja vizinha duma sequência da linha em análise, atribui-se a esta a mesma etiqueta da sequência da linha acima. Quando se verifica que uma sequência da linha em análise é vizinha de mais do que uma sequência da linha anterior, então prossegue-se com uma das etiquetas e colocam-se as etiquetas das sequências referidas, da linha anterior, numa tabela de equivalências com a mesma função da tabela de equivalências do método de etiquetagem de componentes descrito no inicio desta secção.

$$\begin{array}{ccc} P_4(x-1,y-1) & P_3(x,y-1) & P_2(x+1,y-1) \\ P_5(x-1,y) & P(x,y) & P_1(x+1,y) \\ P_6(x-1,y+1) & P_7(x,y+1) & P_8(x+1,y+1) \end{array}$$

Figura 3: Vizinhança dum ponto.

Bow e Kasturi [18] referem um processo de etiquetagem de linhas com seguimento de pixels vizinhos. Iniciando o processo num pixel, verificam-se os seus vizinhos pertencentes ao objecto e segue-se para um deles, respeitando um critério de movimento pré-definido. O critério pode ser, por exemplo, o de respeitar a numeração dada aos pontos vizinhos apresentados na figura 3, seguindo para o ponto de número inferior dentre os pontos $P_1,...,P_8$. A este ponto vizinho vai atribuir-se um código que representará a direcção de aproximação ao passar-se de P para P_i. Este código poderá ser o próprio número i atribuído aos vizinhos do ponto. Ao atingir-se um ponto em que não se possa prosseguir (vizinhos já processados ou pertencendo ao fundo), regressa-se ao ponto de partida, seguindo-se em direcções opostas àquelas com que os pixels estão marcados. Neste movimento de regresso, marcam-se todos os pixels com um nível de cinzento, efectuando a etiquetagem do componente. No algoritmo descrito além de se etiquetar os pixels ligados dum mesmo objecto, determina-se o rectângulo envolvente (coordenadas horizontal e vertical extremas do componente ligado), e a percentagem de ocupação desse rectângulo por parte do componente (relação entre a área efectiva do componente e a área do rectângulo envolvente).

---

[5]Do inglês "run-length"

2.4.1.2 Método de Borrão de Comprimentos de Sequências (RLS[6])

Este é um método de abordagem descendente descrito por Wong, Casey e Wahl [27] e por Antonacopoulos e Ritchings [17]. Este algoritmo considera os pixels brancos com o valor zero, e os pretos com o valor um. Considerando uma sequência x de zeros e de uns de uma linha, o algoritmo RLS retorna uma outra sequência y em que os zeros são convertidos para uns se o número seguido de zeros não for superior a um determinado limiar T_c. Por exemplo, se $T_c = 4$, a sequência x é convertida na sequência y da forma descrita na tabela 1.

Tabela 1: Método de borrão de comprimentos de sequências (RLS)

x	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0
y	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	1

O algoritmo RLS é aplicado horizontalmente e verticalmente, conduzindo a duas imagens de mapa de bits diferentes que são combinadas através duma conjunção lógica. Como a palavra borrão indica, os pixels não pertencentes ao fundo (pretos) invadem as áreas de fundo, formando uma mancha dos objectos. As linhas de caracteres formam, por aplicação horizontal do RLS, áreas rectangulares com largura muito superior à altura (figura 4b). Por outro lado a aplicação vertical do algoritmo RLS transforma as linhas de caracteres em manchas negras de dimensão similar à do rectângulo envolvente das linhas (figura 4c), porque a distância entre linhas é inferior ao limiar T_c. A aplicação horizontal e vertical do RLS transforma as figuras de imagens de qualidade fotográfica em manchas negras de dimensão similar à do rectângulo envolvente da imagem (figura 4b,c), por terem pixels não pertencentes ao fundo muito próximos uns dos outros. Os desenhos de linhas dependendo da sua complexidade, e consequentemente da densidade de pontos, e também da orientação dos desenhos, formarão manchas negras mais ou menos densas, conforme se aplique o algoritmo RLS horizontal ou verticalmente. A conjunção lógica (figura 4d) das duas imagens permite aproximar a dimensão das manchas resultantes do algoritmo RLS da dimensão do rectângulo envolvente das figuras de qualidade fotográfica. O mesmo se verificará entre a largura das linhas de caracteres e a largura da mancha resultante. No caso dos desenhos de linhas obtém-se manchas negras mais ou menos densas, com dimensões variáveis, em função da complexidade dos desenhos. Numa imagem com figuras e texto, o texto será representado por linhas

[6]Do inglês "Run-Length Smearing"

grossas horizontais, enquanto as figuras correspondem predominantemente a blocos pretos aproximadamente rectangulares.

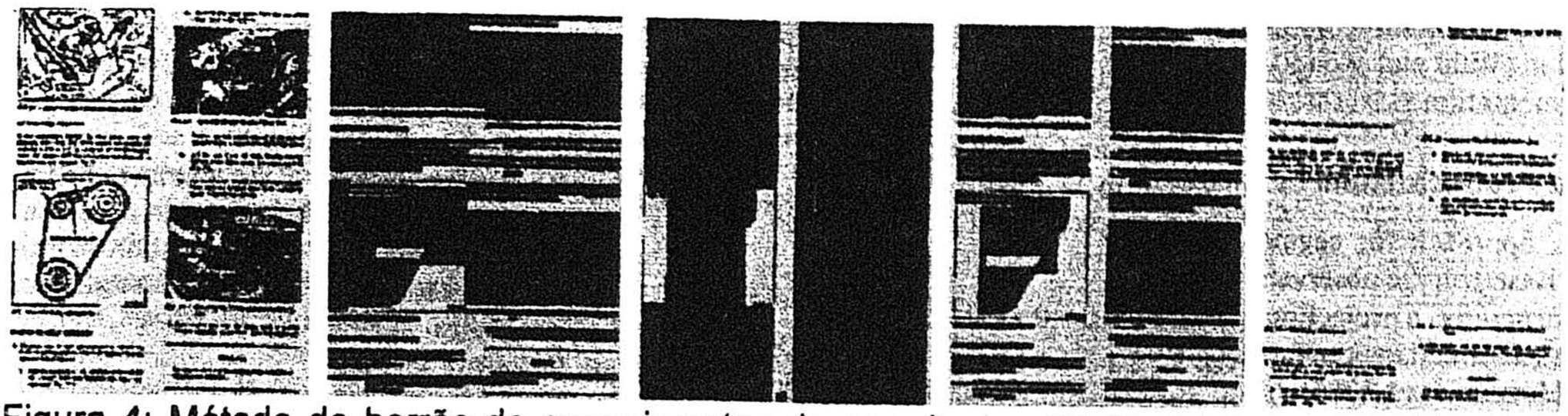


Figura 4: Método de borrão de comprimentos de sequências (RLS). (**a**) Imagem original. (**b**) Aplicação horizontal do algoritmo RLS. (**c**) Aplicação vertical do algoritmo RLS. (**d**) Imagem resultante da conjunção lógica de (b) e (c). (**e**) Blocos de texto obtidos por aplicação deste método.

A etiquetagem de componentes e a sua correspondência com as diferentes regiões negras resultantes do algoritmo RLS, permite classificar os componentes como texto, imagem ou desenhos de linhas.

2.4.1.3 Método de Projecção em Duas Direcções Ortogonais

O método de projecção do número de pixels pretos em duas direcções ortogonais é referido em descrições de aplicações de análise de documentos [17, 19], para segmentação de blocos. A sua utilização baseia-se na consideração de que os documentos são, usualmente, constituídos por regiões rectangulares. É possível verificar na figura 5 que as projecções do número de pixels pretos na direcção do texto, formam um histograma com picos referentes ás linhas, separados por espaços de frequência nula referentes ao espaço entre linhas. As projecções na direcção ortogonal à direcção de escrita formam histogramas com vales abruptos referentes a espaços entre colunas, que separam áreas unimodais referentes a colunas de texto. Este método permite a determinação da distância entre colunas, através dos vales abruptos existentes no histograma de projecções na direcção ortogonal à direcção de escrita. Permite também a determinação da distância entre linhas, através dos espaços do histograma de frequência de projecção nula na direcção de escrita, como foi anteriormente referido. Em [17] recorre-se à determinação da distância entre linhas para estabelecer um limiar que será igual a 2/3 desta distância; este limiar foi utilizado no algoritmo RLS vertical, de forma a não agregar linhas adjacentes.

É possível recorrer a este método para determinação do ângulo de inclinação da imagem relativamente às direcções vertical ou horizontal. A direcção de escrita pode ser determinada efectuando projecções em direcções sucessivas até encontrar um histograma com picos separados por áreas de frequência nula. Pela diferença entre a

direcção de escrita (por exemplo horizontal na escrita ocidental) e a direcção determinada por este processo calcula-se o ângulo de inclinação da imagem.

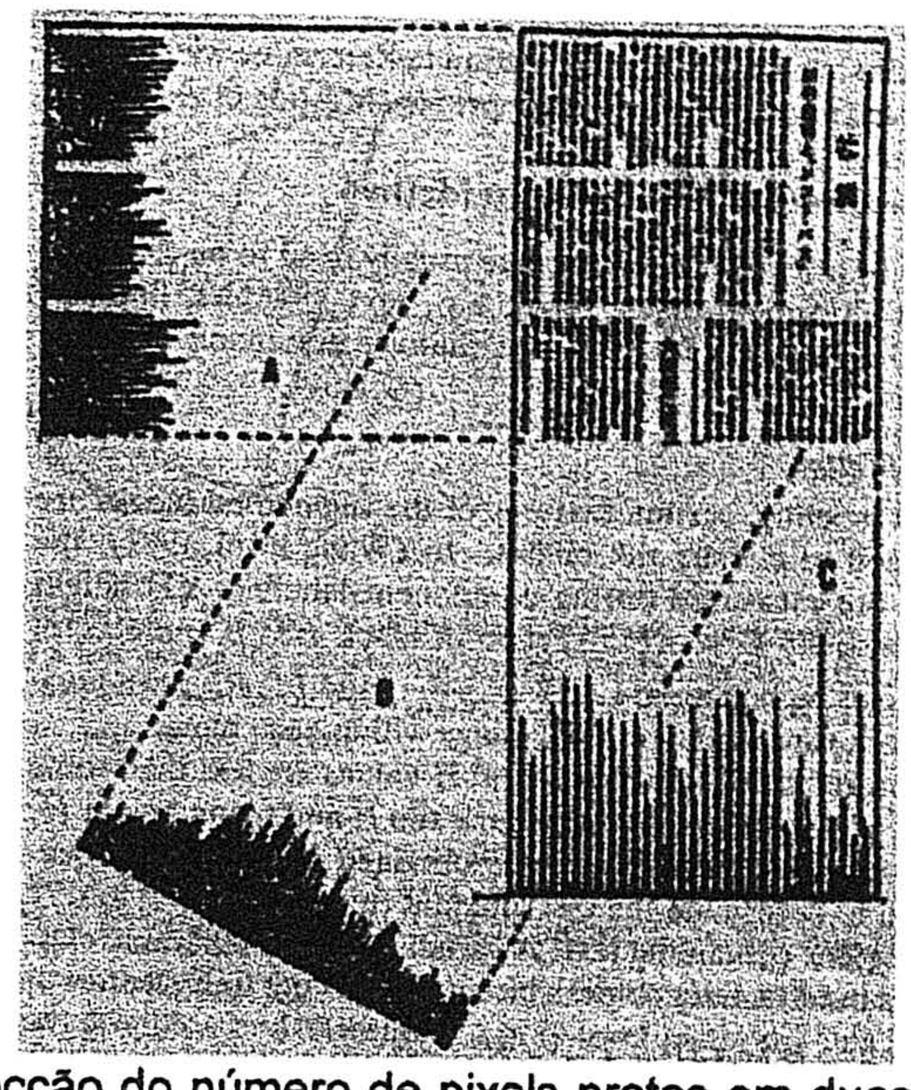


Figura 5: Método de projecção do número de pixels pretos em duas direcções ortogonais. (**A**) Projecção na direcção ortogonal à direcção de escrita. (**B**) Projecção numa direcção não ortogonal relativamente à direcção de escrita. (**C**) Projecção na mesma direcção da direcção de escrita.

2.4.2 Classificação de Blocos

Depois da imagem ser dividida em diversos blocos, é necessário efectuar a sua classificação. Para esse efeito, e recorrendo aos resultados obtidos na fase de separação dos blocos, utilizam-se métodos de análise dos componentes conexos, quantificando determinadas características por forma a classificar os diversos componentes como pertencendo a determinados blocos. Estes componentes podem corresponder a blocos de texto, blocos de desenhos de linhas ou blocos de figuras.

2.4.2.1 Extracção de Características Morfológicas

Os diferentes componentes existentes num documento têm características geométricas e morfológicas que poderão permitir a sua distinção. Os desenhos de linhas são usualmente constituídos por componentes de dimensão superior aos componentes referentes a caracteres. A própria ocupação do rectângulo envolvente dum componente ligado será na maioria dos casos superior em caracteres relativamente a desenhos de linhas. Assim sendo, a extracção de características morfológicas possibilita a classificação de componentes ligados.

Sendo Δx e Δy respectivamente a altura e largura, B a área efectiva, e R o comprimento médio de sequências horizontais dos componentes ligados, Wong, Casey e Wahl [27] definiram as seguintes medidas:

a. Altura do componente $H = \Delta y$;
b. Relação entre a largura e altura $E = \Delta x / \Delta y$;
c. Densidade de ocupação do rectângulo envolvente $S = B / \Delta x \cdot \Delta y$;
d. Comprimento médio de sequências horizontais R.

A avaliação de gráficos de dispersão no plano $R - H$ de imagens documento típicas permite determinar heuristicamente o valor de constantes C_{ij} e os valores médios de R e de H, respectivamente $\overline{R}$ e $\overline{H}$. As constantes C_{ij} em conjunto com os valores médios $\overline{R}$ e $\overline{H}$ permitem definir critérios para classificação dos componentes. Por exemplo, pequenos valores de R e de H correspondem a componentes de texto. Com base nestas características é utilizada a seguinte regra de decisão para separação dos componentes em quatro classes:

- Texto: se $R < C_{21} \times \overline{R}$ e $H < C_{22} \times \overline{H}$;
- Linhas horizontais sólidas: se $R > C_{21} \times \overline{R}$ e $H < C_{22} \times \overline{H}$;
- Gráficos e imagens meio tom: se $E > 1/C_{23}$ e $H > C_{22} \times \overline{H}$;
- Linhas verticais sólidas: se $E < 1/C_{23}$ e $H > C_{22} \times \overline{H}$.

Um processo similar foi utilizado por Bow e Kasturi [18], com base na área dos componentes. Os componentes gráficos grandes têm áreas elevadas relativamente a caracteres. Logo, utilizando o conhecimento sobre o tipo de documentos a serem processados, pode definir-se um limiar de separação entre o texto, ou pelo menos entre gráficos de pequenas dimensões, e os gráficos de grandes dimensões. Para documentos com grandes quantidades de texto relativamente a gráficos, a área mais frequente A_{mp} permite estabelecer um limiar acima do qual os componentes serão classificados como pertencendo a blocos de gráficos. O limiar deve ser estabelecido acima de A_{mp} de forma a não classificar erradamente caracteres de texto. No entanto, se o tamanho de componentes de texto variar, os componentes área mais frequente corresponderão aos caracteres mais pequenos. Neste caso define-se como limiar de separação um valor de área superior ao valor mais elevado entre A_{mp} e A_{av} (área média dos componentes) de forma a incluir os caracteres maiores. Outro critério utilizado, é o do quociente entre o comprimento e a largura do rectângulo envolvente, parâmetro que será superior para elementos gráficos longos. Considera-se por avaliação de caracteres que provavelmente a sua altura não ultrapassará em 10 vezes a largura. Assim sendo, componentes com esta relação serão à partida classificados como gráficos, sendo eliminados de futuras considerações.

2.4.2.2 Medida de Forma por Mapeamento de Distâncias

Uma medida de forma para classificação de blocos é referida por Casey e Wong [19]. A medida de distância $d(x,y,\phi)$ é definida como a distância entre os bordos dum componente, unidos por uma recta, com ângulo ϕ que passa por um ponto de coordenadas (x,y). Esta medida é ilustrada pela figura 6.

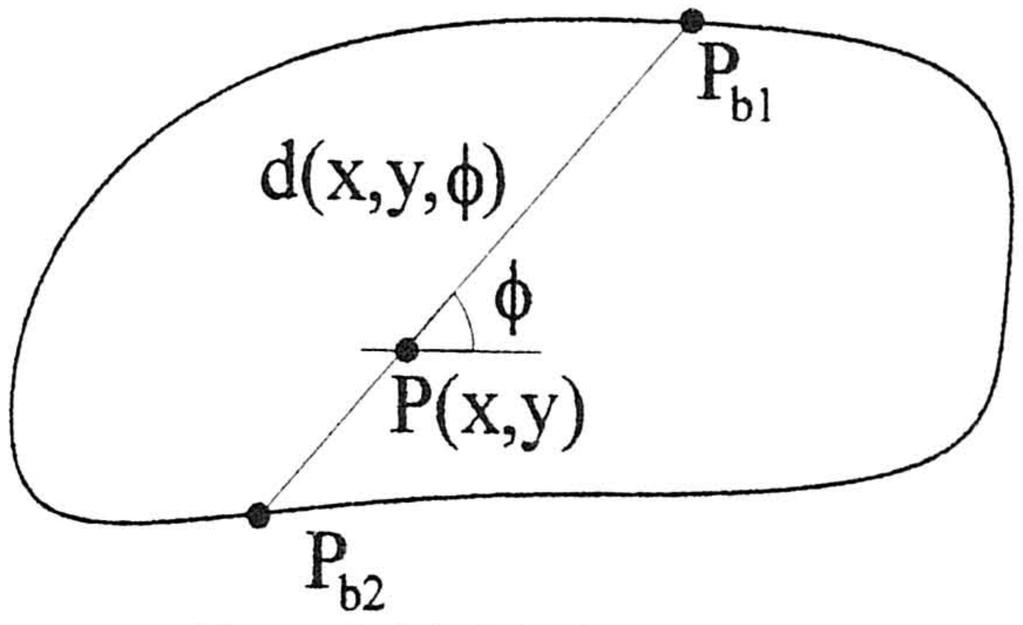


Figura 6: Medida de distância.

Para todos os pontos do componente podem definir-se medidas máximas e mínimas, respectivamente $D_{\max}(x,y)$ e $D_{\min}(x,y)$, para todos os ângulos ϕ possíveis. A excentricidade de cada ponto é determinada pelo quociente entre as medidas máxima e mínima:

$$D_{ecc}(x,y) = D_{\max}(x,y)/D_{\min}(x,y).$$

O valor médio das medidas máximas, mínimas e de excentricidade de todos os pontos do componente são representadas por $\bar{d}_{\max}$, $\bar{d}_{\min}$ e $\bar{d}_{ecc}$.

Verifica-se em geral que, blocos de texto têm elevados valores do factor de forma:

$$f_2 = area/\bar{d}_{\max}{}^2,$$

relativamente a gráficos e figuras. Também se verifica que as figuras têm valores superiores do factor de forma:

$$f_1 = area/\bar{d}_{\min}{}^2,$$

relativamente a gráficos e texto.

2.4.2.3 Método da Transformada de Hough

Bow e Kasturi [18] descrevem a utilização da transformada de Hough para classificar os diferentes blocos, numa abordagem ascendente. A transformada de Hough referida por Rosenfeld e Kak [5], é uma técnica que mapeia todos os pontos pertencentes a uma determinada curva para um ponto de um outro sistema de coordenadas, permitindo a detecção de curvas pretendidas.

A transformada de Hough foi aplicada aos centróides dos componentes ligados para identificar caracteres colineares, que formam frases. Por frases entende-se um conjunto

de caracteres, normalmente num mínimo de três, colocados em posições colineares, que podem formar palavras ou conjuntos de palavras. O domínio de Hough é analisado, em primeiro lugar, nas direcções horizontal e vertical, detectando frases escritas nessas direcções, e só depois se analisam outras direcções, detectando frases escritas em direcções inclinadas relativamente às direcções horizontal e vertical. Na procura de componentes colineares são avaliadas primeiro as frases de comprimento superior, o que ajuda a evitar erros de análise que colocariam determinados caracteres de uma frase noutra mais pequena colocada na sua vizinhança. Este efeito resultaria (se a análise inicial fosse feita para frases mais pequenas) do facto de se considerar um caracter como pertencendo a apenas uma frase, não se considerando novamente esse caracter na análise posterior de outras frases.

O processo descrito, permite detectar componentes ligados colineares que serão classificados como frases, isto é, como blocos de texto. Por eliminação destes blocos obtém-se uma imagem formada por blocos de gráficos.

2.4.2.4 Tamanhos de Componentes e Agrupamento de Ligações

Muitas vezes o texto pode ser classificado analisando o tamanho de componentes. No entanto, se o texto toca em componentes gráficos é necessário utilizar um algoritmo de dois passos, como descrito em [19], começando por determinar o texto duma forma rápida utilizando o tamanho de componentes. Em seguida utiliza-se uma medida de densidade de vizinhança de linha (NLD[7]). Elevados valores de NLD aparecem junto a caracteres, o que permite segmentar os restantes caracteres. Outra forma de classificação, consiste em efectuar o agrupamento de ligações, juntando partes de caracteres em caracteres individuais, baseado na distância entre os centróides dos rectângulos envolventes.

Depois de determinadas as frases colineares como foi referido na secção anterior, Bow e Kasturi [18] descrevem a utilização de um filtro de área para se separarem caracteres que não pertencem à frase. Isto é, impõe-se uma relação de 5 vezes entre a área do componente maior e a área do componente menor, evitando a selecção de componentes não pertencentes à frase mas cujo centróide é colinear com a frase. Além disso, por análise das distâncias entre caracteres, menor que a altura dos mesmos, e entre palavras, menor que 2,5 vezes a altura dos mesmos, efectuam-se agrupamentos dos componentes em palavras. Em seguida as frases são refinadas utilizando algumas heurísticas. Nenhuma frase deve ter menos do que três palavras, e numa linha não pode

---

[7]Do inglês "Neighbourhood Line Density"

haver mais do que duas palavras isoladas. Se as frases não obedecerem a estes regras, são reorganizadas até que só existam frases de texto válidas.

2.5 Reconhecimento de texto

Casey e Wong [19] definiram o reconhecimento de texto como sendo o "processo de mapeamento dum bloco de texto em formato 'bitmap' para um formato computacional de codificação de caracteres (por exemplo: ASCII)". É preciso notar que o próprio termo *reconhecimento de texto,* implica um conhecimento sobre as propriedades da linguagem e sobre a organização do documento, em oposição ao conhecimento único sobre as formas individuais dos padrões dos caracteres.

Para reconhecimento de texto Ejiri et al. [20] referem, resumidamente, dois métodos básicos, um de correspondência de padrões, e outro de análise de estrutura. No primeiro destes métodos a uma área candidata fazem-se corresponder padrões de caracteres ou símbolos, procurando bons índices de correspondência. Este método tem vários problemas relacionados com os tamanhos dos caracteres, rotações, e similitude entre diferentes caracteres. No método de análise de estrutura comparam-se características das áreas candidatas com características de símbolos ou caracteres típicos, utilizando por exemplo informação sobre concavidade e convexidade do contorno do padrão numa área candidata, ou utilizando análise de Fourier para comparar os coeficientes obtidos.

Relacionado com o reconhecimento de caracteres, e com texto impresso (não manuscrito), colocam-se vários problemas, como sejam, a existência de inúmeros tipos de fontes, que podem coexistir num mesmo documento com vários tamanhos, e formatados de diferentes formas, em itálico, negrito, sublinhados com traço único ou duplo ou interrompido; os caracteres podem estar igualmente espaçados, ou terem espaçamento variável; conforme o tipo de máquina de impressão utilizada, impressora de agulhas, de jacto de tinta, laser, térmica, máquina de escrever, máquinas tipográficas, colocam-se muitas variantes no tipo de ponto impresso e traço dos caracteres, já para não falar na crescente utilização de cor. Estas considerações demonstram que métodos de reconhecimento que dependam apenas da informação sobre o padrão dos caracteres tornam-se menos flexíveis e aptos a reconhecer todos os tipos de documento. Dessa forma, os métodos de análise da estrutura dos caracteres serão mais efectivos como referido por Ejiri et al. [20]. Um outro método, referido em [19], não utiliza informação sobre a forma dos caracteres, mas apenas informação sobre o texto em si. Este método apoia-se na utilização de dicionários, de modelos gramaticais, no conhecimento estrutural do texto, e de propriedades contextuais, como seja a distribuição estatística de letras e palavras. Como estas propriedades do texto são independentes da fonte de caracteres, e

têm uma pequena variação entre textos diferentes, permitem a construção de sistemas de codificação de texto independentes da estrutura de padrões de caracteres.

2.6 Reconhecimento de gráficos

Em muitos sistemas de análise de documentos coexistem áreas de texto com áreas de gráficos e de figuras em nível de cinzento ou a cores, sendo tão ou mais importante o reconhecimento de gráficos como de texto. Em documentos de engenharia cujo objectivo seja obter uma representação de entrada num sistema CAD é mais importante o reconhecimento gráfico do que do texto, embora nesse caso seja necessário reconstituir a informação textual (em menor quantidade) manualmente, o que colocará em causa a automatização do sistema. O mesmo se passa com a criação de catálogos técnicos como os de química, ou de electrónica, e com todo o tipo de mapas, como os geográficos, topográficos, geológicos, ou de meteorologia.

O processamento e reconhecimento de gráficos compreende a utilização de todo o tipo de métodos e algoritmos que permitem extrair linhas do esqueleto de objectos alongados e de contornos de objectos cheios, e o seu reconhecimento representando-as num nível apropriado. Essa representação pode ser feita ou por primitivas gráficas básicas (por exemplo rectas, polígonos e elipses) ou utilizando uma representação de mais alto nível traduzindo os gráficos obtidos por símbolos apropriados à aplicação. Esta é uma noção apoiada em várias referências:

- Para Bow e Kasturi [18], "a imagem resultante da separação de blocos de texto contem gráficos e símbolos isolados, que é processada para gerar uma descrição de linhas que representam entidades finas e as fronteiras de componentes gráficos sólidos";
- Por outro lado, Casey e Wong [19] afirmam que "a discussão do processamento de imagens gráficas, só se preocupará com desenhos de linhas, que poderão ser codificadas em diversos níveis de entendimento, e que poderão de facto, passar por esses níveis durante o processo de codificação";
- Também Ejiri et al. [20], Nagasamy e Langrana [24], e Nadler [28] corroboram estas noções nas descrições que efectuam de sistemas de análise de documentos de aplicação geral ou particular, e que utilizam metodologias similares.

No artigo 'Engineering Drawing Processing and Vectorization System' de Nagasamy e Langrana [24], é descrita uma aplicação que, segundo a classificação de Nadler [28], processa desenhos técnicos. Neste exemplo, não são processadas áreas com texto.

O primeiro passo efectuado por estes autores foi a digitalização e pré-processamento. A digitalização devido a problemas de iluminação, de resolução, ou baixa

qualidade do documento originam usualmente imagens documento com ruído, e pequenos buracos e aberturas na estrutura de linhas e componentes sólidos. Para remover pontos de ruído e preencher buracos, os autores empregaram filtros morfológicos que são "fáceis de implementar e baseados em conceitos topológicos locais". Estes filtros foram implementados utilizando janelas de 3x3, cujo resultado é apresentado na figura 7.

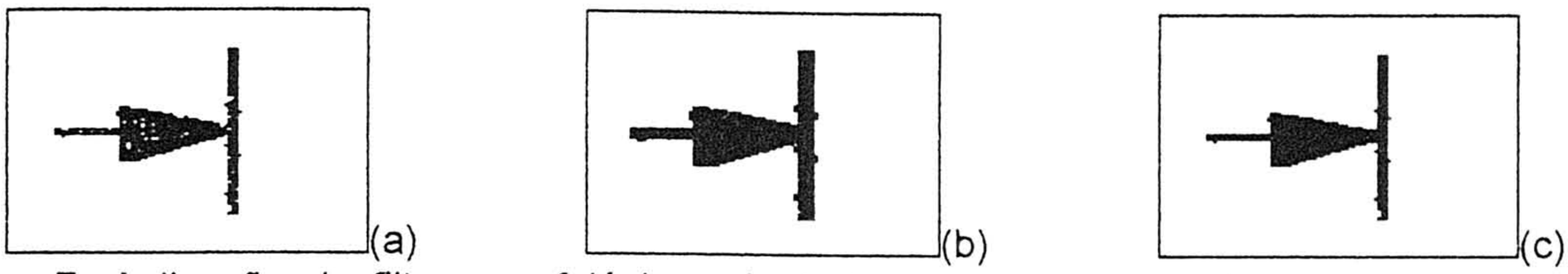


Figura 7: Aplicação de filtros morfológicos de fecho, correspondente à dilatação seguida da erosão. (**a**) Imagem original. (**b**) Imagem resultante da aplicação da dilatação. (**c**) Imagem correspondente ao fecho morfológico, resultante da aplicação da erosão sobre (b).

Depois do pré-processamento, a imagem é considerada como contendo dois tipos de objectos. Estruturas finas, como linhas rectas e curvas, e regiões grossas ou áreas preenchidas, correspondentes a vários símbolos como por exemplo setas que são comuns em desenhos de engenharia. Dessa forma, o próximo passo consistirá na separação de áreas finas das grossas, ilustrado pela figura 8.

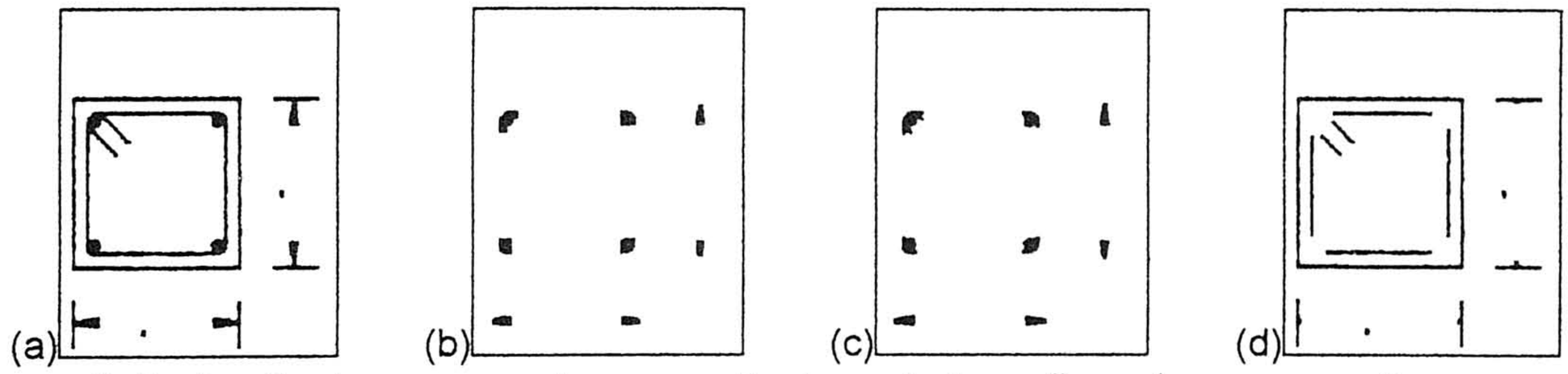


Figura 8: Ilustração do processo de separação das estruturas finas das grossas. Da esquerda para a direita podem ver-se (**a**) a imagem original, (**b**) a imagem resultante do fecho morfológico utilizando a janela $w \times w$, (**c**) a imagem I_G e (**d**) a imagem I_F.

Com o intuito de separar as áreas referidas, considera-se que estruturas com uma largura superior a w pixels são estruturas grossas, com $w = 7$. Para separar estes dois tipos de estruturas recorre-se a uma sequência de passos, começando por se aplicar uma operação morfológica de abertura com uma janela de dimensão $w \times w$ (figura 8b). Como resultado, obtém-se uma imagem com os objectos sólidos de contorno suavizado. Para evitar este efeito, a esta última imagem aplica-se uma dilatação, e em seguida efectua-se a sua conjunção lógica com a imagem que possui ambos os tipos de estruturas (figura 8c). Finalmente subtraindo à imagem com ambos os tipos de estruturas, a imagem com as regiões grossas (I_G) obtém-se uma imagem (figura 8d) apenas com estruturas finas (I_F). Destas duas imagens obtém-se por seguimento de contornos de I_G informação sobre a forma dos objectos sólidos, e efectua-se o adelgaçamento dos componentes de I_F para processamento de linhas.

Finalmente efectua-se a vectorização das estruturas de linhas obtidas. A partir dum vector de pontos que descrevem as linhas, determina-se a informação necessária para descrever essas linhas como primitivas gráficas básicas, ou seja, como segmentos de linha, secções cónicas e splines. Esta representação permite a compressão da imagem documento e sua tradução para utilização com sistemas CAD. Só em último caso se utiliza uma aproximação poligonal. Os autores utilizaram um algoritmo de processamento inicial de preparação para a vectorização seguido de um procedimento de vectorização.

No processamento inicial convertem-se as linhas dum formato de lista de coordenadas para formato de código em cadeia[8] [6, 5], em que se armazenam as coordenadas do primeiro ponto e em seguida o código de deslocamento (figura 9) para o ponto seguinte, obtendo-se assim uma representação da curvatura das linhas. Em conjunto com esta representação também se identificam os pontos dominantes ou críticos [66, 59], que são os pontos extremos de pontos consecutivos da linha que representam segmentos de recta digitais (as condições que definem uma recta digital são referidas na secção de "aproximação poligonal" no capítulo 5), que assim, podem representar uma linha, mantendo a sua curvatura, como representado na figura 10.

```
3  2  1
4     0
5  6  7
```

Figura 9: Código de sequência de pontos.

O procedimento de vectorização foi dividido em dois passos. No primeiro examinam-se os pontos críticos e marcam-se aqueles que poderão ser aproximados por arcos de circunferência, e no segundo aproximam-se as restantes linhas por segmentos de rectas utilizando um algoritmo de aproximação poligonal. São utilizados três critérios para se detectarem pontos que poderão ser aproximados por arcos de circunferência:

- O menor ângulo θ entre dois segmentos de recta, definidos pelos pontos críticos $C_{k-1}C_k$ e por C_kC_{k+1}, é superior a um valor pré-determinado, usualmente 135º;
- O quociente entre os comprimento dos segmentos anteriores não deve exceder um valor λ, dependente da curvatura local; segundo os autores, o valor de λ deve estar compreendido no intervalo [2.0, 3.0];
- O sinal do menor ângulo definido pelos segmentos $C_{k-1}C_k$ e C_kC_{k+1}, medido no sentido do segundo destes segmentos para o primeiro, deve ser igual ao do

[8]Do inglês "chain code"

menor ângulo definido pelos segmentos $C_{k-2}C_{k-1}$ e $C_{k-1}C_k$, medido no sentido do segundo destes segmentos para o primeiro.

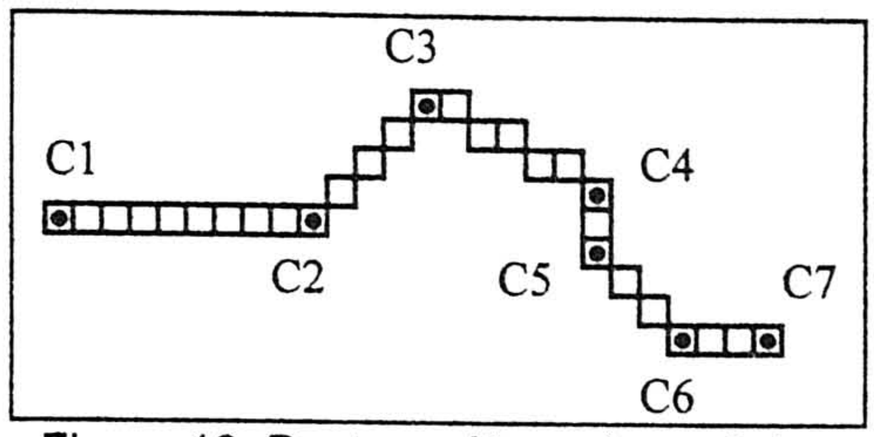


Figura 10: Pontos críticos duma linha.

A primeira condição identifica cantos onde a curvatura será muito pronunciada, seleccionando-os para pontos extremos de arcos; a segunda condição selecciona pontos extremos de arcos de forma que se mantenha, aproximadamente, a continuidade de declive no arco segmentado, e a terceira condição assegura que o arco tenha sempre o mesmo sinal de curvatura, evitando arcos tangentes de curvatura com sinal oposto. Em seguida, cada curva segmentada por este processo é aproximada por uma secção cónica representada pelo polinómio de segundo grau:

$$Ax^2 + Bxy + Cy^2 + Dx + Ey = -1.$$

Os restantes pontos, que não foram aproximados por secções cónicas, são aproximados por segmentos de recta.

Ejiri et al. [20] descrevem sucintamente várias técnicas utilizadas no reconhecimento de gráficos, nomeadamente: técnicas de adelgaçamento; de reconhecimento de tipos de linhas (por exemplo: sólidas, interrompidas, ponto-traço) com reconhecimento local seguido de global com aplicação de regras de sintaxe; codificação de contornos e linhas por código em cadeia. Finalmente sumaria aplicações em diversos campos, de que se destaca as duas que se seguem.

O reconhecimento de diagramas LSI apoia-se nas regras rígidas de desenho destes diagramas, com distâncias mínimas entre pistas e componentes. Surge no entanto a necessidade de distinguir linhas sobrepostas, que representam símbolos ou regiões, o que é feito por processamento e decomposição de ciclos. Os segmentos abertos existentes são considerados pistas de ligação. Utilizando esta filosofia, as linhas finas assim determinadas, são transformadas em linhas com uma determinada largura em função da máscara utilizada na implementação do circuito impresso. Para se transformar as linhas finas em linhas com uma determinada largura, começa-se por detectar os cantos (pontos de elevada curvatura) das linhas, seguindo-se a substituição dos segmentos, definidos por cantos sucessivos, por rectângulos com a largura pré-definida.

Para uma aplicação de reconhecimento de documentos de diagramas de circuitos lógicos, depois de se extraírem as áreas de texto e símbolos, aplica-se uma operação de adelgaçamento, seguida do reconhecimento de linhas para se obter uma lista das suas coordenadas. Em seguida, extraem-se pontos característicos, como sejam, os pontos extremos, de ramificação, de cruzamento, e de cantos. A partir da descrição de pontos, que representam os eixos médios das ligações e componentes do circuito lógico, e da informação relativa aos pontos característicos, extraem-se ciclos formados pelas linhas. Analisando estes ciclos, reconhecem-se componentes nas duas direcções horizontal e vertical, e as suas orientações em cada uma destas direcções (por exemplo da esquerda para a direita ou de cima para baixo). Finalmente, os componentes e caracteres reconhecidos são substituídos por elementos normalizados, e as ligações são transformadas para terem a largura pretendida e serem exactamente verticais e horizontais.

Uma outra aplicação descrita por Bow e Kasturi [18], na fase de reconhecimento gráfico utiliza uma metodologia dividida em 4 fases: separação de componentes gráficos sólidos; adelgaçamento e seguimento de contornos; detecção de segmentos de linha a traço cheio; e detecção de linhas a traço interrompido. Na fase inicial aplica-se à imagem um filtro morfológico de erosão de *N* pixels, seguido da dilatação pelo mesmo número de pixels. Na medida em que a operação de dilatação não é exactamente a inversa da operação de erosão é necessário alguns passos adicionais para se obter a imagem dos componentes sólidos. A imagem de componentes finos é obtida por subtracção da imagem referida à imagem inicial (Nagasamy e Langrana [24] descreveram um processo similar). Na fase seguinte é utilizado um algoritmo de adelgaçamento que marca todos os pixels como pertencendo a um dos quatro tipos: tipo 0 - pixel que não pertence ao esqueleto; tipo 1 - pixel do extremo do esqueleto; tipo 2 - pixel do interior do esqueleto; tipo 3 - pixel nó do esqueleto. Depois de adelgaçar as linhas gera-se uma lista ordenada de todos os pontos, com a localização de junções. Para processamento dos objectos sólidos aplica-se um operador de detecção de orlas à imagem de componentes sólidos, seguindo-se a construção de uma descrição de bordos, através de uma lista ordenada de pontos. Na terceira fase processa-se a lista de pontos das linhas e orlas, para detectar segmentos de recta e curvas. Determina-se o declive duma linha formada por pontos ligados, separando-se a linha em vários segmentos de recta nos pontos em que a variação de declive seja significativa. Um segmento de linha com uma variação contínua do declive é classificado como curva. Os segmentos de curva são aproximados por círculos ou arcos de circunferência, por estimação do centro e raio a partir dos pontos extremos do segmento e do ponto a meio do segmento. Obtém-se desta forma uma

descrição das linhas formadas por segmentos de recta e arcos de circunferência, correspondendo à descrição vectorial dos objectos da imagem.

2.7 CONSIDERAÇÕES FINAIS

Para processar documentos de engenharia é necessário proceder à sua digitalização, recorrendo à resolução (em pontos por polegada) mais apropriada. Na maior parte dos casos, senão mesmo na totalidade, o passo seguinte é a binarização da imagem, de forma a separar o fundo dos objectos.

Neste ponto existem vários blocos da imagem que é necessário separar para processamento independente, por terem características totalmente diferentes. Além de separar, é necessário proceder à sua classificação nos diferentes tipos de blocos: texto, figuras (com diversos níveis cromáticos), e gráficos (desenhos de linhas). A separação e classificação de blocos é efectuada recorrendo a diversos processos, de abordagem descendente ou ascendente, nomeadamente: pelo reconhecimento de componentes conexos e pela sua classificação (por exemplo utilizando medidas morfológicas, ou de densidade de pixels); pela técnica de projecção nas direcções vertical e horizontal; pelo método de borrão de comprimentos de sequências; pela medida de forma por mapeamento de distâncias.

Finalmente é necessário identificar os objectos existentes em cada um destes blocos. Ou seja, é necessário reconhecer os caracteres, palavras e frases dos blocos de texto, reconhecer as primitivas gráficas dos desenhos de linhas, tanto a baixo nível (por exemplo através de rectas, arcos de circunferência, ou splines) como a alto nível (por exemplo reconhecendo rectângulos, estruturas, cotas dimensionais), e processar as imagens de blocos de figuras.

São estes passos, fundamentais no processamento de imagens de documentos, que vão ser utilizados no desenvolvimento deste trabalho: aquisição e pré-processamento de imagens adquiridas, separação e reconhecimento de blocos, e finalmente reconhecimento de desenhos de linhas.

Para o desenvolvimento do processamento de imagens de documentos criou-se um programa de processamento e análise de imagem, com o intuito de suportar todo o trabalho efectuado, que se descreverá no próximo capítulo.

Capítulo 3

Sistema de Processamento e Análise de Imagem

3. SISTEMA DE PROCESSAMENTO E ANÁLISE DE IMAGEM

Para desenvolver todo o trabalho referente à análise de imagens de documentos desenvolveu-se um sistema de processamento e análise de imagem baseado no ambiente Windows para computadores pessoais. Com este capítulo pretende-se apresentar a motivação para o seu desenvolvimento, descrever a sua arquitectura e ilustrar a utilização do sistema.

3.1 MOTIVAÇÃO

Os investigadores, com actividades nas mais diversas áreas que exigem manipulação de imagens, têm a necessidade de recorrer a sistemas apropriados ao seu trabalho, que incluam uma interface de utilizador eficiente e amigável, e naturalmente disponibilizem as funções necessárias. Este sistema poderá ser escolhido dentre aqueles que estão disponíveis, o que melhor se adapte ao campo em questão, ou então, poderá ser construído por forma a tirar partido dos recursos existentes. A primeira opção terá a vantagem da reutilização de código fonte, e a aplicação do esforço duma forma mais direccionada para o objectivo final. No entanto, o desenvolvimento dum sistema terá interesse desde que traga algo de novo, ou traga vantagens para a investigação em curso, ou ainda, se enquadre também no objectivo a atingir (análise de documentos de engenharia).

Avaliando estes conceitos e as alternativas de desenvolvimento, concluiu-se que existia a possibilidade de criar, baseado no ambiente gráfico 'Microsoft Windows', um sistema de processamento de imagem com uma interface de utilizador eficiente e de fácil utilização, similar a outros programas Windows (facilitando a sua utilização). A justificação da escolha deste ambiente gráfico está relacionada com as suas características, que serão posteriormente descritas. A reutilização de código de processamento de imagem previamente desenvolvido para o sistema operativo UNIX, será outro factor a considerar, permitindo alargar mais rapidamente o lote de operações disponíveis, e por outro lado facilitar a adaptação a este sistema (do ponto de vista do desenvolvimento). Este código fonte foi desenvolvido para processamento de imagem no sistema XITE[1] (que será posteriormente descrito).

---

[1] "X-based Image processing Tools and Environment"

Enquanto os recursos de 'hardware' passaram a oferecer mais capacidades por menor custo, as interfaces gráficas do utilizador (GUI)[2] como a do 'Microsoft Windows' tornaram-se interfaces de utilizador que se podem considerar normalizadas. Este sistema operativo oferece: independência dos dispositivos físicos, com portabilidade entre inúmeros sistemas de hardware [7], com interfaces de desenvolvimento estáveis para recursos de entrada / saída (como memória [13, 15, 16], placas gráficas [7], discos, impressoras [12], entre outros); interface de utilizador homogénea, baseada numa norma adoptada por várias empresas, denominado CUA[3], parte duma norma mais alargada da IBM, sobre sistemas de aplicação denominada SAA[4]; finalmente, oferece um grande número de potenciais utilizadores. Por todas estas razões escolheu-se o ambiente gráfico Windows, para servir de suporte arquitectónico e de interface com o utilizador, para o programa aqui referido.

3.2 Arquitectura do Sistema

O sistema desenvolvido tem como objectivo último, o processamento e análise de documentos. No entanto, pretendeu-se que fosse suficientemente genérico para servir de base ao desenvolvimento de um sistema de processamento e análise de imagem de uso geral. Nesta secção será feita a descrição da estrutura do sistema, apoiada na arquitectura de programas Windows, com utilização de código C++ orientado por objectos. Será ainda referido o formato de imagem utilizado pelo programa, e respectiva estrutura de classes baseada no Microsoft Visual C++.

Funcionalmente, e duma forma simplificada, a estrutura de um sistema de processamento e análise de imagem deve contemplar as seguintes opções:

1. leitura ou aquisição de imagens para memória;
2. visualização de imagens;
3. processamento e análise de imagens;
4. armazenamento das imagens resultantes.

Para este efeito é necessário ter uma estrutura de dados que permita manipular a imagem, tanto em memória como em disco. Esta estrutura, para ser portátil e não perder propriedades (resolução, número de níveis de intensidade e a característica cromática da imagem), deve ser independente dos recursos de vídeo. A independência do formato

---

[2]"Graphical User Interface"
[3]"Common User Access"
[4]"Systems Application Architecture"

adoptado não impõe limitações ao tipo de imagem, que resultarão dos recursos físicos existentes. Ou seja, a resolução da imagem, a sua característica cromática, e o número de níveis de intensidade de cada pixel, deve ser uma imposição do sistema, do processo de captação da imagem, ou das limitações de armazenamento, e não da estrutura que a representa, ou da forma de visualização.

A biblioteca de desenvolvimento de programas (SDK[5]) do Windows 3.1 oferece uma estrutura em memória com estas características, que é o formato DIB[6] [12], ilustrado pela figura 11. Este formato é o equivalente em memória do formato BMP[7] [13] utilizado para armazenar imagens em disco. Nesta secção apresentar-se-à de uma forma sumária o formato DIB, fazendo-se uma descrição mais pormenorizada no anexo 1.

O DIB é constituída por um cabeçalho, com toda a informação pertinente para o reconhecimento e manipulação da imagem, como sejam, o número de bits por pixel, as dimensões horizontal e vertical, e o número de entradas da tabela de cores, assim como o tipo de compressão utilizada, entre outra informação. A seguir ao cabeçalho vem a tabela de cores formada por elementos com as intensidades das cores vermelha, verde e azul. Cada ponto da imagem é caracterizado por um valor, que corresponde a uma entrada da tabela de cores, que define a constituição cromática daquele ponto. No byte imediatamente posterior ao final da tabela de cores está colocado o pixel correspondente à última linha, primeira coluna, do mapa de bits da imagem. Sucessivamente, nas posições seguintes estarão os bytes correspondentes aos pixels da última linha, seguindo-se os bytes dos pixels das linhas anteriores, isto é, as linhas são armazenadas em ordem inversa, da última para a primeira linha.

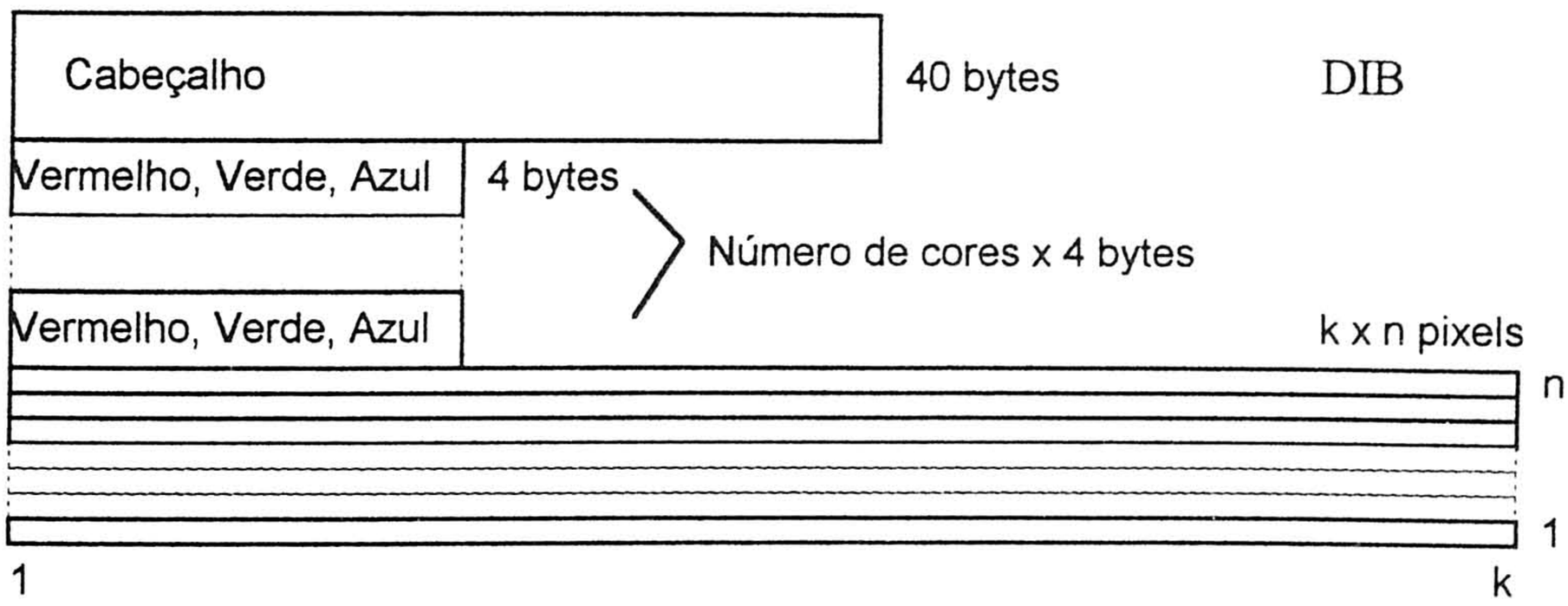


Figura 11: Formato de imagem DIB. Mapa de bits com k colunas e n linhas.

[5]"Software Development Kit"
[6]"Device Independent Bitmap"
[7]"BitMaP"

Os programas Windows obedecem a um determinado conceito de fluxo de processamento. Todos os programas são baseados no conceito de transmissão de mensagens [7], e todos eles têm que ter um ciclo de processamento das mesmas. Sempre que um acontecimento tem lugar, o sistema gera uma mensagem, e passa-a às aplicações interessadas (figura 12), que em conformidade com o objectivo do programa e a sua semântica, efectuam ou não uma acção, libertando em seguida o processador. Como está subentendido nesta última frase, o Windows é um sistema operativo multitarefa não preemptivo[8], o que significa que uma aplicação tem que libertar explicitamente o controlo da unidade de processamento central (CPU[9]), para que outro programa possa ganhar acesso ao CPU. É também por mensagens que as várias aplicações comunicam entre si, e que o próprio sistema operativo comunica com as aplicações. Cada aplicação toma o controlo quando recebe, da fila de mensagens, uma mensagem a ela destinada, passando o controlo a outra quando termina de executar a acção correspondente à mensagem processada.

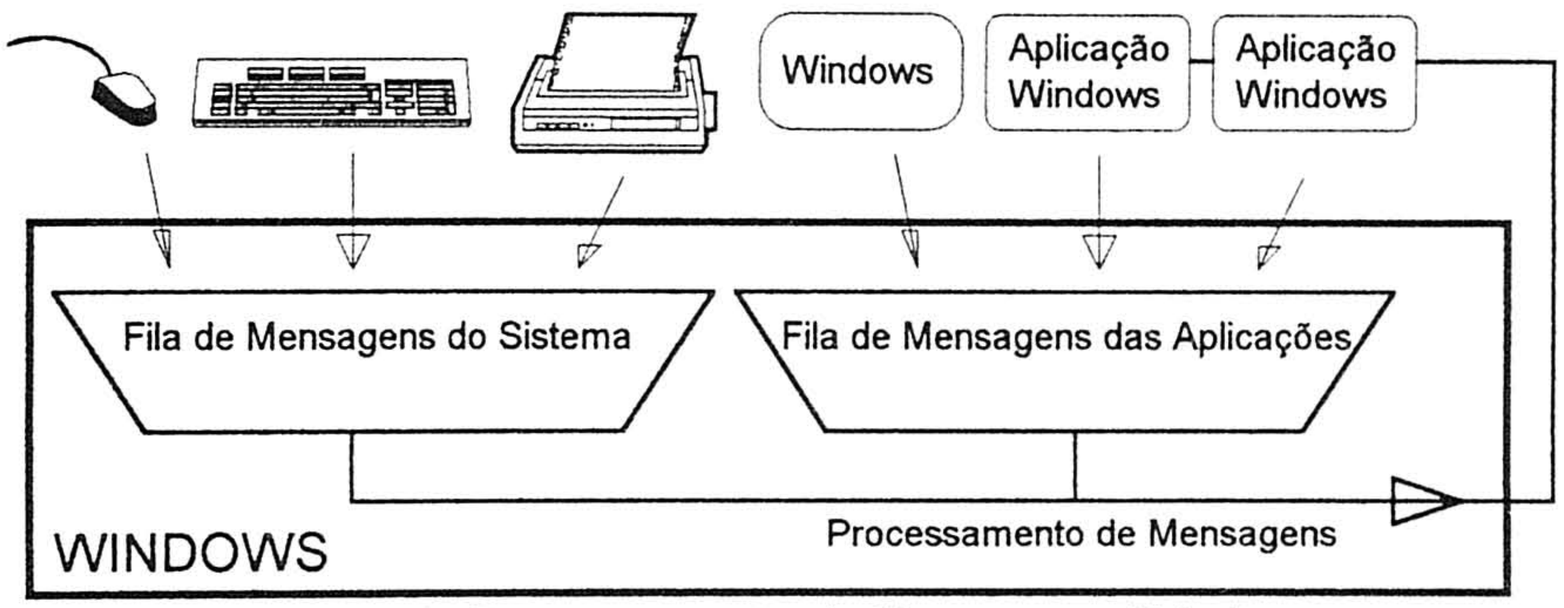


Figura 12: Processamento de Mensagens Windows

Uma aplicação Windows tem uma janela associada, onde se efectua a interacção com o utilizador, através de menus [8], janelas de controlo [9], e caixas de diálogo [10] (alguns destes membros estão representados na figura 15), e onde se visualiza o resultado de acções utilizando a área de cliente da janela de aplicação. Um programa pode ser baseado numa única janela de interacção (interface SDI[10]), associada à área de cliente da janela de aplicação, ou ter múltiplas janelas, uma para cada documento, suportadas pela interface para vários documentos (MDI[11]) [9, 14], denominadas janelas filho. Na figura 15 podemos ver uma janela de aplicação com 3 janelas criança representando 3 documentos diferentes.

---

[8]Do inglês "preemptive"
[9]"Central Processing Unit"
[10]"Single Document Interface"
[11]"Multiple Document Interface"

Uma das capacidades mais determinantes do Windows, é a de traduzir elementos gráficos (fontes, mapas de bits, meta ficheiros, e objectos gráficos) de uma forma independente dos dispositivos [11]. Fazendo-o para sistemas de vídeo monocromáticos, ou para impressoras laser a cores, de tal modo que o programador apenas utiliza uma interface de dispositivo gráfico (GDI[12]) homogénea para todos os dispositivos.

De entre os vários sistemas de desenvolvimento disponíveis para Windows, optou-se pela utilização do Visual C++ (figura 13). Esta opção baseou-se na interface deste programa, acessível e prática, na existência de inúmeras plataformas de ajuda em linha, em papel, e em CD-Rom, nas ferramentas de ajuda ao desenvolvimento de programas (ferramenta CASE[13]) orientados por objectos, e na diversificada capacidade de seguimento e correcção de programas.

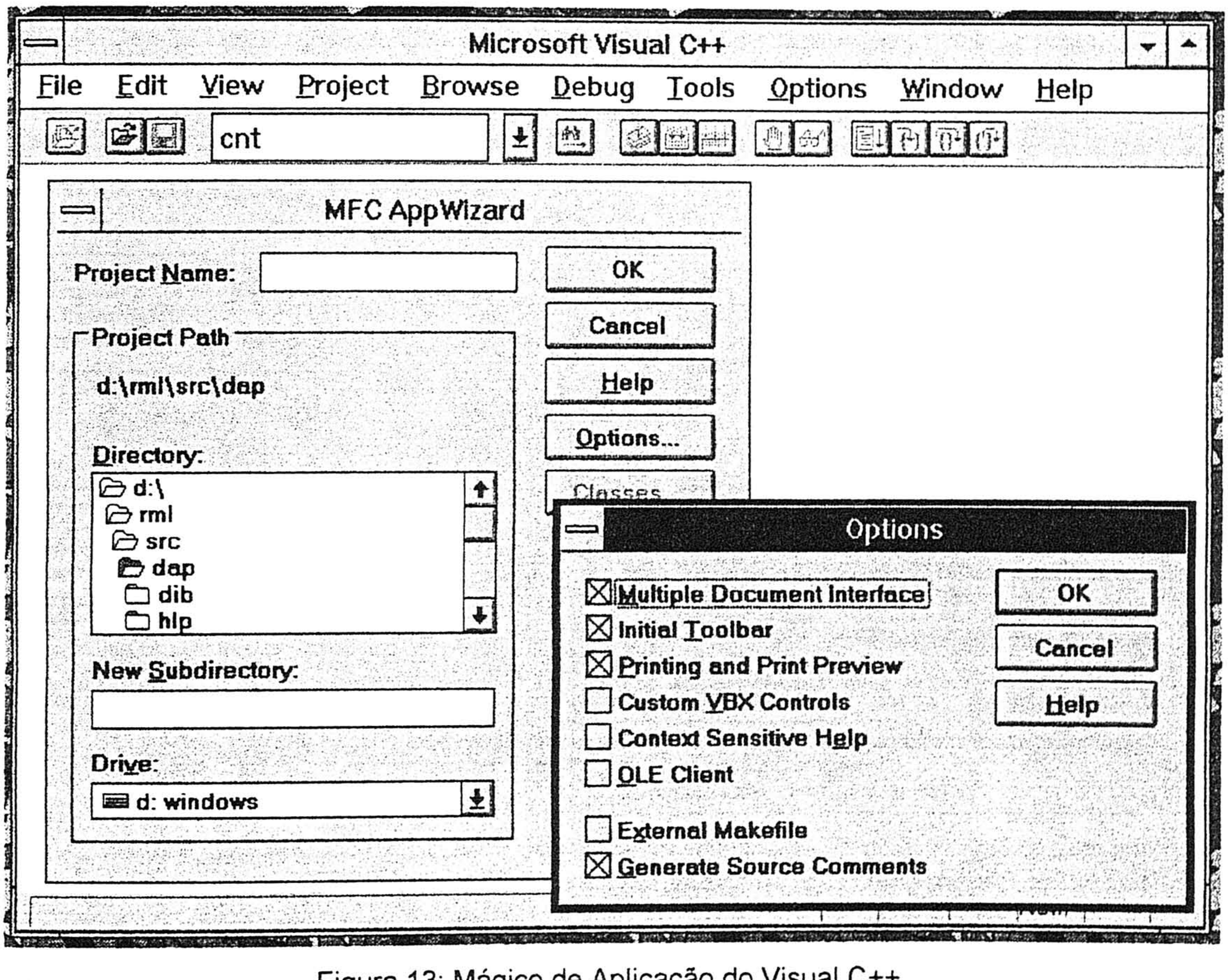


Figura 13: Mágico de Aplicação do Visual C++.

[12]"Graphics Device Interface"
[13]Do inglês "Computer Assisted Software Engineering"

Este sistema tem uma ferramenta CASE que permite montar a estrutura duma aplicação duma forma automática (figura 13). Esta opção, que é denominada mágico de aplicação[14], cria todos os ficheiros necessários para o desenvolvimento do projecto, baseando a estrutura do programa nas bibliotecas de classes básicas (MFC[15]) da Microsoft, em função das opções seleccionadas.

A arquitectura duma aplicação Windows utilizando as classes MFC é constituída por cinco classes fundamentais, a partir das quais se controla o comportamento do programa. A função de cada uma destas classes na construção e controlo do programa, é representada na figura 14. Para iniciar, executar e terminar um programa Windows, construído com base nas classes MFC, existe uma classe fundamental 'CWinApp'. Esta serve de base à criação de qualquer programa, inicializando e associando as classes do programa. É nesta classe que se define, por exemplo, se o programa terá uma interface múltipla de utilizador, qual o tipo de janela principal, de documento, de janelas filho e de janelas de visualização, denominadas janelas *vista*. Derivada de 'CWinApp' criou-se a classe 'CDapApp', que efectua uma inicialização usual, cria um documento padrão do tipo MDI e cria a janela de aplicação.

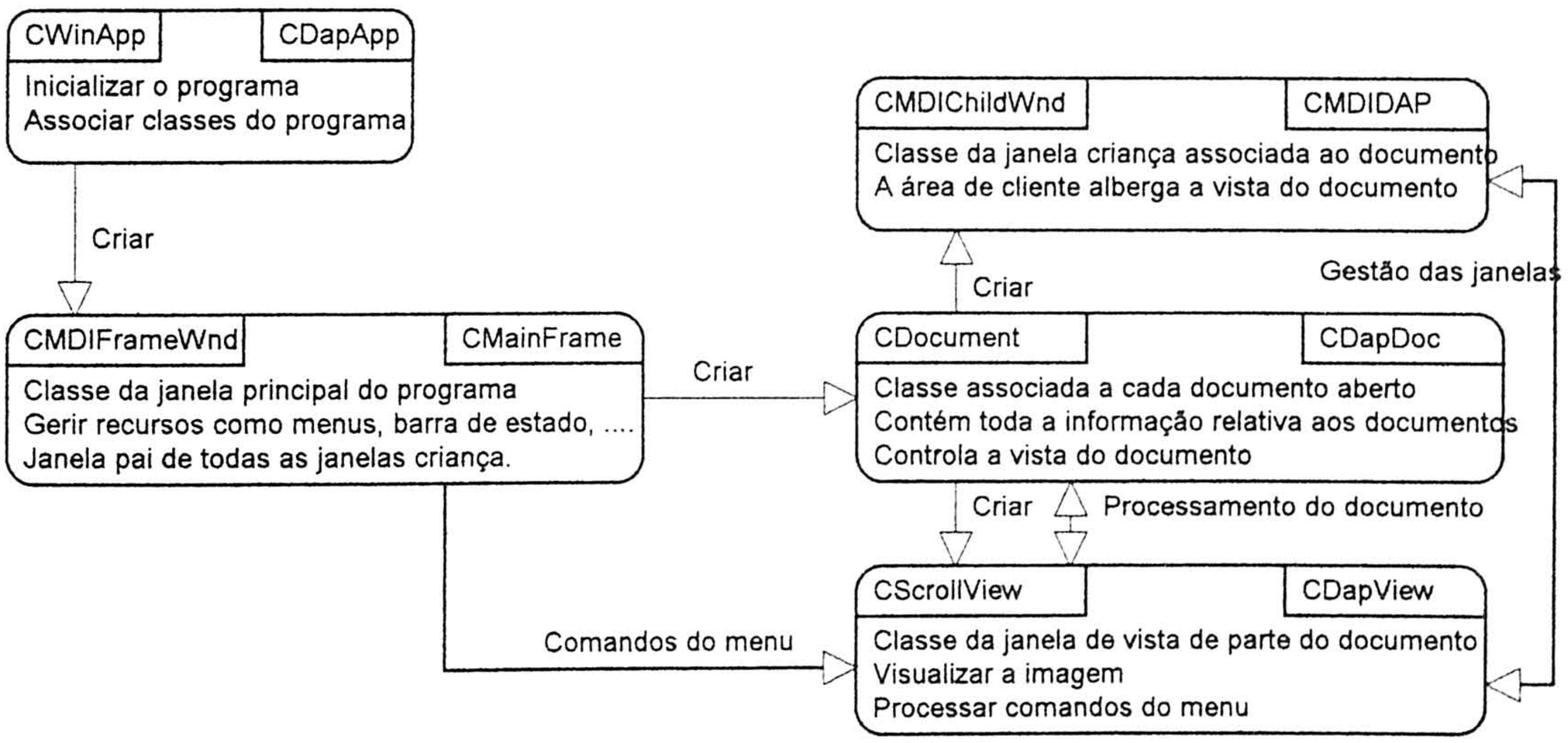


Figura 14: Classes MFC do programa Windows de processamento de imagem. Cada classe tem o nome da classe MFC base no canto superior esquerdo, e o nome da classe herdeira, do programa, no canto superior direito. Em cada classe são indicadas algumas das funções principais. As setas indicam o fluxo, entre classes, de algumas das mensagens principais.

A classe 'CMDIFrameWnd' permite controlar a janela principal da aplicação, que contém o título do programa, os menus, as barras de ferramentas, entre outros recursos.

---

[14]Do inglês "Application Wizard"
[15]"Microsoft Foundation Classes"

A classe 'CMainFrame' descendente de 'CMDIFrameWnd' é a classe associada à janela de aplicação, que permite gerir várias janelas de documento respeitando uma interface do tipo MDI. Na sua criação abre-se uma janela de aplicação com uma barra de ferramentas e uma barra de estado. O seu método construtor inclui código para inicializar algumas variáveis que têm valores por defeito, independentemente do documento a processar, e que se forem alteradas repercutem-se em todos os documentos. É o caso de alguns parâmetros referentes a algoritmos de aproximação de linhas por segmentos de recta, por círculos ou splines, ou ainda o tamanho por defeito de uma janela de convolução. Estas serão variáveis que se podem considerar globais, cuja alteração é efectuada por métodos desta classe. Esta é a classe ascendente (classe pai) de todas as classes relativas às janelas filho.

A classe 'CDocument' está associada aos documentos da aplicação, onde se coloca toda a informação referente ao mesmo. Derivada de 'CDocument' criou-se a classe 'CDapDoc', que servirá de base a todos os objectos documento. Nesta classe são declarados os atributos do documento, ou seja, todas as variáveis que serão necessárias para processar a imagem. Alguns desses atributos são: a própria imagem, isto é, um mapa de bits independente do dispositivo (DIB); o tamanho da imagem, em pixels; a tabela de cores na forma de uma paleta de cores com o formato Windows; alguns apontadores para estruturas que permitem processar listas de pontos. É nesta classe que se introduz o código para criar novos documentos, e para abrir ou gravar documentos. É ainda nesta classe que se controla a parte do documento que vai ser visualizado.

A classe 'CMDIChildWnd' serve para gerir o funcionamento das janelas filho associadas à janela *vista* do documento. A classe 'CMDIDAP' descendente de 'CMDIChildWnd' é a classe associada a cada janela filho da interface de múltiplos documentos (imagens DIB). São estas janelas que servem de janela envolvente de cada *vista*, isto é, envolvente de cada mapa de bits.

A classe 'CScrollView', descendente de 'CView', é a classe base associada às janelas *vista*, isto é, aquela que se utiliza para visualizar a informação pretendida dos documentos. A classe 'CDapView' descendente de 'CScrollView' é a classe associada à janela *vista* deste programa, isto é, associada à área de cliente de cada uma das janelas filho onde se visualiza cada um dos documentos abertos. Com esta classe controla-se a visualização da imagem, toda ou uma parte (utilizando barras de deslocamento), em modo normal ou aumentado, seleccionando correctamente as paletas de cores do sistema em função das tabelas de cores da imagem. Controla-se também, a possibilidade de copiar e colar imagens ou áreas de interesse rectangulares definidas dentro da imagem. Finalmente, uma das principais funções das classes de visualização é

a de terem código que permita processar os documentos, em função dos comandos do programa, introduzidos através dos menus. É portanto, nesta classe que se introduz o código de processamento dos comandos.

A figura 15 ilustra a relação entre o programa e as classes de implementação, podendo-se ver as classes que permitem gerir a janela principal ('CMainFrame'), as janelas filho ('CMDIDAP'), os vários documentos ('CDapDoc') e as respectivas vistas ('CDapView').

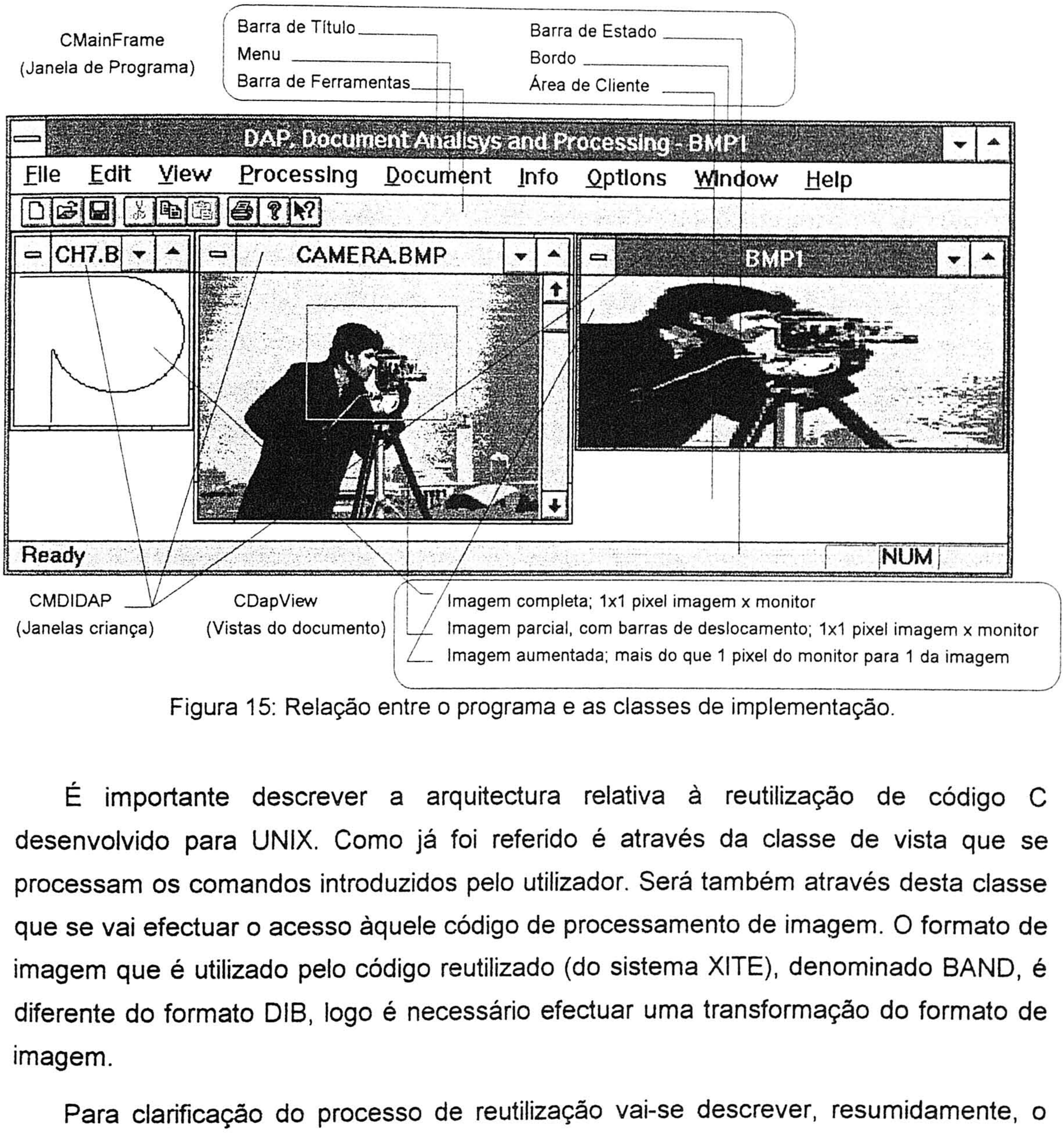


Figura 15: Relação entre o programa e as classes de implementação.

É importante descrever a arquitectura relativa à reutilização de código C desenvolvido para UNIX. Como já foi referido é através da classe de vista que se processam os comandos introduzidos pelo utilizador. Será também através desta classe que se vai efectuar o acesso àquele código de processamento de imagem. O formato de imagem que é utilizado pelo código reutilizado (do sistema XITE), denominado BAND, é diferente do formato DIB, logo é necessário efectuar uma transformação do formato de imagem.

Para clarificação do processo de reutilização vai-se descrever, resumidamente, o sistema XITE, e o formato de imagem utilizado neste sistema. O XITE é um sistema de processamento e análise de imagem, construído como uma colecção de programas desenvolvidos em ambiente UNIX. Estes programas podem ser executados directamente

a partir do sistema operativo UNIX, especificando os argumentos de entrada e saída. Estes argumentos podem ser nomes de ficheiros de imagem, ou parâmetros necessários para os algoritmos de processamento e análise de imagem. Tipicamente, um programa deve transmitir uma mensagem para o utilizador sempre que faltar ou houver erros nos argumentos introduzidos. O programa "xshow" fornecido pelo XITE é uma outra forma de utilizar este sistema. Este programa, com uma interface baseada em menus, permite efectuar a visualização de imagens, utilizar outros programas do sistema, e apresentar as imagens resultantes da operação escolhida. Novos módulos poderão ser construídos e agregados à colecção de programas, recorrendo às bibliotecas de processamento e análise de imagem, de manipulação de imagens em disco e em memória, e de ajuda à criação de aplicações baseadas no sistema de janelas X11.

O formato de imagem BIFF, utilizado pelo XITE, é constituído por 3 partes principais denominadas *info*, *band* e *block*. A primeira destas partes corresponde a um cabeçalho que contém toda a informação necessária para descrever a imagem, nomeadamente o título, o tipo de pixel (pode ser representado por um byte, um inteiro, ou número complexo, entre outros), a resolução horizontal e vertical, e um bloco de texto para descrição da imagem, entre outra informação relevante. A segunda parte que será denominada daqui para a frente por parte de bandas, é constituída por uma ou mais bandas (ou BAND), ou seja, corresponde a uma sequência de bandas. Uma banda corresponde a um mapa de pixels com o valor de cada pixel, isto é, corresponde a um vector bidimensional com os valores de todos os pixels da imagem. O sistema de coordenadas global da banda varia do canto superior esquerdo para o canto inferior direito da imagem de [1,1] para [xsize,ysize], sendo xsize e ysize as dimensões horizontal e vertical da imagem. A terceira parte corresponde a uma colecção de blocos de 512 bytes, cujo conteúdo é livremente escolhido pelo utilizador.

Embora exista a possibilidade de utilizar um único formato de imagem, no entanto, não será viável, porque sendo o formato DIB utilizado pelo SDK do Windows para visualizar mapas de bits (o que acelera e facilita a sua visualização), tornaria penalizador (relativamente à carga computacional) o recurso único ao formato BAND. No caso de se utilizar este formato seria necessário construir rotinas, que permitissem visualizar as imagens, com código suplementar, o que penalizaria a velocidade do programa em si, com custos de desenvolvimento que serão também de considerar. Por outro lado, se se optar pelo formato DIB seria necessário reconverter todas as rotinas de processamento de imagem a reutilizar, o que faria desta, uma opção (reutilização de código fonte) sem sentido.

Colocam-se vários problemas à reutilização, em MsDos, de código fonte desenvolvido para o sistema operativo UNIX. O MsDos divide a memória em blocos de 64 Kbytes, impossibilitando o acesso contínuo a blocos maiores, e a construção de estruturas de dimensão superior. O Windows ultrapassa este problema através do registo de blocos de memória globais, mas continua a ter o problema ao pretender-se utilizar estruturas de memória convencionais, como vectores. Este facto obriga à alteração de módulos que recorram a vectores de dimensão superior a 64 Kbytes. No entanto, este problema não se põe na manipulação de imagens BAND, porque a sua estrutura em memória é formada dinamicamente, por uma lista ligada de vectores do tamanho duma linha (admite-se que a dimensão da linha não ultrapassará 64 Kbytes). Outro problema a considerar será o do tamanho das estruturas simples de dados, do código C. Em UNIX um inteiro é formado por 32 bits, enquanto no MsDos é formado por 16 bits, o que obriga à alteração do código de forma que a maior dimensão da informação a ser albergada pela estrutura simples, não ultrapasse a sua dimensão física, ou seja, é necessário alterar o tipo da estrutura na maior parte dos casos. Verificam-se outros problemas, nomeadamente na impressão para o écran usando a instrução *printf*, que não é compatível com o ambiente Windows, e obrigatoriedade de recorrer a apontadores do tipo *huge*, quando é necessário obter memória de uma forma dinâmica, fora do segmento de dados de 64 Kbytes.

Para efeitos de reutilização de código e com o intuito de estruturar melhor o programa, recorreu-se a um módulo adicional de interface entre o parte do programa puramente Windows, e a parte de reutilização de código desenvolvida em ambiente UNIX. O módulo com esta função denomina-se 'Processing' (figura 16).

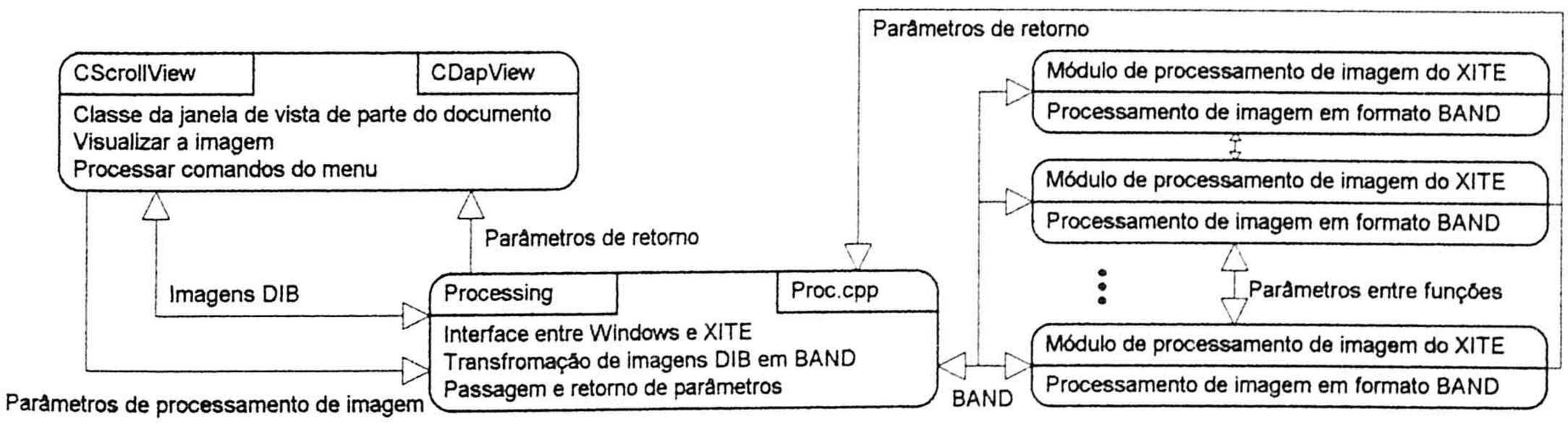


Figura 16: Arquitectura de interface para a reutilização de código XITE, desenvolvido em UNIX.

A ilustração deste componente da arquitectura do sistema está representada na figura 16, onde é visível que um comando escolhido pelo utilizador, começa por ser processado pela classe de visualização, que passa a imagem em formato DIB e os parâmetros necessários para o processamento para o módulo 'Processing'. Neste

módulo dá-se a transformação de DIB para o formato BAND, que é passado juntamente com os parâmetros para os módulos XITE. No retorno destes módulos as imagens BAND serão transformadas para os formatos DIB que serão retornados para a classe associada à janela *vista*.

3.3 As Opções do Programa

Este programa permite processar e analisar imagens, gerir os ficheiros respectivos, modificar modos de visualização, obter informação sobre os mapas de bits, ou simplesmente gerir as janelas filho respeitantes aos vários documentos abertos.

Para alterar o tamanho da imagem, basta alterar o tamanho da janela filho que a alberga. Alterando o tamanho desta janela, a classe de vista recebe uma mensagem e em função da dimensão da janela altera o modo de visualização da imagem. Se a área de cliente da janela for superior, em pixels, à dimensão da imagem, então a imagem será visualizada de uma forma aumentada; se pelo contrário, a área de cliente for inferior, então a imagem será visualizada em modo normal, mas apenas uma parte será visível, sendo necessário recorrer a barras de deslocamento para tornar visível a parte restante. Foi criada a possibilidade de a janela voltar a ter a dimensão adequada à imagem (modo de abertura da mesma), bastando para o efeito, minimizar e depois maximizar a janela. Existe ainda a possibilidade de copiar e colar imagens ou áreas rectangulares da imagem, definindo o rectângulo de interesse por arrastamento do rato, sobre a imagem.

Na figura 15 ilustram-se algumas das possibilidades do programa, que neste caso tem três documentos, sendo o primeiro (da esquerda para a direita) visto de uma forma completa, com uma relação de 1 para 1 entre pixels da imagem e do monitor, o segundo documento é visualizado parcialmente, e o terceiro documento que resultou duma cópia parcial do segundo é visualizado completamente, duma forma aumentada com uma relação de vários pixels do monitor para um pixel da imagem.

A barra de menu é constituída por várias opções representadas por menus de abertura[16], que por sua vez podem ter itens de menu (respeitantes a comandos) ou outros menus de abertura.

No menu de abertura 'File', ilustrado pela figura 17, acedem-se às operações que permitem gerir ficheiros de imagens, lendo imagens (podendo-se procurar o ficheiro pretendido ou escolher de entre uma lista dos quatro últimos ficheiros manipulados),

---

[16]Do inglês "popup" que corresponde a algo que surge inesperadamente

gravando imagens, criando documentos vazios, imprimindo, ou exportando para um ficheiro de formato adequado a um programa CAD.

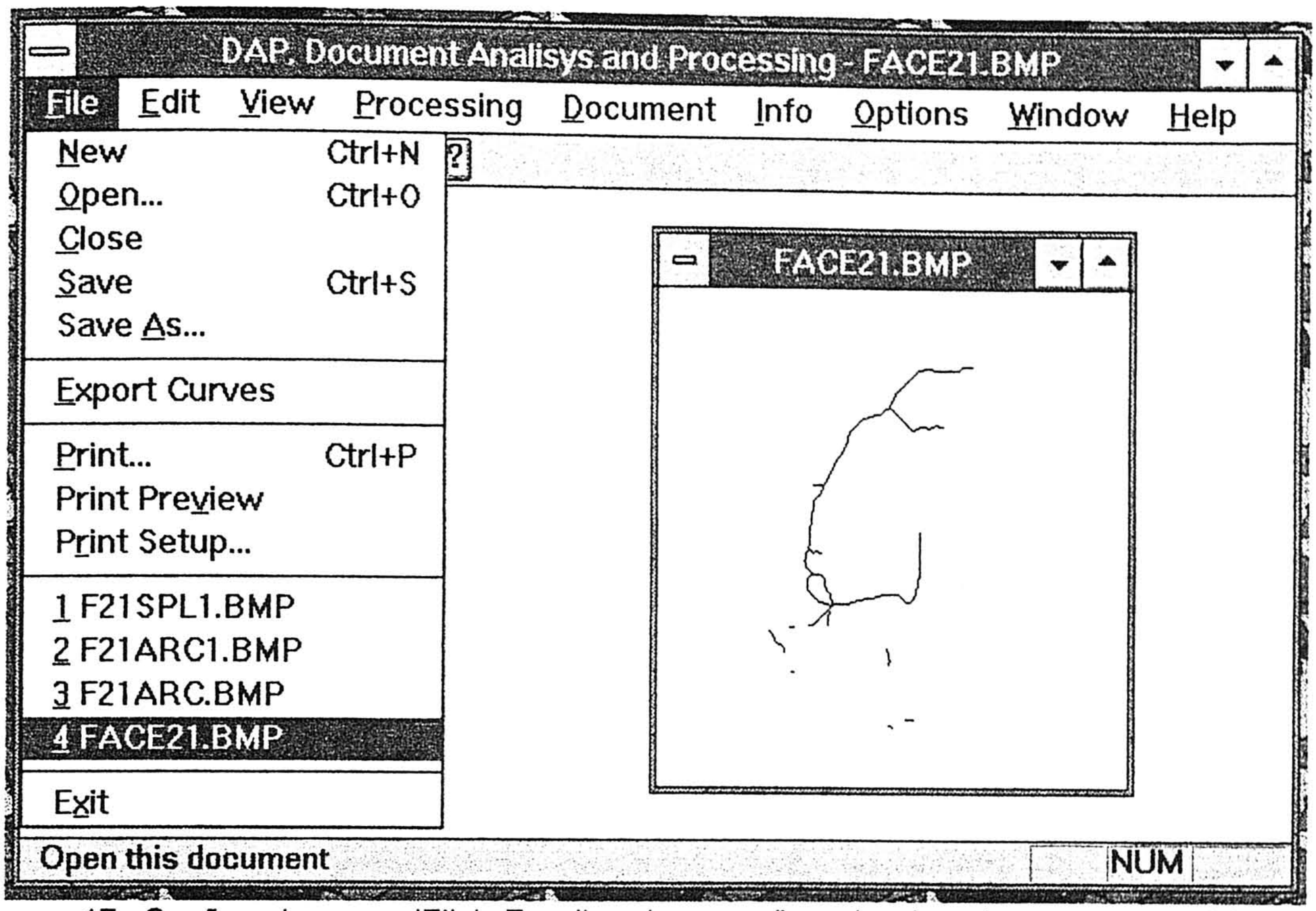


Figura 17: Opções do menu 'File'. Escolhendo a opção seleccionada o programa iria ler a imagem apresentada na janela filho.

É a partir do menu 'Edit', ilustrado pela figura 18, que se torna possível copiar e colar imagens ou parte de imagens. No menu 'View' estão os comandos para controlo do modo de visualização da janela de programa, isto é, a visualização da barra de ferramentas e de estado.

O menu 'Processing' engloba os comandos, apresentados na figura 19, de processamento e análise de imagem, que permitem efectuar operações de convolução, operações morfológicas, operações aritméticas, e operações de adelgaçamento, entre outras.

Figura 18: Opções do menu 'Edit'. Recorrendo à opção 'Copy' copiou-se a área rectangular de interesse da imagem LENA.BMP, efectuando-se a sua colagem nas janelas BMP4 (tamanho real) e BMP5 (imagem aumentada).

A partir do menu 'Document', ilustrado pela figura 20, é possível aplicar comandos de forma a efectuar-se a análise de imagens documento, separando texto de desenhos de linhas, aproximando linhas por polígonos, ou arcos de circunferência, ou detectando cantos que permitam a aproximação por splines, que se descrevem detalhadamente nos dois capítulos seguintes.

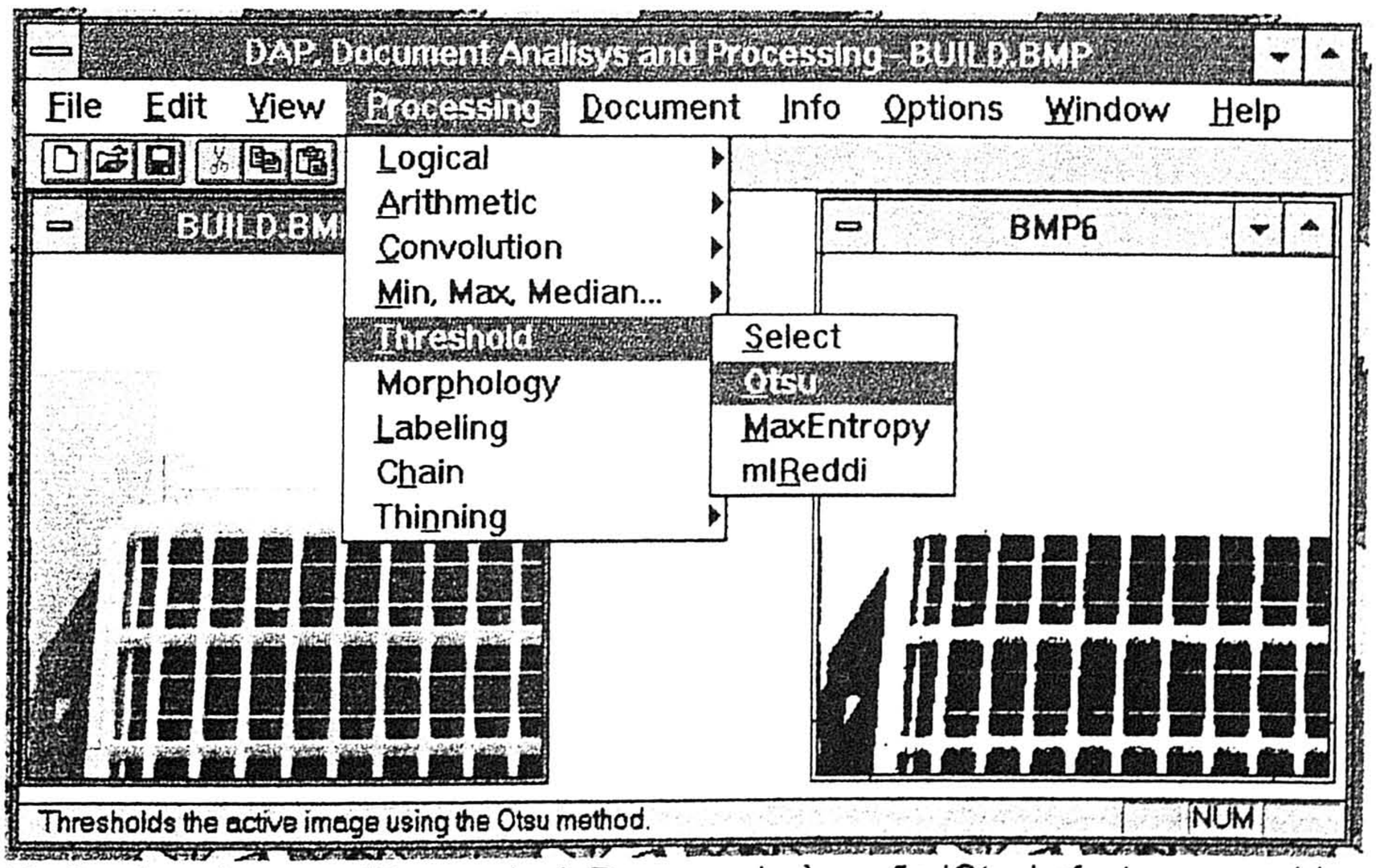


Figura 19: Opções do menu 'Processing'. Recorrendo à opção 'Otsu' efectuou-se a binarização da imagem BUILD.BMP, resultando na imagem apresentada em BMP6.

No menu 'Info' estão os comandos que mostram informação sobre a imagem, e que determina o seu histograma. É no menu 'Options' que se acedem às operações que permitem alterar parâmetros relacionados com algoritmos de processamento de imagem ou de processamento de imagens documento.

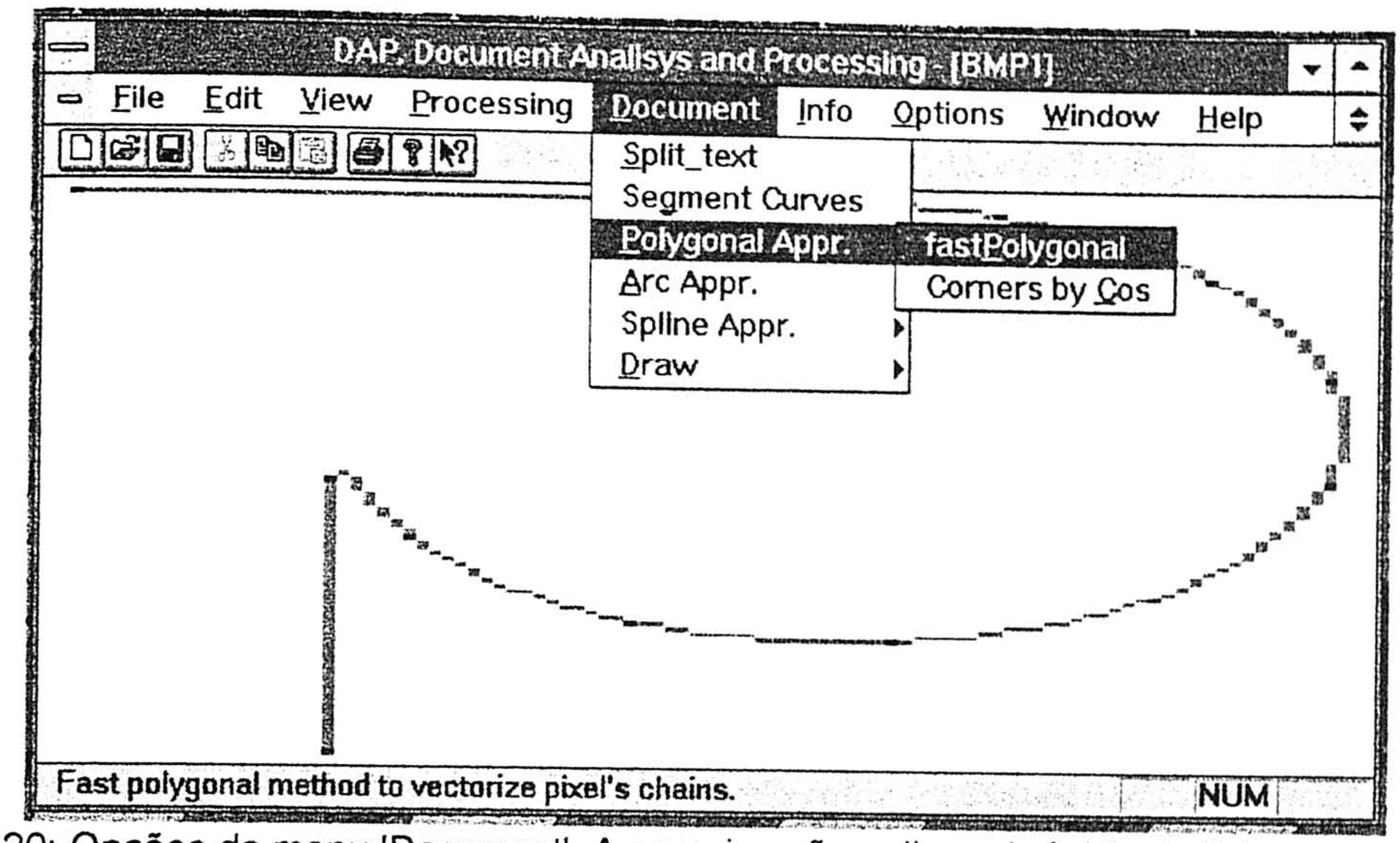


Figura 20: Opções do menu 'Document'. A aproximação poligonal rápida da linha da imagem determina os segmentos representados pelos seus extremos (pixels mais escuros).

Recorre-se ao menu 'Window' para gerir as janelas filho, colocando-as em cascata, ou ocupando toda a área de cliente, possibilitando a activação directa de qualquer uma destas janelas, ou gerindo automaticamente a localização dos ícones representando as janelas minimizadas. Finalmente com o menu 'Help' obtém-se informação de ajuda, ou informação sobre a versão do programa.

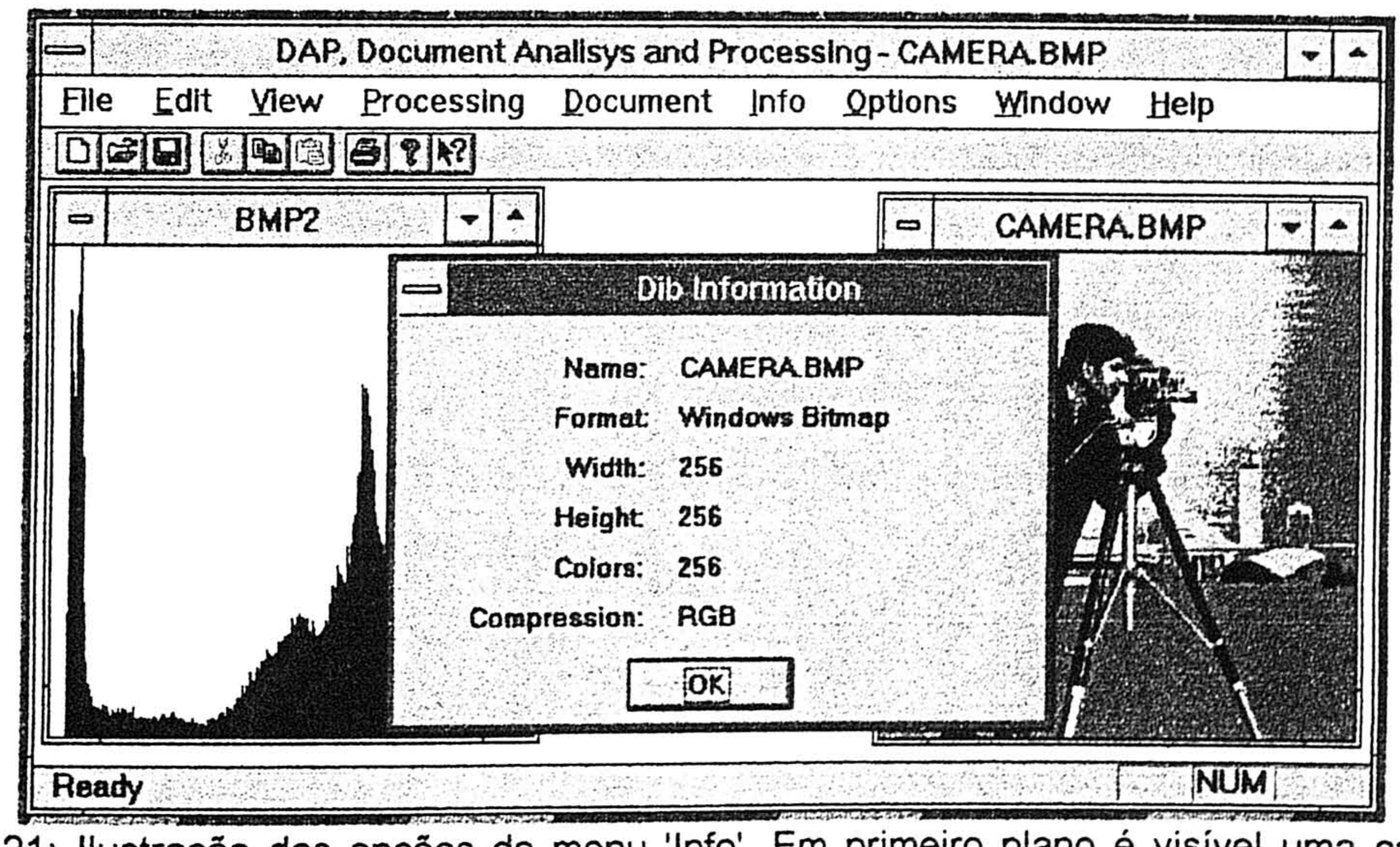


Figura 21: Ilustração das opções do menu 'Info'. Em primeiro plano é visível uma caixa de diálogo com informação sobre a imagem CAMERA.BMP. O histograma desta imagem está representado na janela BMP2.

A aplicação deste sistema em algumas operações simples de análise e processamento de imagem é exemplificada em seguida. Pode-se assim verificar que a utilização directa deste programa, em conjunto com o desenvolvimento de novas operações permitirá efectuar praticamente qualquer tipo de operação sobre imagens.

Para se detectarem orlas de objectos numa imagem, pode-se efectuar a convolução da imagem com as diferentes máscaras 3x3 do operador de Prewitt [5], apresentadas na figura 22. Considerando que o objecto tem valor 0 e o fundo valor 1, a conjunção das imagens resultado de cada uma das convoluções permite obter uma imagem com as orlas, como ilustrado na figura 23.

1	1	1
0	0	0
-1	-1	-1

1	1	0
1	0	-1
0	-1	-1

1	0	-1
1	0	-1
1	0	-1

0	-1	-1
1	0	-1
1	1	0

-1	-1	-1
0	0	0
1	1	1

-1	-1	0
-1	0	1
0	1	1

-1	0	1
-1	0	1
-1	0	1

0	1	1
-1	0	1
-1	-1	0

Figura 22: Máscaras do operador de Prewitt.

A aplicação da diferença aritmética entre duas imagens da mesma cena é um operador básico frequentemente utilizado, nomeadamente para a detecção de movimento, através de diferenças entre duas ou mais imagens consecutivas.

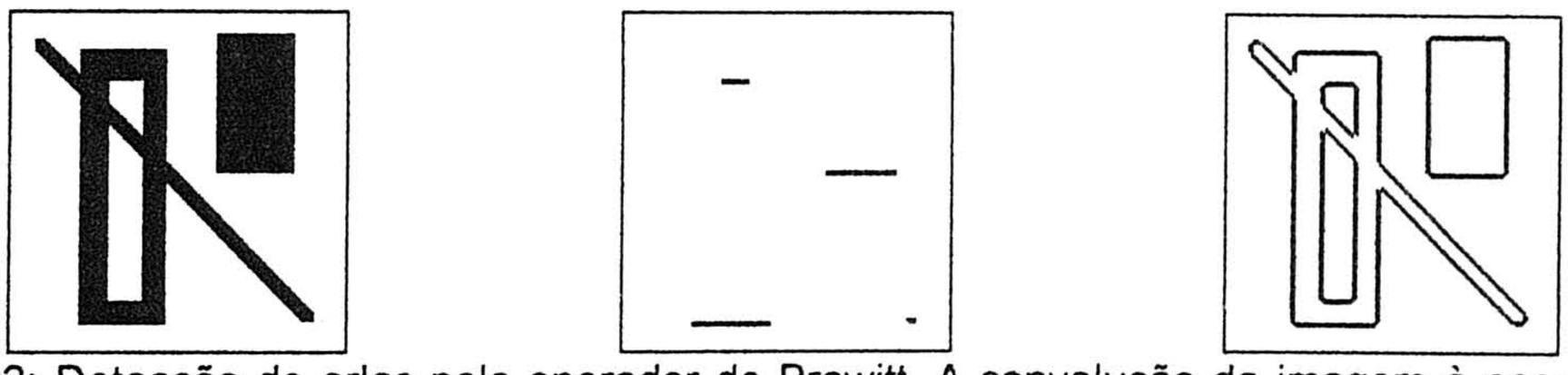


Figura 23: Detecção de orlas pelo operador de Prewitt. A convolução da imagem à esquerda com 1ª janela do operador de Prewitt dá como resultado a imagem central. A imagem à direita corresponde ao resultado final da detecção de orlas.

Sendo $f(x_1,x_2,t)$ a função imagem definida pelas suas coordenadas horizontal x_1 e vertical x_2, no instante t, e f_d a função imagem resultante da diferença entre duas imagens em momentos diferentes, a operação diferença poderá ser equacionada pela expressão [6]:

$$f_d(x_1,x_2,t_1,t_2) = f(x_1,x_2,t_2) - f(x_1,x_2,t_1).$$

A figura 24 ilustra este efeito, podendo verificar-se o movimento do objecto rectangular.

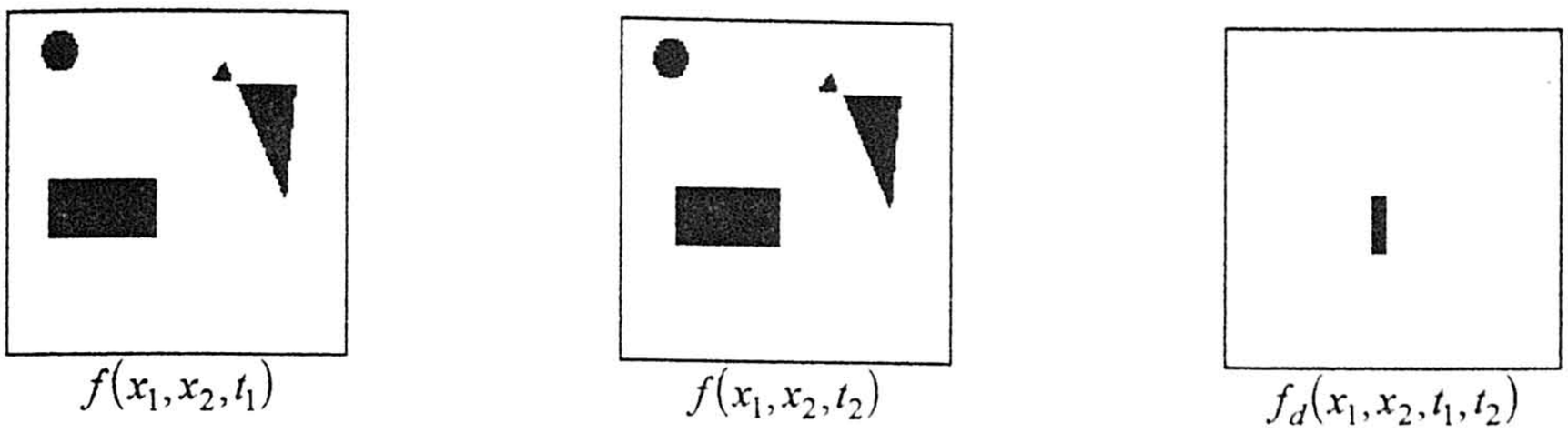


Figura 24: Detecção de movimento por diferença entre duas imagens consecutivas.

A figura 26 ilustra o resultado de várias operações morfológicas em níveis de cinzento, aplicadas sobre a imagem original, representada na figura 25.

Figura 25: Imagem do operador de máquina de filmar.

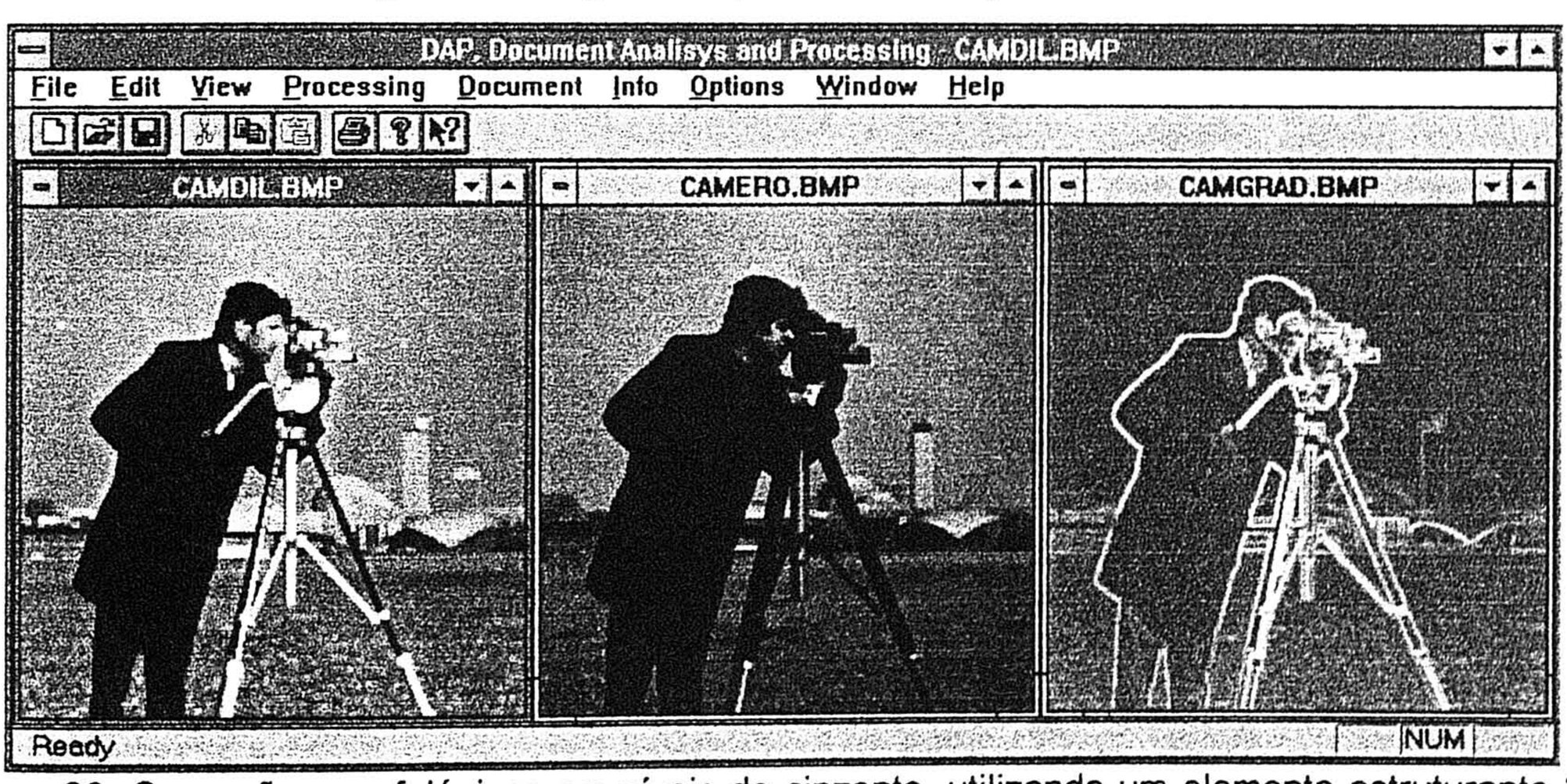


Figura 26: Operações morfológicas em níveis de cinzento, utilizando um elemento estruturante 3x3 com todos os elementos com valor 1. CAMDIL.BMP corresponde à imagem resultante da dilatação, CAMERO.BMP corresponde à imagem resultante da erosão, e CAMGRAD.BMP corresponde à imagem resultante do gradiente [1] $(f \oplus b) - (f \Theta b)$.

O elemento estruturante b utilizado corresponde a uma máscara 3x3 com valores 1. O efeito da operação de dilatação $\oplus$ é o de aumentar o brilho, e reduzir ou eliminar pequenos pormenores escuros. O efeito da operação de erosão Θ é o de diminuir o

brilho, e reduzir ou eliminar pequenos pormenores claros. Finalmente, é visível na figura 26, o efeito do gradiente definido em [1] como $(f \oplus b) - (f \Theta b)$, sendo f a função imagem.

3.4 Considerações Finais

Neste capítulo descreveu-se a arquitectura de um sistema de processamento e análise de imagem construído com o intuito de apoiar o desenvolvimento deste trabalho. O programa referido baseia-se nas classes MFC do Visual C++, e na reutilização de código fonte de processamento e análise de imagem desenvolvido para UNIX. Por último, referiram-se as principais opções de utilização do programa, ilustrando-as com exemplos simples de processamento e análise de imagem.

Este programa serviu de suporte ao desenvolvimento do trabalho que vai ser descrito nos capítulos seguintes. Trabalho de análise de documentos, com processamento inicial de imagens de documentos, separação e classificação de blocos de imagem, e reconhecimento gráfico de desenhos de linhas.

Capítulo 4

ANÁLISE DE DOCUMENTOS DE ENGENHARIA

4. ANÁLISE DE DOCUMENTOS DE ENGENHARIA

No segundo capítulo desta dissertação foi apresentada uma visão geral sobre a análise de documentos, recorrendo para o efeito a algumas publicações relevantes. Neste capítulo pretende-se descrever as metodologias e processos utilizados, no âmbito desta tese de mestrado, para se analisarem documentos de engenharia. Essa análise compreende o processamento e segmentação das imagens documento, e tem como intuito último a obtenção de uma representação vectorial restrita dos desenhos de engenharia, com uma possibilidade de saída para um sistema genérico de desenho.

4.1 AQUISIÇÃO DOS DOCUMENTOS

Como foi descrito no segundo capítulo, na secção referente à aquisição de documentos, a resolução mínima necessária para obter amostras suficientes de pontos impressos, para a análise de documentos de texto usuais é de 240 pontos por polegada. Em determinados casos é aconselhada a aplicação de resoluções superiores, para processar componentes de complexidade superior.

Com base nestes dados foram utilizadas resoluções de 300 dpi e 600 dpi para aquisição dos documentos de engenharia. Considerando fontes de 12 caracteres por polegada, com 0.2 mm por cada ponto impresso, cada caracter terá uma largura aproximada de 0.08 polegadas, o que determina que a largura aproximada de cada caracter, em pontos impressos, seja aproximadamente igual a 10. Se a resolução de aquisição for 300 dpi, vamos obter 2.5 pixels da imagem por cada ponto impresso, ou 25 pixels por caracter. Seguindo o mesmo raciocínio para a resolução de aquisição de 600 dpi, obtemos 50 pixels por caracter. Ambas as resoluções de aquisição são mais do que suficientes para analisar e processar texto, e símbolos, assim como para o processamento de gráficos, onde se exige resoluções ainda inferiores.

A aquisição foi efectuada em 256 níveis de cinzento, e a imagem obtida foi processada com o sistema descrito no capítulo anterior.

4.2 PRÉ-PROCESSAMENTO

Os métodos de aquisição utilizados permitem obter imagens de grande qualidade, dependente fundamentalmente da qualidade do documento fonte. Assim sendo, as operações de processamento inicial restringem-se a operações de redução do ruído, utilizando operadores não lineares, como os de mediana, e de substituição do pixel central da janela pelo nível de cinzento extremo (mínimo ou máximo) mais próximo.

Após a filtragem segue-se a operação de binarização da imagem, obtendo-se objectos negros sobre fundo branco. Para descrever esta operação recorreu-se a uma secção própria, onde serão analisados os resultados.

Outro tipo de processamento inicial utilizado é o morfológico, que permite fechar objectos que devido à aquisição e binarização se tornaram abertos, permitindo igualmente eliminar pontos isolados, que se consideram ruído. Na secção 3.2.1 introduzir-se-ão as operações morfológicas, e descrever-se-à uma metodologia especialmente concebida para a implementação deste tipo de operações.

4.2.1 Operações Morfológicas

A morfologia visa o estudo de formas de objectos, caracterizando a sua topologia (configuração geométrica) e estrutura a partir das suas imagens. É importante notar que no âmbito da topologia se estudam propriedades das imagens que não variam com a deformação (onde se incluem deslocamentos, rotações, e mudanças de escala), desde que não se dêem junções de parte dos objectos, ou mesmo de objectos distintos [1]. Para se caracterizar determinadas imagens, o estudo dos seus atributos geométricos pode ser determinante, como será o caso de desenhos de linhas quando se pretende efectuar o reconhecimento de primitivas gráficas de aproximação das linhas, para posterior alteração recorrendo a programas de desenho.

Para descrever regiões de imagens utilizam-se vários conceitos. Schalkoff [6] e Gonzalez e Woods [1] propõem várias propriedades úteis à descrição, nomeadamente: descritores de elementos geométricos de objectos, como a área e o rectângulo envolvente; momentos geométricos que permitem descrever determinadas características, como o grau de horizontalidade ou verticalidade, ou da divergência horizontal ou vertical; descritores topológicos, como medidas de perímetro, de área, e número de Euler; medidas de textura de natureza estatística, estrutural ou espectral [1]; bordos, esqueletos e cobertura convexa[1] de objectos, que podem ser extraídos recorrendo a operações morfológicas, sendo úteis na descrição e representação de formas de regiões.

Como foi referido, a matemática morfológica pode ser utilizada para extrair informação [1] de componentes da imagem úteis na descrição das suas formas, como sejam os bordos, esqueletos e cobertura convexa dos objectos. Por outro lado, pode ser utilizada para processamento morfológico [1] efectuando filtragem, adelgaçamento ou

---

[1]Do inglês "convex hull"

poda de componentes da imagem, entre outras operações. O processamento morfológico efectua-se utilizando elementos estruturantes através duma operação similar à convolução, com a diferença de se utilizarem operadores da teoria de conjuntos [6] entre o elemento estruturante e a imagem. Nas descrições efectuadas vão-se considerar sempre imagens binárias.

4.2.1.1 Operações Básicas

As operações morfológicas básicas são a erosão e dilatação [1, 6]. Suponha-se um objecto X, representado por um conjunto no espaço a duas dimensões Z^2 de números inteiros, em que cada elemento do conjunto é um vector das coordenadas dos pixels pretos da imagem. Sendo o elemento estruturante B um conjunto de Z^2 (um elemento estruturante usual é representado na figura 27a), $\hat{B}$ corresponde à reflexão do elemento estruturante sobre a origem definida como $\hat{B} = \{x | x = -b_i, b_i \in B\}$ (as coordenadas dos pixels pretos do elemento estruturante tomarão o valor negativo relativamente à origem do elemento), e $\hat{B}_x$ a translação de $\hat{B}$ de tal forma que a sua origem coincida com um ponto x, e sendo x qualquer elemento de Z^2, as operações morfológicas básicas são definidas da seguinte forma:

Erosão, $X \Theta B \stackrel{\Delta}{=} \{x : B_x \subset X\}$;

Dilatação, $X \oplus B \stackrel{\Delta}{=} \{x : \hat{B}_x \cap X \neq \varnothing\}$.

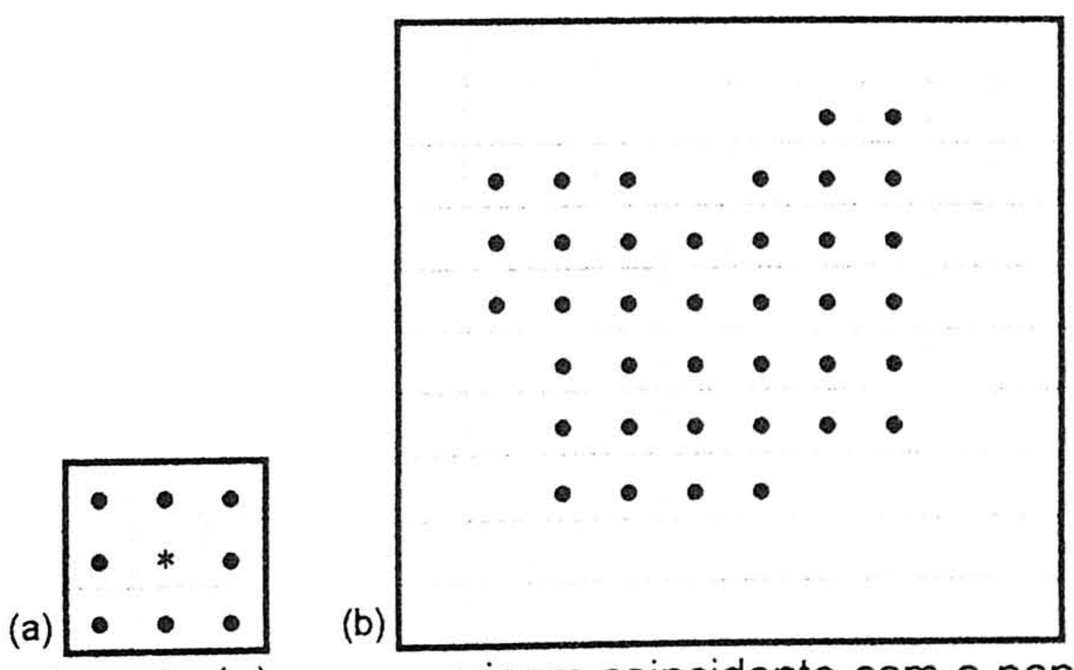


Figura 27: Elemento estruturante (a), com a origem coincidente com o ponto central e imagem original (b), a utilizar na ilustração das operações de dilatação e erosão.

Assim, o resultado da erosão corresponde ao conjunto de pontos da imagem em que o elemento estruturante está totalmente contido no objecto, quando se considera a sua origem coincidente com estes pontos. O resultado da dilatação corresponde aos pontos da imagem em que o elemento estruturante reflectido sobre a origem intersecta o objecto, quando se considera a sua origem coincidente com estes pontos. As operações morfológicas são similares à convolução por se deslocar o elemento estruturante pelos

pontos da imagem, com a diferença de se utilizarem operações da teoria de conjuntos entre o elemento estruturante e a imagem.

Nas figuras 28 e 29 pode observar-se o resultado das operações de erosão e dilatação, aplicadas sobre a imagem original ilustrada pela figura 27b, utilizando o elemento estruturante representado na figura 27a. Os pontos circulares representam o resultado da operação e o padrão cinzento a imagem original. Pode concluir-se que as operações de erosão e dilatação são respectivamente operações de diminuição e expansão de objectos. No caso particular da utilização deste elemento estruturante (figura 27a) correspondem mesmo a operações de mínimo e máximo. Outros elementos estruturantes darão resultados diversos, como se ilustrará posteriormente.

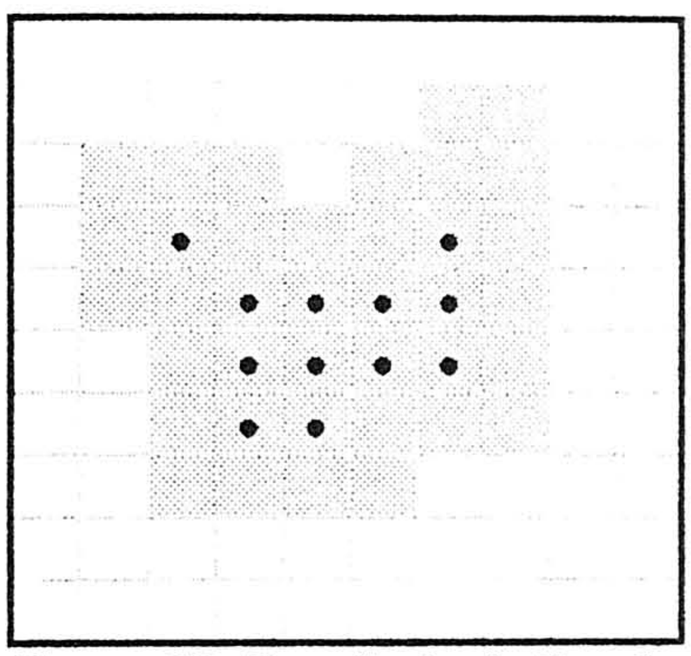


Figura 28: Resultado da Erosão.

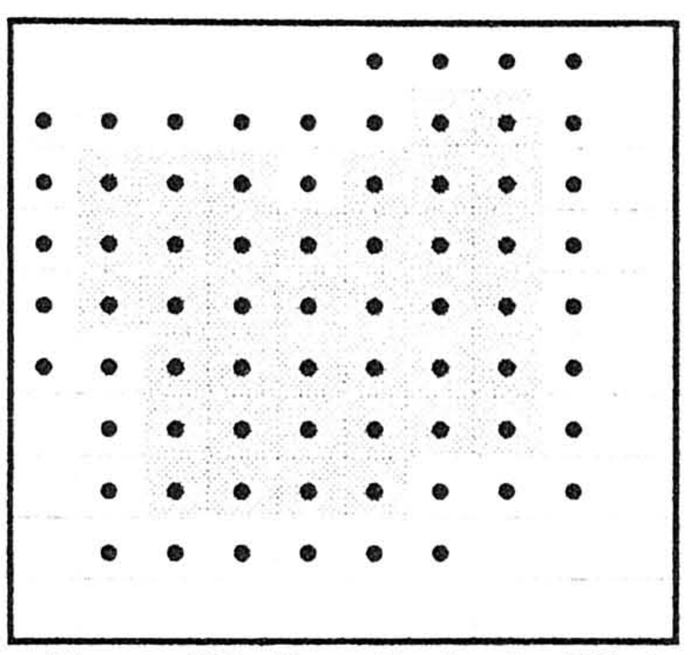


Figura 29: Resultado da Dilatação

Embora estas duas operações morfológicas não sejam inversas, são duais uma da outra relativamente ao complemento e reflexão de conjuntos. Considerando a reflexão de um conjunto definida como $\hat{A} = \{x | x = -a_i, a_i \in A\}$, e o complemento como $A^c = \{x | x \notin A\}$, representando x qualquer elemento de Z^2, as operações duais da erosão e dilatação serão definidas respectivamente por:

$$X \Theta B = \left(X^c \oplus \hat{B}\right)^c,$$

$$X \oplus B = \left(X^c \Theta \hat{B}\right)^c.$$

4.2.1.2 Operações Morfológicas

Neste parágrafo descrevem-se outras funções morfológicas, apresentando as suas definições, e ilustrando algumas destas operações com imagens.

Abertura

A abertura é constituída por uma erosão seguida duma dilatação, tendo o efeito de reduzir o objecto e depois aumentá-lo. A principal característica desta operação é a de quebrar ligações finas entre objectos, eliminar saliências finas e eliminar pequenos objectos (de dimensão horizontal ou vertical inferior à do elemento estruturante).

A abertura morfológica é definida por:

$$A \circ B = (A \ominus B) \oplus B.$$

Fecho

O fecho é constituído por uma dilatação seguida duma erosão, tendo o efeito de aumentar o objecto e depois reduzi-lo. A principal característica desta operação é a de ligar pixels que estejam suficientemente perto uns dos outros, e preencher pequenos buracos, de dimensão horizontal ou vertical inferior à do elemento estruturante. Como buracos consideram-se regiões do fundo completamente rodeadas por pixels do objecto, isto é, regiões em que nenhum dos seus pontos pertence aos extremos da imagem e em que todos os pontos do bordo são vizinhos de pixels do objecto.

O fecho morfológico é definido por:

$$A \bullet B = (A \oplus B) \ominus B.$$

Na figura 30 pode ver-se o efeito das operações de abertura e de fecho. A abertura provoca a eliminação do ponto existente no canto superior direito da imagem, e a quebra de ligação existente entre os dois triângulos. Devido à utilização do elemento estruturante da figura 27a, os cantos que não terminam exactamente duma forma quadrada são filtrados. Por efeito da operação de fecho o buraco existente num dos quadrados é eliminado, passando a ser preenchido por pixels do objecto, e a elipse passa a estar ligada ao componente definido pelos dois triângulos, .

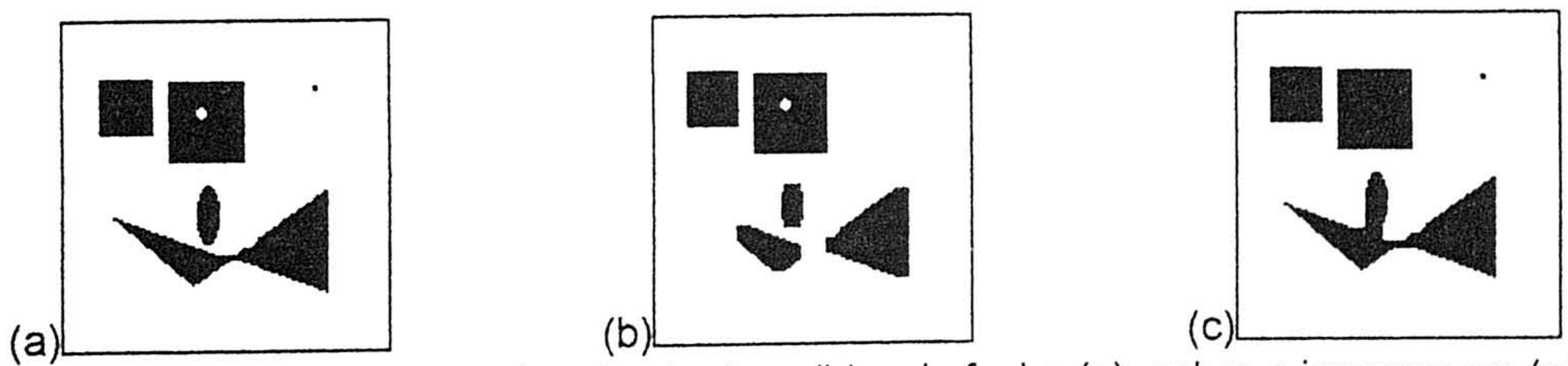


Figura 30: Ilustração de operações de abertura (b) e de fecho (c), sobre a imagem em (a).

Acerto ou Falha[2]

A operação de acerto ou falha efectua a correspondência entre uma forma e a imagem, utilizando dois elementos estruturantes. Um dos elementos estruturantes (B_1) corresponde aos pontos do objecto do qual se pretende encontrar uma correspondência, e o outro (B_2) corresponde aos pontos do fundo da imagem. Se o elemento estruturante B_1 coincidir com um ponto x, e houver coincidência entre os pixels pretos de B_1 e os pontos da imagem na vizinhança de x (definida pelo elemento estruturante), e simultaneamente se houver coincidência entre os pontos do fundo da imagem e os

[2]Do inglês "hit-or-miss"

pontos na vizinhança de x pertencentes ao fundo, então x será um ponto no qual se verificou correspondência entre uma forma e a imagem, através da operação de acerto ou falha.

Definição de acerto ou falha:

$$A * B = (A \Theta B_1) \cap \left(A^c \Theta B_2\right).$$

Adelgaçamento

A operação de adelgaçamento permite tornar um objecto mais fino, retirando os pixels do bordo do objecto. Esta operação pode ser definida, como o fizeram Gonzalez e Woods [1], em função da intersecção entre a imagem original e o complemento dos pontos envolventes:

$$A \otimes B = A \cap (A * B)^c.$$

Esta definição seria suficiente se, com um único elemento estruturante se conseguissem representar todas as formas correspondentes a pontos envolventes. Não sendo possível, utiliza-se um conjunto de elementos estruturantes

$$\{B\} = \left\{B^1, B^2, ..., B^n\right\},$$

simétricos, que aplicados em sequência, permitem detectar todos os pontos envolventes do objecto.

Assim sendo a definição de adelgaçamento será:

$$A \otimes \{B\} = \left(\left(...\left(\left(A \otimes B^1\right) \otimes B^2\right)...\right) \otimes B^n\right).$$

Os mesmos autores apresentam um conjunto de oito elementos estruturantes, representados na figura 31, que permitem efectuar esta operação. Os pontos negros correspondem a pixels pertencentes ao objecto, os quadrados vazios correspondem a pixels do fundo da imagem, e os quadrados com um × correspondem a pixels que não interessam, isto é, que tanto podem pertencer ao fundo como ao objecto.

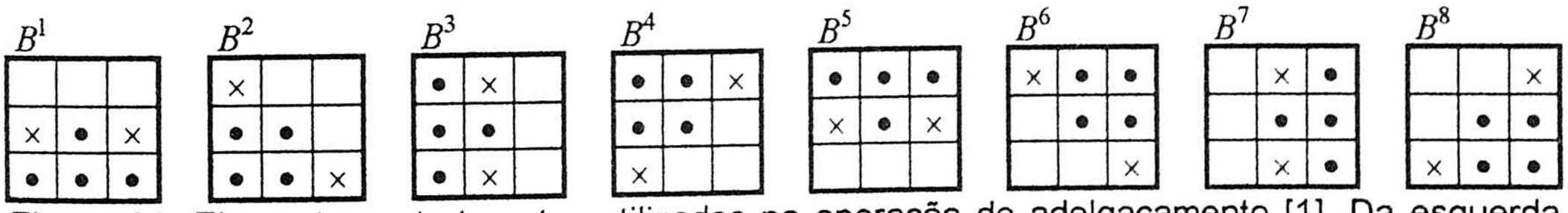


Figura 31: Elementos estruturantes utilizados na operação de adelgaçamento [1]. Da esquerda para a direita os elementos estruturantes B^i são iguais a B^{i-1} rodados de 45º no sentido contrário ao dos ponteiros do relógio.

4.2.1.3 Implementação

As operações morfológicas, como é indicado no primeiro capítulo desta tese, permitem, entre outras transformações, eliminar ruído, preencher falhas, e retirar ramos parasitas resultantes do adelgaçamento de linhas. Para implementar as operações

morfológicas matemáticas sobre imagens binárias foi especialmente concebido um método inspirado numa abordagem de processamento paralelo proposta por Boomgaard e Balen [35]. A principal característica do método reside na representação de cada pixel por um bit numa palavra de 32 bits, o que permite, com o recurso a operações lógicas, processar simultaneamente 32 pixels.

Com base neste artigo foi especialmente desenvolvido um método que permite transformar imagens utilizando operações de erosão, dilatação e acerto ou falha. No entanto, não se recorreu ao formato de imagem proposto em [35], por exigir o desenvolvimento de código específico para acesso e manipulação de pixels e para transformação de imagens. Para além disso, não é desejável introduzir um novo formato de representação de imagens, pois existem já dois formatos diferentes (DIB e BAND). Por outro lado, o recurso a vectores (nos quais se baseia o formato proposto) de dimensões superiores a um segmento (64 kbytes) é impossível em MsDos, e a utilização de estruturas alternativas iria provocar sobrecarga de processamento e perda de significado na utilização do algoritmo descrito no artigo. Tudo isto tornaria o programa menos estruturado e menos compreensível.

Assim sendo, o mapa de bits DIB é transformado numa imagem BAND, constituída por palavras de 16 bits sem sinal, em que cada bit corresponde a um pixel da imagem. A utilização de palavras de 16 bits em vez de palavras de 32 bits deveu-se ao facto de, nas imagens de formato BAND, esse ser o tamanho de números inteiros de maior resolução, utilizados para definir pontos da imagem. Uma imagem de 256x256 pixels está representada na figura 32, sendo representada por 256 linhas de 16 palavras de 16 bits. Na figura 32 está representada a palavra da linha 2, coluna 3 que contém o pixel de coordenadas (33,2) no seu bit mais significativo, e o pixel de coordenadas (48,2) no seu bit menos significativo.

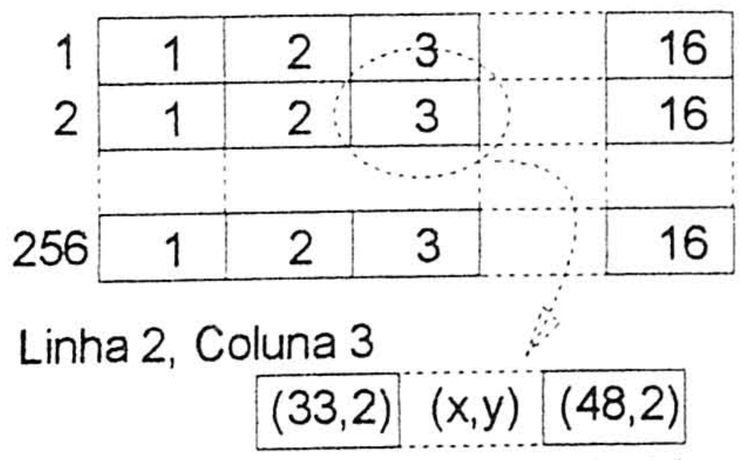


Figura 32: Imagem BAND correspondendo a um mapa de bits, de uma imagem binária com 256x256 pixels. Estão representadas as coordenadas dos pixels contidos na palavra da linha 2, coluna 3.

Como foi referido anteriormente, na dilatação de uma imagem, se o elemento estruturante intersectar pontos do objecto ao fazer coincidir a sua origem com um determinado ponto da imagem, então este ponto da imagem passará também ele a

pertencer ao objecto. Aqui consideram-se os pontos do objecto representados pelo *um* lógico, os pontos do fundo pelo *zero* lógico, e a origem do elemento estruturante o ponto central.

Considera-se, para efeitos de compreensão um elemento estruturante de dimensão 3x3 contendo um *1* lógico em todos os seus 9 elementos. Um ponto da imagem resultado pertencerá ao objecto por dilatação morfológica, se a sua disjunção com todos os seus 8 vizinhos for diferente do *zero* lógico. Se algum dos valores do elemento estruturante for diferente de *um*, basta não considerar a disjunção com o vizinho correspondente.

Este raciocínio torna-se mais complexo se houver uma correspondência de 1 bit para 1 pixel, e se não se pretender ler e analisar bits individuais, mas sim processá-los em conjunto, 16 duma vez. Considere-se b_{in} a imagem a processar, B o elemento estruturante 3x3 com todos os valores iguais a *um*, e (i, j) o vector das coordenadas horizontal e vertical de uma palavra de 16 bits da imagem, a imagem resultante da dilatação nesse ponto é definida por:

$$
\begin{aligned}
b_{out} = {} & b_{in}(i,j) \mid b_{in}(i,j-1) \mid b_{in}(i,j+1) \mid \\
& [(b_{in}(i,j) >> 1) \mid (b_{in}(i-1,j) << 15)] \mid \\
& [(b_{in}(i,j) << 1) \mid (b_{in}(i+1,j) >> 15)] \mid \\
& [(b_{in}(i,j-1) >> 1) \mid (b_{in}(i-1,j-1) << 15)] \mid \\
& [(b_{in}(i,j-1) << 1) \mid (b_{in}(i+1,j-1) >> 15)] \mid \\
& [(b_{in}(i,j+1) >> 1) \mid (b_{in}(i-1,j+1) << 15)] \mid \\
& [(b_{in}(i,j+1) << 1) \mid (b_{in}(i+1,j+1) >> 15)]
\end{aligned}
$$

>>, operação de deslocamento dos bits à direita

<<, operação de deslocamento dos bits à esquerda

|, operação de disjunção ao nível do bit

A primeira linha desta equação corresponde às disjunções entre a palavra (os pixels) em processamento e as palavras (os pixels) imediatamente acima e abaixo (figura 33c). Na segunda linha representa-se a disjunção com os pixels a oeste (figura 33d); na terceira linha a disjunção com os pixels a leste (figura 33e); na quarta linha a disjunção com os pixels a noroeste (figura 33f); na quinta linha a disjunção com os pixels a nordeste (figura 33g); na sexta linha a disjunção com os pixels a sudoeste (figura 33h); e finalmente na sétima linha a disjunção com os pixels a sudeste (figura 33i).

O resultado duma erosão corresponde, como já foi referido, a todos os pontos da imagem em que o elemento estruturante está totalmente contido no objecto quando a sua origem coincide com o ponto em análise.

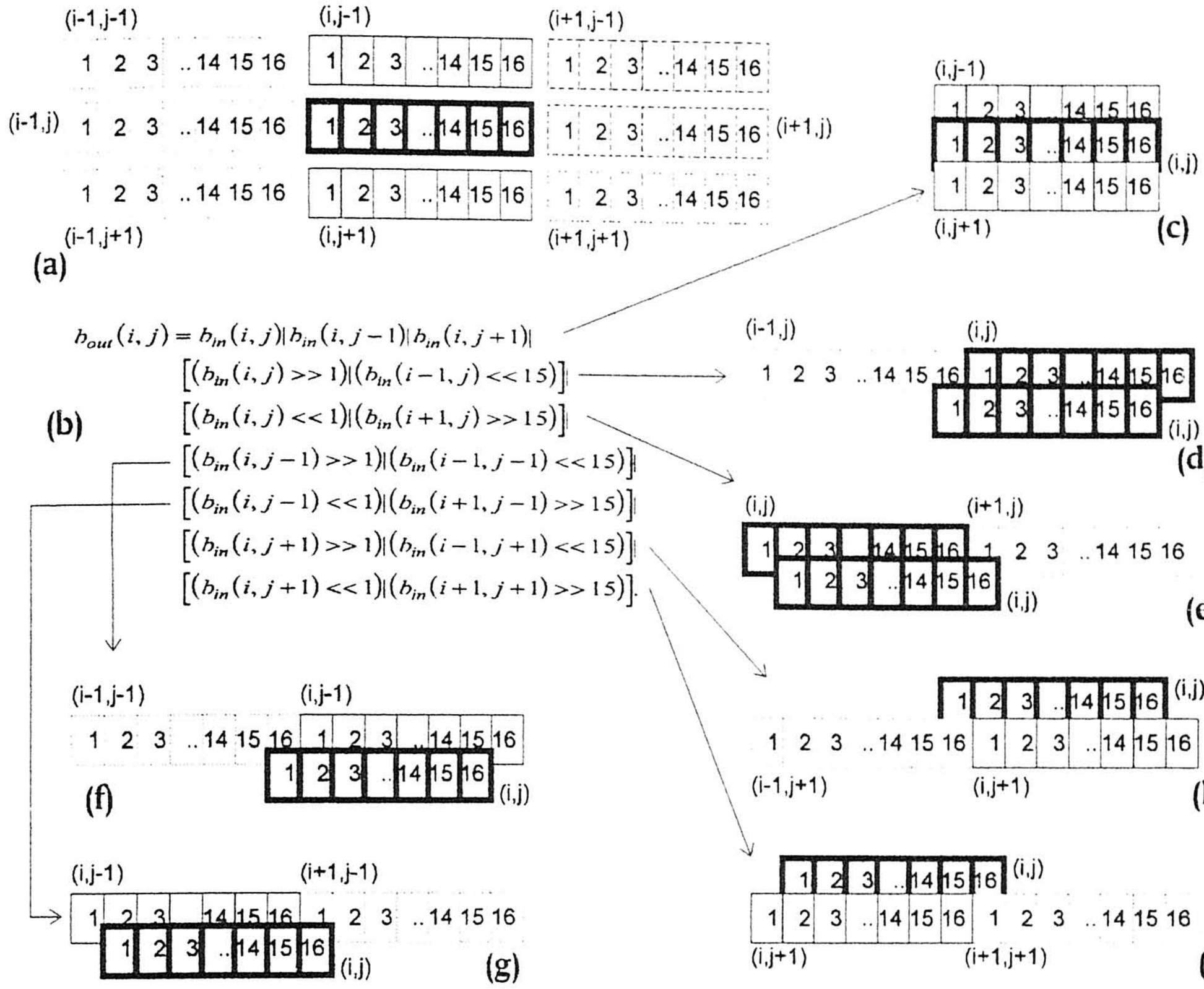


Figura 33: Esta figura relaciona a equação que define a dilatação, com a imagem, isto é, relaciona as palavras vizinhas com aquela que está em processamento (não esquecer que o elemento estruturante é todo ele constituído por valores com *um* lógico). (a) Representa a palavra em processamento e as suas vizinhas, com os bits que as constituem. (b) Equação da dilatação. (c) Relação entre a imagem e a primeira linha da equação; disjunção dos pixels em análise com os seus vizinhos a norte e a sul. A partir daqui considerar a palavra (i, j) como sendo o resultado do processamento da alínea anterior. (d) Segunda linha da equação; disjunção com os vizinhos a oeste. (e) Terceira linha da equação; disjunção com os vizinhos a leste. (f)Quarta linha da equação; disjunção com os vizinhos a noroeste. (g) Quinta linha da equação; disjunção com os vizinhos a nordeste. (h) Sexta linha da equação; disjunção com os vizinhos a sudoeste. (i) Sétima linha da equação; disjunção com os vizinhos a sudeste.

Considere-se os pontos do objecto representados pelo *um* lógico, os pontos do fundo pelo *zero* lógico, a origem do elemento estruturante o ponto central, e um elemento estruturante de dimensão 3x3 contendo um *1* lógico em todos os seus 9 elementos. Um ponto da imagem resultado, pertencerá ao objecto por erosão morfológica, se a sua conjunção com todos os seus 8 vizinhos for igual ao *um* lógico. Se algum dos valores do elemento estruturante for diferente de *um*, basta não considerar a conjunção com o vizinho correspondente.

Se considerarmos b_{in} a imagem a processar, B o elemento estruturante 3x3 com todos os valores iguais a *um*, e (i, j) respectivamente as coordenadas horizontal e vertical

de uma palavra de 16 bits da imagem, a imagem resultante da dilatação nesse ponto será definida por:

$$b_{out} = b_{in}(i,j) \& b_{in}(i,j-1) \& b_{in}(i,j+1) \;\&$$
$$[(b_{in}(i,j) >> 1) | (b_{in}(i-1,j) << 15)] \;\&$$
$$[(b_{in}(i,j) << 1) | (b_{in}(i+1,j) >> 15)] \;\&$$
$$[(b_{in}(i,j-1) >> 1) | (b_{in}(i-1,j-1) << 15)] \;\&$$
$$[(b_{in}(i,j-1) << 1) | (b_{in}(i+1,j-1) >> 15)] \;\&$$
$$[(b_{in}(i,j+1) >> 1) | (b_{in}(i-1,j+1) << 15)] \;\&$$
$$[(b_{in}(i,j+1) << 1) | (b_{in}(i+1,j+1) >> 15)]$$

>>, operação de deslocamento dos bits à direita

<<, operação de deslocamento dos bits à esquerda

&, operação de conjunção ao nível do bit

Assim, como é visível, esta operação é muito semelhante à dilatação por substituição dos operadores de disjunção (|) por operadores de conjunção (&).

Tendo sido implementadas as operações morfológicas básicas de dilatação e erosão, é possível definir qualquer conjunto de outras operações morfológicas, como as operações de abertura, fecho e acerto ou falha, que são operações disponíveis no programa descrito no capítulo anterior.

4.3 BINARIZAÇÃO

Os documentos em questão são fundamentalmente constituídos por um fundo branco, com desenhos e texto em negro, sendo as imagens adquiridas fundamentalmente bimodais. A imagem do documento foi adquirida em níveis de cinzento, com o intuito de ser o sistema a controlar a binarização da imagem, não deixando esse papel ao programa de aquisição.

De entre os muitos processos de binarização de imagens escolheram-se dois métodos. O método de Otsu [32] que Sahoo et al. [33] referem como sendo um dos melhores métodos de binarização quando a imagem é predominantemente bimodal, e as duas classes têm distribuições do nível de cinzento com variâncias aproximadamente iguais. Estes resultados foram confirmados no decorrer deste trabalho, como se pode comprovar com os resultados ilustrados. O outro método escolhido foi o de máxima entropia de Wong e Sahoo [34], cujos resultados dependem não só da distribuição dos níveis de cinzento, mas também das formas das imagens.

4.3.1 Introdução

A binarização é uma técnica de segmentação em que se selecciona um único limiar de classificação, efectuando-se em seguida uma transformação dos níveis de cinzento da imagem, para dois níveis únicos (os pixels da imagem pertencerão ao objecto ou ao fundo). Esta transformação pode ser efectuada por diversas razões [6], nomeadamente:

- Para identificar objectos numa imagem, separando-os do fundo, para detecção de alvos, por exemplo;
- Por pretendermos analisar a forma de objectos, sendo menos importante, neste caso, a intensidade dos pixels do que a preservação da forma dos objectos;
- Para visualizar uma imagem num dispositivo com uma resolução de intensidade de um bit, como uma impressora, tentando manter a aparência da imagem original;
- Para converter uma imagem em que as orlas foram realçados, de forma a obter uma imagem de desenhos de linhas que representem os bordos de objectos, sendo necessário distinguir orlas correspondentes aos extremos dos objectos, e orlas resultantes de sombras ou de variações de iluminação.

Algumas das aplicações desta técnica são: reprodução *meio tom*, reconhecimento automático de alvos, projecto de sistemas de navegação visual para veículos de terra autónomos, aplicações industriais de visão por computador, e análise de imagens de medicina.

Técnicas similares permitem, em vez de se seleccionar um único limiar, seleccionar vários limiares (para, por exemplo, se separarem vários objectos, em diferentes níveis de cinzento, dentro da mesma imagem), denominando-se estas técnicas por limiarização multi-nível. Este tipo de segmentação não foi aplicada nos documentos em análise, por não fazer sentido, já que não existem, em termos cromáticos mais do que dois objectos. Objectos estes que correspondem a pontos impressos sobre papel.

Em geral, as técnicas de selecção dum nível de cinzento limiar, para binarização da imagem são divididas em dois grupos, nomeadamente, binarização global e binarização local. A binarização global é aquela em que se actua sobre toda a imagem, com um único limiar de classificação, enquanto na local, se divide a imagem e se determina um limiar para cada parte da imagem. Por outro lado, as técnicas de binarização podem ainda ser classificadas como pontuais ou regionais. Será pontual, se o limiar de binarização for determinado com base apenas no nível de intensidade de cada pixel, independentemente das características dos seus vizinhos. Será regional, se o limiar de

binarização for determinado com base numa propriedade local (distribuição do nível de cinzento local por exemplo) relacionada com a vizinhança do pixel.

Neste ponto vai ser definida a notação utilizada para representar a imagem, e a binarização da mesma. Seja N o conjunto dos números naturais, (x,y) as coordenadas espaciais duma imagem digitalizada, e $\mathrm{N}_l=\{0,1,...,l-1\}$ o conjunto dos níveis de cinzento. A imagem corresponderá a uma aplicação $f\colon \mathrm{N}\times\mathrm{N}\to\mathrm{N}_l$, sendo $f(x,y)$ o nível de cinzento do pixel de coordenadas (x,y).

Seja $A=\{a_1,a_2\}$ um par de níveis de cinzento com $a_1,a_2\in\mathrm{N}_l$. O resultado de binarizar a função da imagem $f(x,y)$ no limiar $t\in\mathrm{N}_l$ corresponde à aplicação $f_t\colon \mathrm{N}\times\mathrm{N}\to A$, sendo a intensidade do pixel nas coordenadas (x,y) dada por:

$$f_t(x,y)=\begin{cases}a_1, & f(x,y)\le t\\ a_2, & f(x,y)>t\end{cases}.$$

Sendo n_i o número de pixels da imagem com nível de cinzento i, o número total de pixels duma imagem será $n=\sum_{i=0}^{l-1}n_i$. A probabilidade de ocorrência do nível de cinzento i será $p_i=n_i/n$.

4.3.2 Binarização de Documentos pelo Método de 'Otsu'

Neste método Otsu [32] considera imagens com histogramas bimodais, correspondendo a cada moda as classes C_0 e C_1. Otsu utiliza um conjunto de variâncias para definir estas classes, o que lhe permite seleccionar um nível de cinzento limiar t para efectuar a segmentação da imagem.

Considerando $\omega_0=\sum_{i=0}^{t}p_i$ a probabilidade da classe C_0, $\omega_1=1-\omega_0$ a probabilidade da classe C_1, $\mu_T=\sum_{i=0}^{l-1}i\cdot p_i$ a média total, e $\mu_t=\sum_{i=0}^{t}i\cdot p_i$, as médias da classe C_0 e da classe C_1 serão definidas por:

$$\mu_0=\frac{\mu_t}{\omega_0},\qquad \mu_1=\frac{\mu_T-\mu_t}{\omega_1}.$$

As variâncias das duas classes também pesadas das suas probabilidades são dadas pelas seguintes expressões:

$$\sigma_0=\frac{\sum_{i=0}^{t}(i-\mu_0)^2\cdot p_i}{\omega_0},\qquad \sigma_1=\frac{\sum_{i=t+1}^{l-1}(i-\mu_1)^2\cdot p_i}{\omega_1}.$$

As variâncias intra-classes ${\sigma_W}^2$, inter-classes ${\sigma_B}^2$, e total ${\sigma_T}^2$, são expressas por:

$${\sigma_B}^2=\omega_0\cdot\omega_1\cdot(\mu_0-\mu_1)^2,\qquad {\sigma_W}^2=\omega_0\cdot{\sigma_0}^2+\omega_1\cdot{\sigma_1}^2,\qquad {\sigma_T}^2=\sum_{i=0}^{l-1}(i-\mu_T)^2\cdot p_i.$$

Com base nestas variâncias, 'Otsu' definiu três medidas cuja maximização, relativa ao nível de cinzento, permite encontrar um limiar óptimo de binarização:

$$\lambda = \frac{{\sigma_B}^2}{{\sigma_w}^2}, \qquad \eta = \frac{{\sigma_B}^2}{{\sigma_T}^2}, \qquad \kappa = \frac{{\sigma_T}^2}{{\sigma_W}^2}.$$

O problema de determinação de um limiar óptimo de binarização é reduzido ao problema de encontrar um nível de cinzento que optimize um destes critérios. Sendo σ_T constante relativamente ao limiar t, a maximização de η é equivalente à maximização de σ_B. Também o critério κ exige apenas a minimização de σ_W, sendo no entanto, mais complexo, porque além de exigir o cálculo de μ_0 e μ_1 como σ_B, também exige o cálculo de σ_0 e σ_1, para todos os níveis de cinzento.

Assim, o limiar óptimo t^* de binarização, segundo Otsu, corresponde à variável (Arg) que maximiza (Max), para todos os níveis de cinzento, a função η:

$$t^* = Arg \underset{t \in \Gamma}{Max}\, \eta .$$

4.3.3 Binarização de Documentos pelo Método de Máxima Entropia

Este método de selecção dum nível de cinzento limiar para segmentação de imagens, de Wong e Sahoo [34], baseia-se no princípio de máxima entropia. O limiar óptimo é determinado por maximização da entropia *a posteriori*, sujeita a determinadas restrições, que são derivadas de medidas especiais que caracterizam a uniformidade e a forma de regiões da imagem. Para se atingir este objectivo é utilizada a distribuição de níveis de cinzento e a informação espacial da imagem.

Os métodos pontuais de binarização global, relativamente aos regionais, são comparativamente mais simples (determinando o limiar de binarização em função do histograma da imagem, não recorrendo a nenhum tipo de informação sobre a relação entre pixels vizinhos, como por exemplo a média local de níveis de cinzento), mas utilizam apenas uma pequena parte da informação independente fornecida pela imagem. Desta forma, estas técnicas nem sempre serão úteis. Assim sendo, este método propõe a selecção do limiar de binarização utilizando tanto a distribuição dos níveis de cinzento da imagem, como a sua informação espacial, recorrendo a medidas de uniformidade e de forma de regiões.

Segundo o princípio de máxima entropia [34], para uma determinada quantidade de informação, a distribuição de probabilidade que melhor descreve o nosso conhecimento, é aquela que maximiza a entropia de Shannon sujeita a restrições, impostas pela informação conhecida.

Depois de binarizada, uma imagem segmentada no limiar t transforma-se numa imagem de dois níveis. Sendo $F(t) = \sum_{i=0}^{l-1} p_i$ a probabilidade *a posteriori* dos pixels da

imagem com nível de cinzento inferior a t, $F'(t)=1-F(t)$ a probabilidade a posteriori dos pixels da imagem com nível de cinzento superior a t, e assumindo $0\cdot\log_2 0=0$, a entropia de Shannon da imagem binarizada será:

$$H(F(t))=-F(t)\cdot\log_2 F(t)-(1-F(t))\cdot\log_2(1-F(t)).$$

Se não existisse mais nenhuma informação sobre a imagem, procedia-se à maximização da entropia de Shannon para obter o limiar de binarização óptimo. No entanto, a maximização referida, efectuada sem restrições, dá o resultado $F(t)=0.5$. Ou seja, se não houver mais nenhuma informação sobre a imagem, o princípio de máxima entropia sugere um limiar de binarização tal que a percentagem de pixels brancos e pretos seja igual. Este é precisamente o caso em que o histograma é bimodal, com contribuições (número de pixels) iguais de cada modo. No entanto, nem sempre isto se verifica, devendo-se incorporar no processo de selecção do limiar de binarização, mais informação respeitante à imagem.

A informação obtida da distribuição espacial de níveis de cinzento desempenha um papel importante em muitas tarefas de processamento de imagem, como em detecção de orlas, análise de textura e restauração de imagem. Em imagens digitais, a uniformidade de níveis de cinzento e a forma de regiões são aspectos determinantes na separação de objectos do fundo da imagem. A concordância destes dois aspectos entre as imagens originais e as imagens binarizadas é quantificada, pelas medidas de uniformidade e de forma. Estas medidas fornecem informação adicional para a determinação do limiar de informação pelo método de máxima entropia.

4.3.3.1 Medida de Uniformidade

A medida de uniformidade foi originalmente introduzida para avaliar a performance de métodos de segmentação, baseando-se na noção intuitiva de que a uniformidade duma região é tanto maior quanto menor for a variância dos valores da característica dos pixels a ser medida.

Supondo que uma imagem é segmentada em duas classes C_0 e C_1 por binarização no nível de cinzento t, sendo n_i o número de pixels com nível de cinzento i, e sendo μ_0 e μ_1 as médias das duas classes, as variâncias das classes são definidas por:

$${\sigma_0}^2=\sum_{(x,y)\in C_0}(f(x,y)-\mu_0)^2=\sum_{i=0}^{t}(i-\mu_0)^2\cdot n_i,\qquad {\sigma_1}^2=\sum_{(x,y)\in C_1}(f(x,y)-\mu_1)^2=\sum_{i=t+1}^{l-1}(i-\mu_1)^2\cdot n_i.$$

Seja C um factor de normalização, então a medida de uniformidade é definida por:

$$U(t)=1-\frac{{\sigma_1}^2+{\sigma_2}^2}{C}.$$

Dado que a uniformidade regional é uma característica com interesse na segmentação de imagem, poder-se-ía efectuar a binarização baseando-a simplesmente

neste critério, ou seja, o limiar t_1 corresponderia ao argumento que maximizaria a função de uniformidade:

$$t_1 = Arg \underset{t \in N_l}{Max} U(t).$$

O limiar escolhido desta forma, maximiza a uniformidade da imagem binarizada, ou seja, minimiza a soma das variâncias das duas classes segmentadas por este limiar.

4.3.3.2 Medida de Forma

O problema de encontrar um limiar de binarização óptimo para uma imagem poderá estar relacionado com o problema de encontrar orlas ou características relacionadas com a forma dum objecto que o torne distinto do fundo. A forma de um objecto numa imagem é, portanto, uma característica com interesse para separar objectos do fundo da imagem. Uma medida que fornece informação relacionada com a forma (ou seja, com as orlas) dum objecto pode ser construída recorrendo a uma medida de gradiente.

Wong e Sahoo [34] definiram uma medida de gradiente para todos os pixels, definida a partir de várias diferenças D_k entre dois pixels vizinhos do ponto de coordenadas (x, y), considerados em sentidos opostos. A medida de gradiente é dada pela equação:

$$\Delta(x,y) = \sqrt{\sum_{k=1}^{4} {D_k}^2 + \sqrt{2} \cdot D_1 \cdot (D_3 + D_4) - \sqrt{2} \cdot D_2 \cdot (D_3 - D_4)},$$

com $D_1 = f(x+1,y) - f(x-1,y)$, $D_2 = f(x,y-1) - f(x,y+1)$, $D_3 = f(x+1,y+1) - f(x-1,y-1)$, e $D_4 = f(x+1,y-1) - f(x-1,y+1)$.

Sendo C um factor de normalização, e $\overline{f_{N(x,y)}}$ a média dos níveis de cinzento na região de vizinhança $N(x,y)$, determina-se a medida de forma pela expressão:

$$S(t) = \frac{\sum_{(x,y)} sgn\left(f(x,y) - \overline{f_{N(x,y)}}\right) \cdot \Delta(x,y) \cdot sgn\left(f(x,y) - t\right)}{C},$$

com $sgn(x) = \begin{cases} 1 & , x \geq 0 \\ -1 & , x < 0 \end{cases}$.

Nesta medida de forma os gradientes dos pixels pertencentes a C_0 serão considerados positivos e dos pixels pertencentes a C_1 serão considerados negativos $(sgn(f(x,y) - t))$. A medida de forma corresponde a uma soma dos gradientes de todos os pixels, gradientes estes que serão afectados pelo sinal positivo (negativo) se o seu nível de cinzento for superior (não superior) ao nível médio dos seus vizinhos $(sgn(f(x,y) - \bar{f}_{N(x,y)}))$.

O nível de cinzento (argumento) que maximiza a medida de forma, corresponde a um critério de selecção de um limiar de binarização, definido por:

$$t_2 = Arg \underset{t \in N_l}{Max} S(t).$$

De acordo com este critério, se a imagem for binarizada em t_2 reterá o máximo de informação relacionada com as orlas, já que o máximo de $S(t)$ encoraja a segmentação da imagem em pixels com valores elevados de gradiente.

4.3.3.3 **Selecção do Limiar de Binarização**

Segundo este método, assume-se que um limiar de binarização óptimo deve, dentro do possível, originar imagens que retenham o máximo da informação sobre a uniformidade e a forma. Sendo $F(t_1)$ e $F(t_2)$ as probabilidades dos pixels terem valores de nível de cinzento inferiores aos limiares t_1 e t_2 (diferentes um do outro), o limiar t estará compreendido entre aqueles dois limiares, isto é:

$$\min(t_1, t_2) \le t \le \max(t_1, t_2).$$

Ao limiar t escolhido entre t_1 e t_2 corresponderá a probabilidade $F(t)$ que, por definição, é uma função crescente em t, estando compreendida entre:

$$\min\big(F(t_1), F(t_2)\big) \le F(t) \le \max\big(F(t_1), F(t_2)\big).$$

Estas inequações fornecem restrições para o problema mencionado, de maximização da entropia *a posteriori*. O limiar óptimo de binarização, será então definido, a partir da entropia $H(F(t))$, por:

$$t^* = Arg \underset{t \in N_l}{Max}\, H\big(F(t)\big),$$

de tal forma que $\min(t_1, t_2) \le t^* \le \max(t_1, t_2)$.

Sendo t_e o nível de cinzento em que $F(t_e) = 0.5$, ou seja, em que a entropia de Shannon é máxima, o limiar de binarização definido será determinado por:

$$t^* = \begin{cases} \max(t_1, t_2), & F(t_1) \in [0, 0.5] \;,\; F(t_2) \in [0, 0.5] \\ \min(t_1, t_2), & F(t_1) \in [0.5, 1] \;,\; F(t_2) \in [0.5, 1] \\ t_e, & F(t_1) \in [0, 0.5] \;,\; F(t_2) \in [0.5, 1] \\ t_e, & F(t_1) \in [0.5, 1] \;,\; F(t_2) \in [0, 0.5] \end{cases}.$$

O limiar de binarização corresponderá à entropia máxima se, entre as entropias de t_1 e t_2 se encontrar a entropia máxima (duas equações de baixo). Se as entropias de t_1 e t_2 estiverem abaixo da máxima, então o limiar óptimo toma o valor do máximo dentre t_1 e t_2, que corresponde ao máximo da entropia para estes dois limiares. Se as entropias de t_1 e t_2 estiverem acima da máxima, então o limiar óptimo toma o valor do mínimo dentre t_1 e t_2, que corresponde ao máximo da entropia para estes dois limiares. Consegue-se assim maximizar a entropia de Shannon restringida pela medida de uniformidade dos níveis de cinzento e pela medida de forma.

4.3.4 Comparação entre os métodos referidos

Para avaliação de resultados foram utilizados dois métodos, um quantitativo, com comparação entre as medidas de uniformidade e de forma entre imagens binarizadas utilizando os dois métodos descritos, outro qualitativo, com uma comparação visual entre estas imagens.

Os valores das medidas de uniformidade e de forma terão que ser normalizados para permitir a comparação entre as diversas imagens. Para este efeito, os factores de normalização *'C'* das duas medidas serão considerados iguais com o valor correspondente ao número *'N'* de pixels da imagem. Não se considera para efeito de normalização o número de níveis de cinzento, por este ser constante e igual a 256.

Pode verificar-se o bom resultado visual de ambos os processos de binarização (figura 34), que deram origem a um limiar praticamente idêntico, sendo aqui representadas pela mesma imagem binária. O limiar do método de Otsu foi 154 e o outro 155 resultante da medida de uniformidade máxima (entropia 10483). A medida de forma máxima levaria ao limiar 130 (entropia 9222), na figura 34, e a máxima entropia 54440 levaria ao limiar 242. Pelo critério deste método seleccionou-se o limiar resultante da medida de uniformidade máxima.

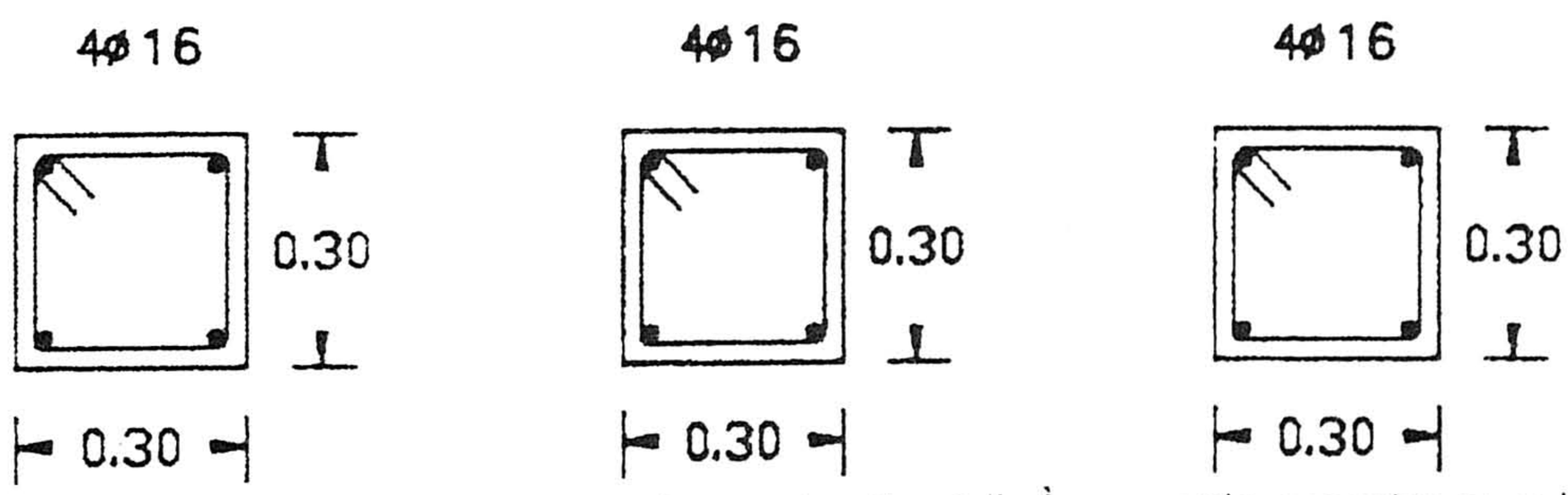


Figura 34: Imagem de uma estrutura de construção civil. À esquerda encontra-se a imagem digitalizada com uma resolução de 300 dpi em 256 níveis de cinzento. Ao centro é visível a imagem resultado da binarização de Otsu e de uniformidade máxima e à direita a imagem resultado da binarização de medida de forma máxima.

Na imagem resultado da aplicação dos dois processos de binarização (figura 35, central) não houve quebras nas linhas, nem junção das mesmas. Ambas as imagens estão representadas pela mesma figura por terem sido segmentadas com um limiar praticamente idêntico. O limiar do método de Otsu foi 154 e o de máxima entropia foi 155, resultante da medida de uniformidade máxima (entropia 5225). A medida de forma máxima levaria ao limiar 133 (entropia 4722), na figura 35 (à direita), e a máxima entropia 22770 levaria ao limiar 242. Pelo critério do método de máxima entropia seleccionou-se o limiar resultante da medida de uniformidade máxima.

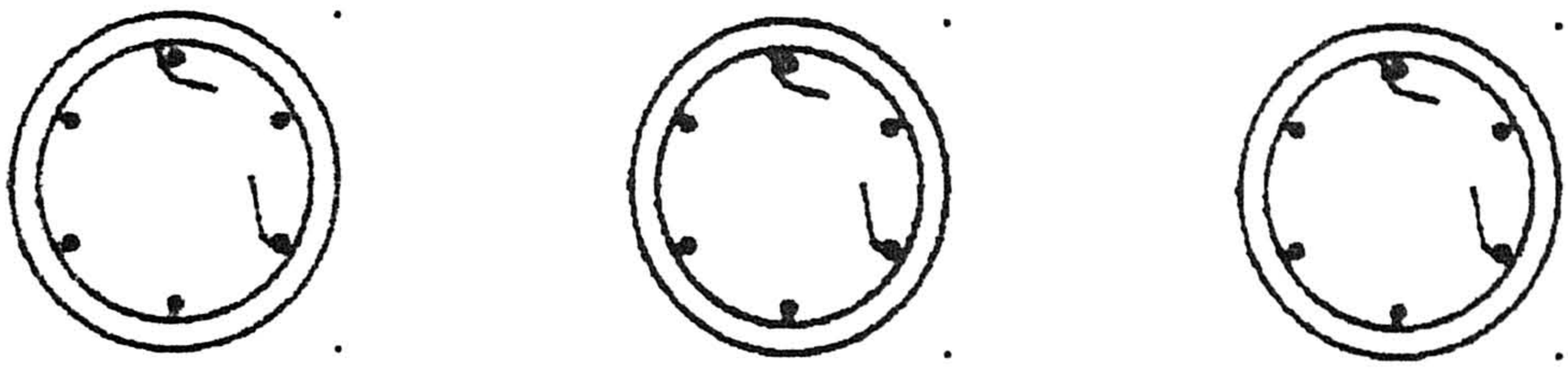


Figura 35: Imagem de uma estrutura de construção civil. À esquerda encontra-se a imagem digitalizada com uma resolução de 300 dpi em 256 níveis de cinzento. Ao centro é visível a imagem resultado da binarização de Otsu e de uniformidade máxima e à direita a imagem resultado da binarização de medida de forma máxima.

Com ambas as imagens se obtêm limiares similares, resultantes dos métodos descritos, o que está relacionado com o facto de terem sido adquiridas do mesmo documento, com o mesmo processo de digitalização. Pode concluir-se que para estas imagens, os processos de selecção de um limiar pelo método de Otsu, ou pela uniformidade máxima, dão bons resultados, por se detectarem os objectos existentes, por não se verificarem quebras nas linhas, nem a sua junção. O método de Otsu é bastante mais rápido, conseguindo resultados similares, daí se poder optar por este método relativamente ao método de máxima entropia. A análise destes métodos com imagens de âmbito mais genérico é apresentada no anexo 3 numa publicação apresentada na forma de poster em [23].

4.4 Segmentação de Página

A imagem documento é constituída por pixels negros, que constituem diferentes objectos, sobre um fundo branco. Os objectos são fundamentalmente de dois tipos: gráficos referentes aos desenhos, e texto referente às legendas e cotas. Em seguida descreve-se o processo utilizado para separação destes objectos, obtendo-se uma imagem com pixels negros, classificados como blocos de gráficos, e outra com pixels classificados como blocos de texto, ambas com fundo branco.

4.4.1 Separação do Texto dos Desenhos de Engenharia

Um documento depois de digitalizado, binarizado, e pré-processado (operações morfológicas) é constituído por um conjunto de pontos pretos em fundo branco. Estes pixels representam elementos gráficos e caracteres, sendo necessário separar os gráficos do texto para posterior processamento. Os caracteres e elementos gráficos são representados por um conjunto de pixels ligados, formando componentes ligados. Normalmente, e como já foi referido no capítulo de abordagem ao tema de análise de imagens de documentos, os caracteres formam componentes ligados de dimensão

inferior aos dos elementos gráficos, sendo no entanto complicado distingui-los se tiverem dimensões similares. Assim sendo, procuraram-se características que permitissem distinguir os componentes ligados de uns e de outros. Estas características serão referidas posteriormente, pois de seguida, descrever-se-á o processo que permitiu obter os diferentes componentes ligados.

Um componente ligado (definido no segundo capítulo), é um conjunto de pontos que têm pelo menos um vizinho pertencente a este conjunto [5]. Considera-se que dois pontos são vizinhos quando estão ligados, ou seja, quando a distância espacial digital entre os dois pontos é mínima [6]. Usualmente definem-se dois tipos de vizinhança de um ponto: pixels 4-vizinhos são representados na figura 36 por P_1, P_3, P_5, e P_7, são pixels adjacentes nas direcções norte, sul, este e oeste; pixels 8-vizinhos, representados na figura 36 por $P_1 ... P_8$, incluem os 4-vizinhos e os pixels imediatamente adjacentes nas direcções noroeste, sudoeste, nordeste e sudeste.

$$\begin{matrix} P_4(x-1,y-1) & P_3(x,y-1) & P_2(x+1,y-1) \\ P_5(x-1,y) & P(x,y) & P_1(x+1,y) \\ P_6(x-1,y+1) & P_7(x,y+1) & P_8(x+1,y+1) \end{matrix}$$

Figura 36: Vizinhança dum ponto.

Para determinar os diferentes componentes ligados existentes numa imagem, Rosenfeld e Kak [5] descreveram um algoritmo de etiquetagem de componentes, que foi utilizado no desenvolvimento do trabalho. Linha a linha, procuram-se pontos de componentes; encontrando-se um primeiro ponto de um componente (pixel negro) atribui-se-lhe uma etiqueta (um nível de cinzento diferente de zero), atribuindo-se a mesma etiqueta aos pontos dessa linha que lhe estiverem ligados, e uma nova etiqueta a pontos que não lhe estiverem ligados. Na linha seguinte, quando for encontrado um ponto negro, começa-se por verificar se existe um 8-vizinho já processado (vizinhos a noroeste, norte, nordeste e oeste, considerando um deslocamento da esquerda para a direita e de cima para baixo), atribuindo-se neste caso a mesma etiqueta a este ponto e a todos os que lhe estiverem ligados nesta linha. Se não existir um 8-vizinho processado anteriormente então atribui-se uma nova etiqueta. Se vier a verificar-se que um componente com uma determinada etiqueta tem um vizinho já processado com uma etiqueta diferente, então as duas etiquetas darão entrada numa tabela de correspondências, que permitirá, numa passagem posterior pela imagem, atribuir uma

única etiqueta a esses componentes. Pode prescindir-se desta segunda passagem transformando a imagem através de uma tabela de cores (do tipo LUT[3]).

O algoritmo implementado, além de efectuar a etiquetagem de componentes determina características geométricas do componente, tais como as coordenadas máximas e mínimas do componente (rectângulo envolvente), e o número de pixels que o constituem (área), de que é extraída a característica de densidade de pontos do componente no rectângulo envolvente.

Para classificar os componentes ligados de forma a poder distinguir caracteres de elementos gráficos as imagens foram avaliadas visualmente. Chegou-se à conclusão que uma das características que permitiria efectuar a distinção seria a forma dos caracteres, o que levaria a processos de reconhecimento de padrões, e por consequência lógica, ao próprio reconhecimento dos caracteres e não à sua separação da parte gráfica. Por outro lado, verificou-se que existe uma maior densidade de pixels nas áreas de componentes correspondentes a caracteres, símbolos, e gráficos complexos. Para efeito de aproximação de linhas por primitivas gráficas básicas (sem reconhecimento de alto nível), pretende-se isolar desenhos de linhas, pelo que é aceitável extrair caracteres juntamente com símbolos, até porque o reconhecimento destes se pode efectuar por processos similares ao reconhecimento de caracteres. Para quantificar essa relação foi determinada uma medida de fracção entre a área do componente e a área do rectângulo envolvente, que se denomina densidade de área:

$$Dens_{área} = \frac{A_{comp}}{A_{rect}}.$$

É importante notar neste ponto, que existem componentes gráficos (segmentos de recta) com uma densidade de valor um, superior à dos caracteres, mas precisamente por ser muito elevada podem ser imediatamente classificados como elementos gráficos.

O problema seguinte seria a determinação de um limiar que permitisse separar os dois tipos de componentes. Para esse efeito, não considerando os componentes de densidade de área superior a 0.9 (segmentos de recta perfeitos, segmentos de recta com buracos ou com ligeiras falhas) construiu-se um histograma de medidas de densidade de área para todos os componentes, aplicando-se o método de Otsu para determinação de um limiar de segmentação.

Este processo de separação de componentes de texto de componentes gráficos tem resultados aceitáveis, desde que os caracteres não estejam ligados a elementos

[3]Do inglês "look-up table"

gráficos, e não existam elementos gráficos de densidade muito alta (não considerando segmentos de recta) como pequenos rectângulos ou símbolos.

Na figura 37 ilustra-se todo o processo descrito, estando representada uma imagem original (a), o resultado da etiquetagem de componentes ligados (b), o resultado da separação do texto (c), e finalmente a imagem apenas com elementos gráficos (d).

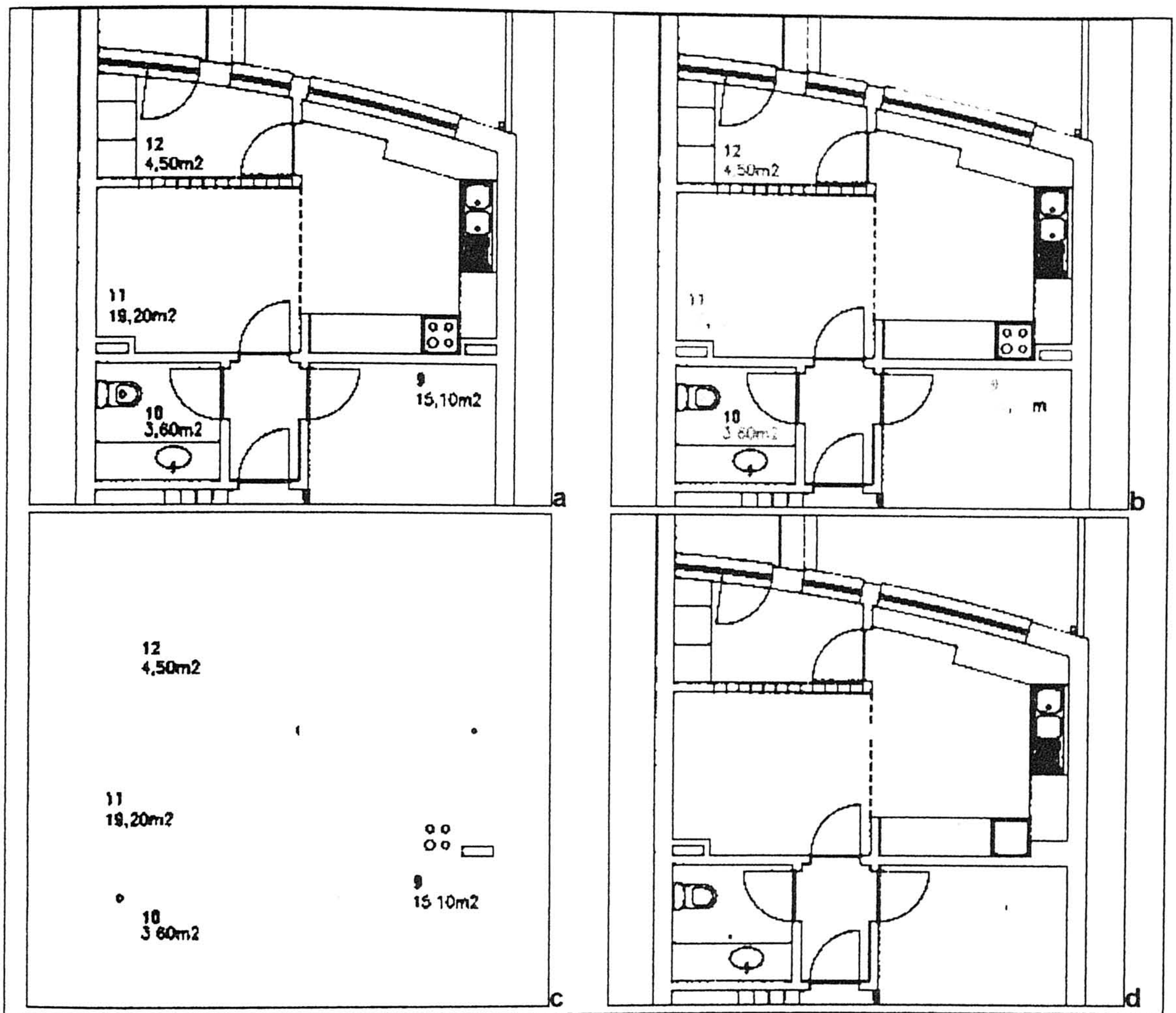


Figura 37: Ilustração do processo de separação de caracteres de elementos gráficos. (a) Imagem correspondente a uma planta de um edifício. (b) Etiquetagem de componentes ligados. (c) Separação de texto de gráficos. (d) Elementos gráficos resultantes.

Pode ver-se na figura 38 um gráfico, relativo à imagem da figura 37a, que a cada etiqueta (eixo horizontal) faz corresponder um valor de densidade de área (eixo vertical). Para aquela imagem o limiar obtido foi $Dens_{área} = 0.3737$, levando à classificação representada na imagem da figura 37c.

É visível na figura 37 que todos os elementos de texto foram correctamente classificados, com excepção das vírgulas isoladas. É de notar que foram igualmente classificados como blocos de texto, alguns blocos de gráficos, correspondentes a

elementos de elevada densidade de área, isto é, correspondentes a símbolos gráficos ou pequenos segmentos de recta.

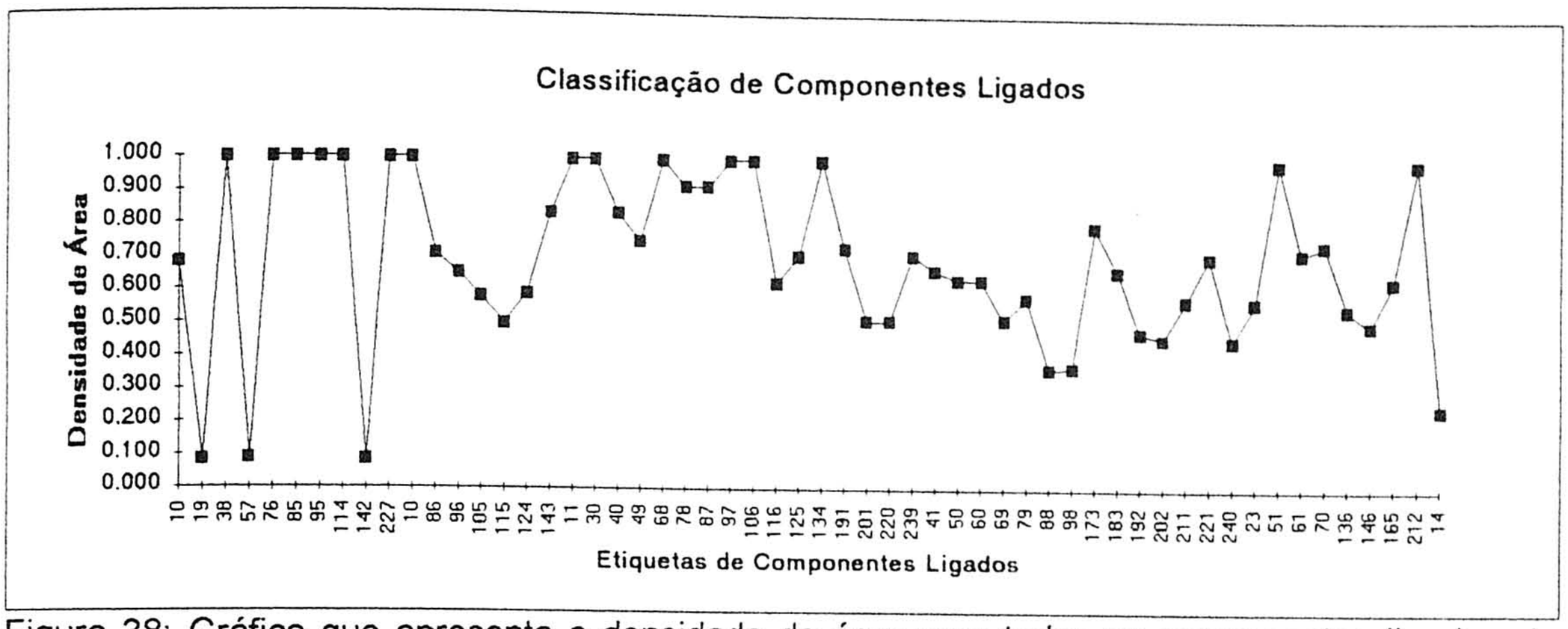


Figura 38: Gráfico que apresenta a densidade de área para todos os componentes ligados da imagem anterior. Os componentes são referidos pela sua etiqueta, sem ordem específica.

A figura 39 ilustra a aplicação deste processo a uma imagem de uma estrutura de construção civil, podendo-se verificar visualmente que todos os caracteres foram classificados correctamente, com excepção das vírgulas entre números, por terem uma densidade de área próxima de 1, sendo à partida consideradas como elementos gráficos. Por outro lado, todos os elementos gráficos foram classificados correctamente.

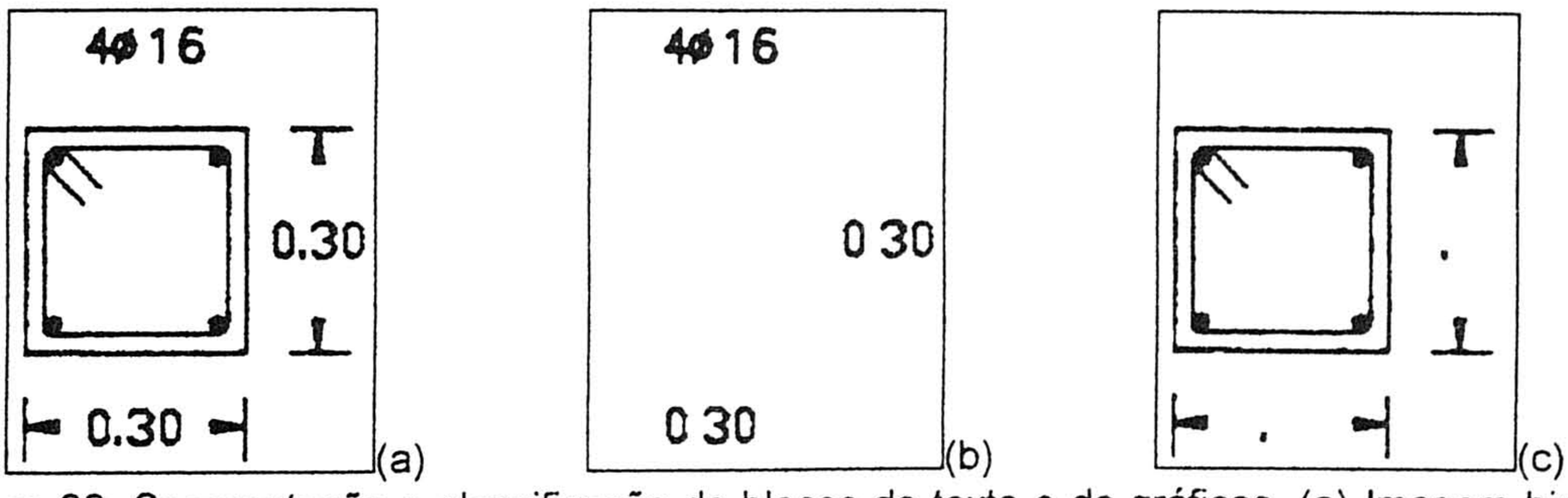
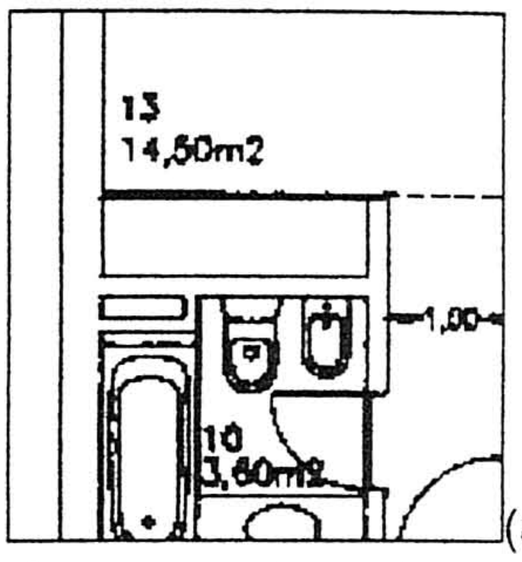


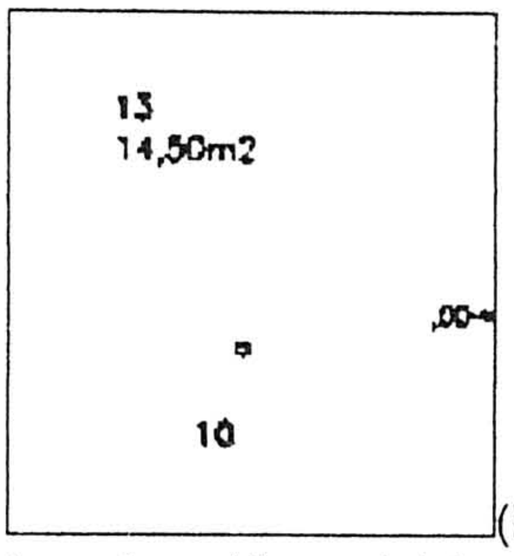
Figura 39: Segmentação e classificação de blocos de texto e de gráficos. (**a**) Imagem binária original (**b**) Imagem resultado da segmentação e classificação de blocos, com extracção dos blocos de texto (**c**) Imagem subtracção entre a imagem original e a imagem com os componentes de texto.

Numa outra imagem, apresentada na figura 40, este método não classificou correctamente vários elementos de texto, por estarem ligados a elementos gráficos. A ligação entre caracteres e elementos gráficos, transforma-os num único componente ligado, não podendo ser segmentados através do processo descrito, conduzindo portanto, a uma classificação errada de um dos elementos. Existe um elemento gráfico que, por estar ligado a elementos de texto (dois zeros do lado direito da imagem), foi classificado erradamente como elemento de texto.

Através do processo de segmentação e classificação de blocos de texto e blocos de gráficos, obtém-se resultados aceitáveis, como foi verificado para as imagens apresentadas, desde que a distribuição das medidas de densidade de área dos blocos (excluindo os componentes com densidade superior a 0.9) seja aproximadamente bimodal (limiarização de Otsu). É necessário, para uma classificação correcta, que os elementos de texto estejam separados dos elementos gráficos. Este problema não é ultrapassável pelo método descrito, sendo necessário recorrer à edição da imagem, para eliminar a ligação referida.



(a)



(b)

Figura 40: Segmentação e classificação de blocos de texto e de gráficos. (**a**) Imagem binária original (**b**) Imagem resultado da segmentação e classificação de blocos, com extracção dos blocos de texto.

4.5 Considerações Finais

Neste capítulo introduziram-se algumas noções de processamento inicial de imagens de documentos digitalizados, e estabeleceram-se alguns conceitos que permitem seleccionar resoluções de digitalização. No processamento inicial referem-se fundamentalmente operações morfológicas.

Em seguida descreveram-se dois processos de binarização de imagem e efectuou-se a sua avaliação em termos práticos.

Finalmente, desenvolveu-se um método de segmentação de partes da imagem do documento de forma a separar texto de desenhos de linhas.

Foram referidos métodos a aplicar inicialmente sobre uma imagem de documentos de engenharia, de forma a prepará-la para posterior reconhecimento gráfico, com o objectivo de descrever as linhas digitais por primitivas gráficas base.

Capítulo 5

RECONHECIMENTO DE GRÁFICOS

5. RECONHECIMENTO DE GRÁFICOS

Um dos objectivos deste trabalho é o reconhecimento de primitivas gráficas de desenhos de linhas em desenhos de engenharia, e a produção de um ficheiro compatível com um programa CAD. Neste capítulo descreve-se todo o processo que permite reconhecer essas primitivas, e produzir, como saída, o referido ficheiro. Os documentos de engenharia utilizados são desenhos de estruturas, de pequenas dimensões, constituídos fundamentalmente por linhas finas, símbolos e texto pertencentes a legendas ou cotas dimensionais.

Inicialmente é necessário proceder a um pré-processamento para obter linhas finas de largura unidimensional, sem ramos não pertencentes à estrutura gráfica.

O passo seguinte consiste no reconhecimento de linhas ligadas, obtendo-se uma descrição composta por uma lista de coordenadas. A partir desta lista vão reconhecer-se as primitivas gráficas básicas que a constituem, como sejam os segmentos de recta, arcos de circunferência, ou descrições de linhas a partir de curvas *spline*. Selecciona-se um processo apropriado ao reconhecimento de gráficos de desenhos de engenharia, e estuda-se o seu desempenho.

Finalmente transforma-se a informação obtida a partir do reconhecimento efectuado, para um formato adequado à entrada num sistema CAD.

5.1 PRÉ-PROCESSAMENTO

Operações de pré-processamento correspondem àquelas que são aplicadas com o intuito de preparar a imagem, para posteriormente se efectuar a aproximação das linhas dos desenhos por primitivas gráficas básicas. Com este objectivo, as principais operações desenvolvidas foram as de adelgaçamento de linhas, eliminação de ramos sem interesse (poda morfológica) e reconhecimento dessas linhas, obtendo-se um conjunto de listas de pontos representando os elementos gráficos por um conjunto de pontos, ainda sem uma descrição das primitivas gráficas existentes.

5.1.1 Adelgaçamento

Depois da eliminação de ruído, do pré-processamento, e da remoção de texto, a imagem documento é formada por regiões, que representam linhas com uma certa espessura. Para se extrair as primitivas gráficas que melhor aproximarão estas linhas, é necessário obter uma descrição fina das mesmas, sem destruir a conectividade de componentes ligados.

Esta operação, denominada adelgaçamento, corresponde à redução de uma região conexa. O esqueleto de um objecto (conjunto de pixels ligados) pode ser definido [1, 5, 57] através da transformação de eixo médio (MAT[1]). Para se efectuar esta transformação é necessário calcular a distância mínima de todos os pontos de uma região para o bordo da mesma. O eixo médio será composto por todos os pontos do objecto cuja distância mínima para o bordo se verifique para mais do que um ponto do bordo. A MAT depende da definição de distância utilizada. Se se considerar a distância euclideana, será possível, em todos os pontos do eixo médio centrar uma circunferência totalmente contida na região e que seja tangente ao bordo em pelo menos dois pontos distintos. A implementação directa desta definição exige a determinação da distância de todos os pontos interiores para todos os pontos do bordo, sendo por isso de peso computacional elevado.

A maioria dos autores têm proposto operações de adelgaçamento iterativas que vão sucessivamente eliminando pontos do bordo, reduzindo o objecto e mantendo a sua conectividade, aproximando assim, o resultado do eixo médio.

Rosenfeld e Kak [5] propuseram um algoritmo que elimina pontos do bordo se forem considerados *simples*, e não forem pontos extremos (apenas têm um vizinho pertencente ao objecto), avaliando sempre consecutivamente os bordos em direcções opostas, isto é, Norte - Sul, Este - Oeste. Um ponto é simples se o conjunto dos seus 8-vizinhos, pertencentes ao objecto, formarem um componente ligado, isto é, se entre quaisquer dois pontos do conjunto de pixels da vizinhança, pertencentes ao objecto, existir um caminho (considerando qualquer ponto, existe sempre um outro que lhe é vizinho). A definição de componente ligado foi estabelecida no segundo capítulo e referida no quarto. Um ponto final (extremo) tem apenas um vizinho, logo, haverá apenas um componente ligado no conjunto dos seus pixels vizinhos, pertencentes ao objecto, sendo portanto um ponto simples. No entanto, não se eliminam pontos extremos porque levaria à redução de regiões finas até um ponto único, não coincidindo dessa forma, o resultado obtido com o eixo médio do objecto. Resumindo, este processo propõe uma avaliação sucessiva de pontos de bordos opostos (Norte - Sul, Este - Oeste), eliminando-os se forem pontos simples e não forem pontos extremos. Notar que a avaliação dos 8-vizinhos implica a utilização de janelas 3x3. A figura 41 ilustra a definição de ponto simples. Enquanto na primeira janela, o ponto central tem como vizinhos dois componentes ligados, um na linha de cima e outro na de baixo, na segunda janela o ponto central tem dois componentes ligados como vizinhos, um na primeira coluna e outro na terceira. A

---

[1]Do inglês 'medial axis transformation'.

última janela mostra como vizinho do ponto central dois pontos, sendo estes vizinhos um do outro, logo, formando um único componente ligado.

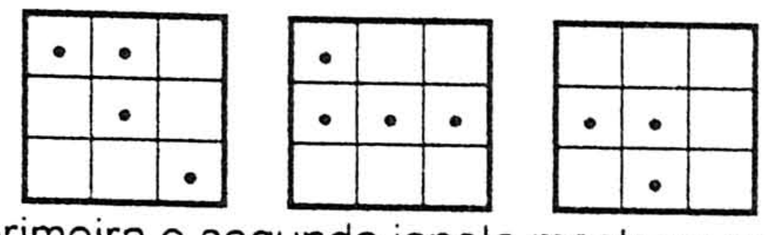


Figura 41: Pontos simples. A primeira e segunda janela mostram pontos centrais simples, ao contrário da terceira.

Gonzalez e Woods [1] apresentam um processo de adelgaçamento de regiões binárias, considerando pontos das regiões com valor *um* lógico e pontos do fundo com valor *zero* lógico. Este processo é iterativo, eliminando-se pontos dos bordos, por aplicação de duas condições, e repetindo-se o processo na iteração seguinte, até que não existam mais alterações ao bordo. Um ponto é considerado como pertencendo ao bordo se tiver pelo menos um vizinho com valor *zero*.

Cada iteração é composta por 4 passos: marcam-se os pontos dos bordos que satisfaçam a primeira condição (tabela 2); apagam-se todos os pontos marcados no primeiro passo; marcam-se todos os pontos dos bordos que satisfaçam a segunda condição (tabela 2); apagam-se todos os pontos marcados por efeito da segunda condição.

Tabela 2: Condições de eliminação de pontos dos bordos.

Condição 1	Condição 2
(a) $2 \le N(p_1) \le 6$	(a) $2 \le N(p_1) \le 6$
(b) $S(p_1) = 1$	(b) $S(p_1) = 1$
(c) $p_2 \cdot p_4 \cdot p_6 = 0$	(c') $p_2 \cdot p_4 \cdot p_8 = 0$
(d) $p_4 \cdot p_6 \cdot p_8 = 0$	(d') $p_2 \cdot p_6 \cdot p_8 = 0$

Considerando a vizinhança do ponto em análise, definida da forma representada na figura 42, $N(p_1)$ é o número de vizinhos de p1 pertencentes ao objecto e $S(p_1)$ é o número de transições de zero para um na sequência ordenada de vizinhos de p2..p9.

p9	p2	p3
p8	p1	p4
p7	p6	p5

Figura 42: Definição da vizinhança no processo de adelgaçamento apresentado por Gonzalez e Woods.

A condição (a) evita que se eliminem pontos finais de sequências *finas* ou que se efectue a erosão para o interior duma região. As figuras 43i e 43ii mostram duas alternativas em que esta condição é violada.

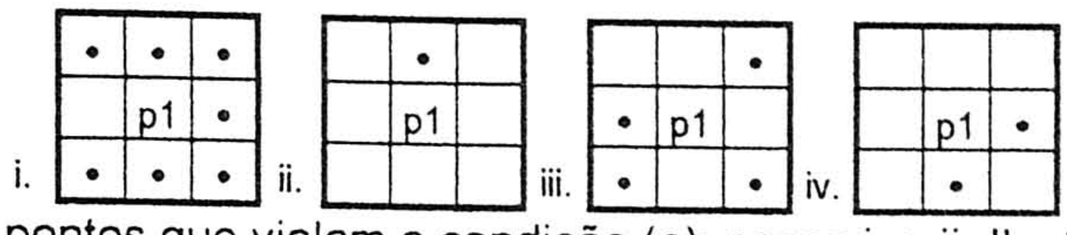


Figura 43: Ilustração de pontos que violam a condição (a): casos i, e ii. Ilustração de pontos que violam a condição (b): casos iii, e iv.

Verifica-se a condição (b) quando os pontos vizinhos de p1 formam mais do que um componente 4-ligado, o que levaria, por eliminação deste ponto, à perda de 4-conectividade. As figuras 43iii e 43iv mostram duas alternativas em que esta condição é violada.

Para a figura 43i temos $N(p_1)=7$, para a figura 43ii $N(p_1)=1$, para a figura 43iii $N(p_1)=4$ e $S(p_1)=3$, e para a figura 43iv $N(p_1)=2$ e $S(p_1)=2$.

As condições (c) e (d) são satisfeitas simultaneamente pelo mínimo conjunto de valores: (p4=0 *ou* p6=0) ou (p2=0 *e* p8=0). Assim, um ponto que satisfaça estas condições, assim como as condições (a) e (b) é um ponto do bordo Este ou Sul (figura 44i), ou um canto Noroeste (figura 44ii). De uma forma similar, as condições (c') e (d') são satisfeitas simultaneamente pelo mínimo conjunto de valores: (p2=0 *ou* p8=0) ou (p4=0 *e* p6=0). Estas condições, juntamente com (a) e (d) correspondem a pontos dos bordos Norte ou Oeste (figura 44iii), ou a um canto Sudeste (figura 44iv).

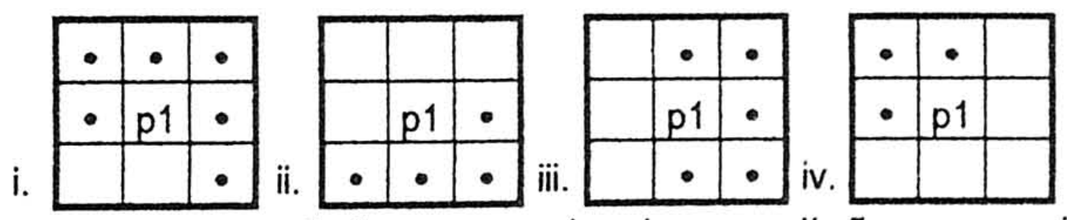


Figura 44: Ilustração de pontos que satisfaçam a primeira condição: casos i, e ii. Ilustração de pontos que satisfaçam a segunda condição: casos iii, e iv.

Lawrence O'Gorman [58] propõe um método de adelgaçamento denominado adelgaçamento k×k, que permite efectuar a redução de objectos, eliminando um pixel, ou blocos de pixels, de cada vez. Para este efeito são utilizadas janelas de dimensão kxk, com $k \geq 3$, e eliminados blocos de dimensão (k-2)x(k-2), podendo-se assim reduzir o número de iterações, aumentando no entanto, a espessura do resultado e a quantidade de ramos parasitas.

As condições a respeitar por este processo são:

- a conectividade das regiões deve ser mantida ao serem transformadas em estruturas finas;
- as estruturas resultantes da operação devem conter o mínimo de pixels possível, para que respeitem a 8-conectividade;
- devem manter-se os locais aproximados do final de linhas;
- as estruturas resultantes devem aproximar-se do eixo médio;
- os ramos parasitas devem ser reduzidos ao mínimo.

Em seguida, define-se a janela que é utilizada nesta operação de adelgaçamento [58] e os critérios a respeitar para eliminar pontos da imagem ou blocos. Sendo k a dimensão de um dos lados da janela W a utilizar na operação de adelgaçamento, a janela $W(x,y,k)$ é definida como sendo a região da imagem de dimensão kxk, centrada em (x_c,y_c) e em que as coordenadas horizontal e vertical tomam os seguintes valores:

$$(x,y)=(x_c,y_c)+r,$$

com $\left\{-\left\lfloor k-1/2\right\rfloor \le r \le \left\lfloor k/2\right\rfloor\right\}$, em que $\lfloor\ \rfloor$ designa a truncagem inteira. A figura 45 apresenta várias janelas, os valores que *r* toma, e o ponto central de cada uma das janelas.

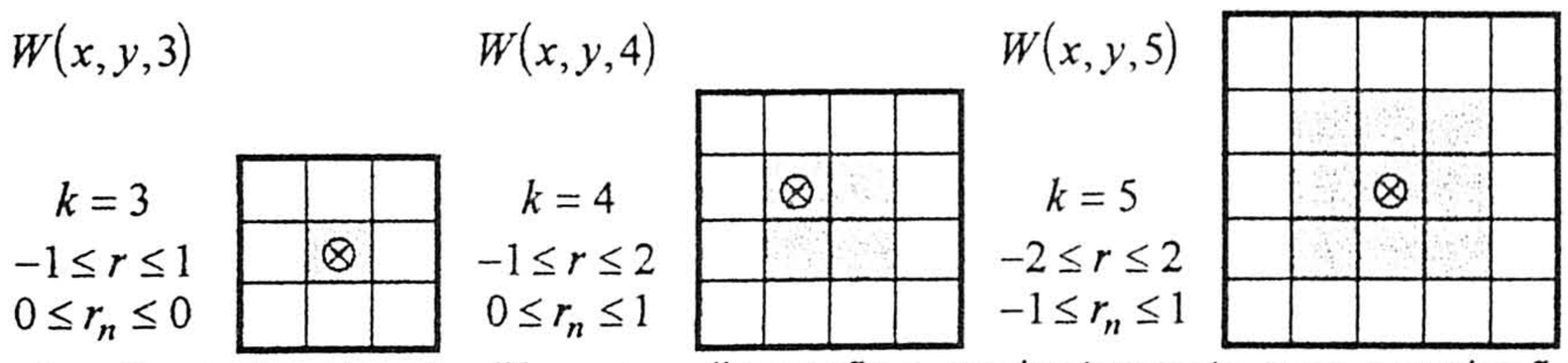


Figura 45: Ilustração de janelas de diferentes dimensões, conjuntamente com a variação de *r*. O ponto central da janela de coordenadas (x_c,y_c) está representado pelo símbolo ⊗. A fundo mais escuro estão representados os núcleos.

O núcleo $R(x,y,k)$ a eliminar, de dimensões $(k-2)\times(k-2)$, é a região da janela envolvida pelo perímetro, ou seja, envolvida pelo bordo da janela. A origem do núcleo e da janela coincidem, e as coordenadas horizontal e vertical (x,y) tomam os seguintes valores:

$$(x,y)\leftarrow((x_c,y_c)+r_n),$$

com $\left\{-\left\lfloor k-3/2\right\rfloor \le r_n \le \left\lfloor k-2/2\right\rfloor\right\}$. Na figura 45 pode ver-se com um fundo mais escuro o núcleo de cada uma das janelas kxk com k entre 3 e 5.

A vizinhança de uma janela (figura 46) corresponde aos pontos do perímetro dessa janela. É composta por $4(k-1)$ pixels que contém o núcleo, e é designada por $n(i)$, com $i=0,\ldots,4(k-1)-1$. Esta vizinhança corresponde à sequência de pixels ao longo do perímetro, com inicio no canto superior esquerdo.

Os cantos (figura 46) das janelas correspondem aos quatro pontos situados no topo esquerdo e direito, e na base esquerda e direita da janela, sendo representados na vizinhança pelos índices 0, $k-1$, $2(k-1)$, e $3(k-1)$.

Os pixels dos lados (figura 46) correspondem aos pixels da vizinhança que não são cantos. Existem 4 lados com pixels ligados, separados pelos cantos, designados por norte, sul, este e oeste.

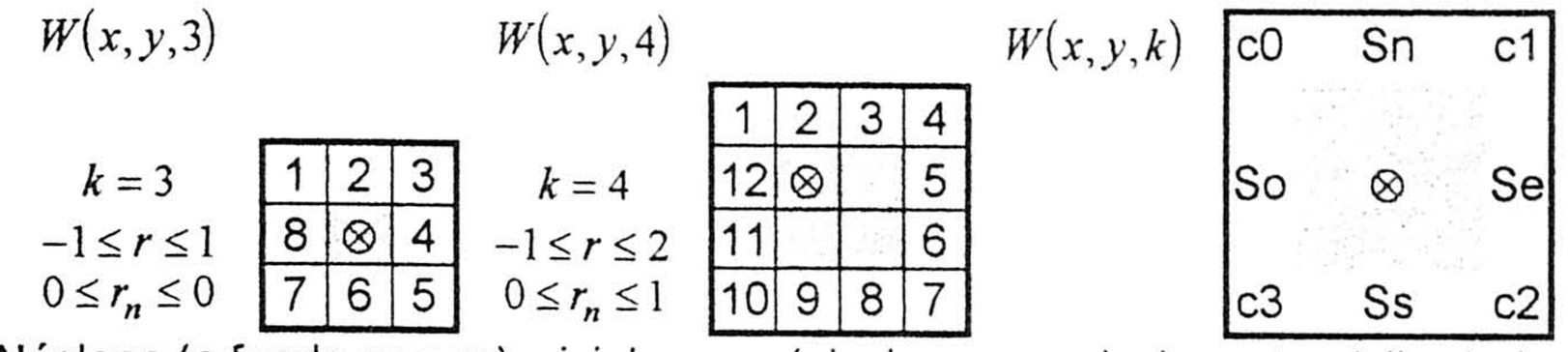


Figura 46: Núcleos (a fundo escuro), vizinhanças (pixels numerados), cantos (ci) e lados (Sdir) de janelas 3x3, 4x4 e kxk.

Para uma janela $W(x,y,k)$, pode proceder-se à eliminação do seu núcleo $R(x,y,k)$ se este for composto por pixels da região (*um* lógico), desde que as condições da tabela 3 se verifiquem. Numa janela definem-se as seguintes medidas, para estabelecimento dos critérios de adelgaçamento:

- $\phi_0(\eta)$ corresponde ao máximo comprimento de pixels do fundo, pertencentes à vizinhança, que sejam 4-ligados;
- $\phi_1(\eta)$ corresponde ao máximo comprimento de pixels da região, pertencentes à vizinhança, que sejam 8-ligados.
- $\chi(\eta)$ é o número de sequências conexas de pixels da região, pertencentes à vizinhança (tendo esta pelo menos um pixel do fundo), denominada conectividade do núcleo, que pode ser determinada pela seguinte função, considerando-se $n = 0,1,2,3$ e $\eta(-1) = \eta(4k-5)$:

$$\chi(\eta) = \frac{1}{2} \sum_{\substack{i=0 \\ \{i,i-1\} \neq n(k-1)}}^{4(k-1)-1} \left|\eta(i) - \eta(i-1)\right| + \frac{1}{2} \sum_{m=n(k-1)} \left|\eta(m+1) - \eta(m-1)\right| + \sum_{m=n(k-1)} \eta(m) \cdot \left|\eta(m) - \eta(m-1)\right| \cdot \left|\eta(m) - \eta(m+1)\right|.$$

Nesta última medida, o primeiro termo conta o número de transições nos lados do bordo, ou seja, não considera os cantos; o segundo termo conta as transições através dos cantos, ou seja, entre os vizinhos dos cantos, e o terceiro termo conta o número de cantos isolados pertencentes à região.

Tabela 3: Critério de adelgaçamento kxk.

Critério de Adelgaçamento	
Condição 1	$\chi(\eta) = 1$
Condição 2	$\phi_1(\eta) > k-2$
Condição 3	$\phi_0(\eta) > k-2$

Relativamente ao critério de adelgaçamento pode afirmar-se que a condição 1 permite que se mantenha a conectividade da estrutura, porque a vizinhança conterá uma única sequência ligada de pixels pertencentes à região; a condição 2 mantém pontos finais de linhas, e a condição 3 impede a erosão para o interior de regiões.

São visíveis os resultados (figura 47) da aplicação do algoritmo de adelgaçamento de Rosenfeld e Kak (imagem ao centro) e do algoritmo de adelgaçamento kxk (imagem à direita). Ambos os algoritmos mantiveram a forma circular dos três objectos da imagem, com linhas finas, em que todos os pontos pertencem ao bordo do objecto. Esta representação, por ser de linhas sem espessura, é adequada para aproximação por primitivas gráficas, que será posteriormente descrita.

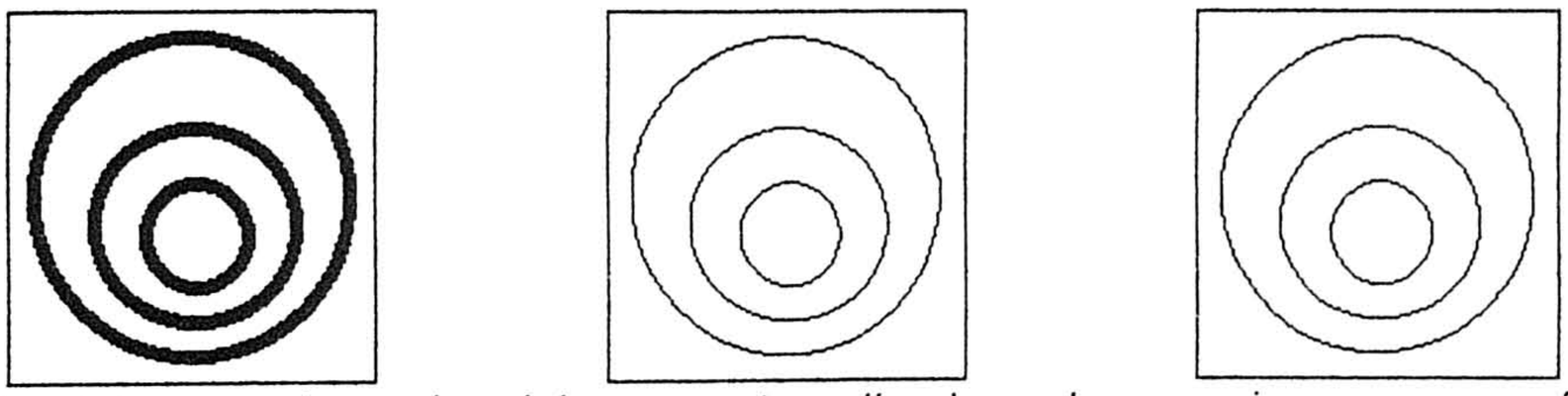


Figura 47: Os dois operadores de adelgaçamento aplicados sobre uma imagem com vários círculos (à esquerda). A imagem central resulta do adelgaçamento de Rosenfeld e Kak, e a imagem à direita do adelgaçamento kxk.

5.1.2 Método Morfológico de Poda[2]

Após se efectuar o adelgaçamento sobre os objectos, podem existir pixels que, por não pertencerem ao eixo médio interessa eliminar. Estes pixels denominam-se componentes parasitas. Para eliminar esses componentes parasitas desenvolveu-se uma operação morfológica a que se deu o nome de poda, pela semelhança com o acto de podar árvores eliminando ramos sem interesse para o desenvolvimento das mesmas.

Gonzalez e Woods [1] propuseram uma técnica morfológica de poda, assumindo que um ramo parasita não excede 3 pixels. A operação é baseada na eliminação sucessiva de pixels extremos de ramos parasitas, o que, como é lógico, também vai

[2]Do inglês 'pruning'.

eliminar pixels extremos de cadeias de pontos de ramos não parasitas do objecto. No entanto, este efeito é limitado a ramos de dimensão 3. Esta eliminação consegue-se pelo adelgaçamento de um objecto, em três passos, utilizando elementos estruturantes que apenas eliminem pontos extremos, obtendo-se assim a imagem X_1. O passo seguinte consiste em repor o objecto original, sem os ramos parasitas, o que se consegue através de 3 dilatações sucessivas dos pontos extremos de X_1, nunca ultrapassando os limites do objecto original. Finalmente, a imagem resultante da operação de poda é obtida por união entre a imagem X_1 e a imagem resultante das dilatações condicionais.

Sendo $\{B\}=\left\{B^1,B^2,B^3,B^4,B^5,B^6,B^7,B^8\right\}$, a definição de poda será:

1. $X_1 = A \otimes \{B\}$, adelgaçamento de A;
2. $X_2 = \bigcup_{k=1}^{8}\left(X_1 * B^k\right)$, determinação dos pontos extremos de X_1;
3. $X_3 = (X_2 \oplus H) \cap A$, Dilatação condicionada por A;
4. $X_4 = X_1 \cup X_3$, Resultado final da operação.

A imagem original é representada por A e a imagem resultado por X_4. Os elementos estruturantes são apresentados na figura 48. Os pontos negros correspondem a pixels pertencentes ao objecto, os quadrados vazios correspondem a pixels do fundo da imagem, e os quadrados com um × correspondem a pixels indiferentes, isto é, que tanto podem pertencer ao fundo como ao objecto.

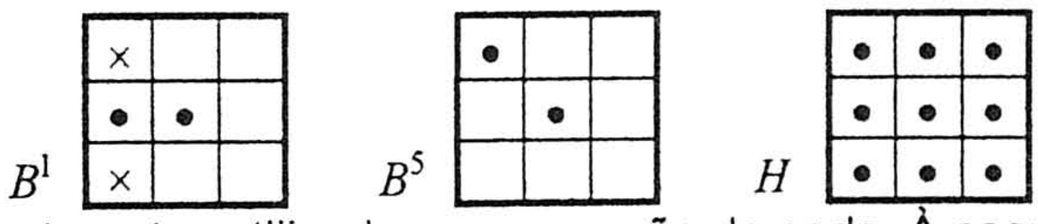


Figura 48: Elementos estruturantes utilizados na operação de poda. À esquerda é apresentado o elemento estruturante B^1, sendo B^i igual a B^{i-1} rodado de 90º, com $i=\{2,3,4\}$; ao centro o elemento estruturante B^5, sendo B^j igual a B^{j-1} rodado de 90º, com $j=\{6,7,8\}$; à direita o elemento estruturante H.

Na figura 49 ilustra-se a aplicação da operação morfológica de poda. A imagem à esquerda é formada por linhas de espessura de um pixel, com alguns componentes que saem da estrutura principal, de comprimento inferior a 3 pixels, que são considerados ramos parasitas. Aplicando a poda morfológica aqui descrita obtém-se uma imagem resultado (à direita), sem os referidos ramos.

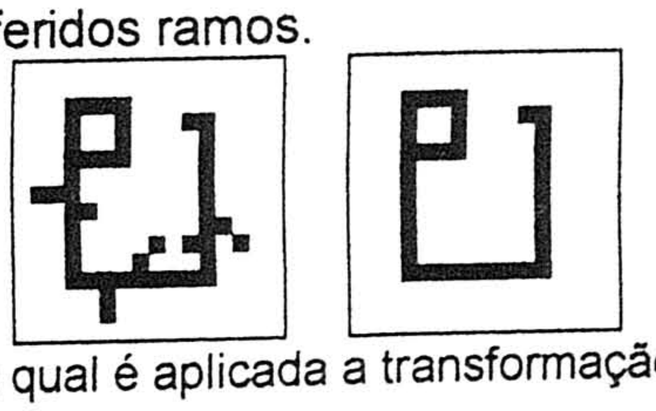


Figura 49: Imagem (esquerda) à qual é aplicada a transformação morfológica de poda (direita).

Gonzalez e Woods [1] referem a utilização, por vários autores, de um outro elemento estruturante, que no entanto pode conduzir à quebra de conectividade do objecto por

eliminação incorrecta de pixels. Na figura 50 está representado o elemento estruturante. A operação de poda com este elemento estruturante pode originar a quebra de conectividade em casos especificos. Essa quebra será provocada pela eliminação de um determinado ponto. A figura 51 ilustra este efeito, já que o ponto que liga os dois blocos quadrados será eliminado pelo adelgaçamento, utilizando o elemento estruturante B^2 (figura 50), transformando uma imagem com um único componente ligado numa imagem com dois componentes ligados. Depois de eliminado aquele ponto, e de terminado o adelgaçamento, na operação seguinte (reconhecimento de pontos extremos) não seria encontrado nenhum ponto extremo, logo a reconstrução condicional da imagem não reconheceria o pixel eliminado como válido para esta imagem. O mesmo não aconteceria com a operação de poda descrita em [1], já que aquele ponto nunca seria considerado um ponto extremo.

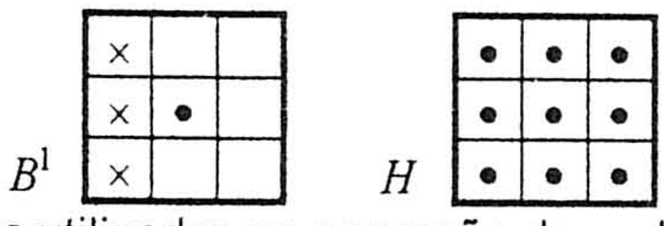


Figura 50: Elementos estruturantes utilizados na operação de poda. À esquerda é apresentado o elemento estruturante B^1, sendo B^i igual a B^{i-1} rodado de 90º, com $i=\{2,3,4\}$; à direita o elemento estruturante H.

5.1.3 Lista de Pontos

Neste fase do processamento as imagens documento já se apresentam de uma forma bastante condensada. Isto é, já se procedeu à eliminação de texto, da maior parte do ruído aleatório, já se reduziu a espessura das linhas, e eliminaram-se ramos parasitas. Resumindo, as imagens resultantes do processamento efectuado até este ponto são formadas por linhas finas, das quais se pretendem extrair as primitivas gráficas que melhor as representarão.

Para se efectuar a análise de sequências de pontos ligados é necessário ter uma representação desses pontos, ou seja, é necessário saber quais os pontos constituintes de cada uma das sequências ligadas. As sequências podem ser representadas simplesmente pelas coordenadas de listas de pontos ligados, ou por processos, amplamente divulgados na literatura [1, 5, 6], de códigos em cadeia, que são representações mais compactas de listas de pontos ligados, que permitem:

- poupar espaço no armazenamento de descrições de linhas, relativamente à lista de pontos que a constituem;
- efectuar a caracterização de linhas, reconhecendo, por exemplo a sua orientação predominante;

- classificar segmentos de linha como curvas (mudanças de declive consecutivas, e muito próximas), ou rectas.
- determinar o declive de segmentos da linha, e a curvatura dos pontos por subtracção entre declives anteriores e posteriores ao ponto.

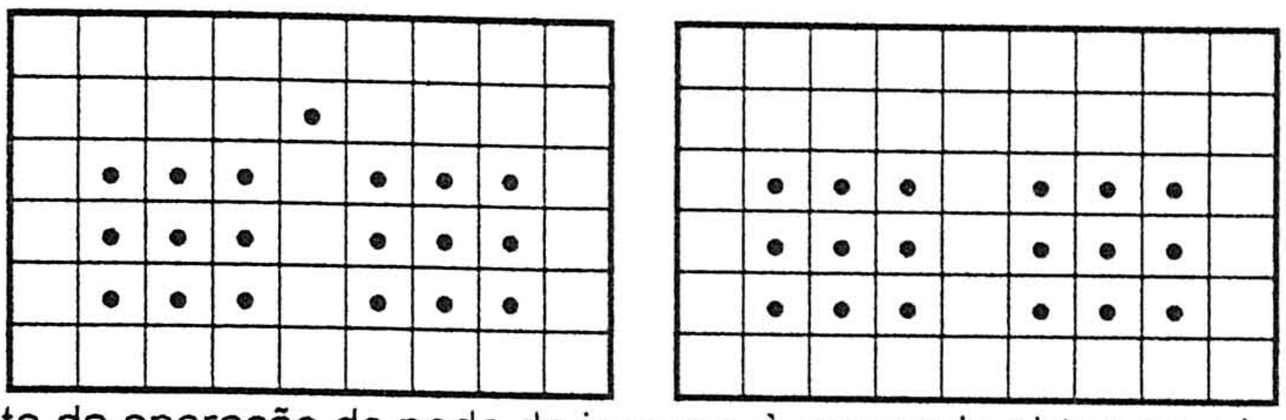


Figura 51: Por efeito da operação de poda da imagem à esquerda obtemos a imagem à direita.

O código em cadeia absoluto, ou simplesmente código em cadeia, indica a partir de um ponto qual a direcção do ponto seguinte da lista de pontos ligados [1, 5]. Os códigos variam, conforme o tipo de vizinhança considerada (ver secção de etiquetagem de componentes ligados), denominando-se de código em cadeia 4-direccional ou 8-direccional (ilustrados na figura 52).

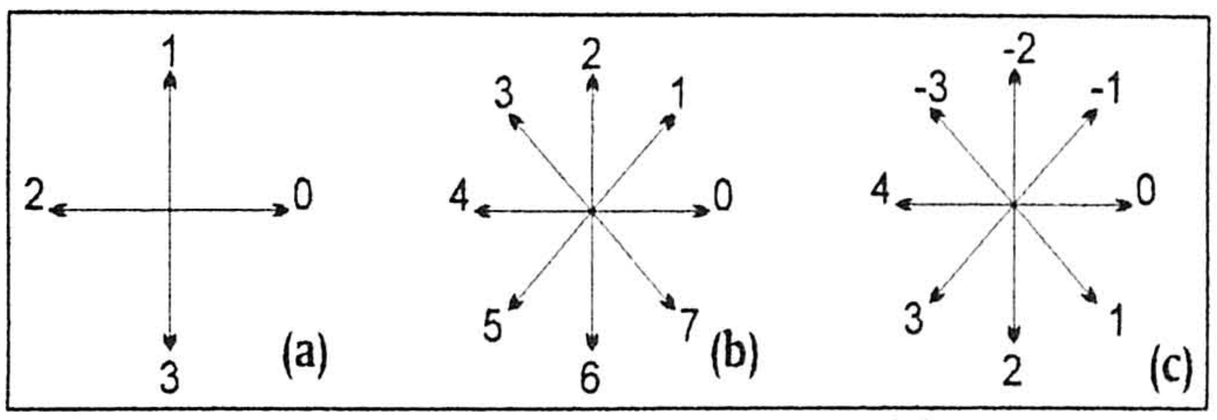


Figura 52: Códigos em cadeia de sequências de pontos. (a) Código em cadeia 4-direcional. (b) Código em cadeia 8-direcional. (c) Código em cadeia relativo 8-direcional.

Outra codificação possível, baseada nesta, é a de código em cadeia relativo. Neste caso, indica-se o código da direcção a tomar, relativamente à direcção anterior. Ou seja, se se mantiver a direcção o código será 0, se mudar 45º de direcção o código será 1, se mudar 135º de direcção o código será -3, e assim sucessivamente, respeitando a imagem da figura 52c.

Na figura 53 ilustra-se a codificação absoluta e relativa para uma imagem com uma lista de pontos ligados. A procura de um vizinho do ponto em análise dá-se respeitando as direcções ascendentes do código de sequência (de 0 para 7).

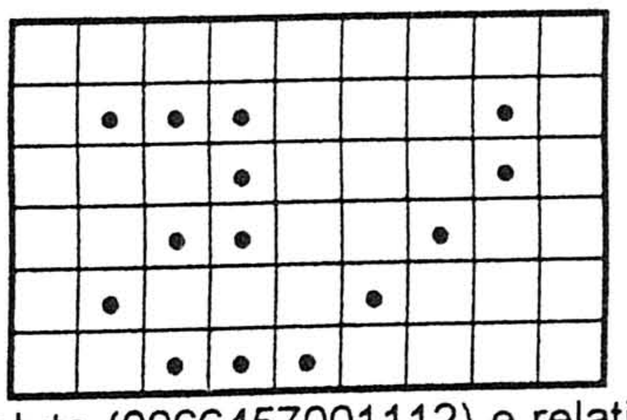


Figura 53: Código em cadeia absoluto (0066457001112) e relativo (0060457001002) de listas de pontos ligados, iniciando-se a sequência no pixel mais à esquerda e mais acima.

Schalkoff [6] refere um outro método de codificação em cadeia (método de Badi'i e Majd), baseado em segmentos de recta, na mudança de direcção de um segmento para outro (para a direita ou para a esquerda), e na diferença entre os tamanhos desses segmentos (menor, igual ou maior). Estas classificações são qualitativas e não quantitativas, não permitindo a partir delas, efectuar a extracção de primitivas gráficas que descrevam as sequências. Os códigos estão representados na tabela 4.

Tabela 4: Códigos de sequência do método de Badi'i e Majd

Direcção relativa	Tamanho relativo	Código inteiro
Direita	Menor	1
Direita	Igual	2
Direita	Maior	3
Esquerda	Menor	4
Esquerda	Igual	5
Esquerda	Maior	6

A codificação da sequência de pontos da figura 53, segundo este método é apresentada na tabela 5. Cada código corresponde a um segmento de recta, indicando a mudança relativa para o próximo segmento. Consideram-se as coordenadas do canto superior esquerdo (0,0) e do canto inferior direito (8,5). Exemplificando, o código 6 do segmento 5 indica que o próximo segmento sofrerá uma viragem à esquerda relativamente a este, e que será maior.

Tabela 5: Código de Badi'i e Majd para a figura 53.

Segmento	Condição relativa	Código
Segmento 1, de (1,1) a (3,1)	direita, igual	2
Segmento 2, de (3,1) a (3,3)	direita, menor	1
Segmento 3, de (3,3) a (2,3)	esquerda, igual	5
Segmento 4, de (2,3) a (1,4)	esquerda, igual	5
Segmento 5, de (1,4) a (2,5)	esquerda, maior	6
Segmento 6, de (2,5) a (4,5)	esquerda, maior	6
Segmento 7, de (4,5) a (7,2)	esquerda, menor	4

O processo de extracção de linhas implementado neste trabalho, considera uma linha composta por dois pontos extremos e por um conjunto de pontos intermédios, ou seja, considera um conjunto de pontos conexos com dois pontos extremos. O primeiro passo do processo de extracção de linhas é o de encontrar um ponto das linhas (pixel negro), começando no canto superior esquerdo da imagem e fazendo uma busca da esquerda para a direita, e de cima para baixo. A partir deste primeiro ponto procuram-se vizinhos, que pertençam às linhas, começando pelo vizinho de código de cadeia 0 e indo sucessivamente até ao de código 7. A partir do primeiro vizinho seleccionado, avança-se

até encontrar um ponto que seja extremo. Neste algoritmo considera-se um ponto extremo, um ponto que apenas tenha um vizinho (ponto final duma sequência de pontos ligados), ou um ponto que tenha mais de dois vizinhos (ponto de entroncamento ou cruzamento entre sequências de pontos ligados). Depois de encontrado um ponto extremo, regressa-se ao ponto inicial, determinando-se a sua condição de extremo, ou não. Se for extremo, então a sequência de pontos ligados está terminada, senão é necessário procurar mais um vizinho, e a partir daí encontrar outro ponto extremo. Os pontos extremos que pertençam a cruzamentos entre linhas, são pontos extremos de outras linhas, sendo candidatos a procuras posteriores de pontos ligados. Cada uma destas linhas origina a formação de uma lista de pontos, e todas estas listas, correspondendo a linhas que têm uma ligação entre si, compõem uma estrutura conexa de pontos. Uma estrutura destas, composta por várias linhas está ilustrada na figura 54. No componente ligado superior, cada linha é definida pelo ponto final da linha e pelo ponto de cruzamento entre elas (ponto representado pelo fundo mais escuro). O componente ligado inferior, corresponde a uma única linha com dois pontos extremos, que coincidem com os pontos finais da mesma.

Na próxima secção vai descrever-se a forma de aproximar linhas por segmentos de recta, partindo-se da descrição de pontos aqui referida.

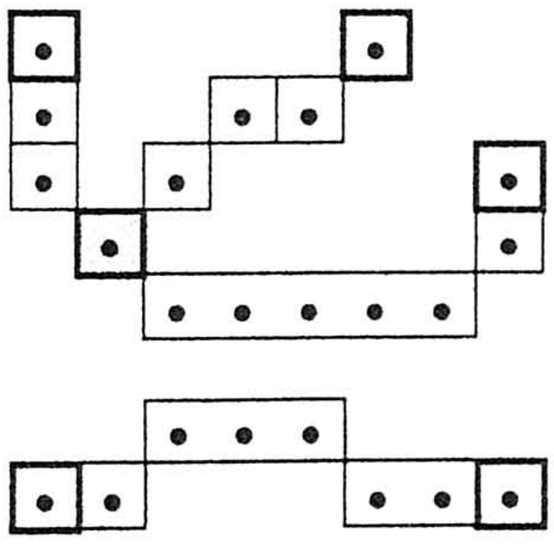


Figura 54: Linhas e pontos extremos, estando estes representados com um bordo mais grosso.

5.2 Aproximação Poligonal de Linhas

Aproximação poligonal de linhas é a denominação da operação de substituição de partes das linhas por segmentos de recta, desde que essas partes se aproximem de uma recta, respeitando um determinado critério, que se pode basear por exemplo num limiar da distância (considerada na perpendicular) máxima da recta aos pontos aproximados da linha. O objectivo principal desta operação é a captura da essência da forma (da sua curvatura) de uma linha, utilizando o menor número de segmentos de recta possível [1]. Embora seja um problema, em geral, não trivial, existem soluções de complexidade média, que juntamente com a característica da simplicidade da representação (dois pontos extremos), a tornam numa abordagem largamente utilizada.

Em imagens com linhas predominantemente formadas por segmentos rectilíneos (com pequena variação do declive), esta aproximação leva a uma descrição mais compacta relativamente à descrição de toda a sequência de pontos, ou mesmo relativamente à própria imagem. No entanto, quando as linhas são predominantemente curvas, para que a representação tenha alguma qualidade é necessário um grande número de segmentos, o que aumenta a memória necessária para armazenamento da informação, relativamente a linhas com menor variação do declive. Esse aumento chega mesmo a produzir descrições que ocupam maior espaço do que as descrições de lista de pontos (por exemplo códigos de sequência), ou mesmo maior espaço do que o da imagem. Nesta secção referem-se vários processos de aproximação poligonal e descrevem-se dois métodos utilizados, efectuando-se uma análise do seu desempenho.

Rosenfeld e Kak [5] consideram um declive, no contexto da definição de rectas digitais, como sendo um conjunto de pixels ligados, não havendo mudança de direcção no deslocamento consecutivo de um extremo para outro, isto é, são pontos ligados com código em cadeia igual. O comprimento de um declive corresponde ao número de movimentos necessários para efectuar o deslocamento consecutivo entre os pixels extremos do declive. Considerando estes conceitos definiram-se em [5] as seguintes condições de rectilineadade duma linha digital, ou seja, duma linha composta por pontos consecutivamente ligados, de coordenadas inteiras:

1. no máximo existirão dois declives diferentes;
2. os dois diferirão de 45º;
3. um dos declives terá comprimento 1;
4. o outro declive será composto por sequências de no máximo dois comprimentos diferentes (não é necessário verificar-se nos extremos), que apenas poderão diferir em uma unidade de comprimento.

A figura 55 ilustra estas condições, mostrando casos em que elas são violadas, e outros em que não o são. A figura 55a representa uma recta digital porque respeita as condições anteriores, enquanto 55b não satisfaz a condição 1, 55c não satisfaz a condição 4, na medida em que as duas sequências diferem de 2 unidades de comprimento, e 55d não satisfaz a condição 3 porque ambos os declives têm comprimento superior a 1.

Este critério de rectilineadade digital poderia ser utilizado para aproximação poligonal, conduzindo a resultados com segmentos exactamente rectos (segundo esta definição), constituídos pelos próprios pontos da linha. Tornaria, no entanto, o processo bastante pesado computacionalmente e resultaria numa solução com demasiados segmentos de recta.

O que a maior parte dos métodos utilizados tenta fazer, é encontrar partes das linhas que tenham um declive mais ou menos constante. Para este efeito utilizam-se duas abordagens, separando os métodos de aproximação poligonal em sequenciais ou iterativos [66]. Um método sequencial, começa a sua análise a partir de um ponto e vai juntando os pontos seguintes da linha, até que o conjunto não satisfaça um dado critério de rectilineadade. Os métodos iterativos, a partir de dois pontos (por exemplo os dois extremos numa linha aberta, ou os dois mais distantes numa linha fechada), verificam se o segmento de recta assim definido satisfaz um critério para aproximação da linha, aceitando o segmento em caso afirmativo, e partindo o segmento em dois em caso negativo.

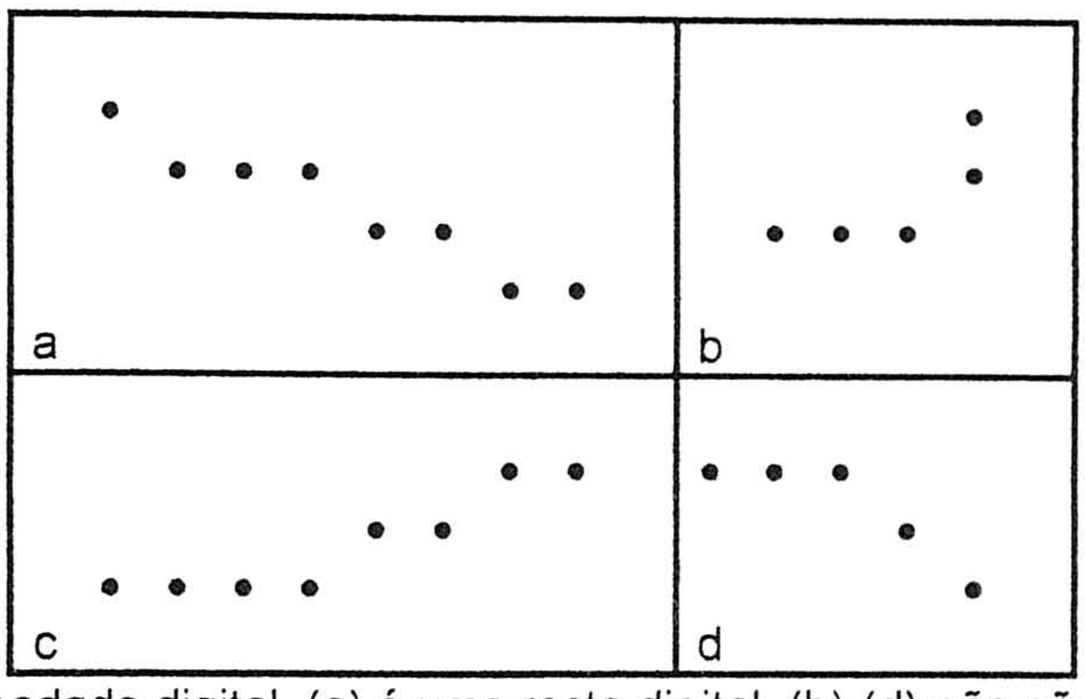


Figura 55: Rectilineadade digital. (a) é uma recta digital. (b)-(d) não são rectas digitais.

Um método de abordagem sequencial é por exemplo, aquele que determina o valor da curvatura (variação do declive) num determinado ponto, decidindo em função dum limiar estabelecido se esse ponto será ou não dominante. Pontos dominantes serão aqueles que descrevem a forma da linha de um modo aproximado. Numa aproximação poligonal, estes pontos serão aqueles em que a mudança de declive é tal que permite seleccioná-los para extremos de segmentos de recta. No caso da curvatura não exceder o limiar, então juntamos este ponto ao segmento de recta, e seguimos para a avaliação do próximo ponto. Este método é referido por Rosenfeld e Kak [5] e utilizado de uma forma similar por Wu e Wang [66], e será descrito com detalhe na sub secção seguinte, por ter sido um dos métodos utilizados.

Outro método de abordagem sequencial é o método de faixas de Leung e Yang [46], que agrega o maior número de pontos consecutivos, duma linha, desde que os mesmos estejam contidos numa faixa de largura $2 \cdot d$, sendo d a distância entre a recta central e as rectas exteriores da faixa. Pode ver-se na figura 56 que os pontos a-d serão aproximados pelo segmento de recta [o1,o2], segmento este que corresponde à recta central da faixa, delimitada pelas rectas exteriores, distantes de d da recta central.

Gonzalez e Woods [1], e Wu e Wang [66] referem métodos similares de aproximação poligonal de abordagem iterativa. Em [1] mede-se a distância (medida na

perpendicular à recta) máxima do segmento de recta candidato (definido pelos pontos extremos da linha) para a linha; se essa medida for inferior a um limiar aceita-se este segmento de aproximação, senão divide-se a linha em dois segmentos (no ponto de distância máxima) e repete-se o processo. Na figura 57 ilustra-se a aproximação de uma linha por partição de segmentos, baseado neste critério. Na figura 57, o segmento que une P1 a P2 será dividido em dois segmentos que unirão P1 a Pi, e Pi a P2, porque a distância do segmento à linha será máxima no ponto Pi, e ultrapassará o limiar. Em [66] além de se utilizar o método descrito, também se utiliza uma medida de curvatura c_i definida como sendo igual à divisão entre a distância do segmento de recta candidato a um ponto da linha e o comprimento da mesma. Aceita-se a aproximação se a curvatura máxima não exceder um limiar de curvatura, ou divide-se o segmento em dois se exceder o limiar e for um máximo local. A definição de curvatura será:

$$c_i = d_i / l_i .$$

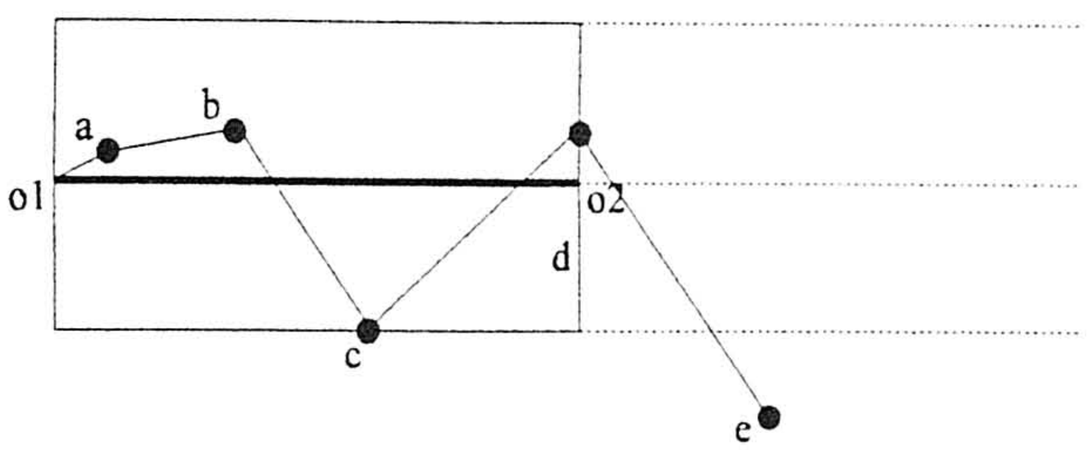


Figura 56: Ilustração do método de aproximação poligonal utilizando faixas dinâmicas.

Os métodos desenvolvidos neste trabalho vão ser descritos nas próximas duas sub secções, correspondendo a abordagens sequenciais. O primeiro destes métodos recorre a uma noção de k-curvatura (definida adiante) referida por diversos autores. O segundo método tem uma aplicação relativamente simples, utiliza um critério diferente relacionado com a área entre a linha e o segmento de recta de aproximação, e é potencialmente rápido.

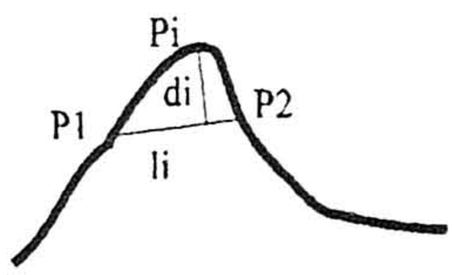


Figura 57: Método iterativo de aproximação poligonal.

5.2.1 Detecção de Cantos pelo Método do Coseno

Wu e Wang [66] descrevem uma forma de detectar cantos de linhas, isto é, de detectar pontos dominantes que descrevem a forma de uma curva de um modo apropriado para a sua percepção visual e também para o seu reconhecimento. Neste processo atribui-se um valor de curvatura a todos os pontos da linha, localizando-se em seguida os cantos. Os cantos serão os pontos com curvatura superior a um limiar.

Considerando o ponto P_i, o ângulo θ_i será a diferença entre os declives das duas rectas que se intersectam em P_i, e que passam uma por P_{i+k} e outra por P_{i-k}, conforme ilustrado na figura 58, sendo P_{i+k} o ponto k vezes posterior a P_i e P_{i-k} o ponto k vezes anterior a P_i. O ângulo θ_i denomina-se *k-curvatura* da linha no ponto P_i. Neste método [66] a definição de curvatura da linha no ponto P_i é dada por:

$$c_i = \cos\theta_i .$$

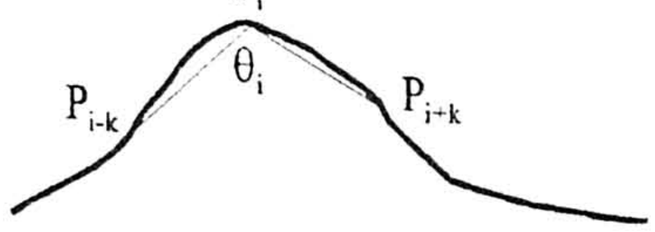


Figura 58: Ângulo de curvatura da linha num ponto, pelo método do coseno.

Em função da representação de pontos das linhas (secção anterior) disponível, com a utilização de códigos de sequência, pode efectuar-se uma pré-selecção de candidatos a pontos dominantes, considerando apenas os pontos em que haja mudança de direcção. Na figura 59 pode ver-se um conjunto de pontos que formam a seguinte descrição de códigos de sequência, começando no ponto mais à esquerda: 11000. O único ponto em que à mudança de direcção é o ponto em que o código passou de 1 para 0 (fundo mais escuro). Este ponto é portanto, um candidato a ponto dominante, ou canto.

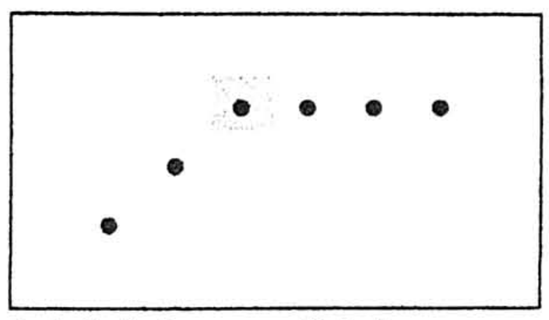


Figura 59: Selecção inicial de candidatos a pontos dominantes.

Considerando a curvatura c_i dos pontos candidatos a cantos, vão determinar-se aqueles que ultrapassam um determinado limiar, que assim serão classificados como cantos. No método implementado neste trabalho adoptou-se uma abordagem sequencial. Em vez da determinação inicial de todas as curvaturas, começa-se por considerar o primeiro ponto da linha como primeiro ponto extremo do primeiro segmento, e em seguida determina-se a curvatura do segundo ponto, se não ultrapassar o limiar (definido previamente pelo utilizador), determina-se a curvatura do terceiro ponto e assim sucessivamente até que ultrapasse o limiar, passando esse ponto a ser o segundo ponto extremo do segmento, e o primeiro ponto extremo do próximo segmento.

O processo repete-se até que não existam mais pontos para processar, obtendo-se assim uma representação poligonal da curva.

Na figura 60 ilustra-se a aplicação da detecção de cantos pelo método do coseno. A utilização de um ângulo de curvatura limiar aproximadamente igual a 36º, com k=6, permite efectuar a aproximação poligonal das linhas da imagem original detectando

apenas um canto visível (figura 60). Um outro canto correspondente a um ângulo de curvatura próximo de 90º, é detectado utilizando um limiar do coseno igual a -0.2 (101.5º), e ilustrado pela figura 60. Finalmente, para se detectar um canto no outro componente da imagem, é necessário utilizar um limiar do coseno igual a -0.95 (161.8º) e ilustrado pela figura 60. Note-se que os ângulos de curvatura nos cantos referidos com k=6, e ilustrados na figura 60 de (b) a (d), são respectivamente 33º, 101.3º e 158.2º.

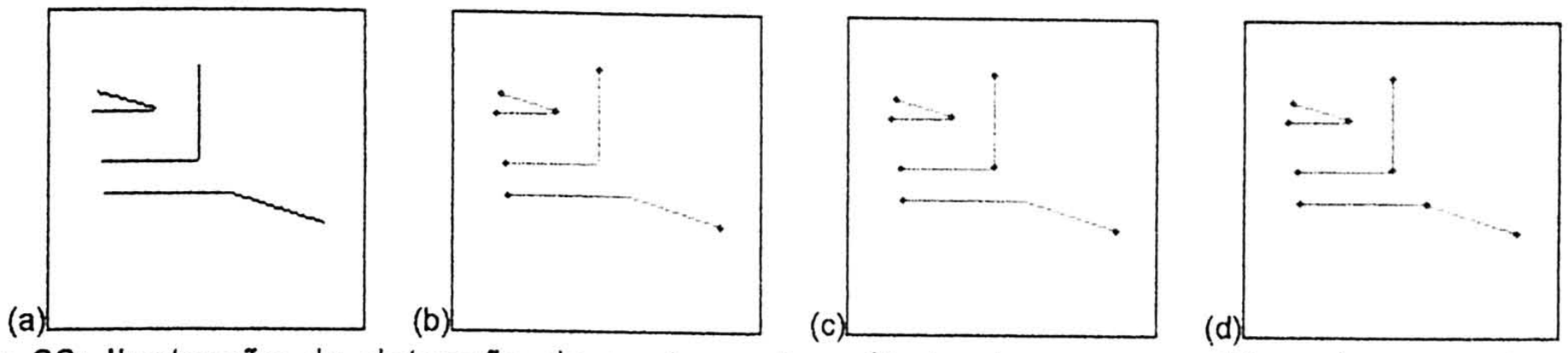


Figura 60: Ilustração da detecção de cantos pelo método do coseno, utilizando um valor de k=6. (a) Imagem original. (b) Detecção de cantos com cos=0.8 (c) Detecção de cantos com cos=-0.2 (d) Detecção de cantos com cos=-0.95.

Na figura 61 ilustra-se a aplicação da detecção de cantos pelo método do coseno, sobre a imagem da esquerda. A utilização de um ângulo de curvatura limiar aproximadamente igual a 150º (cos=-0.866), com k=6, permite efectuar a detecção de cantos das linhas da imagem, representados na figura 61 central. Na figura 61 à direita é apresentada a imagem resultado da aproximação poligonal das suas linhas. Os dois componentes (aproximadamente rectângulos) de maiores dimensões da imagem central da figura 61, foram aproximados por segmentos, de tal forma que houve uma compressão de dados do componente exterior de 704 para 6 pontos (taxa de compressão igual 99.15%) e do componente interior de 606 para 15 pontos (taxa de compressão igual 97.52%) na representação das suas linhas.

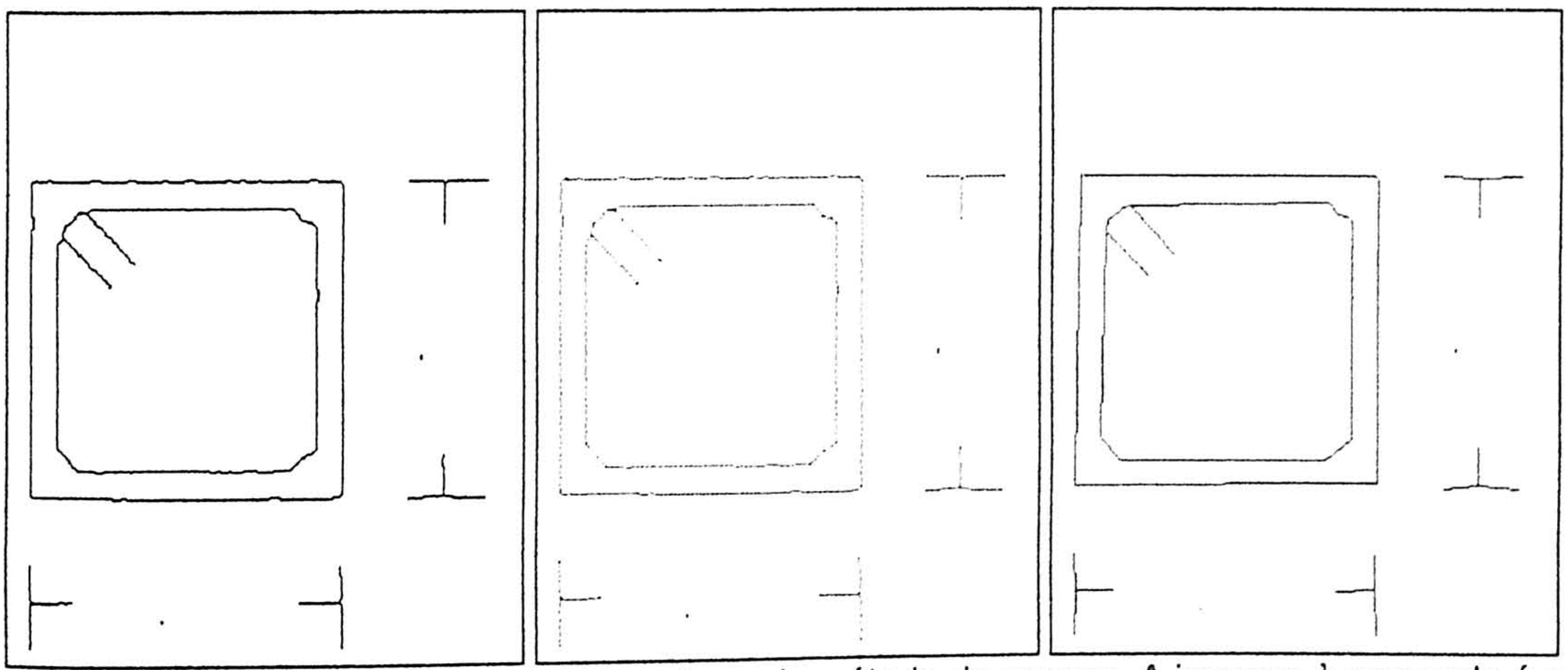


Figura 61: Ilustração da detecção de cantos pelo método do coseno. A imagem à esquerda é a imagem original. A imagem central corresponde às linhas mais os cantos detectados (mais escuros). A imagem à direita corresponde à aproximação poligonal.

Na figura 62 ilustra-se a aplicação da detecção de cantos pelo método do coseno, sobre imagem da esquerda. A utilização de um ângulo de curvatura limiar aproximadamente igual a 15º (cos=0.96), com k=10, permite efectuar a detecção de cantos das linhas da imagem, representados na figura 62 ao centro, a que corresponde a aproximação poligonal representada à direita. Os dois componentes de maiores dimensões da imagem da figura 62, foram aproximados por segmentos de tal forma que houve uma compressão de dados do componente exterior de 502 para 18 pontos (taxa de compressão igual 96.41%) e do componente interior de 549 para 42 pontos (taxa de compressão igual 92.35%) na representação das suas linhas.

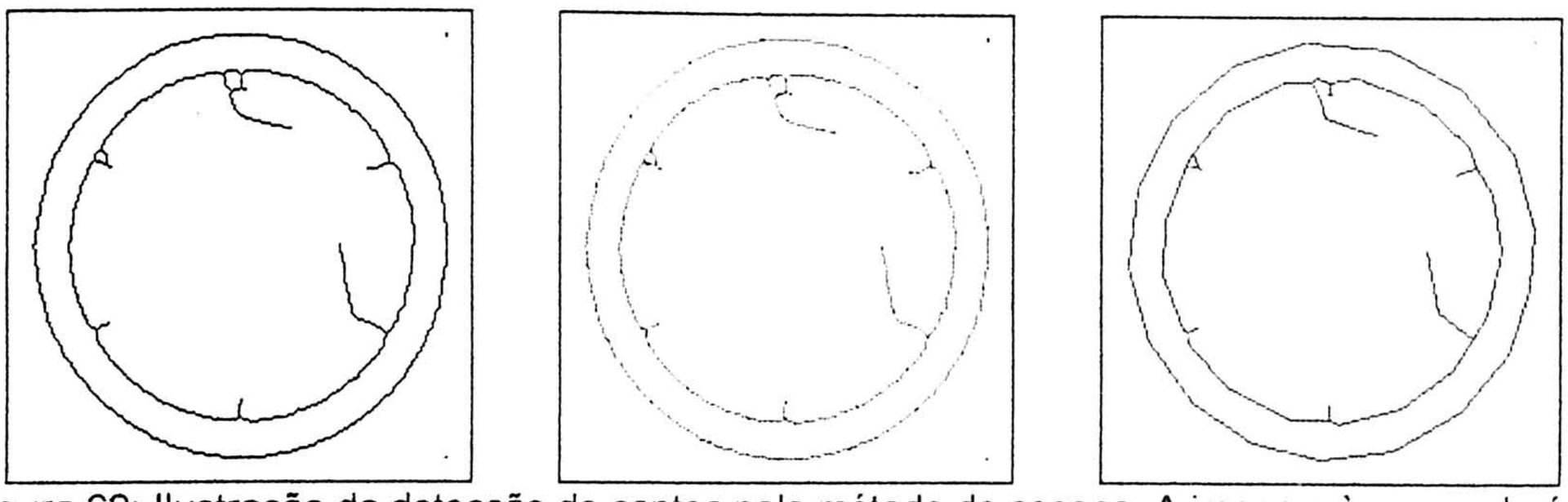


Figura 62: Ilustração da detecção de cantos pelo método do coseno. A imagem à esquerda é a imagem original. A imagem central corresponde às linhas mais os cantos detectados (mais escuros) utilizando um limiar do coseno igual a -0.9, com k=6. A imagem à direita corresponde à aproximação poligonal.

5.2.2 Método de Rápida Aproximação Poligonal

O método de rápida aproximação poligonal de Wall e Danielsson [54] é um método de abordagem sequencial, que vai avaliando pontos sequencialmente a partir do ponto inicial, até que um determinado critério não seja satisfeito. O critério baseia-se na área entre o segmento de recta (entre o ponto inicial e o ponto em avaliação) e a linha (definida por todos os pontos da linha entre o ponto inicial e o ponto em avaliação). Mais concretamente, baseia-se na área de desvio por unidade de comprimento do segmento de recta candidato, sendo a área de desvio a diferença entre as áreas acima e abaixo do segmento de recta de aproximação delimitada pela curva.

O segmento de recta seleccionado é definido pelo ponto inicial e pelo último ponto que passar o teste, por avaliação sucessiva dos pontos que se sigam ao inicial. O critério utiliza um parâmetro T, denominado máxima área de desvio por unidade de comprimento do segmento de aproximação. Neste método impõe-se que os extremos dos segmentos de recta de aproximação pertençam à linha. Neste caso o erro de aproximação vai ser maior do que se se permitisse uma escolha mais livre (em [46] apresenta-se um método em que os extremos do segmento de aproximação não coincidem com pontos da linha a

aproximar), ganhando-se no entanto, em velocidade de processamento, por se restringir o campo de selecção.

A figura 63 mostra graficamente a área entre uma linha e um segmento de aproximação dessa linha, isto é, ilustra a área de desvio entre a aproximação e a curva.

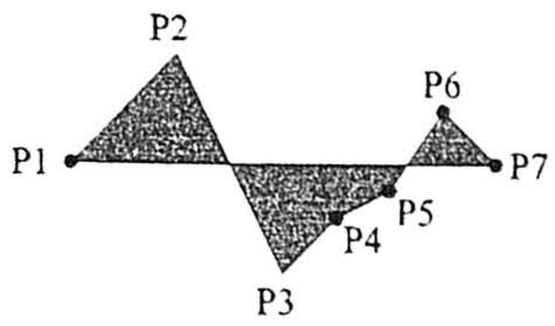


Figura 63: Ilustração da área entre um segmento de recta de aproximação (segmento [P1,P7]) e a própria linha aproximada, definida pelos pontos consecutivos P1 a P7.

Em seguida descreve-se o método de cálculo da área de desvio, durante o processo de aproximação. No decorrer da abordagem sequencial de aproximação é necessário determinar a área de desvio num determinado ponto, para se tomar a decisão de continuar, ou não, para o ponto seguinte. Se continuarmos para o ponto seguinte e com o intuito de acelerar o processo, actualiza-se a área de desvio a partir da informação do ponto anterior.

O processo vai ser descrito com a ajuda da ilustração da figura 64, em que se considera o ponto inicial coincidente com a origem do sistema de coordenadas. Se P_{i-1} ainda respeitar o critério utilizado, então vamos verificar sequencialmente o próximo ponto. A área de desvio do segmento $[O,P_i]$ relativamente à linha será igual à área de desvio do segmento anterior $[O,P_{i-1}]$ mais a área de desvio incremental, que é a área compreendida entre os lados do triângulo definido pelo novo segmento de recta ($[O,P_i]$, ou pelo respectivo vector $\overline{L}_i$), pelo segmento de recta anterior ($[O,P_{i-1}]$, ou pelo respectivo vector $\overline{L}_{i-1}$), e pelo segmento relativo ao deslocamento de um ponto para o outro ($[P_{i-1},P_i]$, ou pelo respectivo vector $\overline{\Delta}_i$). Sendo L_i o módulo do respectivo vector, d_i a distância, com sinal, do segmento $[O,P_i]$ ao ponto P_{i-1}, a área de desvio incremental $\Delta f_i/2$, será definida pela equação:

$$\frac{\Delta f_i}{2} = \frac{L_i \cdot d_i}{2}.$$

Sendo o cálculo de Δf_i efectuado pela equação equivalente:

$$\Delta f_i = L_i \cdot d_i = \pm\left|\overline{L}_{i-1} \times \overline{\Delta}_i\right| = x_i \cdot \Delta y_i - y_i \cdot \Delta x_i .$$

A área de desvio $f_i/2$ no ponto P_i, correspondente à diferença entre a área acima e abaixo do segmento de recta de aproximação (entre o ponto inicial e P_i) delimitada pela curva, será definida por:

$$\frac{f_i}{2} = \frac{\sum_{r=1}^{i} \Delta f_r}{2} = \frac{f_{i-1} + \Delta f_i}{2}.$$

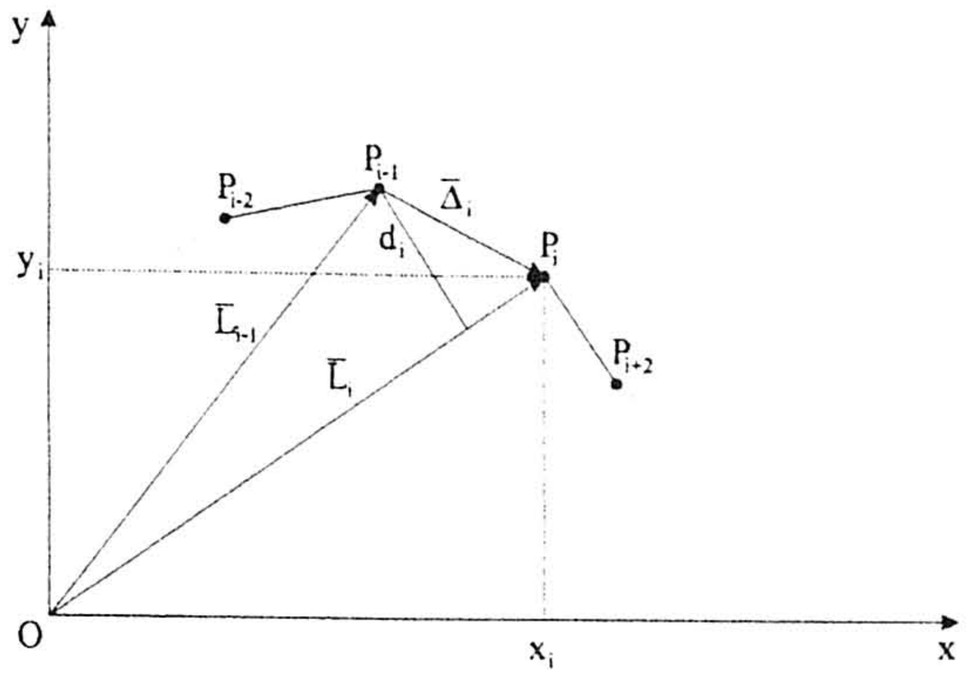


Figura 64: Ilustração do processo de rápida aproximação poligonal.

O critério de selecção de um ponto da linha aproximada pelo segmento será:

$$|f_i| / L_i \leq T,$$

sendo o T a máxima área de desvio por unidade de comprimento do segmento de aproximação,

A partir destes conceitos básicos o algoritmo de rápida aproximação poligonal é constituído pelos seguintes passos:

[i] $i = 2$.

[ii] Transformar o sistema de coordenadas de modo a fazer coincidir o ponto inicial P_1 com a origem dos eixos.

[iii] $f_i = 0$.

[iv] Determinar o deslocamento $\overline{\Delta}_i = (\Delta x_i, \Delta y_i)$ e passar para o próximo ponto $P_i = P_{i-1} + \overline{\Delta}_i = (x_i, y_i)$.

[v] Determinar a área de desvio f_i, por acumulação da área de desvio incremental: $\Delta f_i = x_i \cdot \Delta y_i - y_i \cdot \Delta x_i$; $f_i = f_{i-1} + \Delta f_i$.

[vi] Calcular o comprimento L_i do segmento de recta actual, desde o ponto inicial até P_i: $L_i = \sqrt{x_i^2 + y_i^2}$.

[vii] Aplicar o teste: $|f_i| \leq T \cdot L_i$

[viii] **Se** o teste foi satisfeito, incrementar i e voltar para [i]; **senão,** foi encontrado o mais longo segmento permitido, ou seja, uma linha desde a origem até P_{i-1}. P_{i-1} é tomado como ponto inicial para o próximo segmento, e volta-se ao inicio.

[x] Terminar quando não houver mais pontos para processar.

No final da determinação de todos os segmentos de aproximação da curva, a diferença entre a área do polígono e a área da curva será delimitada por:

$$T/2 \cdot \sum_{r=1}^{n} L_r ,$$

sendo n o número de segmentos de recta de aproximação.

Para curvas convexas, a distância máxima $\varepsilon_{\max}$ entre a curva e o polígono será maior quando a área de desvio corresponder a um triângulo, sendo neste caso:

$|f_i| = L_i \cdot \varepsilon_{\max} \leq L_i \cdot T \equiv \varepsilon_{\max} \leq T$.

Para simplificar o algoritmo convém utilizar apenas aritmética inteira, de forma a tornar mais rápido o processamento computacional. Isto consegue-se modificando o critério de modo a evitar o cálculo de raízes quadradas: $f_i^2 \leq L_i^2 \cdot T^2$.

Na figura 65 estão representadas duas imagens originais, formadas por desenhos de linhas. A aproximação das linhas das duas imagens pelo métodos de rápida aproximação poligonal é ilustrada pela figura 66.

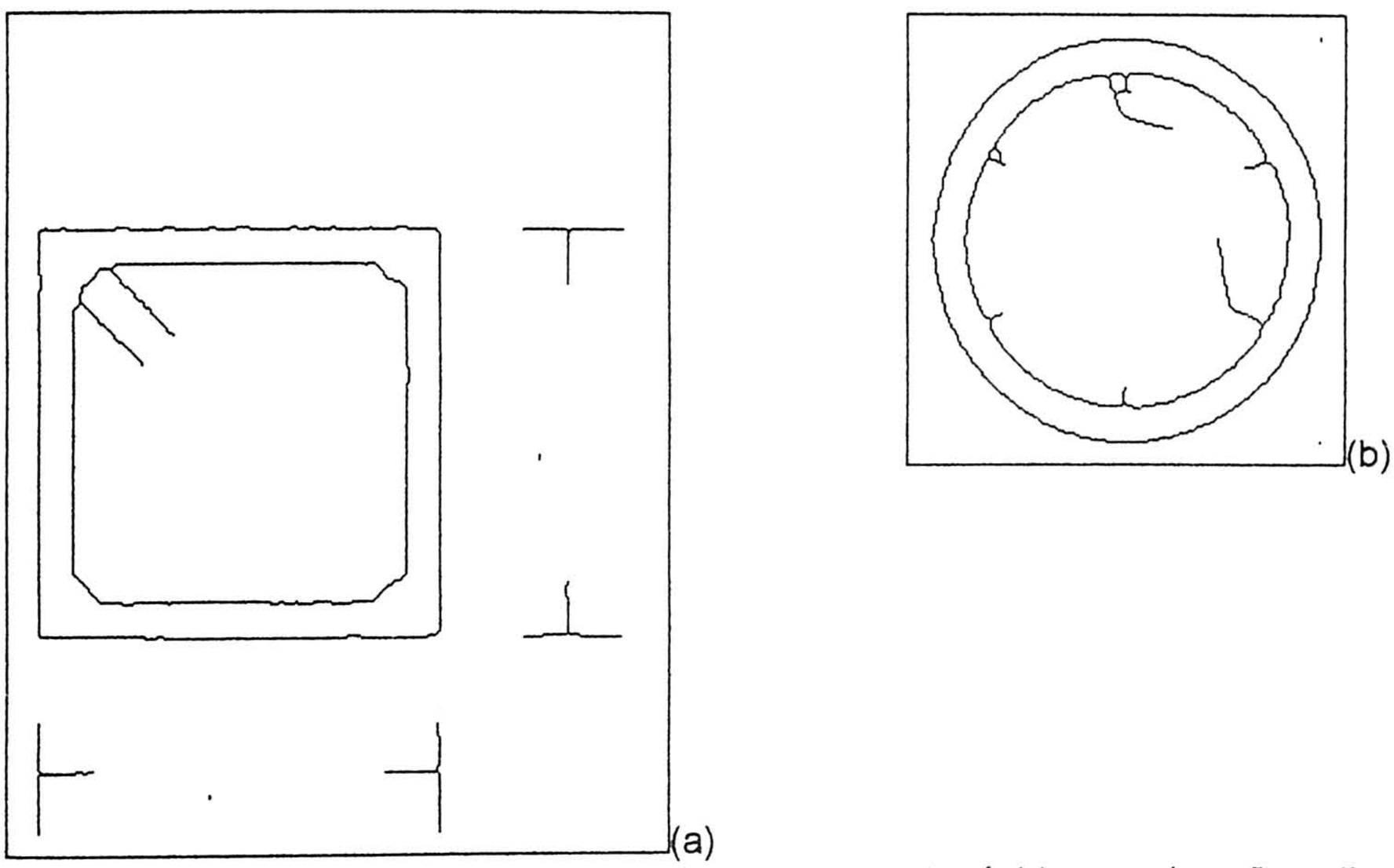


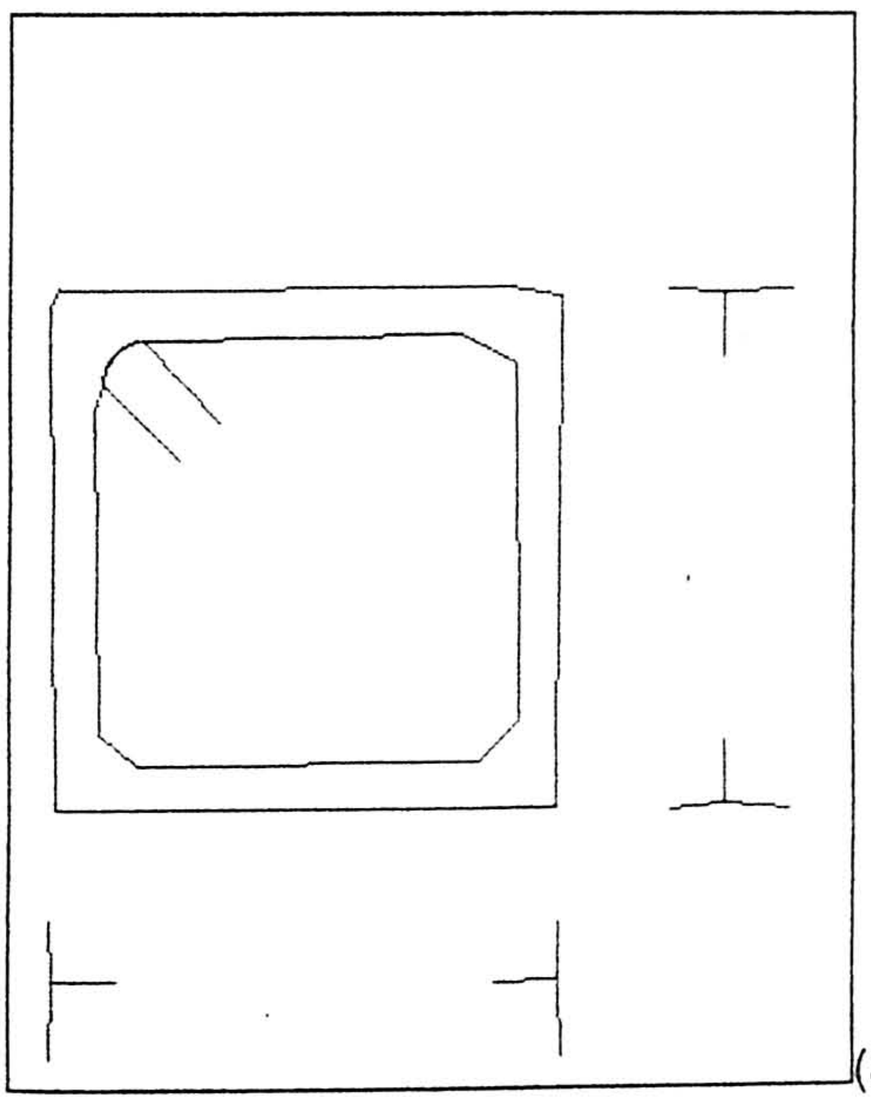
Figura 65: Imagens originais para ilustração do método de rápida aproximação poligonal.

O resultado de efectuar a rápida aproximação poligonal das linhas das imagens originais está apresentado na figura 66. Na aplicação do método sobre ambas as imagens foi utilizado o parâmetro da área de desvio máximo de 2.0. É visível que a imagem na figura 66(a) é mais aproximada da imagem original, o que seria de esperar, por ser fundamentalmente formada por linhas com pequena variação de declive. A imagem da figura 66(b) não se aproxima, visualmente, tanto da imagem original porque esta é formada por linhas em que o declive varia de uma forma quase contínua. Na primeira imagem (a) a compressão de dados do componente exterior foi de 704 para 7

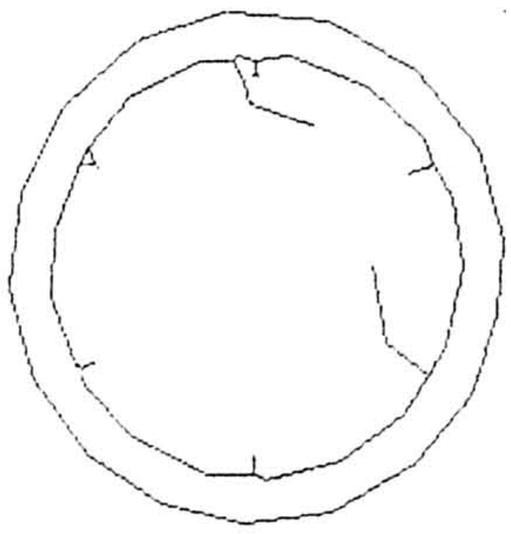
pontos (taxa=99.01%), e do componente interior foi de 606 para 14 (taxa=97.69%). Na segunda imagem a compressão de dados do componente exterior foi de 502 para 18 (taxa=96.41%), e do componente interior foi de 549 para 42 (taxa=92.35%).

5.2.3 Comparação de Métodos

As ilustrações apresentadas, sobre os dois métodos de aproximação poligonal mostram que é possível, com ambos, obter bons resultados de aproximação de linhas por segmentos de recta. Para a imagem da figura 65a ambos os métodos apresentam resultados visíveis e de compressão similares. Para a imagem da figura 65b os resultados visuais são similares, com taxas de compressão iguais. Um método de detecção de cantos, tem dificuldade em encontrar cantos em linhas cuja variação de declive seja aproximadamente contínua, embora possa encontrar cantos se se efectuar uma adaptação detalhada dos parâmetros do algoritmo (limiar de curvatura e distância k). Por seu lado, o método de rápida aproximação poligonal, obteve bons resultados para as duas imagens, com o recurso ao mesmo valor limiar. Este método, baseado na área de desvio entre o segmento de recta e a linha, será de uma forma geral preferível, por recorrer a um critério que poderá ser eventualmente escolhido como uma constante, desde que não se exija sempre que a curvatura da descrição poligonal esteja demasiado próxima (neste caso poderíamos que ter que utilizar outros valores para o limiar T) da curvatura da linha. Com o método de detecção de cantos implementado, será sempre necessário escolher limiares de curvatura que se adaptem a todas as linhas da imagem, para se conseguir uma descrição aproximada.



(a)



(b)

Figura 66: Imagens obtidas por aplicação do método de rápida aproximação poligonal, com T=2.0.

5.3 Aproximação de Linhas Utilizando Arcos de Circunferência

O que acontece ao aproximarmos linhas por polígonos, é que determinados segmentos das linhas ficam mal representados por corresponderem essencialmente a curvas. Verifica-se neste caso, uma filtragem da curva representada por poucos segmentos de recta, ou um aumento da memória de armazenamento e da complexidade de representação pelo aumento do número de segmentos de recta curtos. Em ambos os casos, a extracção de características, relacionadas com a forma da curva pode-se tornar demasiado complexa, ou mesmo impossível.

Tentando ultrapassar estas dificuldades, utilizam-se funções com representação gráfica curva (com variação contínua do declive) para aproximar estes segmentos de linhas. As funções de arcos de circunferência, de curvas de Bézier e de Splines têm sido as mais utilizadas. No entanto, a sua utilização tem sido mais intensiva em programas vectoriais de desenho do que em processamento de imagem.

É ao problema de aproximação de linhas por funções curvas que se tenta dar solução, recorrendo-se, com este objectivo, a um algoritmo de aproximação de linhas por arcos de circunferência, e por splines.

Os algoritmos de desenho de arcos de circunferência são, usualmente, bastante rápidos, pelo que a sua utilização na aproximação de linhas, respeitando um critério pré-definido, permite definir um método de aproximação de linhas com arcos de circunferência. Nesta secção vai descrever-se um método, com uma abordagem deste tipo.

5.3.1 Aproximação de Linhas por Arcos de Três Pontos

A aproximação de linhas utilizando arcos de circunferência pode ser fundamental para a descrição de linhas resultantes de imagens de documentos de engenharia, que tipicamente contêm inúmeras circunferências e arcos de circunferência.

Albano [38] descreve um método para aproximação de contornos por arcos cónicos, e Nagasamy e Langrana [24] e Thomas e Chan [53] estimam o centro e raio de uma circunferência que melhor aproxima um conjunto de pontos.

Galton [43] por sua vez, descreve um algoritmo para desenho de arcos de circunferência a partir de três pontos do arco, que tipicamente poderia ser incluído num programa de desenho. No entanto, por ser uma abordagem diferente, e por não parecer muito penalizadora, devido à velocidade de desenho destes algoritmos, adaptou-se o processo à aproximação de linhas por arcos de circunferência. Esta adaptação escolhe os dois pontos extremos, e um ponto intermédio da curva para formar o arco, e a partir

daí determina-se a área entre a linha e o arco de aproximação. Utilizando esta ideia poder-se-ía escolher o arco logo que a área fosse inferior a um dado limiar, ou então, procurar o ponto intermédio que minimize essa área.

Em seguida vai descrever-se o processo de desenho de um arco utilizando três pontos [43]. Os três pontos utilizados serão os dois pontos extremos do arco e um intermédio, o que permite uma certa flexibilidade na manipulação de arcos por alteração deste ponto.

A principal inovação, segundo os autores, deste algoritmo é de não impor uma curvatura mínima, porque as suas variáveis não dependem do raio, logo, não são ilimitadas. O raio de um arco de circunferência tende para o infinito quando o arco tende para o segmento de recta. Ou seja, em última análise, este algoritmo possibilita mesmo o desenho de segmentos de recta.

Recorrendo a uma equação de circunferência, este algoritmo permite desenhar arcos de circunferência definidos por três pontos recorrendo a aritmética de números inteiros. Seja (x_0, y_0) um ponto inicial, e (x_A, y_A) um qualquer ponto denominado ponto fixo diferente do ponto inicial. Para qualquer par de constantes positivas α e β diferentes de zero, existe uma circunferência ou recta passando pelo ponto inicial definida por

$$\alpha \cdot v^2(x,y) - \beta \cdot u^2(x,y) = \gamma ,$$

sendo γ uma constante, e $u(x,y)$ e $v(x,y)$ funções definidas pelas equações:

$$u(x,y) = \sqrt{(x_0 - x)^2 + (y_0 - y)^2} ,$$

$$v(x,y) = \sqrt{(x_A - x)^2 + (y_A - y)^2} .$$

A função da circunferência assim definida é equivalente à equação da circunferência $(x_0 - x)^2 + (y_0 - y)^2 = {R_0}^2$ se $\alpha \neq \beta$, e é equivalente à equação da recta $y = m \cdot x + b$ se $\alpha = \beta$, sendo os parâmetros x_0, y_0, R_0, m e b constantes dependentes dos três pontos que definem o arco.

A figura 67 ilustra o significado de diversas variáveis da função. As constantes α e β são denominadas por factores de curvatura. A utilização desta função para o desenho de arcos exige a determinação do ponto fixo e dos factores de curvatura. O cálculo destes parâmetros faz-se pela resolução do sistema formado pelas três equações da circunferência resolvidas nos três pontos do arco.

A solução do sistema leva ao ponto fixo determinado pelas equações:

$$x_A = x_0 - \frac{C_0}{\tau},$$

$$y_A = y_0 - \frac{C_1}{\tau},$$

sendo τ uma constante, e os valores de C_0 e C_1 determinados por $C_0 = (y_0 - y_1) \cdot u^2(x_2, y_2) - (y_0 - y_2) \cdot u^2(x_1, y_1)$, e $C_1 = (x_0 - x_1) \cdot u^2(x_2, y_2) - (x_0 - x_2) \cdot u^2(x_1, y_1)$.

A solução do sistema dá-nos os factores de curvatura determinados pelas equações:

$\alpha = u^2(x_2, y_2)$, correspondendo ao quadrado da distância entre os pontos extremos;

$\beta = v^2(x_2, y_2) - v^2(x_0, y_0)$, correspondendo à diferença entre os quadrados das distâncias dos pontos extremos ao ponto fixo.

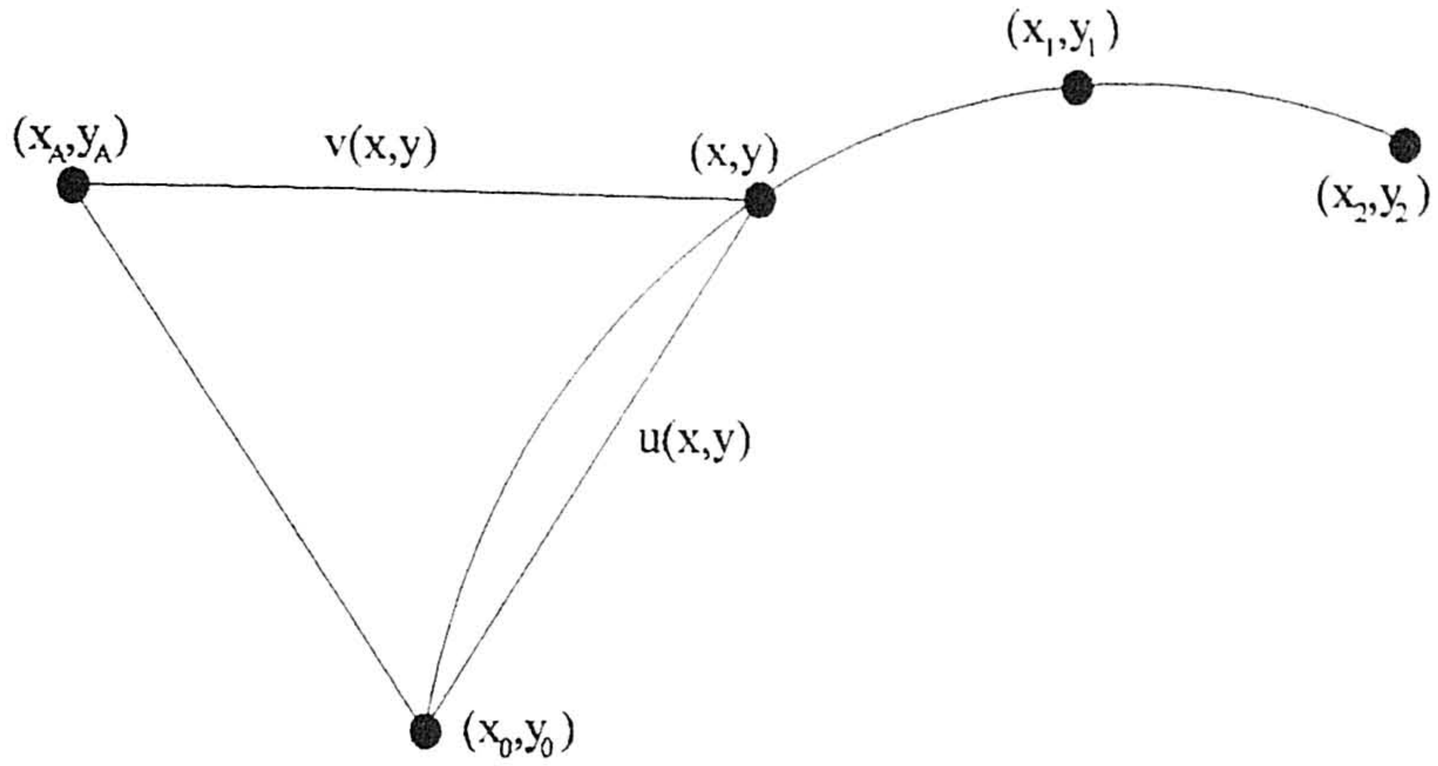


Figura 67: Relação entre os parâmetros da função da circunferência.

As equações apresentadas até ao momento são exactas e contínuas, podendo ser utilizadas para determinar os pontos discretos do arco. Mas neste caso o algoritmo seria ineficiente (para gerar arcos em imagens digitais) porque geraria informação desnecessária (parte decimal das coordenadas de cada ponto), e não aproveitaria a informação de cada ponto para gerar o seguinte. É um algoritmo baseado nestas premissas que se pretende, de modo a aumentar a eficiência, indo ser descrito em seguida.

Ao desenhar uma circunferência, o ponto seguinte será sempre um dos 8-vizinhos do actual. A utilização da informação referente à tangente e à direcção de desenho, relativa a cada octante, permite seleccionar apenas dois pontos seguintes possíveis. Apenas um destes pontos será escolhido, por minimização do erro de quantificação, definido como resultado da diferença entre a função da circunferência quantificada nas coordenadas reais (x, y), e a mesma função quantificada nas coordenadas inteiras $(\hat{x}, \hat{y})$.

A definição do erro será então:

$\varepsilon(\hat{x}, \hat{y}) = \alpha \cdot v^2(\hat{x}, \hat{y}) - \beta \cdot u^2(\hat{x}, \hat{y}) - \gamma$.

Como $\alpha \cdot v^2(x_0, y_0) - \beta \cdot u^2(x_0, y_0) = \gamma \Leftrightarrow \alpha \cdot v^2(x_0, y_0) = \gamma$, logo γ é conhecido, e o erro poderia ser quantificado em cada ponto gerado utilizando a função anterior. No entanto, torna-se mais eficiente determinar o erro a partir do ponto anterior, utilizando equações diferenciais de erro, permitindo uma implementação incremental da equação de erro.

O erro de deslocamento do ponto actual para o seu vizinho resulta da subtracção entre os erros nesses pontos. Assim sendo, vamos determinar o erro de deslocamento para o vizinho a leste:

$$\begin{aligned}\Delta\varepsilon_{x+}(\hat{x}) &= \varepsilon(\hat{x}+1,\hat{y}) - \varepsilon(\hat{x},\hat{y}) = \alpha\cdot v^2(\hat{x}+1,\hat{y}) - \beta\cdot u^2(\hat{x}+1,\hat{y}) - \gamma - \varepsilon(\hat{x},\hat{y}) \\ &= \alpha\left[(x_A-\hat{x}-1)^2+(y_A-\hat{y})^2\right] - \beta\left[(x_0-\hat{x}-1)^2+(y_0-\hat{y})^2\right] - \gamma - \varepsilon(\hat{x},\hat{y}) \\ &= \alpha\left[(x_A-\hat{x})-1\right]^2 + \alpha(y_A-\hat{y})^2 - \beta\left[(x_0-\hat{x})-1\right]^2 - \beta(y_0-\hat{y})^2 - \gamma - \varepsilon(\hat{x},\hat{y}) \\ &= \alpha(x_A-\hat{x})^2 + \alpha - 2\alpha(x_A-\hat{x}) + \alpha(y_A-\hat{y})^2 - \beta(x_0-\hat{x})^2 - \beta + 2\beta(x_0-\hat{x}) - \beta(y_0-\hat{y})^2 - \gamma - \varepsilon(\hat{x},\hat{y}) \\ &= \alpha\left[1-2(x_A-\hat{x})\right] - \beta\left[1-2(x_0-\hat{x})\right] + \alpha\left[(x_A-\hat{x})^2+(y_A-\hat{y})^2\right] - \beta\left[(x_0-\hat{x})^2+(y_0-\hat{y})^2\right] - \gamma - \varepsilon(\hat{x},\hat{y}) \\ &= \alpha\left[1-2(x_A-\hat{x})\right] - \beta\left[1-2(x_0-\hat{x})\right]\end{aligned}$$

De uma forma similar determinavam-se os erros de deslocamento para os quatro vizinhos:

$$\Delta\varepsilon_{x+}(\hat{x}) = \varepsilon(\hat{x}+1,\hat{y}) - \varepsilon(\hat{x},\hat{y}) = \alpha\left[1-2(x_A-\hat{x})\right] - \beta\left[1-2(x_0-\hat{x})\right];$$

$$\Delta\varepsilon_{x-}(\hat{x}) = \varepsilon(\hat{x}-1,\hat{y}) - \varepsilon(\hat{x},\hat{y}) = \alpha\left[1+2(x_A-\hat{x})\right] - \beta\left[1+2(x_0-\hat{x})\right];$$

$$\Delta\varepsilon_{y+}(\hat{y}) = \varepsilon(\hat{x},\hat{y}+1) - \varepsilon(\hat{x},\hat{y}) = \alpha\left[1-2(y_A-\hat{y})\right] - \beta\left[1-2(y_0-\hat{y})\right];$$

$$\Delta\varepsilon_{y-}(\hat{y}) = \varepsilon(\hat{x},\hat{y}-1) - \varepsilon(\hat{x},\hat{y}) = \alpha\left[1+2(y_A-\hat{y})\right] - \beta\left[1+2(y_0-\hat{y})\right].$$

Na prática utiliza-se um segundo conjunto de funções de diferença do erro, que relacionam a própria diferença de deslocamento de um ponto com a diferença de erro anterior. A diferença de erro ao incrementar x de uma unidade será:

$$\Delta\varepsilon_{x+}(\hat{x}\pm1) = \alpha\left[1-2(x_A-\hat{x}-1)\right] - \beta\left[1-2(x_0-\hat{x}-1)\right] = \Delta\varepsilon_{x+}(\hat{x}) \pm 2(\alpha-\beta).$$

De uma forma similar determinaram-se os outros erros incrementais:

$$\Delta\varepsilon_{x+}(\hat{x}\pm1) = \Delta\varepsilon_{x+}(\hat{x}) \pm 2(\alpha-\beta);$$

$$\Delta\varepsilon_{x-}(\hat{x}\pm1) = \Delta\varepsilon_{x-}(\hat{x}) \mp 2(\alpha-\beta);$$

$$\Delta\varepsilon_{y+}(\hat{y}\pm1) = \Delta\varepsilon_{y+}(\hat{y}) \pm 2(\alpha-\beta);$$

$$\Delta\varepsilon_{y-}(\hat{y}\pm1) = \Delta\varepsilon_{y-}(\hat{y}) \mp 2(\alpha-\beta).$$

Para desenhar uma circunferência é necessário considerar as mudanças de octante, ao incrementar cada ponto. Por exemplo, se estivermos no primeiro octante, $\partial x/\partial y < -1$.

Logo, se o desenho se efectuar no sentido contrário ao do movimento dos ponteiros do relógio, y aumentará mais rapidamente do que x. O ponto seguinte terá a coordenada vertical incrementada, e a coordenada horizontal decrementada ou não.

Em cada octante uma das coordenadas (variável de passo) é sempre incrementada (ou decrementada) enquanto a outra (variável de teste) se mantém inalterada ou é incrementada (ou decrementada), de modo a minimizar o erro relativo à quantificação discreta das coordenadas. As funções de diferença do erro permitem manter a informação relativa ao octante em processamento, da forma apresentada na tabela 6,

considerando que o desenho se efectua no sentido contrário ao do movimento dos ponteiros do relógio.

Tabela 6: Condições de cada octante definidas pelas funções de diferença do erro, considerando o desenho dos arcos no sentido contrário ao do movimento dos ponteiros do relógio.

Octante	Funções de diferença do erro.		
$[0,\frac{\pi}{4}[$	$\Delta\varepsilon_{x-}(\hat{x})>0$	$\Delta\varepsilon_{y+}(\hat{y})<0$	$\lvert\Delta\varepsilon_{x-}(\hat{x})\rvert>\lvert\Delta\varepsilon_{y+}(\hat{y})\rvert$
$[\frac{\pi}{4},\frac{\pi}{2}[$	$\Delta\varepsilon_{x-}(\hat{x})\geq 0$	$\Delta\varepsilon_{y+}(\hat{y})<0$	$\lvert\Delta\varepsilon_{x-}(\hat{x})\rvert\leq\lvert\Delta\varepsilon_{y+}(\hat{y})\rvert$
$[\frac{\pi}{2},\frac{3\pi}{4}[$	$\Delta\varepsilon_{x-}(\hat{x})<0$	$\Delta\varepsilon_{y-}(\hat{y})>0$	$\lvert\Delta\varepsilon_{x-}(\hat{x})\rvert<\lvert\Delta\varepsilon_{y-}(\hat{y})\rvert$
$[\frac{3\pi}{4},\pi[$	$\Delta\varepsilon_{x-}(\hat{x})<0$	$\Delta\varepsilon_{y-}(\hat{y})\geq 0$	$\lvert\Delta\varepsilon_{x-}(\hat{x})\rvert\geq\lvert\Delta\varepsilon_{y-}(\hat{y})\rvert$
$[\pi,\frac{5\pi}{4}[$	$\Delta\varepsilon_{x+}(\hat{x})>0$	$\Delta\varepsilon_{y-}(\hat{y})<0$	$\lvert\Delta\varepsilon_{x+}(\hat{x})\rvert>\lvert\Delta\varepsilon_{y-}(\hat{y})\rvert$
$[\frac{5\pi}{4},\frac{3\pi}{2}[$	$\Delta\varepsilon_{x+}(\hat{x})\geq 0$	$\Delta\varepsilon_{y-}(\hat{y})<0$	$\lvert\Delta\varepsilon_{x+}(\hat{x})\rvert\leq\lvert\Delta\varepsilon_{y-}(\hat{y})\rvert$
$[\frac{3\pi}{2},\frac{7\pi}{4}[$	$\Delta\varepsilon_{x+}(\hat{x})<0$	$\Delta\varepsilon_{y+}(\hat{y})>0$	$\lvert\Delta\varepsilon_{x+}(\hat{x})\rvert<\lvert\Delta\varepsilon_{y+}(\hat{y})\rvert$
$[0,\frac{\pi}{4}[$	$\Delta\varepsilon_{x+}(\hat{x})<0$	$\Delta\varepsilon_{y+}(\hat{y})\geq 0$	$\lvert\Delta\varepsilon_{x+}(\hat{x})\rvert\geq\lvert\Delta\varepsilon_{y+}(\hat{y})\rvert$

Para que o algoritmo utilize apenas variáveis inteiras é necessário determinar um ponto fixo, também ele de coordenadas inteiras. Como τ pode tomar qualquer valor constante, poder-se-á escolher o valor 1, conduzindo neste caso a um valor muito elevado para o ponto fixo (variável de armazenamento com mais bits do que os das coordenadas dos pontos do arco). Uma outra escolha permite que as coordenadas do ponto fixo sejam arredondadas, mantendo os três pontos no arco desenhado, mesmo que o arco definido no espaço real não coincida exactamente com um deles. Este valor de τ é definido por Galton [43] como:

$$\tau=\frac{\max(|C_0|,|C_1|)}{2(|x_0-x_1|+|y_0-y_1|)}.$$

O algoritmo para o desenho do arco no primeiro quadrante será constituído pelos seguintes passos:

[i] Inicializar $\varepsilon(\hat{x},\hat{y})=0;\ \hat{x}=x_0;\ \hat{y}=y_0$.

[ii] Inicializar as variáveis C_0, C_1, τ, x_A, y_A, α, β, $\Delta\varepsilon_{x-}(\hat{x})$ e $\Delta\varepsilon_{y+}(\hat{y})$ com os valores definidos pelas funções acima descritas.

[iii] Desenhar o ponto (x,y).

[iv] Incrementar y, que é a variável passo no primeiro octante.

[v] Actualizar o erro acumulando a diferença $\Delta\varepsilon_{y+}(\hat{y})$.

[vi] Actualizar a diferença de erro $\Delta\varepsilon_{y+}(\hat{y}+1)$ acumulando $2(\alpha-\beta)$.

[vii] Se o erro de decrementar $\hat{x}$ ao erro de manter inalterada a variável, então decrementa-se actualizando-se os erros. A representação desta condição será $\left|\varepsilon(\hat{x},\hat{y})\right|>\left|\varepsilon(\hat{x},\hat{y})+\Delta\varepsilon_{x-}(\hat{x})\right|$.

[viii] Verifica-se a condição do primeiro octante (tabela 6) e termina-se o desenho se não verificar, ou então continua-se o desenho em [iii].

Este algoritmo necessita de poucas operações aritméticas para o desenho de cada arco, não precisando de aritmética de vírgula flutuante, sendo portanto muito rápido.

Para determinar o arco que melhor aproxima a curva desenham-se arcos escolhendo sempre os dois pontos extremos da curva e variando o ponto intermédio. Para que a aproximação seja óptima, relativamente a um critério escolhido (ver mais à frente), este ponto poderia ser qualquer um da curva, mas para acelerar o processo escolheram-se como candidatos os pontos em que haja mudança de direcção (método também utilizado na aproximação poligonal). Penaliza-se o erro de aproximação relativamente à velocidade da mesma.

O critério de determinação do arco de aproximação, poderia basear-se, por exemplo, na distância do arco à linha, ou na área compreendida entre o arco e a linha, ou ainda na curvatura dos pontos da linha, entre outros parâmetros. Na aproximação poligonal de linhas concluiu-se que o método de rápida aproximação poligonal é aplicável de uma forma mais genérica, e que a aproximação poligonal obtida é visualmente melhor, se não houver uma escolha detalhada dos parâmetros. Assim sendo, por forma a manter alguma coerência entre os métodos de aproximação utilizados, baseou-se o critério de aproximação por arcos na área compreendida entre a linha e o arco de aproximação. É preciso notar que não se tenta aproximar uma curva por vários arcos em função de um critério, mas que se vai determinar o arco que melhor aproxime aquela curva, recorrendo ao critério de minimização da área entre o arco e a linha. É uma abordagem diferente que exige que as linhas sejam segmentados à partida, e que parte do princípio que em documentos de engenharia existem arcos que é necessário descrever, não interessando tanto a descrição aproximada das curvas por vários arcos. A segmentação de uma linha corresponde à divisão da linha em conjuntos de pixels da linha; cada conjunto será conexo, isto é, será constituído por pixels ligados, existindo sempre um caminho (formado por pixels do conjunto) entre quaisquer dois pixels. A área compreendida entre a linha e o arco de aproximação é determinada por contagem do número de pixels envolvidos pela linha e pelo arco.

Poderão perguntar-se, neste momento, como se efectua a segmentação da linha. Esta é efectuada com base no método descrito por Nagasamy e Langrana [24], com o qual se pretende dividir a linha em conjuntos de pontos, que formarão linhas aproximadamente rectilíneas, ou então fundamentalmente curvas. A descrição deste método foi apresentada na secção "reconhecimento de gráficos" no segundo capítulo. Nó entanto, neste trabalho recorre-se a um reconhecimento inicial de segmentos de recta, e em seguida aplica-se o método recorrendo aos extremos de segmentos detectados e não aos pontos críticos da linha.

Resumindo, o algoritmo implementado neste trabalho retorna um conjunto conexo de pontos, que representa uma linha, correspondente a um arco de circunferência. Este é o arco de aproximação de uma linha de pontos, determinado por minimização da área compreendida entre a linha e o arco. Neste algoritmo, portanto, não se aproxima uma linha com vários arcos de circunferência, aproxima-se a linha por um único arco de circunferência. Isto porque, em princípio não haverá interesse na representação de segmentos de linha por sucessivos arcos de circunferência, haverá sim, interesse na divisão das linhas em outras linhas, umas com pontos críticos de curvatura elevada que representarão extremos de segmentos de recta, e outras com pontos críticos de curvatura baixa, com uma variação aproximadamente contínua do declive, e que serão aproximados por arcos de circunferência.

Na figura 68 é ilustrada a aproximação de linhas por arcos de três pontos. É visível a aproximação quase exacta (coincidente com os pontos) das linhas recta e circular (na zona inferior da imagem). Na aproximação da recta a área entre a linha e o arco é zero, enquanto na aproximação da linha circular a área entre a aproximação e a linha é de 11 pixels. Na outra linha aproximadamente circular (na zona direita da imagem) a área entre o arco e a linha é de 70 pixels, correspondendo à área mínima de um arco de três pontos que aproxime esta linha. A relação entre a área e o número de pixels das linhas é uma medida uniformizada que pode ser comparada entre linhas diferentes (utilizada na rápida aproximação poligonal), que será para a recta 0, para a linha circular inferior 11/85=0.13, e para a linha circular superior 70/87=0.80. Esta medida é nitidamente superior para a aproximação da linha que, visualmente, menos se aproxima de um arco de circunferência, ou de um segmento de recta.

(a) (b)

Figura 68: Aproximação de linhas com arcos de circunferência. (a) Imagem original. (b) Aproximação das linhas da imagem original por arcos de circunferência representados num tom mais escuro.

A aplicação do método de segmentação de linhas e o método de aproximação de linhas com arcos de circunferência serão utilizados num método global de reconhecimento de primitivas gráficas, e serão melhor ilustrados na secção "método global" neste capítulo.

5.4 Aproximação de Linhas Utilizando Splines

Como já foi referido, a aproximação de linhas corresponde à determinação de uma descrição das linhas utilizando primitivas geométricas. O processo mais simples de aproximar uma linha recorre a um conjunto de segmentos de recta adjacentes, que pelo seu declive e comprimento consigam representar aproximadamente a forma da linha, isto é, a sua curvatura. Este é o processo de aproximação poligonal já referido neste capítulo. No entanto, não é adequado se quisermos descrever linhas fundamentalmente curvas, com alterações aproximadamente contínuas de declive. Neste caso a aproximação poligonal levaria a um grande número de segmentos de recta de pequena dimensão, perdendo-se a suavidade da curva, isto é, a sua continuidade. A suavidade duma curva pode ser descrita em função da sua continuidade [2, 39], ou mesmo em função de critérios de julgamento visual [39].

Para se conseguir uma aproximação mais fidedigna de linhas curvas, recorre-se a processos que utilizam primitivas gráficas mais complexas, como sejam arcos de circunferência (secção anterior), e funções cuja representação gráfica seja curva [3]. A representação duma linha curva, com diferentes pontos de inflexão e variações acentuadas de curvatura pode ser difícil de descrever recorrendo a uma única função, porque apresentam um peso computacional elevado. No entanto, qualquer curva pode ser descrita recorrendo a diferentes funções para descrição de diferentes partes da curva (segmentos de curva), podendo estas ser descritas por funções polinomiais, sendo mais usuais as de grau 3, já que polinómios de grau superior têm natureza oscilatória não sendo boas opções para a aproximação de linhas. Em [2] define-se a suavidade da curva de aproximação em função da passagem entre segmentos de curva. Se a curva nunca for quebrada, então existe continuidade de primeira ordem, isto é, segmentos de curva consecutivos encontram-se nos extremos. Por outro lado a continuidade de segunda ordem verifica-se quando no ponto de transição entre diferentes partes da curva o declive é igual, isto é, a derivada da função que aproxima os dois segmentos tem o mesmo valor naquele ponto. Se as curvas que aproximam dois segmentos tiverem a mesma curvatura no ponto de intersecção então existe continuidade de segunda ordem.

As funções polinomiais base, utilizadas habitualmente são definidas através polinómios de Lagrange [3], polinómios cúbicos de Hermite [3], polinómios de Bernstein [3], utilizados para construir curvas de Bézier [2], e polinómios B-Spline [2, 3].

Neste capítulo vão descrever-se: polinómios de Bernstein para construção de curvas de Bézier; polinómios B-Spline para a construção de curvas B-Spline; uma nova família de curvas spline denominadas spline angular ou A-Spline; um processo de detecção de cantos e representação de linhas recorrendo a B-Splines.

5.4.1 Definição de Curvas de Bézier

Estas curvas foram desenvolvidas pelo engenheiro francês Bézier para desenho de automóveis da Renault [2]. Neste método, um conjunto de pontos de uma linha é aproximado por uma curva que resulta da soma de funções polinomiais definidas por aqueles pontos, denominados pontos de controlo.

Sendo t um parâmetro, que se considera variar entre 0 e 1 excepto quando referido explicitamente, $x(t)$ e $y(t)$ as funções de variação das coordenadas horizontal e vertical, respectivamente, em função do parâmetro, então a função paramétrica duma curva é definida por:

$$\overline{P}(t) = \left(x(t), y(t)\right).$$

Considerando $n+1$ pontos de controlo $\overline{p}_k = (x_k, y_k)$, com $0 \le k \le n$, a curva de Bézier de aproximação dos pontos de controlo é definida [2, 3] por

$$\overline{P}(t) = \sum_{k=0}^{n} \overline{p}_k \cdot B_{k,n}(t),$$

sendo $B_{k,n}(t)$ o polinómio de Bernstein de grau n definido no intervalo $0 \le t \le 1$, cuja equação é:

$$B_{k,n}(t) = \frac{n!}{k!(n-k)!} \cdot t^k \cdot (1-t)^{n-k}.$$

A curva de Bézier tem algumas propriedades que a tornam útil na modelação de linhas curvas, nomeadamente:

1. os pontos extremos da curva de Bézier correspondem aos pontos extremos da linha poligonal definida pelos pontos de controlo;
2. a direcção da tangente à curva de Bézier nos pontos extremos coincide com a direcção do segmento de recta definido pelo ponto de controlo extremo e pelo ponto de controlo imediatamente anterior.

Estas propriedades fazem com que estas curvas sejam facilmente modeladas mantendo a sua suavidade. Consegue-se manter a continuidade de ordem zero fazendo coincidir os pontos de controlo extremos de dois segmentos de curva adjacentes. Mantém-se a continuidade de primeira ordem se o segmento de recta extremo de duas secções adjacentes coincidir.

Em seguida vão-se determinar vários polinómios de Bernstein, e ilustrar a sua utilização. As curvas de Bézier definidas por dois pontos de controlo ($n+1=2$) serão constituídas por polinómios de primeiro grau ($n=1$) com a seguinte definição:

$B_{0,1} = 1-t$; $\qquad B_{1,1} = t$.

Com o parâmetro t a variar entre 0 e 1, a curva de Bézier respectiva é ilustrada na figura 69a, e definida pela equação paramétrica:

$$\overline{P}(t)=\left(x_0(1-t)+x_1t, y_0(1-t)+y_1t\right)=\left((x_1-x_0)t+x_0,(y_1-y_0)t+y_0\right).$$

Se em vez de dois forem três os pontos de controlo $(n+1=3)$, as curvas de Bézier serão definidas por polinómios de segundo grau $(n=2)$ com a seguinte definição:

$B_{0,2}=(1-t)^2;\qquad B_{1,2}=2t(1-t);\qquad B_{1,2}=t^2$

Com o parâmetro t a variar entre 0 e 1, a curva de Bézier respectiva é ilustrada na figura 69b, e definida pela equação paramétrica:

$$\overline{P}(t)=\left((x_2-2x_1+3x_0)t^2+(2x_1-2x_0)t+x_0,(y_2-2y_1+3y_0)t^2+(2y_1-2y_0)t+y_0\right).$$

Para quatro pontos de controlo obtém-se polinómios de terceiro grau $(n=3)$ com as seguintes funções:

$B_{0,3}=(1-t)^3;\qquad B_{1,3}=3t(1-t)^2;\qquad B_{2,3}=3t^2(1-t);\quad B_{1,2}=t^3$

Com o parâmetro t a variar entre 0 e 1, a curva de Bézier respectiva é ilustrada na figura 69c, e definida pela equação paramétrica:

$$\overline{P}(t)=\begin{pmatrix}(x_3-3x_2+3x_1-x_0)t^3+(3x_2-6x_1+3x_0)t^2+(3x_1-3x_0)t+x_0\\(y_3-3y_2+3y_1-y_0)t^3+(3y_2-6y_1+3y_0)t^2+(3y_1-3y_0)t+y_0\end{pmatrix}.$$

Para cinco pontos de controlo obtêm-se polinómios de quarto grau $(n=4)$ com as seguintes funções:

$B_{0,4}=(1-t)^4;\qquad B_{1,4}=4t(1-t)^3;\qquad B_{2,4}=6t^2(1-t)^2;\quad B_{3,4}=4t^3(1-t);\qquad B_{4,4}=t^4$

Com o parâmetro t a variar entre 0 e 1, a curva de Bézier respectiva é ilustrada na figura 69d, e definida pela equação paramétrica:

$$\overline{P}(t)=\begin{pmatrix}(x_4-4x_3+6x_2-4x_1+x_0)t^4+(4x_3-12x_2+12x_1-4x_0)t^3+(6x_2-12x_1+6x_0)t^2+(4x_1-4x_0)t+x_0\\(y_4-4y_3+6y_2-4y_1+y_0)t^4+(4y_3-12y_2+12y_1-4y_0)t^3+(6y_2-12y_1+6y_0)t^2+(4y_1-4y_0)t+y_0\end{pmatrix}.$$

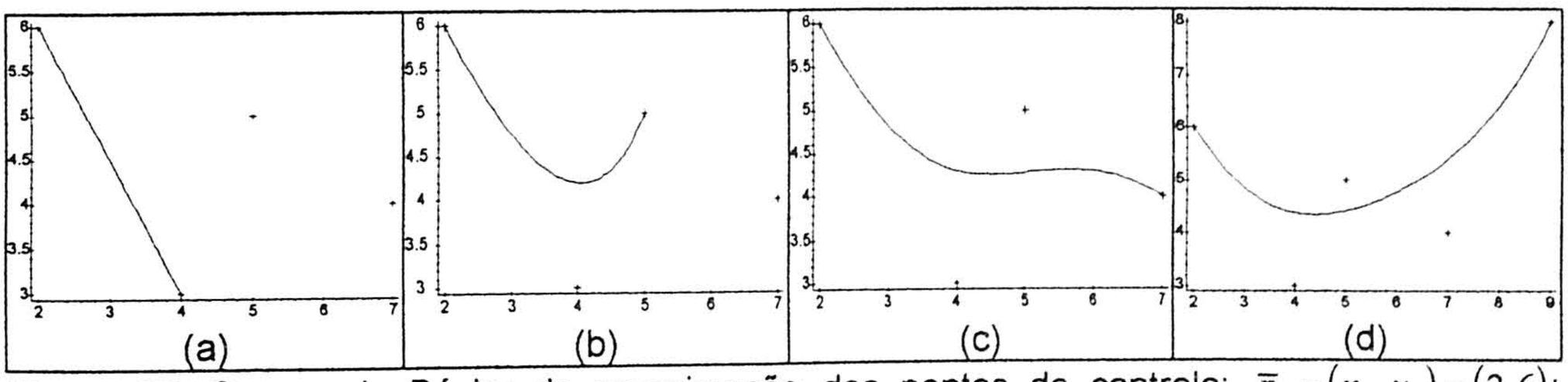


Figura 69: Curvas de Bézier de aproximação dos pontos de controlo: $\overline{p}_0=(x_0,y_0)=(2,6)$; $\overline{p}_1=(x_1,y_1)=(4,3)$; $\overline{p}_2=(x_2,y_2)=(5,5)$; $\overline{p}_3=(x_3,y_3)=(7,4)$; $\overline{p}_4=(x_4,y_4)=(9,8)$. (a) Curva de Bézier com dois pontos de controlo e função paramétrica $\overline{P}(t)=(2t+2,-3t+6)$. (b) Três pontos de controlo e função paramétrica $\overline{P}(t)=\left(-t^2+4t+2,5t^2-6t+6\right)$. (c) Quatro pontos de controlo e função paramétrica $\overline{P}(t)=\left(2t^3-2t^2+6t+2,-8t^3+15t^2-9t+6\right)$. (d) Cinco pontos de controlo e função paramétrica $\overline{P}(t)=\left(-3t^4+8t^3-6t^2+8t+2,+16t^4-32t^3+30t^2-12t+6\right)$.

Na figura 69(a-d) podem observar-se curvas de Bézier de aproximação de 2 a 5 pontos de controlo, e as respectivas funções paramétricas de cada curva. São curvas

suaves, sem oscilações erráticas, e que se situam sempre dentro do polígono fechado definido pelos pontos de controlo.

5.4.2 Definição de Curvas B-Spline

Hearn e Baker [2] fazem uma analogia entre uma curva spline e um fio flexível utilizado para produzir uma curva suave através de vários pontos de controlo. As curvas spline utilizam funções polinomiais para aproximar segmentos de curva, respeitando três ordens de continuidade.

Considerando $n+1$ pontos de controlo $\overline{p}_k = (x_k, y_k)$, com $0 \le k \le n$, a curva de B-Spline de aproximação dos pontos de controlo é definida [2] por:

$$\overline{P}(t) = \sum_{k=0}^{n} \overline{p}_k \cdot N_{k,m}(t).$$

Sendo $N_{k,m}(t)$ os polinómios B-Spline de grau $m-1$, variando o parâmetro entre 0 e $n-m+2$, e dividindo o intervalo de variação do parâmetro em $n+m$ intervalos, separados pelos pontos $t_0, t_1, \ldots, t_{n+m}$, assumindo $0/0 = 0$, a função polinomial B-Spline será definida recursivamente por:

$$N_{k,1}(t) = \begin{cases} 1, & t_k \le t < t_{k+1} \\ 0, & t < t_k \wedge t \ge t_{k+1} \end{cases}$$

$$N_{k,m}(t) = \frac{t - t_k}{t_{k+m-1} - t_k} N_{k,m-1}(t) + \frac{t_{k+m} - t}{t_{k+m} - t_{k+1}} N_{k+1,m-1}(t).$$

Os diferentes pontos $t_0, t_1, \ldots, t_{n+m}$ de separação dos $n+m$ intervalos de t são denominados pontos de quebra, e os correspondentes pontos da curva B-Spline denominam-se pontos nó. A curva B-Spline vai depender da definição dos pontos de quebra, que pode ser feita de diversas formas, nomeadamente:

- Escolhendo um espaçamento idêntico entre pontos de quebra [2]:
  $t_j = j \cdot \frac{n-t+2}{n+t}$, $0 \le j \le n+t$.
- Hearn e Baker [2] definem, com espaçamentos diferentes, os seguintes pontos de quebra com valores inteiros:

$$t_j = \begin{cases} 0, & j < m \\ j - m + 1, & m \le j \le n \\ n - m + 2, & j > n \end{cases}.$$

- Plastock e Kalley [3], com o parâmetro t a variar entre 0 e $n-m'+1$, definiram um conjunto de $(m'+1)+(n-m')+(m'+1) = n+m'+2$ pontos de quebra:

$$t_0, \ldots, t_{n+m'+1} = \left\{ \underbrace{0, \ldots, 0}_{m'+1}, 1, 2, \ldots, n-m', \underbrace{n-m'+1, \ldots, n-m'+1}_{m'+1} \right\},$$

que asseguram, por repetição dos pontos de quebra extremos, que os pontos extremos da spline coincidam com os pontos extremos da linha poligonal formada pelos pontos de controlo.

Embora em [3] se utilize uma definição diferente da apresentada em [2], que temos vindo a seguir, as duas definições poderão ser reduzidas à mesma forma. Plastock e Kalley [3] definem as funções polinomiais $N_{k,m'}(t)$ de grau m' com $n+1$ pontos de controlo $(\overline{p}_0,...,\overline{p}_n)$, e com o parâmetro t a variar entre 0 e $n-m'+1$. Definem também, um conjunto de $n+m'+2$ pontos de quebra. No entanto, ambas as definições são equivalentes, bastando para o verificar, considerar $m'=m-1$, passando as duas definições a ser exactamente iguais.

Resumindo, uma curva B-Spline é formada por funções polinomiais B-Spline, que representam polinómios que se assumem de grau $m-1$. Sendo $n+1$ o número de pontos de controlo, o parâmetro t varia entre 0 e $(n+1)-(m-1)=n-m+2$, definindo-se $n+m$ intervalos do parâmetro, separados pelos $n+m+1$ pontos de quebra. A estes pontos de quebra correspondem os nós da curva B-Spline.

As funções polinomiais B-Spline, por definição, são nulas em determinados intervalos, logo a alteração de um desses polinómios reflecte-se localmente na curva B-Spline, entre os parâmetros em que a curva não é nula. As funções polinomiais B-Spline, por definição, não aumentam de grau (relativo aos pontos de quebra), por aumento do número pontos de controlo, ou seja, o que aumenta são os polinómios necessários para definir a curva. Estas propriedades serão úteis na modelação interactiva de curvas.

Tal como com as curvas Bézier, a especificação de um maior número de pontos de controlo numa dada posição faz com que a curva se aproxime desse ponto. Uma curva B-Spline é desenhada dentro da envolvente convexa dos pontos de controlo, o que evita oscilações erráticas da curva.

As curvas B-Spline que recorrem a funções polinomiais quadráticas e cúbicas são as mais utilizadas, sendo definidas em seguida as funções quadráticas, com cinco pontos de controlo.

Escolhendo cinco pontos de controlo, $n=4$, e procurando polinómios B-Spline de segundo grau, $m=3$, definem-se $n+m=7$ intervalos, separados pelos pontos de quebra $t_0,...,t_7$, e com o parâmetro t a variar entre 0 e $n-m+2=3$.

Para o primeiro ponto de controlo, $k=0$.

$$N_{0,1}(t)=\begin{cases}1, & t_0 \le t < t_1\\ 0, & t \ge t_1\end{cases}$$

$$N_{0,2}(t)=\frac{t-t_0}{t_1-t_0}N_{0,1}+\frac{t_2-t}{t_2-t_1}N_{1,1}$$

$$N_{0,3}(t)=\frac{t-t_0}{t_2-t_0}N_{0,2}+\frac{t_3-t}{t_3-t_1}N_{1,2}$$

Conhece-se a equação de $N_{0,2}(t)$, e pode determinar-se a equação de $N_{1,2}(t)$.

$$N_{1,1}(t)=\begin{cases}1, & t_1\le t<t_2\\ 0, & t<t_1\wedge t\ge t_2\end{cases}$$

$$N_{1,2}(t)=\frac{t-t_1}{t_2-t_1}N_{1,1}+\frac{t_3-t}{t_3-t_2}N_{2,1}$$

Sendo,

$$N_{2,1}(t)=\begin{cases}1, & t_2\le t<t_3\\ 0, & t<t_2\wedge t\ge t_3\end{cases}$$

Obtém-se,

$$N_{0,3}(t)=\frac{(t-t_0)^2}{(t_2-t_0)(t_1-t_0)}N_{0,1}+\left(\frac{t-t_0}{t_2-t_0}\cdot\frac{t_2-t}{t_2-t_1}+\frac{t_3-t}{t_3-t_1}\cdot\frac{t-t_1}{t_2-t_1}\right)N_{1,1}+\frac{(t_3-t)^2}{(t_3-t_1)(t_3-t_2)}N_{2,1}.$$

De forma similar determinam-se as funções polinomiais relacionadas com os pontos de controlo seguintes.

$$N_{1,3}(t)=\frac{(t-t_1)^2}{(t_3-t_1)(t_2-t_1)}N_{1,1}+\left(\frac{t-t_1}{t_3-t_1}\cdot\frac{t_3-t}{t_3-t_2}+\frac{t_4-t}{t_4-t_2}\cdot\frac{t-t_2}{t_3-t_2}\right)N_{2,1}+\frac{(t_4-t)^2}{(t_4-t_2)(t_4-t_3)}N_{3,1}.$$

$$N_{2,3}(t)=\frac{(t-t_2)^2}{(t_4-t_2)(t_3-t_2)}N_{2,1}+\left(\frac{t-t_2}{t_4-t_2}\cdot\frac{t_4-t}{t_4-t_3}+\frac{t_5-t}{t_5-t_3}\cdot\frac{t-t_3}{t_4-t_3}\right)N_{3,1}+\frac{(t_5-t)^2}{(t_5-t_3)(t_5-t_4)}N_{4,1}.$$

$$N_{3,3}(t)=\frac{(t-t_3)^2}{(t_5-t_3)(t_4-t_3)}N_{3,1}+\left(\frac{t-t_3}{t_5-t_3}\cdot\frac{t_5-t}{t_5-t_4}+\frac{t_6-t}{t_6-t_4}\cdot\frac{t-t_4}{t_5-t_4}\right)N_{4,1}.$$

$$N_{4,3}(t)=\frac{(t-t_4)^2}{(t_6-t_4)(t_5-t_4)}N_{4,1}.$$

A especificação dos pontos de quebra permite determinar as expressões explícitas destas funções polinomiais. Assim, se for utilizada a segunda forma de especificação descrita acima, os pontos de quebra terão os valores apresentados na tabela 7.

Tabela 7: Pontos de quebra.

índice j	$m=3$	$n=4$	Pontos de Quebra
$j=0$	<	<	$t_0=0$
$j=1$	<	<	$t_1=0$
$j=2$	<	<	$t_2=0$
$j=3$	=	<	$t_3=j-m+1=1$
$j=4$	>	=	$t_4=j-m+1=2$
$j=5$	>	>	$t_5=n-m+2=3$
$j=6$	>	>	$t_6=n-m+2=3$
$j=7$	>	>	$t_7=n-m+2=3$

Assim, as funções polinomiais terão as seguintes equações.

$0\le t\le 3$

$$N_{0,3}(t)=\left(t^2-2t+1\right)N_{2,1}=\begin{cases}t^2-2t+1, & 0\le t<1\\ 0, & t\ge 1\end{cases}$$

$$N_{1,3}(t)=\left(-\frac{3}{2}t^2+2t\right)N_{2,1}+\left(\frac{1}{2}t^2-2t+2\right)N_{3,1}=\begin{cases}-\frac{3}{2}t^2+2t, & 0\le t<1\\ \frac{1}{2}t^2-2t+2, & 1\le t<2\\ 0, & t\ge 2\end{cases}$$

$$N_{2,3}(t)=\left(\frac{1}{2}t^2\right)N_{2,1}+\left(-t^2+3t-\frac{3}{2}\right)N_{3,1}+\left(\frac{1}{2}t^2-3t+\frac{9}{2}\right)N_{4,1}=\begin{cases}\frac{1}{2}t^2, & 0\le t<1\\ -t^2+3t-\frac{3}{2}, & 1\le t<2\\ \frac{1}{2}t^2-3t+\frac{9}{2}, & 2\le t<3\\ 0, & t\ge 3\end{cases}$$

$$N_{3,3}(t)=\left(\frac{1}{2}t^2-t+\frac{1}{2}\right)N_{3,1}+\left(-\frac{3}{2}t^2+7t-\frac{15}{2}\right)N_{4,1}=\begin{cases}\frac{1}{2}t^2-t+\frac{1}{2}, & 1\le t<2\\ -\frac{3}{2}t^2+7t-\frac{15}{2}, & 2\le t<3\\ 0, & t<1\wedge t\ge 3\end{cases}$$

$$N_{4,3}(t)=\left(t^2-4t+4\right)N_{4,1}=\begin{cases}t^2-4t+4, & 2\le t<3\\ 0, & t<2\wedge t\ge 3\end{cases}$$

Estas funções, como se pode verificar, só estão definidas em determinados intervalos, respeitando a propriedade local referida anteriormente. A ilustração das mesmas é apresentada na figura 70.

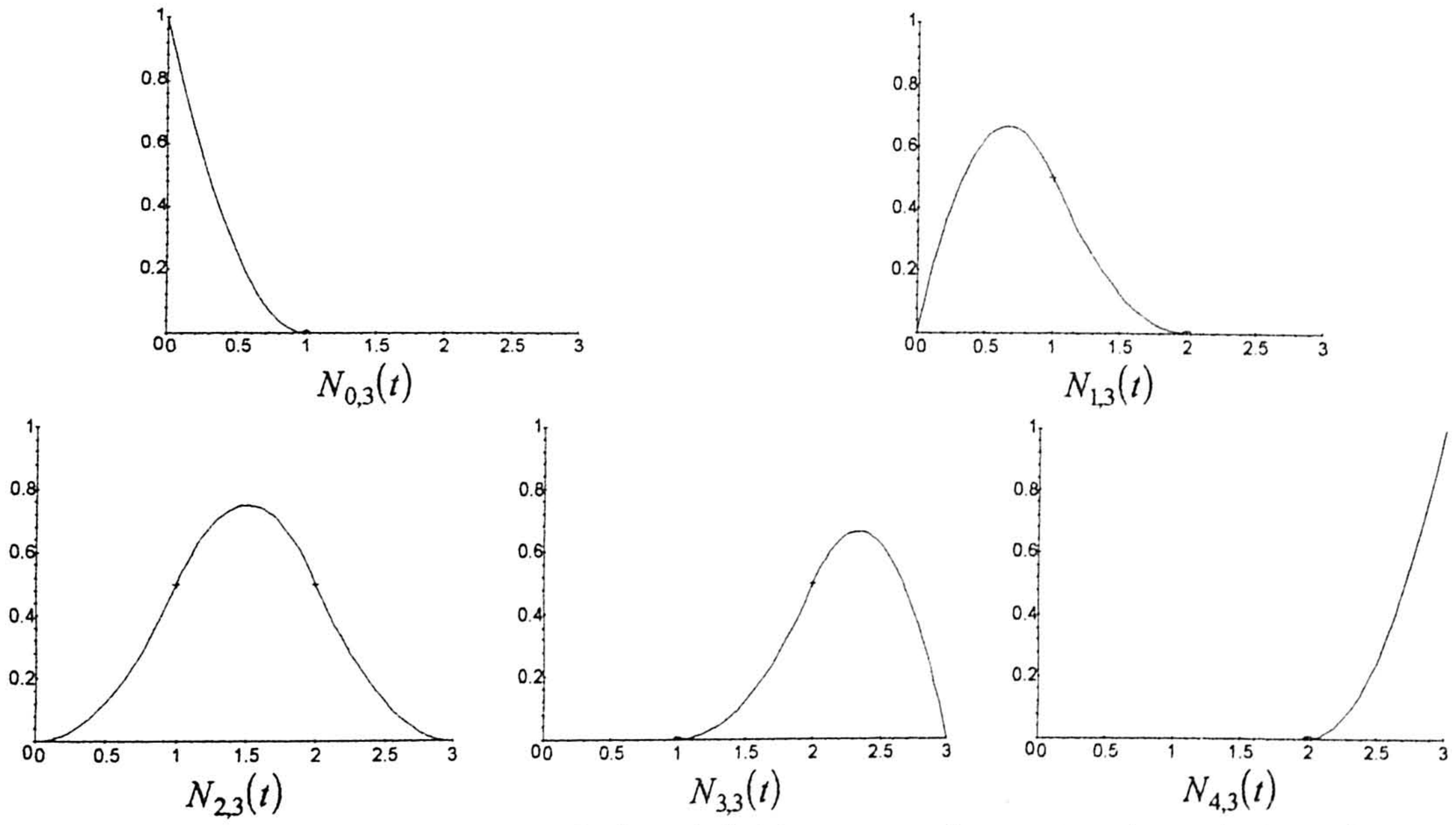


Figura 70: Polinómios quadráticos B-Spline definidos por quatro pontos de controlo, e 8 pontos de quebra.

Em seguida vão determinar-se os polinómios da curva B-Spline de aproximação de cinco pontos de controlo $\overline{p}_k=(x_k,y_k)$, com $0\le k\le n=4$. Sejam $\overline{p}_0=(2,6)$, $\overline{p}_1=(4,3)$, $\overline{p}_2=(5,5)$,

$\overline{p}_3 = (7,4)$ e $\overline{p}_4 = (9,8)$ os pontos de controlo, a curva B-Spline $\overline{P}(t) = (x(t), y(t)) = \sum_{k=0}^{n} \overline{p}_k \cdot N_{k,m}(t)$ será definida por:

$$x(t) = x_0 \cdot N_{0,3} + x_1 \cdot N_{1,3} + x_2 \cdot N_{2,3} + x_3 \cdot N_{3,3} + x_4 \cdot N_{4,3} = \begin{cases} -\frac{3}{2}t^2 + 4t + 2, & 0 \le t < 1 \\ \frac{1}{2}t^2 + 4, & 1 \le t < 2 \\ t^2 - 2t + 6, & 2 \le t \le 3 \\ 0, & t < 0 \wedge t > 3 \end{cases}, \text{ e}$$

$$y(t) = y_0 \cdot N_{0,3} + y_1 \cdot N_{1,3} + y_2 \cdot N_{2,3} + y_3 \cdot N_{3,3} + y_4 \cdot N_{4,3} = \begin{cases} 4t^2 - 6t + 6, & 0 \le t < 1 \\ -\frac{3}{2}t^2 + 5t + \frac{1}{2}, & 1 \le t < 2 \\ \frac{9}{2}t^2 - 19t + \frac{49}{2}, & 2 \le t \le 3 \\ 0, & t < 0 \wedge t > 3 \end{cases}.$$

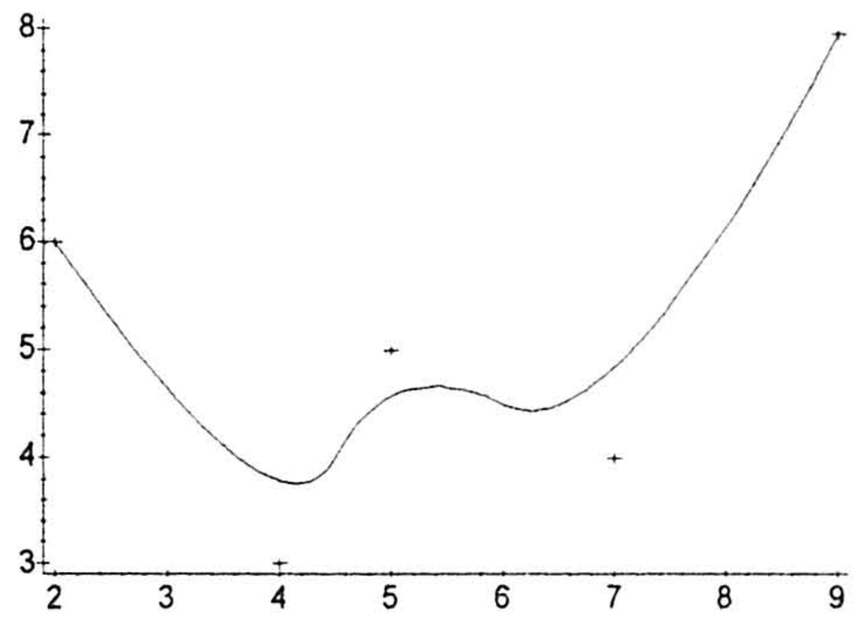


Figura 71: Curva B-Spline de aproximação de cinco pontos de controlo.

Na figura 71 pode observar-se a curva B-Spline, definida pelas equações anteriores, a aproximar um conjunto de cinco pontos de controlo. Esta curva tal como a de Bézier é suave, mantendo-se definida dentro da envolvente convexa determinada pelos pontos de controlo, e os seus extremos coincidem com os pontos de controlo extremos, notando-se no entanto uma grande diferença relativamente à curva de Bézier, já que se aproxima mais dos mesmos pontos de controlo, tendo valores de curvatura muito mais acentuados. Esta aproximação aos pontos de controlo, com curvaturas mais acentuadas em determinadas zonas, advém da definição da curva com vários polinómios, cada um nulo num intervalo do parâmetro da curva.

5.4.3 Definição de Splines Angulares

Uma outra família de curvas spline foi definida por Alia Et Al. [39] para interpolação de pontos de controlo, denominando-se splines angulares ou simplesmente A-Spline. Permitem construir uma curva entre dois pontos consecutivos baseada em quatro pontos, isto é, baseada nos dois pontos a interpolar, e nos dois pontos imediatamente anterior e posterior. A spline angular de interpolação de vários pontos é construída por

conjugação de vários segmentos A-Spline entre pares de pontos. As splines angulares gozam de diversas propriedades, como sejam: flexibilidade, por gerarem um grande número de formas; todos os seus parâmetros são determinados apenas com base nos pontos de interpolação, não sendo necessário definir mais nenhuma variável; as curvas geradas são suaves havendo continuidade, de ordem zero por segmentos contíguos se tocarem, e de primeira ordem porque o declive na junção de segmentos é igual; a alteração de um ponto de controlo apenas altera a curva localmente, o que facilita o seu controlo interactivo; a forma da curva não é alterada por transformações geométricas (deslocamento, rotação, mudança de escala); gera um arco de circunferência se os pontos de controlo pertencerem a uma circunferência.

A definição de splines angulares é feita recorrendo a considerações geométricas, que serão facilmente compreendidas se acompanhadas pela ilustração da figura 72.

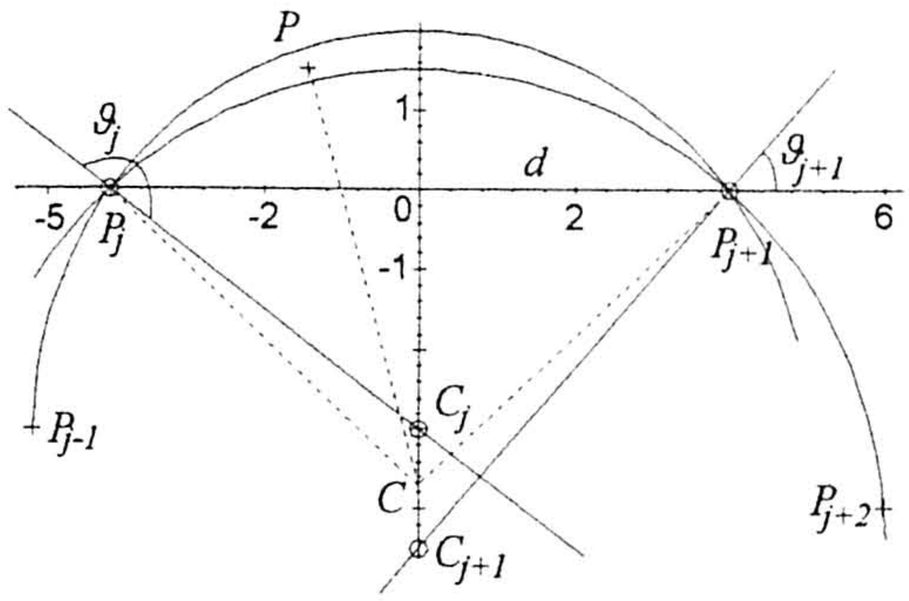


Figura 72: Definição de splines angulares.

Dados 4 pontos consecutivos a interpolar P_{j-1}, P_j, P_{j+1} e P_{j+2} no plano xy, os centros das duas circunferências definidas por $\{P_{j-1},\ P_j,\ P_{j+1}\}$ e por $\{P_j,\ P_{j+1},\ P_{j+2}\}$ serão respectivamente C_j e C_{j+1}. Seja C um ponto do segmento de recta $[C_j,\ C_{j+1}]$, variando a sua posição entre os extremos deste segmento. A A-Spline corresponde ao conjunto de pontos P, que em conjunto com os pontos C definem segmentos de recta, que nos seus limites coincidem com os segmentos definidos pelos pontos extremos de interpolação e respectivos centros de circunferência, isto é, coincidem com os segmentos $\left[C_j,P_j\right]$ e $\left[C_{j+1},P_{j+1}\right]$. O comprimento dos segmentos $[C,P]$ é $\left|C,P\right|=\left|C,P_j\right|=\left|C,P_{j+1}\right|$.

É uma curva compreendida entre os arcos de circunferência $\{P_{j-1},\ P_j,\ P_{j+1}\}$ e $\{P_j,\ P_{j+1},\ P_{j+2}\}$, que corresponderia a um arco de circunferência se C fosse fixo. No entanto ao mesmo tempo que C varia de C_j para C_{j+1}, o declive do segmento definido por C e P vai variar, o que determinará uma curva entre os pontos P_j e P_{j+1}.

A definição paramétrica da curva $P(\alpha)=\left(x(\alpha),y(\alpha)\right)$, considerando um sistema de coordenadas com a origem no ponto central do segmento entre P_j e P_{j+1}, sendo metade do comprimento deste segmento definido por d, e com P_{j-1} e P_{j+2} abaixo do eixo vertical (figura 72) será:

$$x(\alpha) = \rho(\alpha)\cos(\alpha),$$
$$y(\alpha) = \rho(\alpha)sen(\alpha) + c(\alpha),$$

sendo α o parâmetro da curva, correspondendo ao ângulo de variação do segmento de recta $[C,P]$, $\rho(\alpha) = \overline{CP} = \overline{CP_j} = \sqrt{d^2 + c^2(\alpha)}$ o comprimento do segmento, e $c(\alpha)$ uma função que permite determinar C em função do ângulo α variando entre $\left[yc_j, yc_{j+1}\right]$. O parâmetro α varia, por definição, entre $\left[\vartheta_j, \vartheta_{j+1}\right]$, que são os declives dos segmentos extremos $\left[C_j, P_j\right]$ e $\left[C_{j+1}, P_{j+1}\right]$.

A consideração geométrica de que a função de correspondência entre as direcções de varrimento da A-Spline e o ponto C é igual a $c(\alpha) = -d\tan\alpha'$, definindo a variável $\alpha' = \frac{\pi - \vartheta_j - \vartheta_{j+1}}{\vartheta_j - \vartheta_{j+1}} \cdot \alpha + \frac{2\vartheta_j\vartheta_{j+1} - \pi\vartheta_{j+1}}{\vartheta_j - \vartheta_{j+1}}$, permite determinar,

$$\rho(\alpha) = \sqrt{d^2 + c^2(\alpha)} = d\sqrt{1 + \tan^2\alpha'} = {}^{d}\!/_{\cos\alpha'},$$

e assim definir as equações paramétricas da spline angular, com α variando no intervalo $\left[\vartheta_j, \vartheta_{j+1}\right]$, da seguinte forma:

$$x(\alpha) = d \cdot {}^{\cos\alpha}\!/_{\cos\alpha'},$$
$$y(\alpha) = d \cdot {}^{(\sin\alpha - \sin\alpha')}\!/_{\cos\alpha'}.$$

Considerando os pontos $P_{j-1} = (-5,-3)$, $P_j = (-4,0)$, $P_{j+1} = (4,0)$ e $P_{j+2} = (6,-4)$, os centros das circunferências definidas por $\left\{P_{j-1}, P_j, P_{j+1}\right\}$ e $\left\{P_j, P_{j+1}, P_{j+2}\right\}$ são respectivamente $C_j = (0,-3)$ e $C_{j+1} = (0,4.5)$, metade da distância entre P_j e P_{j+1} será $d = 4$, os ângulos inicial e final do parâmetro α serão $\vartheta_j = 2.5$ e $\vartheta_j = 0.84$. Assim sendo, a equação paramétrica da spline angular será:

$$P(\alpha) = (x(\alpha), y(\alpha)) = \left(4\frac{\cos\alpha}{\cos(-0.11952\alpha + 0.9404)}, 4\frac{\sin\alpha - \sin(-0.11952\alpha + 0.9404)}{\cos(-0.11952\alpha + 0.9404)}\right).$$

Na figura 73 ilustra-se o desenho da spline angular definida por esta equação, podendo ver-se claramente que a curva está compreendida entre as duas circunferências descritas na definição, de centros $C_j = (0,-3)$ e $C_{j+1} = (0,4.5)$.

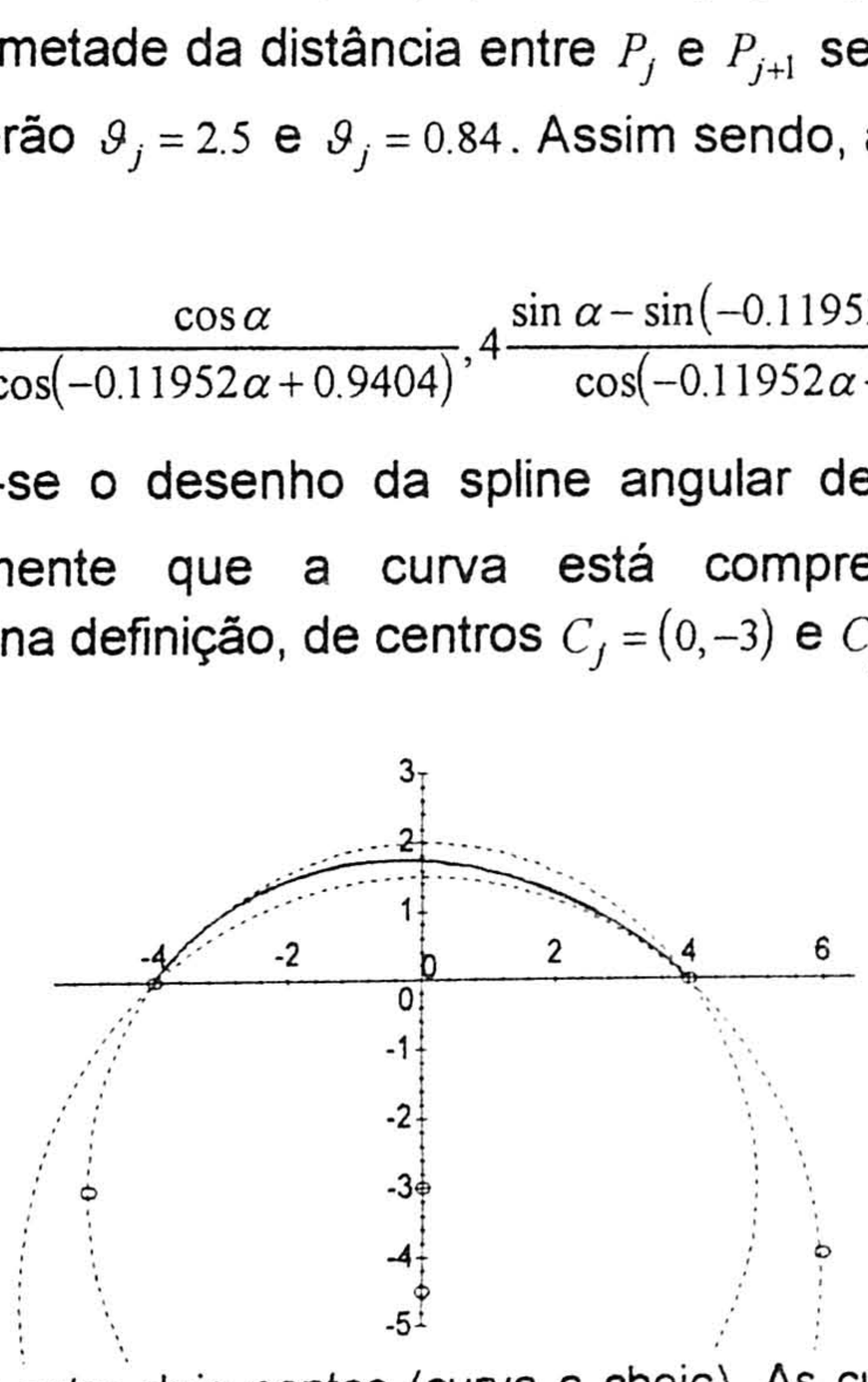


Figura 73: Spline angular entre dois pontos (curva a cheio). As curvas a traço interrompido representam as circunferências descritas na definição.

5.4.4 Detecção de Cantos Recorrendo a Splines Cúbicas

Depois de verificada a utilidade das splines para aproximar curvas, por serem suaves, flexíveis, e com bom controlo interactivo, vai ser ilustrada a aplicação destas curvas na detecção de cantos e representação de linhas. A utilização de B-Splines, aproximando curvas digitais, para detectar cantos e depois para representar essas curvas entre os cantos foi proposta por Medioni e Yasumoto [49]. Neste trabalho foi implementado este método de detecção de cantos.

A representação de curvas digitais por B-Splines, permite efectuar uma suavização dessas curvas, eliminando efeitos da quantização na determinação da curvatura dos pontos, sendo a curvatura definida como a variação do declive dessa curva.

Seja S uma curva definida na sua forma paramétrica, se t for o parâmetro, e se a variação de x for determinada por uma função $f(x)$ e a variação de y por uma função $g(x)$, a sua equação será:

$S = \left(f(t), g(t)\right)$.

O declive duma curva num ponto A em $t = t_1$ será:

$$\left[\frac{dy}{dx}\right]_{t=t_1} = \left[\frac{dg/dt}{df/dt}\right]_{t=t_1}.$$

A curvatura num ponto é definida como sendo a derivada do declive sobre o comprimento do arco da curva:

$$C_v(t) = \frac{\dfrac{d^2y}{dx^2}}{\left(1+\left(\dfrac{dy}{dx}\right)^2\right)^{3/2}} = \frac{\dfrac{df}{dt}\cdot\dfrac{d^2g}{dt^2} - \dfrac{dg}{dt}\cdot\dfrac{d^2f}{dt^2}}{\left(\left(\dfrac{df}{dt}\right)^2+\left(\dfrac{dg}{dt}\right)^2\right)^{3/2}}.$$

Se considerarmos a aproximação duma curva entre dois pontos A e B feita pela função polinomial $x = f(t) = a_1t^3 + b_1t^2 + c_1t + d_1$ e $y = g(t) = a_2t^3 + b_2t^2 + c_2t + d_2$, em que o parâmetro t varia entre 0 (A) e 1 (B), a curvatura no ponto $A(t=0)$ será:

$$C_v(0) = \frac{\left(3a_1t^2+2b_1t+c_1\right)\cdot\left(6a_2t+2b_2\right) - \left(3a_2t^2+2b_2t+c_2\right)\cdot\left(6a_1t+2b_1\right)}{\left(\left(3a_1t^2+2b_1t+c_1\right)^2+\left(3a_2t^2+2b_2t+c_2\right)^2\right)^{3/2}}\Bigg|_{t=0} = 2\frac{c_1\cdot b_2 - c_2\cdot b_1}{\left(c_1{}^2+c_2{}^2\right)^{3/2}}.$$

Foi definida a curvatura de um polinómio cúbico que é a representação polinomial de menor ordem que mantém a continuidade de ordem zero e de primeira ordem, entre segmentos de curva. Assim ao recorrer-se a B-Spline cúbicas para aproximar curvas já se tem a forma de detectar cantos.

Sendo a equação paramétrica da B-Spline cúbica $\overline{P}(t) = \left(x(t), y(t)\right)$, de aproximação de quatro pontos de controlo, definida por pontos de quebra igualmente espaçados:

$$x(t)=\tfrac{1}{6}\left(-x_{j-1}+3x_j-3x_{j+1}+x_{j+2}\right)t^3+\tfrac{1}{2}\left(x_{j-1}-2x_j+x_{j+1}\right)t^2+\tfrac{1}{2}\left(-x_{j-1}+x_{j+1}\right)t+\tfrac{1}{6}\left(x_{j-1}+4x_j+x_{j+1}\right),$$
$$y(t)=\tfrac{1}{6}\left(-y_{j-1}+3y_j-3y_{j+1}+y_{j+2}\right)t^3+\tfrac{1}{2}\left(y_{j-1}-2y_j+y_{j+1}\right)t^2+\tfrac{1}{2}\left(-y_{j-1}+y_{j+1}\right)t+\tfrac{1}{6}\left(y_{j-1}+4y_j+y_{j+1}\right).$$

Então a curvatura duma curva B-Spline cúbica no ponto $x_j(t=0)$ será determinada pela equação:

$$C = \frac{\left(x_{+1}-x_{-1}\right)\left(y_{-1}-y+y_{+1}\right)-\left(y_{+1}-y_{-1}\right)\left(x_{-1}-x+x_{+1}\right)}{\left(x_{+1}-x_{-1}\right)^2+\left(y_{+1}-y_{-1}\right)^2}.$$

Para detectar cantos duma curva é necessário detectar pontos dessa curva de elevada curvatura. Neste método, para se evitar a quantificação que advém do facto de avaliar curvas digitais, recorreu-se a uma suavização da curva, aproximando-a por uma B-Spline. Para um dado ponto $P_j=\left(x_j,y_j\right)$ da curva digital existe um ponto $(t=0)$ da B-Spline que lhe corresponde, sendo possível determinar a distância $(\delta=\left(\delta_x,\delta_y\right)_{t=0}=\left(d_1-x_j,d_2-y_j\right)=\tfrac{1}{6}P_{j-1}-\tfrac{1}{3}P_j+\tfrac{1}{6}P_{j+1})$ entre estes pontos. Para estabelecer uma medida de canto para um dado ponto, vai-se recalcular a curvatura da B-Spline de aproximação dos pontos deslocados, podendo determinar-se uma nova distância entre os pontos das duas B-Spline. Ou seja, vai-se utilizar os pontos $P_j+\delta$ para obter uma nova B-Spline, com uma dada curvatura C_v', e um novo deslocamento δ'. Os novos pontos de interpolação da B-Spline serão:

$$P_j'=P_j+\delta=P_j+\tfrac{1}{6}P_{j-1}-\tfrac{1}{3}P_j+\tfrac{1}{6}P_{j+1}=\tfrac{1}{6}P_{j-1}+\tfrac{2}{3}P_j+\tfrac{1}{6}P_{j+1};$$
$$P_{j-1}'=P_{j-1}+\delta\Big|_{P_{j-1}}=P_{j-1}+\tfrac{1}{6}P_{j-2}-\tfrac{1}{3}P_{j-1}+\tfrac{1}{6}P_j=\tfrac{1}{6}P_{j-2}+\tfrac{2}{3}P_{j-1}+\tfrac{1}{6}P_j;$$
$$P_{j+1}'=P_{j+1}+\delta\Big|_{P_{j+1}}=P_{j+1}+\tfrac{1}{6}P_j-\tfrac{1}{3}P_{j+1}+\tfrac{1}{6}P_{j+2}=\tfrac{1}{6}P_j+\tfrac{2}{3}P_{j+1}+\tfrac{1}{6}P_{j+2}.$$

Substituindo x por y, b_2' será igual a b_1', c_2' igual a c_1', e d_2' igual a d_1', obtendo-se os seguintes parâmetros do polinómio B-Spline de aproximação:

$$b_1'=b_1\Big|_{x_j=x_j';x_{j-1}=x_{j-1}';x_{j+1}=x_{j+1}'}=\tfrac{1}{2}\left(x_{j-1}'-2x_j'+x_{j+1}'\right)=\tfrac{1}{12}x_{j-2}+\tfrac{1}{6}x_{j-1}-\tfrac{1}{2}x_j+\tfrac{1}{6}x_{j+1}+\tfrac{1}{12}x_{j+2};$$
$$c_1'=c_1\Big|_{x_{j-1}=x_{j-1}';x_{j+1}=x_{j+1}'}=\tfrac{1}{2}\left(x_{j+1}'-x_{j-1}'\right)=\tfrac{1}{3}\left(x_{j+1}-x_{j-1}\right)+\tfrac{1}{12}\left(x_{j+2}-x_{j-2}\right);$$
$$d_1'=d_1\Big|_{x_j=x_j';x_{j-1}=x_{j-1}';x_{j+1}=x_{j+1}'}=x(t)=\tfrac{1}{6}\left(x_{j-1}'+4x_j'+x_{j-1}'\right)=\tfrac{1}{36}x_{j-2}+\tfrac{2}{9}x_{j-1}+\tfrac{1}{2}x_j+\tfrac{2}{9}x_{j+1}+\tfrac{1}{36}x_{j+2}.$$

E a curvatura será: $C_v'=2\left(c_1'\cdot b_2'-c_2'\cdot b_1'\right)\Big/\left(c_1'^2+c_2'^2\right)^{3/2}$.

Um segundo deslocamento para a nova B-Spline será:

$$\delta'=\left(\delta_x',\delta_y'\right)_{t=0}=\left(d_1'-x_j',d_2'-y_j'\right)=\tfrac{1}{36}P_{j-2}+\tfrac{1}{18}P_{j-1}-\tfrac{1}{6}P_j+\tfrac{1}{18}P_{j+1}+\tfrac{1}{36}P_{j+2}.$$

E finalmente o deslocamento total δ_t, desde o ponto da curva digital P_j até ao ponto correspondente da segunda curva B-Spline será:

$$\delta_t=\delta+\delta'=\left(\delta_{tx},\delta_{ty}\right)=\tfrac{1}{36}P_{j-2}+\tfrac{2}{9}P_{j-1}-\tfrac{1}{2}P_j+\tfrac{2}{9}P_{j+1}+\tfrac{1}{36}P_{j+2}.$$

Neste momento estão determinadas as formas de cálculo de todas as variáveis necessárias para detectar cantos, recorrendo a este método. Como já foi referido, convém efectuar uma suavização das curvas digitais ao determinar a curvatura dos pontos, para evitar efeitos relativos à quantificação das coordenadas, logo das curvaturas. Neste método utiliza-se para suavização, duas aproximações sucessivas por curvas B-Spline, obtendo resultados menos pronunciados de curvatura nesta descrição, e mais pronunciados na distância (δ_t) desta curva para a curva digital. Em função de toda esta descrição, estabeleceu-se o seguinte critério de determinação de cantos:

- A distância total da curva digital à segunda B-Spline de aproximação deve ser maior do que um dado limiar d_c, isto é, $\delta_t > d_c$; critério que advém da utilização de uma curva de suavização e da subsequente utilização dos cantos para descrição da curva digital utilizando uma B-Spline.
- A curvatura da B-Spline de aproximação deve ser superior (inerente à definição de canto) a um dado limiar c_c, isto é, $C'_v > c_c$.
- A curvatura (C'_v) da B-Spline de aproximação deve ser um máximo local, para evitar cantos correspondentes a pontos vizinhos.

É importante notar, no entanto, que uma lista de cantos não é suficiente para representar a forma duma curva digital, já que curvas suaves fechadas com pequena variação da curvatura poderão não ser representadas. Assim sendo, é necessário obter mais pontos que permitam guiar a curva de aproximação, que se pretende que seja uma B-Spline. Ou seja, é necessário estabelecer um outro limiar de curvatura (c_t) que permita detectar entre cantos, pontos de curvatura significativa (pontos SCP) que servirão de pontos de controlo para a B-Spline de aproximação. O significado disto, é que se torna necessário estabelecer um quarto critério que permita detectar pontos SCP:

- A curvatura da B-Spline de aproximação deve ser superior a um dado limiar $c_t < c_c$, isto é, $c_t < C'_v \leq c_c$.

Na figura 74 é representada uma B-Spline cúbica de aproximação de uma curva digital, representada por vários pontos de controlo. Os pontos de controlo têm as coordenadas (0,0), (1,1), (2,3), (3,6), (4,4), (5,5), (6,6), (7,1), (8,0) e (9,1), podendo-se verificar que é uma aproximação suave em que se mantém a continuidade de posição e de declive entre os vários segmentos da curva, representados sucessivamente por traço cheio e interrompido.

5.5 Armazenamento da Informação Vectorial para Entrada num Sistema CAD

Para armazenar a informação vectorial detectada anteriormente, é necessário ter um formato de um ficheiro que sirva de entrada para um sistema CAD. Poderia recorrer-se a um formato que fosse considerado normalizado, e utilizado por diversos sistemas CAD, ou então, encontrar um sistema CAD utilizado maioritariamente e escolher um formato que este sistema aceitasse. Um formato que tenta ser normalizado é o formato IGES[3] [24]. Por outro lado, um sistema ao qual recorrem a maioria dos utilizadores do meio CAD, que trabalham com desenho técnico, é o programa AutoCAD da AutoDesk.

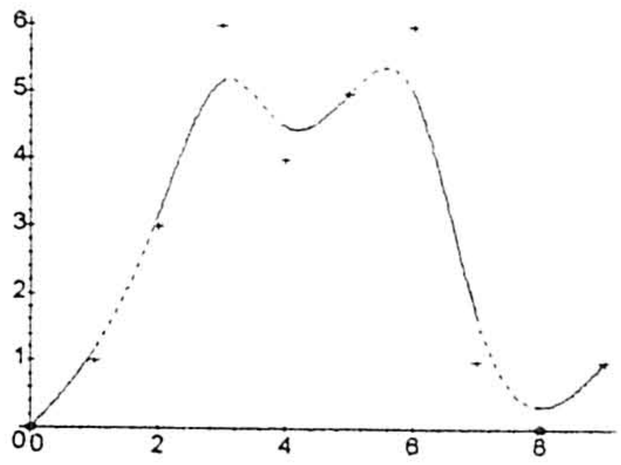


Figura 74: Curva B-Spline cúbica de aproximação de uma curva digital representada por vários pontos de controlo.

O AutoCAD tem vários formatos de ficheiros de saída e entrada [68], entre os quais os formatos de ficheiros de intercâmbio de desenhos em modo texto DXF e IGES. O formato DXF é bastante popular, sendo utilizado em aplicações de desenho vectorial, nomeadamente em ambiente Windows. Assim, por ser um formato bastante utilizado por outras aplicações, por ser menos complexo e não trazer nenhuma limitação quanto às entidades permitidas relativamente ao formato IGES, e por permitir atingir os objectivos propostos nesta tese, este foi o formato escolhido.

A estrutura genérica deste formato é composta por 5 secções principais. Secção de Cabeçalho, descrevendo características genéricas do desenho, como por exemplo a versão do AutoCAD para o qual foi feito o desenho, data de criação do desenho, dimensões do papel e dimensões dos desenho no sistema de coordenadas do mundo. Secção de Tabelas, que permitem definir por exemplo o tipo de linhas utilizado para desenhar, os diferentes níveis de desenho, o estilo associado ao texto, a vista do desenho, o sistema de coordenadas, a configuração da janela de vista, o estilo associado às dimensões do desenho (como a escala, ou a dimensão do texto), e a identificação da aplicação que alterou o desenho. Secção de Blocos, com a definição de entidades bloco, que permitem definir bibliotecas de símbolos associados aos desenhos

[3]Do inglês "Initial Graphics Exchange Specification"

técnicos. Secção de Entidades, que é a secção fundamental deste formato, que contém a descrição das entidades que formam o desenho, como sejam os arcos de circunferência, os círculos, as rectas, os pontos, ou o texto, entre outras. Finalmente, um ficheiro neste formato tem que terminar com a secção de fim do ficheiro.

A única secção obrigatória num ficheiro em formato DXF é a secção de entidades, podendo a informação referente a todas as outras secções ser definida por defeito, pelo programa que ler o ficheiro. Foi esta a abordagem adoptada no desenvolvimento do algoritmo de exportação da descrição das curvas.

As entidades que podem ser exportadas pelo programa de processamento e análise de imagem são: Point, Line, Circle e Arc. Todas as entidades têm como primeiro parâmetro o nível em que são desenhadas. A entidade Point é descrita pelas suas 3 coordenadas num eixo tridimensional, e pelo ângulo do eixo dos xx quando o ponto foi desenhado. A entidade Line é descrita pelos seus 2 pontos extremos, isto é, pelas 6 coordenadas num eixo tridimensional que definem esses pontos extremos. A entidade Circle é descrita pelo seu centro e raio, isto é, pelas coordenadas num eixo tridimensional que definem esse ponto e pelo valor do raio. A entidade Arc é descrita pelo seu centro, pelo raio, e pelos ângulos inicial e final, isto é, pelas coordenadas num eixo tridimensional que definem o centro, pelo valor do raio, e pelos valores dos ângulos inicial e final, medidos no sentido contrário ao do movimento dos ponteiros do relógio. Existe um conjunto de variáveis, opcionais, comuns a todas as entidades, que são o tipo de linha de desenho da entidade, a elevação (característica tridimensional), a espessura da linha, e a cor.

A exportação das rectas e dos pontos é evidente porque a informação com que elas são descritas está disponível como resultado do reconhecimento gráfico. A exportação de arcos de circunferência e de círculos exige a transformação da informação que caracteriza estas entidades sobre a forma de três pontos, para a informação exigida pelo formato DXF.

Se uma circunferência estiver definida por três pontos (x_j, y_j), então as coordenadas horizontal x_c e vertical y_c do seu centro e o raio r podem ser determinados pelo seguinte sistema de três equações a três incógnitas:

$$\begin{cases} (x_1 - x_c)^2 + (y_1 - y_c)^2 = r^2 \\ (x_2 - x_c)^2 + (y_2 - y_c)^2 = r^2 \\ (x_2 - x_c)^2 + (y_2 - y_c)^2 = r^2 \end{cases},$$

obtendo-se os seguintes resultados:

$$y_c = \frac{1}{2} \frac{x_1^2 x_2 - x_1^2 x_3 + x_2^2 x_3 - x_2^2 x_1 + x_3^2 x_1 - x_3^2 x_2 + y_1^2 y_2 - y_1^2 y_3 + y_2^2 y_3 - y_2^2 y_1 + y_3^2 y_1 - y_3^2 y_2}{x_1 y_3 - x_1 y_2 + x_2 y_1 - x_2 y_3 + x_3 y_2 - x_3 y_1};$$

$$x_c = \frac{x_1^2 - x_2^2 + y_1^2 - y_2^2 - 2y_1y_c + 2y_2y_c}{2x_1 - 2x_2};$$

$$r^2 = x_1 - x_c^2 + y_1 - y_c^2.$$

Além de determinar estes parâmetros da circunferência, é necessário determinar os ângulos inicial e final, ao exportar arcos de circunferência. Sendo $\Delta x_j = x_j - x_c$ e $\Delta y_j = y_j - y_c$, o declive da recta definida pelo centro e pelo ponto P_j é determinado por $\tan^{-1}(\Delta y_j / \Delta x_j)$, compreendido no intervalo $]-\frac{\pi}{2}, \frac{\pi}{2}[$. Pode determinar-se o quadrante em que se encontra o ponto P_j relativamente ao centro da circunferência por avaliação do sinal das distâncias Δx_j e Δy_j, da forma indicada na tabela 8.

Tabela 8: Quadrantes duma circunferência.

Quadrante	Δx_j	Δy_j
1°	>0	>0
2°	<0	>0
3°	<0	<0
4°	>0	<0

Para avaliar o quadrante em que se encontra o ponto, de forma a classificar correctamente os ângulos inicial e final, tem que se tomar em consideração, o movimento positivo dos mesmos, atribuindo ao ângulo final um valor superior ao inicial, no sentido contrário ao dos ponteiros do relógio.

É preciso notar que as coordenadas de todos os pontos estão armazenadas no programa de processamento de imagem, considerando um sistema de coordenadas cuja origem se encontra no canto superior esquerdo, e cuja unidade é o pixel de imagem. Assim sendo, é necessário efectuar uma transformação para um sistema de coordenadas do CAD, cuja origem se encontra no canto inferior esquerdo, e cuja unidade varia com a configuração. Para que as unidades coincidam com o CAD, ao ser transformadas, é necessário introduzir um parâmetro que representa a definição da imagem em pontos por polegada (DPI). Antes de determinar o centro, raio e ângulos inicial e final, transformam-se as coordenadas dos três pontos de pixels para polegadas, num sistema com a origem no canto inferior esquerdo, efectuando todas as transformações correctamente.

O desenho da imagem original, representda na figura 75, com dois segmentos de recta e um arco, foi correctamente detectado pelo sistema de processamento de imagem. Na segunda imagem podem ver-se os extremos dos segmentos recta detectados pelo método de rápida aproximação poligonal, utilizando um parâmetro de 0.5. A terceira imagem corresponde à aproximação das linhas por dois segmentos de recta e um arco de circunferência, podendo-se ver os pontos aproximados pelo arco. O ficheiro DXF de exportação destas curvas para um sistema CAD é apresentado em

anexo, sendo os ângulos inicial e final do arco respectivamente -157.30888° e 79.202606 °.

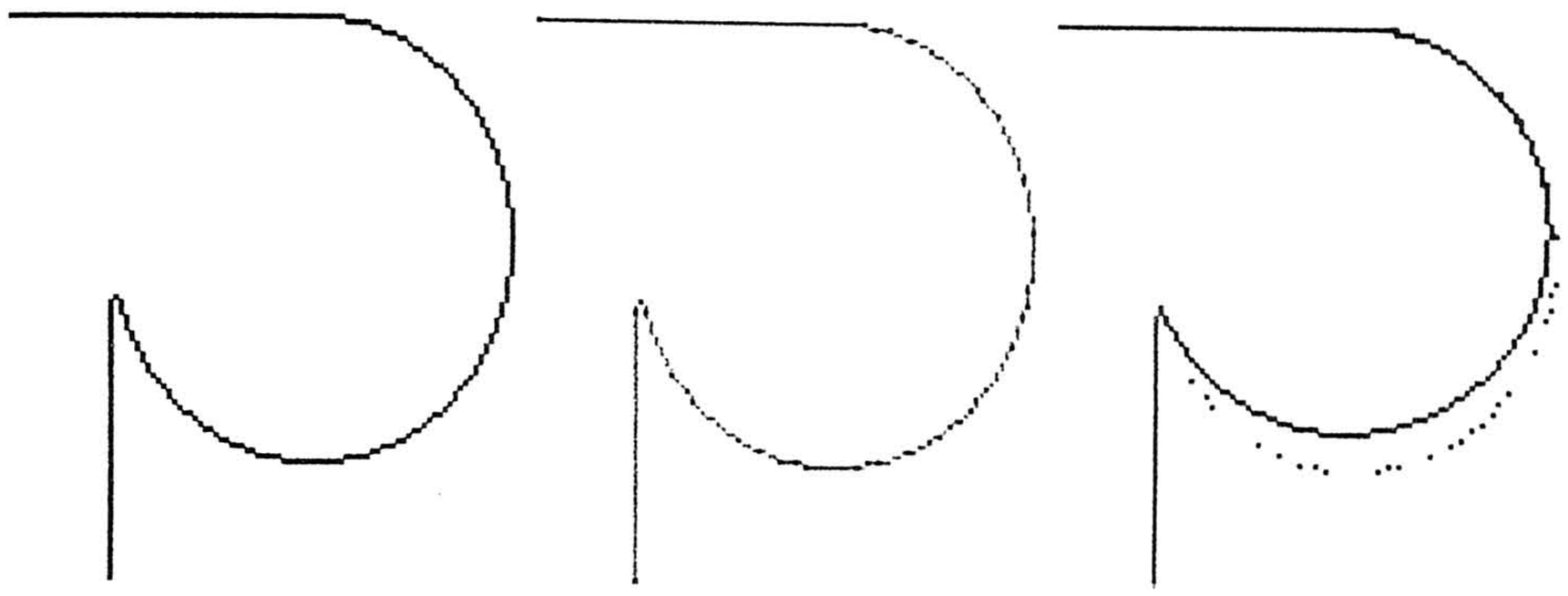


Figura 75: Imagem reconhecida pelo sistema de processamento de imagem como tendo dois segmentos de recta, e um arco de circunferência. Lida pelo AutoCAD correctamente.

5.6 Método Global

O método global de análise de imagens de documentos é constituído por um conjunto de passos que permite detectar as primitivas gráficas básicas das linhas de uma imagem de um desenho de engenharia, que serão ilustrados nesta secção. As imagens originais são representadas pela figura 76.

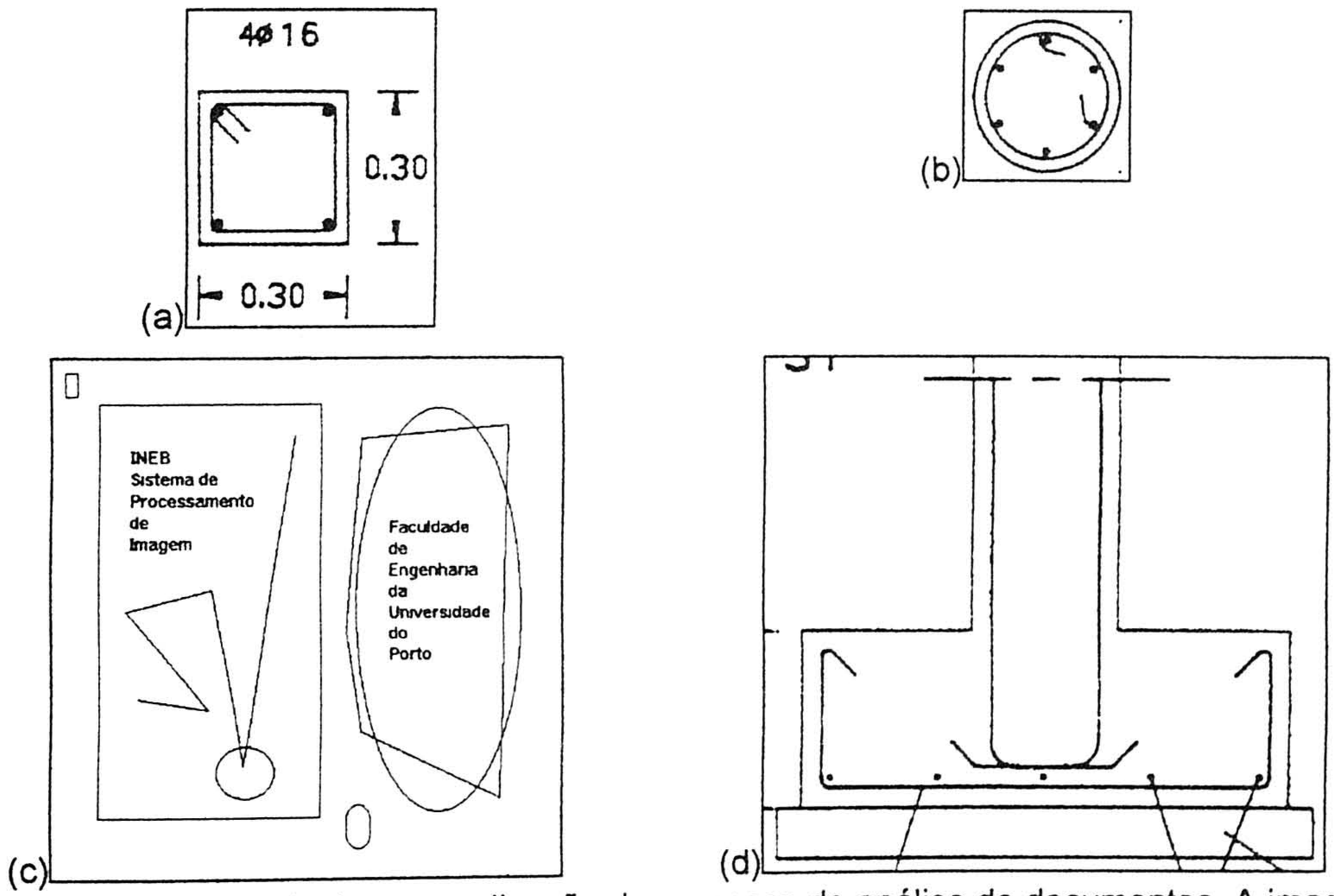


Figura 76: Imagens originais para aplicação do processo de análise de documentos. A imagem (a) será denominada Estrutura1, (b) será denominada Estrutura2, (c) será denominada Teste, (d) será denominada Estrutura3.

O primeiro passo corresponde à binarização das imagens Estrutura1 e Estrutura 2, ambas adquiridas em 256 níveis de cinzento com uma resolução de 300 dpi. A imagem Teste foi sintetizada em modo binário com espessura de linha de 1 pixel. A imagem Estrutura3 foi adquirida com uma resolução de 300 dpi, em dois níveis cromáticos. A binarização das imagens da figura 76 (a) e (b) foram binarizadas seleccionando um limiar pelo método de Otsu, estando as imagens resultado ilustradas pela figura 77.

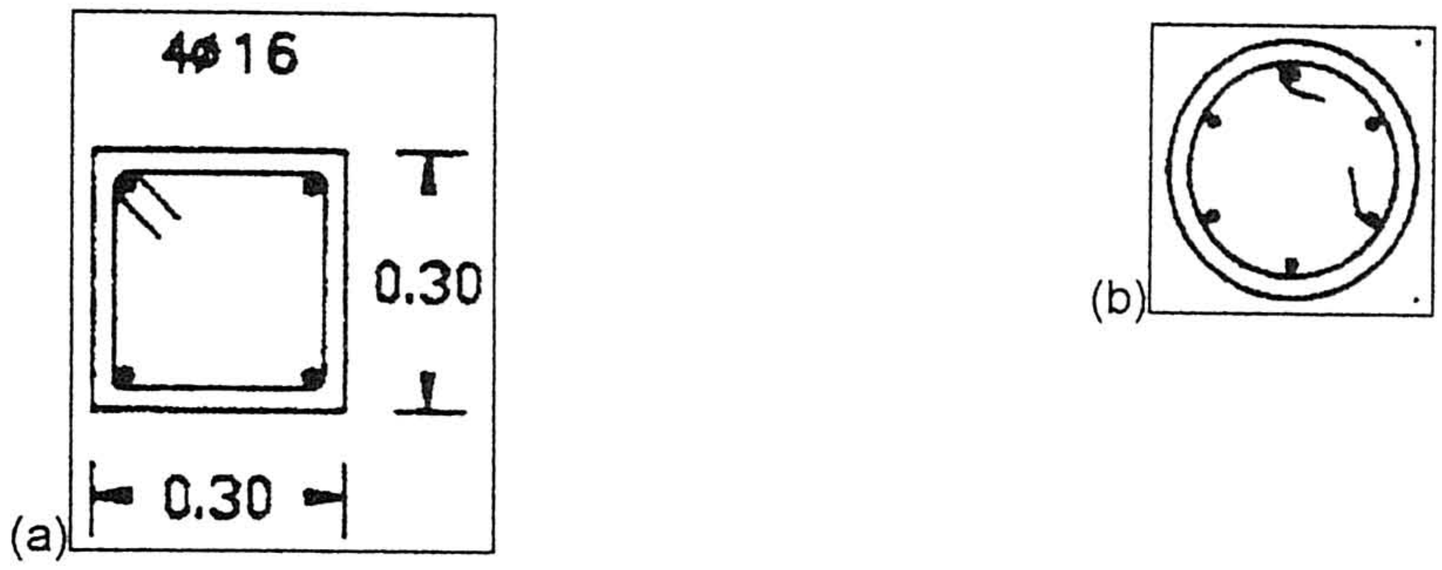


Figura 77: Imagens binarizadas.

Estas imagens binárias (figura 76c,d e 77a) são constituídas por blocos de texto e blocos gráficos, com excepção da imagem Estrutura2 binária (figura 77b). A estas imagens vai aplicar-se o método de separação dos blocos de texto, precedido duma etiquetagem de componentes, obtendo-se as imagens da figura 78.

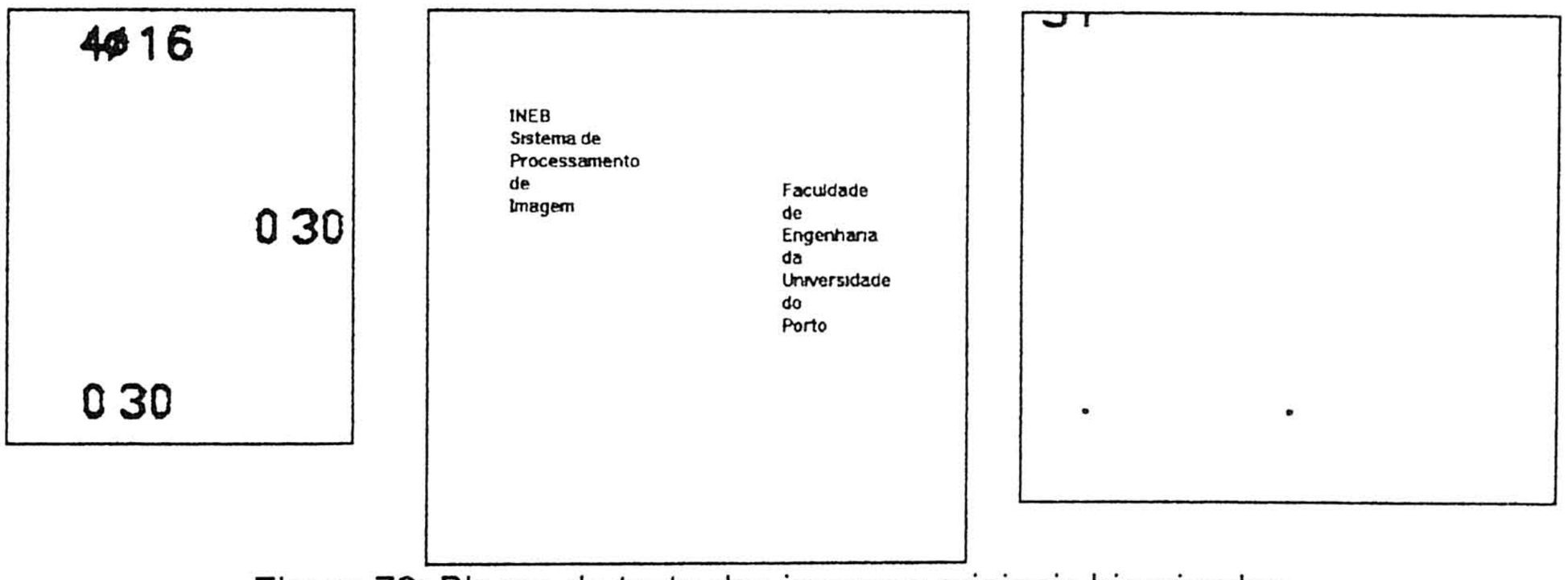


Figura 78: Blocos de texto das imagens originais binarizadas.

Aplicando a operação de disjunção lógica (os pixels pretos correspondem ao zero lógico) entre as imagens binarizadas e a negação das imagens com blocos de texto obtêm-se as imagens com blocos de gráficos, ilustradas pela figura 79.

Às imagens com blocos de gráficos é aplicado um conjunto de operações que permitem obter linhas adelgaçadas, já preparadas para se efectuar o seu reconhecimento. Aplicam-se, portanto, operações de fecho morfológico, adelgaçamento e poda morfológica (com excepção da imagem sintetizada), obtendo-se as imagens representadas pela figura 80.

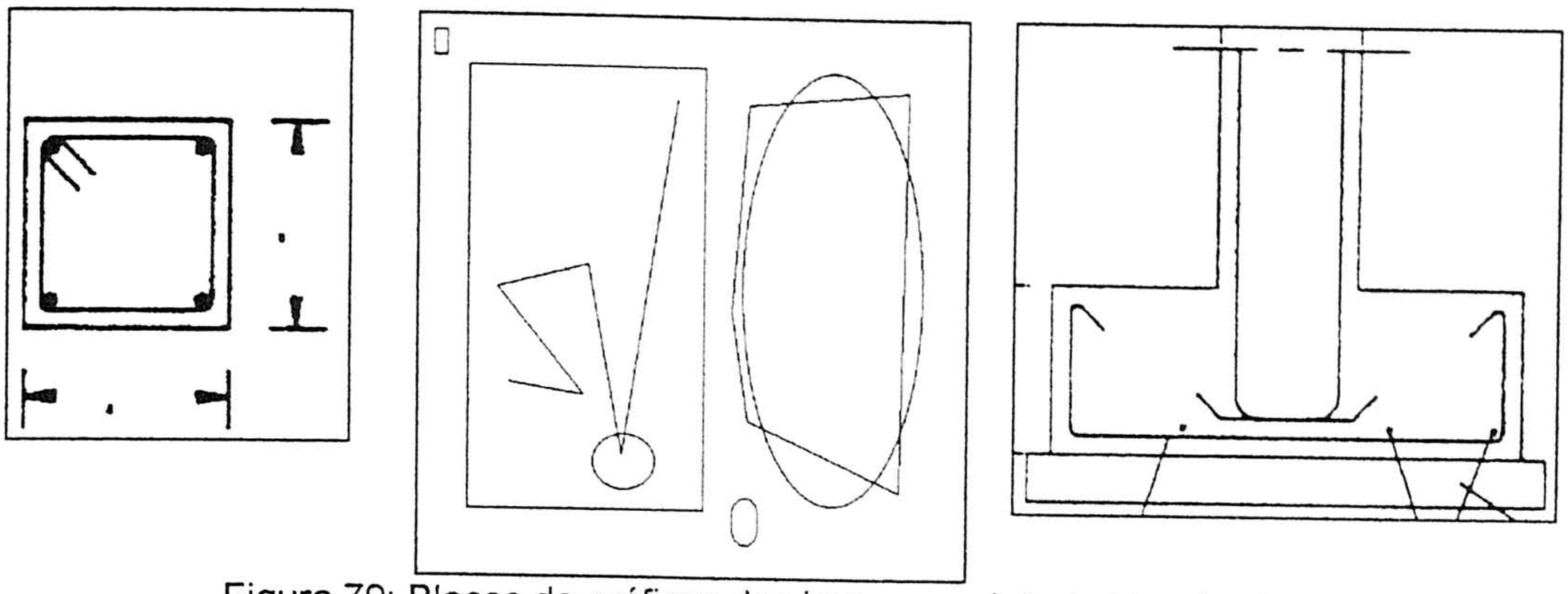


Figura 79: Blocos de gráficos das imagens originais binarizadas.

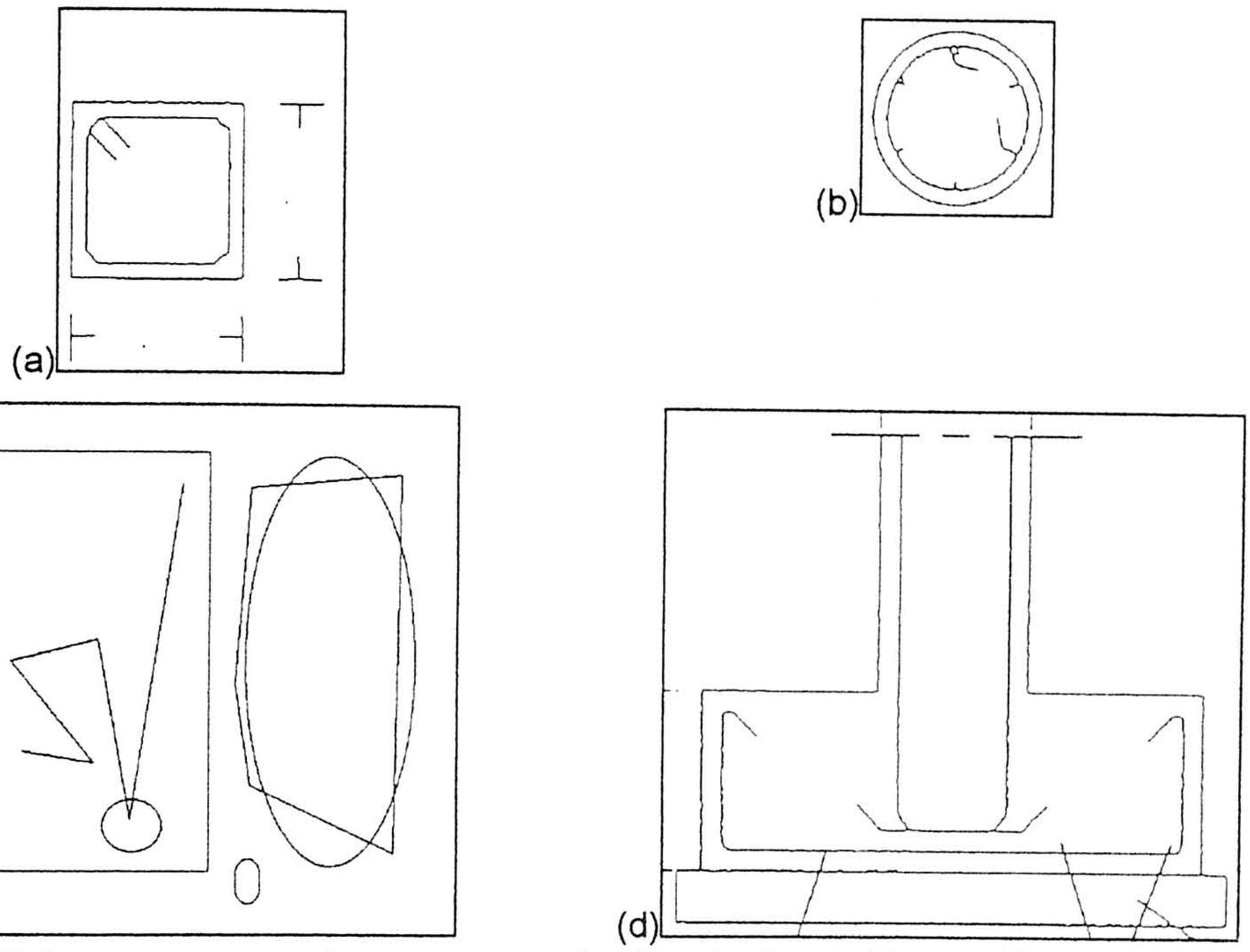


Figura 80: Imagens preparadas para o reconhecimento de primitivas gráficas básicas.

A imagem do documento digitalizado foi processada por diversos métodos que permitem binarizar a imagem, e separar o texto das linhas do desenho; estes métodos são ilustrados nesta secção, mas foram descritos detalhadamente no capítulo anterior.

Para reconhecimento gráfico é necessário efectuar um processamento inicial, que permita posteriormente detectar os segmentos de recta e arcos de circunferência que melhor representam as linhas. O primeiro passo consiste no adelgaçamento das linhas, passando estas a ser representadas pelo seu eixo médio. Em seguida, é necessário eliminar ramos parasitas, que são provocados pelas operações de adelgaçamento, e que não pertencem à descrição de eixo médio pretendido (figura 80). Finalmente, o último passo do processamento inicial corresponde à detecção de pontos do objecto, ligados entre si. Obtém-se assim, uma lista de componente ligados, cada um deles constituído

por listas de pontos correspondentes a linhas dos desenhos. Estas linhas vão ser analisadas individualmente, procurando as melhores primitivas gráficas que as descrevam. Resumindo, no final desta fase de pré-processamento obtém-se um conjunto de listas de pontos, cada uma delas respeitante a linhas que não têm cruzamentos com outras linhas.

Estas linhas, representadas por listas de pontos, poderão ser aproximadas apenas por segmentos de recta, isto é, poderão ser alvo duma aproximação poligonal, ou então, serão separadas em vários segmentos, cada um deles correspondendo a um conjunto de pontos ligados da linha que será aproximada por segmentos de recta ou por arcos de circunferência. Se o objectivo for o de obter apenas uma aproximação poligonal, então basta recorrer ao método de rápida aproximação poligonal, ou ao processo de detecção de cantos pelo método do coseno, para que as curvas digitais passem a ser descritas apenas pelos extremos dos vários segmentos de recta que descrevem a sua forma. A figura 81 ilustra a aproximação poligonal das imagens da figura 80, por segmentos de recta.

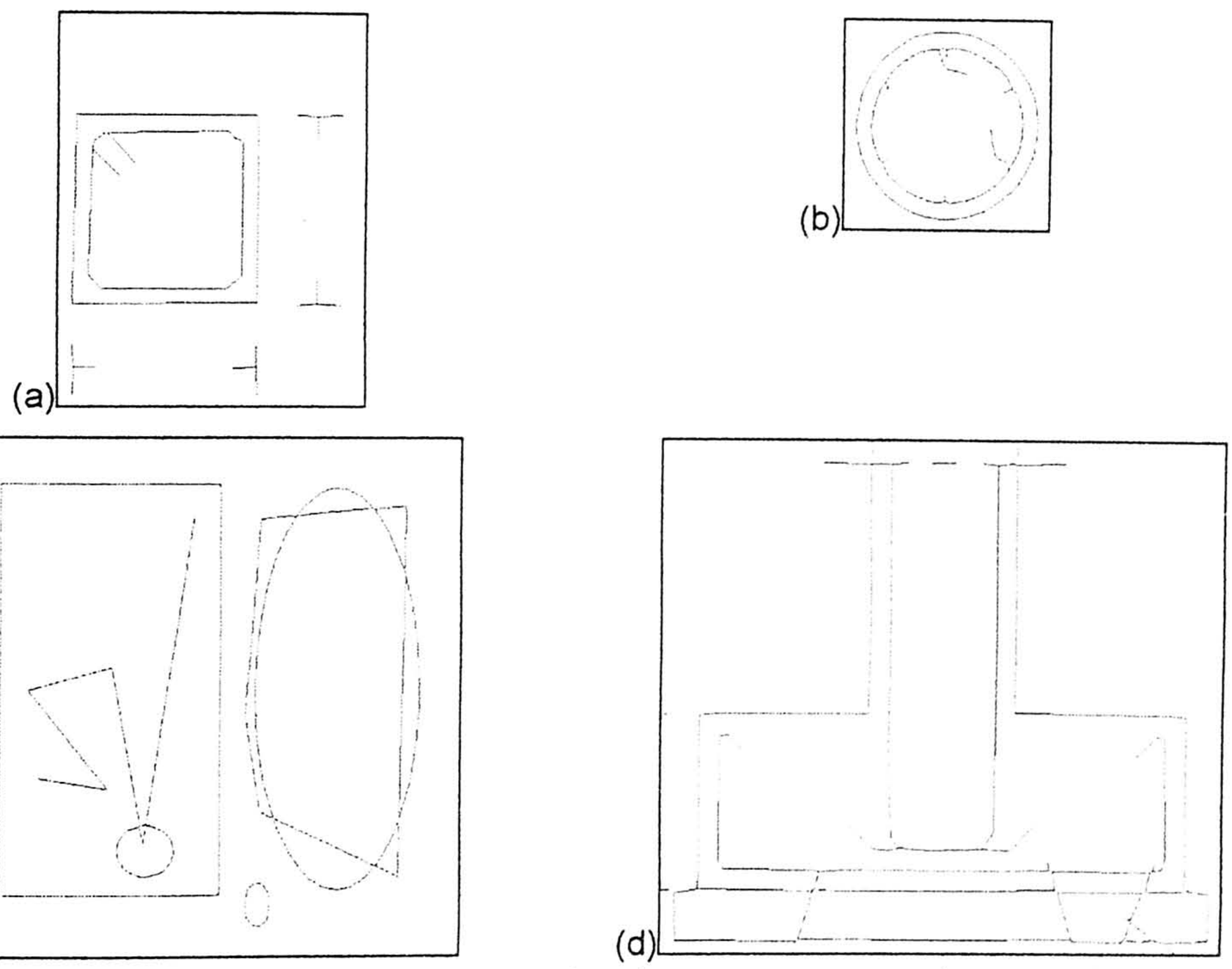


Figura 81: Imagens de desenhos de linhas aproximadas por segmentos de rectas. (a) Detecção de cantos com limiar do coseno igual a -0.866, e k=6. Nas outras imagens utilizou-se o método de rápida aproximação poligonal com o parâmetro T igual a (b) 0.6 (c) 0.8 e (d) 2.0.

Note-se que um dos objectivos desta tese, é o de obter também uma descrição por arcos de circunferência de zonas das curvas digitais. Para se atingir este objectivo, recorre-se ao algoritmo de aproximação de linhas por arcos de três pontos, sendo no

entanto necessário, começar por detectar os segmentos das linhas que serão melhor descritos por arcos de circunferência. Depois de detectados estes segmentos, utilizando o programa de processamento de imagem desenvolvido, efectua-se a aproximação das linhas digitais por segmentos de recta e por arcos de circunferência. Este processo é ilustrado pelas imagens resultado apresentadas na figura 82.

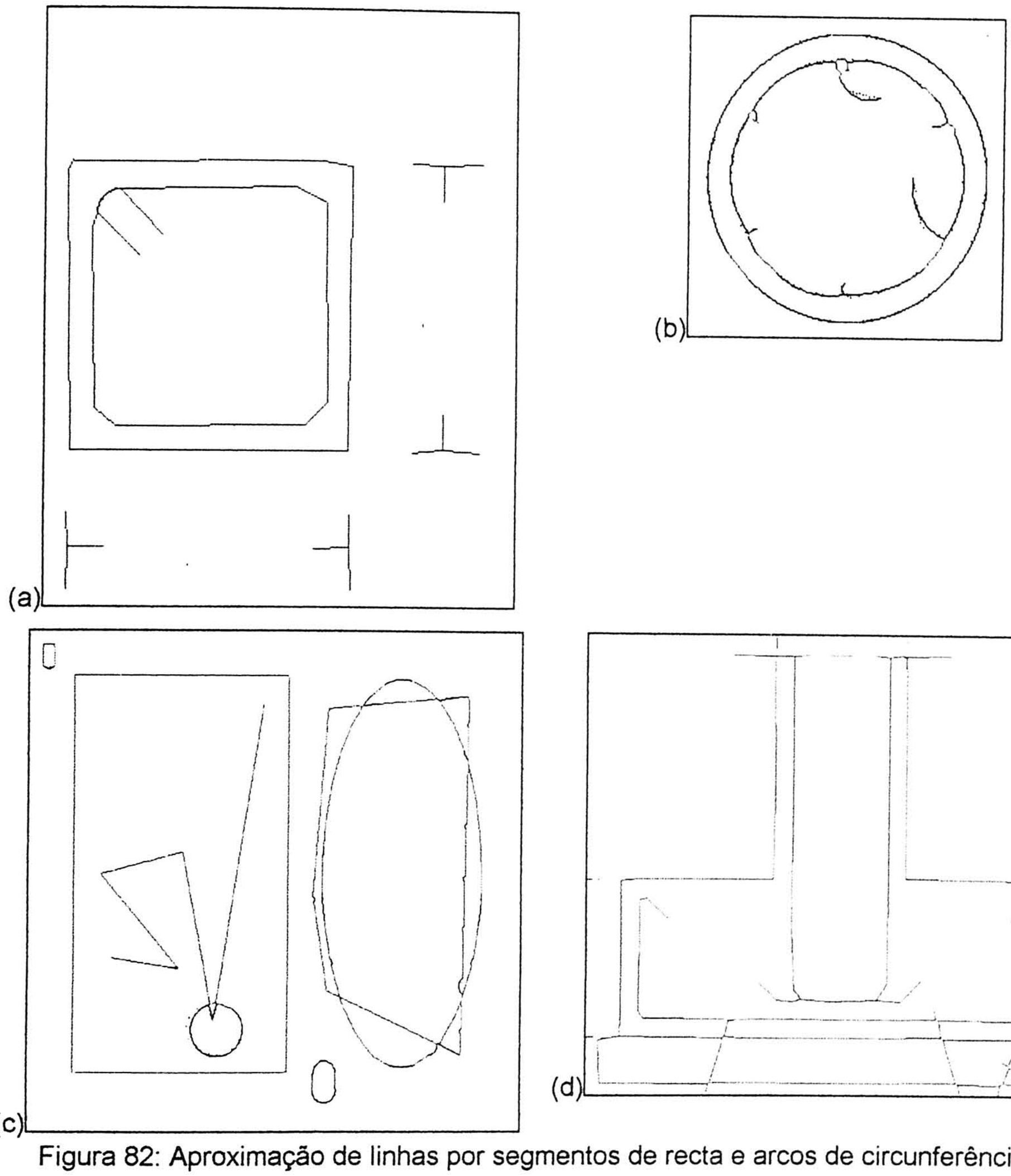


Figura 82: Aproximação de linhas por segmentos de recta e arcos de circunferência.

As descrições de linhas por arcos de circunferência e segmentos de recta destas imagens, foram exportadas para um ficheiro em formato DXF. No programa de CAD os segmentos de recta e os arcos de circunferência podem ser alterados, da forma mais conveniente para o utilizador. Nesta secção ficou ilustrado o processo global de análise de documentos de engenharia, desde a imagem adquirida até à imagem das primitivas gráficas reconhecidas pelo sistema, e exportadas para um ficheiro DXF.

Capítulo 6

Conclusão

6. CONCLUSÃO

Como ficou ilustrado na última secção do capítulo anterior, concebeu-se um processo global de análise de documentos de engenharia, com reconhecimento de primitivas gráficas básicas dos desenhos de linhas existentes nos documentos. Este processo mostrou-se de aplicação prática aceitável por permitir a segmentação de blocos de texto de desenhos de linhas dos documentos, e posteriormente a aproximação das linhas por segmentos de recta e arcos de circunferência. Além disto, no reconhecimento de gráficos efectuou-se uma abordagem à interpolação de linhas por curvas de Bézier e curvas Spline, e implementou-se um método de detecção de cantos recorrendo a curvas Spline.

Implementaram-se e avaliaram-se vários métodos de processamento inicial da imagem documento para posterior segmentação de blocos, e da imagem formada por desenhos de linhas para posterior reconhecimento de primitivas gráficas, nomeadamente de processamento morfológico, binarização, e adelgaçamento (método de processamento paralelo) de objectos gráficos das imagens documento.

O recurso a um formato compatível com um sistema CAD foi conseguido, tendo-se verificado a possibilidade de exportar primitivas gráficas obtidas por aproximação de linhas digitais.

As curvas Spline não são contempladas pelo formato compatível com um sistema CAD, logo, optou-se por não incluir esta descrição no método global. No entanto, as splines contribuem para o estudo do reconhecimento gráfico de linhas, na medida em que correspondem a uma descrição gráfica suave de linhas digitais. A sua principal aplicação é feita na modelação das formas de objectos, detectadas por operadores de detecção de bordos, para posterior armazenamento (compressão da informação) e reconhecimento.

Neste trabalho implementou-se um sistema de processamento e análise de imagem baseado no ambiente gráfico Windows, que permitiu desenvolver todos os métodos de análise de documentos aqui descritos, mas que permite um processamento genérico de imagens.

As possibilidades de posterior desenvolvimento no domínio da análise de documentos de engenharia são muito vastas. O processamento inicial de imagens digitalizadas poderia ser aprofundado, nomeadamente na binarização das imagens, analisando e avaliando outros métodos de selecção de limiares de segmentação globais, regionais ou adaptativos.

A segmentação de blocos de imagens de documentos, poderia só por si, corresponder a um trabalho detalhado, que definisse várias formas de segmentação, adaptadas a estruturas rectangulares de página, ou não, podendo inclusive descriminar blocos que se sobreponham. O reconhecimento de texto será outra das áreas englobadas pela análise de documentos que pode ser aprofundada em trabalhos posteriores.

O reconhecimento gráfico de mais alto nível, permitiria associar o reconhecimento de primitivas gráficas básicas, da sua estrutura, e do significado de objectos gráficos dependentes do contexto do documento, nomeadamente na percepção de símbolos gráficos e de cotas dimensionais.

No reconhecimento de primitivas gráficas, a utilização de curvas da família das splines, possibilitaria descrever linhas digitais, por curvas suaves. Esta poderá ser uma descrição com interesse para bordos de objectos, pela compressão da informação, e para o reconhecimento de formas. O desenvolvimento de outros métodos de aproximação de linhas recorrendo a arcos de circunferência e segmentos de recta, e a sua avaliação, permitiria obter uma visão mais alargada sobre os diversos métodos de aproximação de linhas, e sobre os melhores métodos para imagens de documentos específicos ou para a descrição de diferentes objectos gráficos.

Um dos principais contributos desta tese foi o de englobar variadissímos processos de análise de imagem, e aplicá-los num único direccionamento relacionado com a análise de documentos de engenharia. Permitiu também, avaliar as possibilidades de aproximação gráfica de linhas digitais, recorrendo não só a aproximações poligonais (mais comuns) mas também a aproximações por arcos de circunferência, e efectuar uma abordagem à aproximação por curvas da família das splines. Com o método de

segmentação de blocos obtiveram-se bons resultados de classificação de componentes de texto e de componentes de gráficos, embora pudesse ser desenvolvido no sentido de segmentar blocos sobrepostos, ou de classificar correctamente componentes de texto classificados como componentes gráficos, recorrendo por exemplo a medidas de colinearidade de componentes. A transferência de informação vectorial para um ficheiro compatível com programas CAD, permite fazer uma ponte entre o processamento e análise de imagens em forma de mapas de bits, e programas de desenho vectorial.

Neste trabalho condensaram-se diversos métodos de análise de imagens de documentos, e concebeu-se uma metodologia de processamento global, que em conjunto com as sugestões de desenvolvimento poderão servir de base a trabalhos posteriores.

Pode concluir-se que foram atingidos os objectivos principais deste trabalho, ao conceber-se um processo de análise de documentos de engenharia, que permite a partir da imagem digitalizada, segmentar diferentes blocos do documento, efectuar o reconhecimento de primitivas gráficas, e exportar a informação obtida para um sistema CAD.

ANEXOS

Anexo A

A. FORMATO DIB

A. FORMATO DIB

Uma imagem DIB representa um mapa de bits e é constituída pela estrutura BITMAPINFO (figura 1), que descreve as dimensões e cores da imagem seguida por uma matriz de pixels, cada uma com a dimensão definida no cabeçalho.

```
typedef struct tagBITMAPINFO {
    BITMAPINFOHEADER  bmiHeader;
    RGBQUAD           bmiColors[1];
}BITMAPINFO
```

Figura A.1: Estrutura BITMAPINFO

A estrutura BITMAPINFO é constituída pelo cabeçalho BITMAPINFOHEADER (figura 2) e pela tabela de valores RGB[1], representada pela estrutura QUAD (figura 3), que representam as cores, de cada valor de intensidade de pixel. A descrição dos campos do cabeçalho, dum mapa de bits DIB é a seguinte:

- 'biSize' define o tamanho do cabeçalho em bytes, sempre 40; 'biWidth' representa a largura da imagem, em pixels;
- 'biHeight' representa a altura da imagem, em pixels; 'biPlanes' especifica o número de planos da imagem e é sempre igual a um;
- 'biBitCount' representa o número de bits por pixel, podendo tomar os valores 1, 4, 8, ou 24;
- 'biCompression' especifica o tipo de compressão da imagem, podendo tomar o valor BI_RGB que determina um mapa de bits sem compressão, BI_RLE8 ou BI_RLE4 que especifica uma compressão do tipo "código de comprimento de sequências" (RLE[2]) para imagens respectivamente com 8 ou 4 bits por pixel;
- 'biSizeImage' representa o número de bytes ocupados pelos pixels da imagem, igual a:
  $\frac{(nWidth \times nHeight)}{(8 / biBitCount)}$;
- 'biXPelsPerMeter' e 'biYPelsPerMeter' especificam, respectivamente, a resolução horizontal e vertical da imagem, em pixels por metro do dispositivo alvo para o mapa de bits, informação que pode ser utilizada para escolher a imagem que melhor se adapte a um dispositivo de visualização;

---

[1]"Red, Green and Blue"
[2]"Run Length Encoded"

- 'biClrUsed' especifica o número de índices de cor da tabela de cores que realmente são utilizados pelo mapa de bits;
- 'biClrImportant' especifica o número de índices de cor da tabela de cores que são considerados importantes para a visualização.

```
typedef struct tagBITMAPINFOHEADER {
    DWORD   biSize;
    LONG    biWidth;
    LONG    biHeight;
    WORD    biPlanes;
    WORD    biBitCount;
    DWORD   biCompression;
    DWORD   biSizeImage;
    LONG    biXPelsPerMeter;
    LONG    biYPelsPerMeter;
    DWORD   biClrUsed;
    DWORD   biClrImportant;
} BITMAPINFOHEADER;
```

Figura A.2: Estrutura BITMAPINFOHEADER

Finalmente, é necessário referir a estrutura da matriz de pixels. Esta é constituída por 'nHeight' linhas em ordem inversa, isto é, da última para a primeira linha. Cada linha tem 'nWidth' pixels, que traduzido em bits, corresponde ao número de bits por pixel vezes a largura da imagem aproximada ao número seguinte de 4 bytes de tamanho, isto é, alinhado ao DWORD seguinte. Para se determinar o tamanho de cada linha, em bytes, é necessário efectuar o seguinte cálculo: $((nWidth \times biBitCount)+31)/32*4$.

```
typedef struct tagRGBQUAD {
    BYTE    rgbBlue;
    BYTE    rgbGreen;
    BYTE    rgbRed;
    BYTE    rgbReserved;
} RGBQUAD;
```

Figura A.3: Estrutura RGBQUAD

Resumindo, um mapa de bits com a estrutura dum DIB, é constituído por um cabeçalho (BITMAPINFOHEADER), seguido pela tabela de cores (elementos RGBQUAD) e depois pela matriz de pixels.

Um ficheiro com o formato BMP [4] tem exactamente a mesma estrutura, antecedida por 14 bytes, representados no SDK pela estrutura de cabeçalho dum ficheiro BMP, BITMAPFILEHEADER (figura 4), com a seguinte informação:

- 'bfType' contém os códigos ASCII das letra 'B' e 'M';

- 'bfSize' representa o tamanho do ficheiro, em bytes;
- quatro bytes reservados;
- 'bfOffBits' representa o número de bytes desde o início do ficheiro até ao início da matriz de pixels, isto é, corresponde à soma dos tamanhos da estrutura de cabeçalho dum ficheiro BMP (14 bytes) com estrutura de cabeçalho dum mapa de bits DIB (40 bytes), com o tamanho da tabela de cores (dependente do número de bits por pixel e do número de cores usadas).

```
typedef struct tagBITMAPFILEHEADER {
    UINT    bfType;
    DWORD   bfSize;
    UINT    bfReserved1;
    UINT    bfReserved2;
    DWORD   bfOffBits;
} BITMAPFILEHEADER;
```

Figura A.4: Estrutura BITMAPFILEHEADER

Anexo B

B. OPÇÕES DE UTILIZAÇÃO

B. Opções de Utilização

Neste anexo descrevem-se as opções existentes no programa, que permitem processar e analisar imagens, gerir os ficheiros respectivos, modificar modos de visualização, obter informação sobre os mapas de bits, ou simplesmente gerir as janelas criança respeitantes aos vários documentos abertos. Isto vai ser feito seguindo a barra de menu, ou simplesmente menu, e respectivas opções, com a descrição de algumas operações não existentes no menu.

A barra de menu é constituída por várias opções representadas por menus de abertura[1], que por sua vez podem ter itens de menu (respeitantes a comandos) ou outros menus de abertura. Esta barra é constituída pelos seguintes menus de abertura:

- **'File'**, onde se acedem às operações que permitem gerir os ficheiros das imagens;
- **'Edit'**, a partir do qual se torna possível copiar e colar imagens ou parte de imagens;
- **'View'**, onde estão os comandos para controlo do modo de visualização da janela de programa;
- **'Processing'**, onde se acedem às operações de processamento e análise de imagem;
- **'Document'**, a partir do qual se torna possível aplicar comandos sobre a imagem de forma a conseguir-se efectuar a análise de imagens documento;
- **'Info'**, onde estão os comandos que mostram informação sobre a imagem;
- **'Options'**, onde se acedem às operações que permitem alterar parâmetros relacionados com algoritmos de processamento de imagem ou de processamento de imagens documento;
- **'Window'**, para gerir as janelas criança, e os ícones representando essas janelas minimizadas;
- **'Help'**, para se obter informação de ajuda, ou informação sobre a versão do programa, recorrendo à "janela acerca de"[2].

O menu de abertura 'File' tem as seguintes opções:

- **'New'**, cria um novo documento vazio;
- **'Open'**, abre um documento existente no disco;

---

[1]Do inglês "pop-up" que corresponde a algo que surge inesperadamente
[2]Do inglês "About window"

- **'Close'**, fecha um documento aberto, ou seja, uma janela criança associada a um documento;
- **'Save'**, grava um documento aberto;
- **'Save As'**, grava um documento aberto, alterando-lhe algumas características;
- **'Export Curves'**, grava as linhas e círculos, detectadas na fase de processamento de documentos de engenharia, com o formato 'DXF', apropriado para entrada num programa CAD, mais propriamente no Autocad;
- **'Print'**, imprime uma imagem;
- **'Print Preview'**, permite visualizar a forma como a imagem vai ser impressa;
- **'Print Setup'**, para configurar a impressora;
- **'Recent File'**, corresponde a uma lista das quatro imagens abertas mais recentemente;
- **'Exit'**, para abandonar e fechar o programa.

O menu de abertura 'Edit' tem as seguintes opções:

- 'Undo', não está implementado;
- 'Cut', não está implementado;
- **'Copy'**, permite copiar uma imagem, ou uma área rectangular da mesma, colocando-a em memória, para posterior colagem;
- **'Paste'**, cola (coloca) uma imagem no ponto de inserção, ou numa janela criança aberta, substituindo a imagem lá existente, ou noutro programa.

O menu de abertura 'View' tem as seguintes opções:

- **'Toolbar'**, menu de verificação[3], para visualização ou não da barra de ferramentas;
- **'Status Bar'**, menu de verificação, para visualização ou não da barra de estado;

O menu de abertura 'Processing' tem as seguintes opções:

- **'Logical'**, menu de abertura, para processamento de imagens com operações lógicas;
  - **'Not'**, operação lógica de negação da imagem activa;
  - **'And'**, operação lógica de conjunção entre a imagem activa, e a imagem escolhida para operando (ver mais à frente, no menu de opções);
  - **'Or'**, operação lógica de disjunção entre a imagem activa, e a imagem escolhida para operando;
  - **'Xor'**, operação lógica de disjunção entre a imagem activa, e a imagem escolhida para operando;

---

[3]Do inglês "checked" e que corresponde a um sinal de verificação que é colocado no menu

- **'Arithmetic'**, menu de abertura, para processamento de imagens com operações aritméticas;
  - **'Addition'**, adição entre a imagem activa e a imagem operando, mantendo os limites dos níveis cromáticos por aproximação linear;
  - **'Subtraction'**, subtracção entre a imagem activa e a imagem operando, mantendo os limites dos níveis cromáticos por aproximação linear;
  - **'Multiplication'**, multiplicação entre a imagem activa e a imagem operando, mantendo os limites dos níveis cromáticos por aproximação linear;
  - **'Division'**, divisão entre a imagem activa e a imagem operando, mantendo os limites dos níveis cromáticos por aproximação linear;
- **'Convolution'**, menu de abertura, para efectuar operações de convolução;
  - **'Convolve'**, efectua a convolução entre a imagem activa e janelas de convolução pre-definidas utilizando o menu 'options';
  - **'Crossing'**, efectua uma detecção de atravessamentos de zero;
  - **'Grad. Sobel'**, efectua uma operação de detecção de gradientes, utilizando as máscaras de Sobel;
- **'Min, Max, Median...'**, operações de detecção de mínimo, máximo, mediana, e similares utilizando uma janela de tamanho estabelecido no menu 'options';
  - **'Min'**, operação de mínimo local utilizando uma janela de dimensão definida no menu 'Options';
  - **'Max'**, operação de máximo local utilizando uma janela de dimensão definida no menu 'Options';
  - **'ClosestMinMax'**, operação local utilizando uma janela de dimensão definida no menu 'Options', em que se substitui o pixel central pelo valor mais próximo do máximo ou do mínimo;
  - **'Median'**, operação de mediana local utilizando uma janela de dimensão definida no menu 'Options';
- **'Threshold'**, menu de abertura com comandos que permitem efectuar a limiarização da imagem;
  - **'Select'**, permite seleccionar um limiar de binarização da imagem, e em função deste efectuar a operação de limiarização;
  - **'Otsu'**, retorna o limiar de binarização determinado pelo método de 'Otsu' [33], e em função deste efectua a operação de limiarização da imagem;
  - **'MaxEntropy'**, retorna o limiar de binarização determinado pelo método de "máxima entropia" [34], e em função deste efectua a operação de limiarização da imagem;

- **'mlReddi'**, retorna um número de valores de limiarização determinados pelo método de 'Reddi', tratando-se portanto, de uma operação de limiarização da imagem por vários níveis;

- **'Morphology'**, operações morfológicas, com imagens binárias e em nível de cinzento, utilizando máscaras cujo tamanho e conteúdo são estabelecidos no menu 'options';
  - **'Dilation'**, operação morfológica de dilatação;
  - **'Erosion'**, operação morfológica de erosão;
  - **'Open'**, operação morfológica de abertura;
  - **'Close'**, operação morfológica de fecho;
  - **'Hit-Miss'**, operação morfológica de "acerto e falha";
  - **'Prunning'**, operação morfológica de poda;
- **'Labeling'**, operação que cataloga todos os componentes ligados existentes na imagem com "etiquetas" diferentes, isto é, com diferentes níveis cromáticos, associando esta informação sob a forma de listas, ao documento para posterior análise da imagem;
- **'Chain'**, operação que coloca todos os pontos de uma cadeia de pontos ligados, numa lista associada documento para posterior análise da imagem;
- **'Thinning'**, menu de abertura para adelgaçamento de objectos escuros existentes na imagem;
  - **'Bones'**, adelgaçamento de objectos utilizando um algoritmo descrito por Rosenfeld e Kak em [5];
  - **'thinKxK'**, adelgaçamento de objectos utilizando um algoritmo definido por O'Gorman em [58];

O menu de abertura 'Document' tem as seguintes opções:

- **'Split_text'**, para separar componentes ligados que correspondam a texto de outros que correspondam a gráficos;
- **'Segment Curves'**, para separar listas de pontos em listas de segmentos de recta e de segmentos de circunferência (depois de 'Polygonal Appr');
- **'Polygonal Appr'**, menu de abertura para aproximação poligonal de listas de pontos (depois de 'Chain');
  - **'fastPolygonal'**, aproximação de linhas de pontos por segmentos de recta recorrendo ao algoritmo descrito em [54];
  - **'Corners by Cos'**, aproximação de linhas de pontos por segmentos de recta recorrendo ao algoritmo descrito por Wu e Wang em [66];
- **'Arc Appr.'**, menu de abertura para aproximação de pontos por arcos de circunferência;

- **'Arc of circunference'**, aproximação de linhas de pontos por arcos de circunferência, utilizando um algoritmo construído por mim, e baseado num outro de desenho de arcos de circunferência de Galton [43];

- **'Spline Appr.'**, menu de abertura para aproximação de listas de pontos ligados utilizando curvas do tipo 'spline';
  - **'Corner Detection'**, detecção de cantos de um conjunto de pontos, utilizando uma aproximação desses pontos por uma "spline";
- **'Draw'**, menu de abertura para desenho da imagem, utilizando a aproximação do documento determinada ao longo do programa;
  - **'Draw Chain'**, desenha as listas de pontos ligados determinados pelo comando 'Chain';
  - **'Draw Appr.'**, desenha os segmentos de recta e arcos de circunferência determinados pelas operações de aproximação de listas de pontos ligados;

O menu de abertura 'Info' tem as seguintes opções:

- **'Info'**, abre uma janela de dialogo com informação sobre a imagem (formato, tamanho, dimensões horizontal e vertical, ...);
- **'Histogram'**, determina o histograma da imagem;

O menu de abertura 'Options' tem as seguintes opções:

- **'Kernel Dimension'**, para definição das dimensões horizontal e vertical da máscara de convolução que irá ser utilizada em diversas operações;
- **'Kernel'**, para definição do conteúdo da máscara de convolução que irá ser utilizada em diversas operações;
- **'Spline Parameters'**, definição dos parâmetros utilizados em 'Corner Detection';
- **'Cos Corner Parameters'**, definição dos parâmetros utilizados em 'Corners by Cos';
- **'Fast Poly Parameters'**, definição dos parâmetros utilizados em 'fastPolygonal';
- **'Arithmetic Operand'**, define a janela criança activa como sendo a que contém o documento que servirá de operando nas operações que necessitem de uma segunda imagem;

O menu de abertura 'Window' tem as seguintes opções:

- **'New Window'**, para abertura de uma nova janela criança vazia;
- **'Cascade'**, para colocação das janelas criança em cascata;
- **'Tile'**, para colocação das janelas criança de forma a ocuparem toda a área de cliente da janela de programa;
- **'Arrange Icons'**, para arranjar os ícones;

O menu de abertura 'Help' tem as seguintes opções:

- **'Index'**, não implementado;
- **'Using Help'**, não implementado;
- **'About DIBLOOK...'**, para abertura da janela de diálogo "acerca de".

Anexo C

C. FICHEIRO DXF

C. FICHEIRO DXF

```
  0
SECTION
  2
ENTITIES
  0
LINE
  8
0
 10
0.006667
 20
0.36
 30
0.0
 11
0.216667
 21
0.36
 31
0.0
  0
LINE
  8
0
 10
0.073333
 20
0.183333
 30
0.0
 11
0.07
 21
0.176667
 31
0.0
  0
LINE
  8
0
 10
0.07
 20
0.176667
 30
0.0
 11
0.07
 21
0.006667
 31
0.0
  0
ARC
  8
0
 10
0.192475
 20
0.23315
 30
0.0
 40
0.129137
 50
202.69112
 51
79.202606
  0
ENDSEC
  0
EOF
```

Anexo D

D. PUBLICAÇÕES

THRESHOLDING SELECTION METHODS: A COMPARATIVE STUDY

Rui Manuel Lima*, Aurélio J. C. Campilho

Departamento de Engenharia Electrotécnica e de Computadores

Faculdade de Engenharia da Universidade do Porto

Rua dos Bragas 4099 Porto Codex, Portugal

email: gpai@fe.up.pt

Abstract

Four threshold selection methods are presented and evaluated relative to three images using an uniformity and shape measure. The methods used are the Otsu Method, the Maximum Entropy Method, the Uniform Error Method and finally the Minimum Error Method. This work and results herein described were a partial requirement of the course "Application of Digital Image Processing" of the Master Degree in Electrical and Computer Engineering.

1. Introduction

Thresholding is a basic image segmentation operation on a gray scale image. When a single threshold t is used, two distinctive set of pixels are labelled: label b_0 when pixel gray values are lower than t; label b_1 for those pixels with higher gray value. If several thresholds are involved, several regions are obtained, and the method is usually referred as multi-level thresholding.

Although many researchers avoid to use these methods for their lack of contextual or structural information, yet in many situations, when objects have to be extracted from a scene, thresholding still presents itself as an important tool that can be easily implemented and executed at high speeds.

The automatic selection of a single threshold value is the main issue of several proposed methods. In this paper four methods will be reviewed and quantitatively evaluated.

In a thresholding operation, using a single threshold t, an image f with N×N pixels and l grays level values is converted in a binary image f_t with two values b_0 and b_1 i.e., from

$$f: \mathrm{N}\times\mathrm{N}\to\Gamma=\{0,1,\ldots,l-1\} \qquad (\mathrm{x},\mathrm{y})\mapsto f$$

results

$$f_t: \mathrm{N}\times\mathrm{N}\to\mathrm{B}=\{b_0,b_1\} \qquad t,b_0,b_1\in\Gamma$$

$$(\mathrm{x},\mathrm{y})\mapsto f_t=\begin{cases} b_0 & ,f(x,y)\le t \ (t\text{ - threshold}) \\ b_1 & ,f(x,y)>t \end{cases}$$

* Supported by a grant from Programa CIÊNCIA

2. Threshold Selection Method

In general, the thresholding operations that transform a gray level image into a binary image, are classified as global thresholding and local thresholding. A global thresholding technique thresholds the entire image with a single threshold value, whereas a local thresholding technique partitions the image into sub-images and automatically selects a threshold value for each of these sub-images. If the threshold value is determined solely from the gray level of each pixel, then the threshold method is point-dependent. If the threshold value is determined from a local property (e.g., the local gray level distribution) in the neighbourhood of each pixel, then the threshold method is region-dependent. The selection of a threshold is the main question of the thresholding operation. The choice of an appropriate threshold is easy when the histogram of gray level values has two well-separated peaks, however many problems do appear making difficult the automation of this task. Trying to overcome these problems, several methods have been proposed. In this paper, four methods will be briefly presented:

- Method 1, Otsu Method [8];
- Method 2, Maximum Entropy Method [2];
- Method 3, Uniform Error Method [3];
- Method 4, Minimum Error Method [10].

2.1 Otsu Method

For most real life images, it is often difficult to detect a valley in the histogram of gray level values, and as result the selection of the adequate threshold is not an easy task.

Otsu [8] proposed a thresholding method based on the discriminant analysis for images with gray level histograms with broad and flat valleys or when the two modes are extremely unequal in height.

In this method, the threshold operation is regarded as the partitioning of the pixels of an image into two classes b_0 and b_1 (e.g., objects and background), as described in section 1, at gray level t. Let ${\sigma_W}^2$, ${\sigma_B}^2$ and ${\sigma_T}^2$ be the within-class variance, between-class variance, and the total variance, respectively. An optimal threshold can be determined by maximizing one of the following criterion functions with respect to t:

$$\lambda = \frac{{\sigma_B}^2}{{\sigma_W}^2}, \qquad \eta = \frac{{\sigma_B}^2}{{\sigma_T}^2}, \text{ and} \qquad \kappa = \frac{{\sigma_T}^2}{{\sigma_W}^2}.$$

Where: ${\sigma_B}^2 = \omega_0 \cdot \omega_1 \cdot (\mu_0 - \mu_1)^2$, ${\sigma_W}^2 = \omega_0 \cdot {\sigma_0}^2 + \omega_1 \cdot {\sigma_1}^2$, and ${\sigma_T}^2 = \sum_{i=0}^{l-1} (i - \mu_T)^2 \cdot p_i$

$$\omega_0 = \sum_{i=0}^{t} p_i, \qquad \omega_1 = 1 - \omega_0, \qquad \mu_T = \sum_{i=0}^{l-1} i \cdot p_i, \qquad \mu_t = \sum_{i=0}^{t} i \cdot p_i, \qquad \mu_0 = \frac{\mu_t}{\omega_0}, \qquad \mu_1 = \frac{\mu_T - \mu_t}{\omega_1},$$

$${\sigma_0}^2 = \frac{\sum_{i=0}^{t} (i - \mu_0)^2 \cdot p_i}{\omega_0}, \text{ and} \qquad {\sigma_1}^2 = \frac{\sum_{i=t+1}^{l-1} (i - \mu_1)^2 \cdot p_i}{\omega_1}$$

Of the above three criterion functions, η is the simplest. Thus, the optimal threshold t is:

$$t^* = Arg \underset{t \in \Gamma}{Min}\, \eta .$$

The problem of thresholding is reduced to the optimization problem of searching for a threshold t that maximizes the criterion measures given above. As σ_T^2 is constant with respect to t, maximization of η is equivalent to maximisation of σ_B^2.

2.2 Maximum Entropy Method

The optimal threshold value is determined by maximizing the *a posteriori* entropy subject to certain inequality constraints which are derived by means of special measures (described in section 4) characterizing uniformity and shape of the regions in the image.

The *a posteriori* probability of the pixels with gray values less than t, which is the probability of b_0, is given by

$$F(t) = \sum_{i=0}^{t} \frac{fr_i}{N_T} = \sum_{i=0}^{t} p_i ,$$

with fr_i as the frequency of gray level i and p_i as the probability of the gray level i in the image, being N_0 the total number of pixels. Similarly, the *a posteriori* probability of all those pixels with gray values greater than or equal to t, which is the probability of b_1, is given by $F'(t) = 1 - F(t)$. In the maximum entropy principle proposed in [2] it is involved the maximization of the Shannon entropy of the binary image, given by

$$H\big(F(t)\big) = -F(t)\cdot log_2 F(t) - \big(1 - F(t)\big)\cdot log_2\big(1 - F(t)\big).$$

If there is no *a priori* knowledge about the image, the maximum entropy principle suggests that the threshold value could give rise to equal a posteriori probabilities of black and white pixels (i.e. $F(t) = 0.5$). However, in this method the selection of the threshold value is constrained by two measures, that are derived from the spatial information, which are the uniformity measure ($U(t)$) and the shape measure ($S(t)$), defined later in this article. If t is the gray level that maximizes the Shannon's entropy, t_1 is the gray level that maximizes the uniformity measure, t_2 is the gray level that maximizes the shape measure, e.g.:

$$t \mapsto F(t) = 0.5, \quad t_1 = Arg \underset{t \in \Gamma}{Max}\, U(t), \qquad t_2 = Arg \underset{t \in \Gamma}{Max}\, S(t),$$

and the threshold value will be selected by:

$$t^* = \begin{cases} max(t_1, t_2), & F(t_1) \in [0, 0.5] \;,\; F(t_2) \in [0, 0.5] \\ min(t_1, t_2), & F(t_1) \in [0.5, 1] \;,\; F(t_2) \in [0.5, 1] \\ t, & F(t_1) \in [0, 0.5] \;,\; F(t_2) \in [0.5, 1] \\ t, & F(t_1) \in [0.5, 1] \;,\; F(t_2) \in [0, 0.5] \end{cases}.$$

2.3 Uniform Error Method

This method equalizes the probability of misclassification in an image containing two classes. For a given threshold T, the estimate of the background area is $\alpha(T)$ and the estimate of the object area is $1-\alpha(T)$. In both the background and object areas, there will be specific fractions of pixels above and below the chosen threshold. Let $p(T)$, $q(T)$ denote the fraction of pixels that are "white", i.e., whose gray levels $>T$, in the background and in the object, respectively. Thus, for a given threshold T, $p(T)$ is the fraction of pixels in the background that are misclassified, and $1-q(T)$ is the fraction of pixels of the object that are misclassified. The uniform error threshold is the gray level T such that,

$$p(t) = 1 - q(t).$$

Defining the following three probabilities

$$a = \text{Prob}\{\text{one pixel be "white" i.e. pixel with a gray value} > \text{T}\},$$

$$b = \text{Prob}\{\text{two adjacent pixels are "white"}\}$$

$$c = \text{Prob}\{\text{four neighboring pixels are "white"}\}$$

we can write equations for a, b and c in terms of α, p and q:

$$a = \alpha \cdot p + (1-\alpha)\cdot q,\ b = \alpha \cdot p^2 + (1-\alpha)\cdot q^2,\ c = \alpha \cdot p^4 + (1-\alpha)\cdot q^4.$$

The uniform error threshold criterion is $p = 1 - q \underset{\phi = p+q}{\Leftrightarrow} \phi - 1 = 0$. Rewriting the equations to a, b and c we obtain:

$$\phi^2 = \frac{b^2 - c}{a^2 - b};$$

Dunn in [3], suggested an efficient algorithm that uses a single pass throughout the image to analyse all the 2×2 neighbourhoods. To find the uniform error threshold, the gray level t is selected such that $\phi = 1$. In practice, t is selected such that $|\phi - 1| < \varepsilon$ for some suitable small error ε .

2.4 Minimum Error Method

In the minimum error method, the gray level histogram is viewed as an estimate of the probability density function $p(g)$ of the mixture population comprising of the gray levels of the object and background pixels. It is usually assumed that each of the two components $p_i(g)$ of the mixture is normally distributed with mean m_i and standard deviation s_i and *a priori* probability P_i, that is, $p(g) = P_1 \cdot p_1(g) + P_2 \cdot p_2(g)$. The probability of misclassifying a pixel is

$$E(T) = P_2 \cdot E_1(T) + P_1 \cdot E_2(T) = P_2 \cdot \int_{-\infty}^{T} p_2(g)dg + P_1 \cdot \int_{T}^{\infty} p_1(g)dg$$

where T is the threshold, and E_1 is the probability of misclassifying an object pixel as belonging to the background, and E_2 is the probability of misclassifying a background pixel as belonging to the object . For a normal distribution, the minimization of this error gives rise to the equation:

$$\left(\sigma_1^{\,2}-\sigma_2^{\,2}\right)\cdot T^2+\left[2\cdot\left(\mu_1\cdot\sigma_2^{\,2}-\mu_2\cdot\sigma_1^{\,2}\right)\right]\cdot T+\left[\mu_2^{\,2}\cdot\sigma_1^{\,2}-\mu_1^{\,2}\cdot\sigma_2^{\,2}+2\cdot\sigma_1^{\,2}\cdot\sigma_2^{\,2}\cdot ln\left(\frac{\sigma_2\cdot P_1}{\sigma_1\cdot P_2}\right)\right]=0.$$

The solution of this quadratic equation will give the threshold T.

3. Evaluation of Thresholding Methods

In digital images, uniformity and shape measures used to evaluate the quality of the separation of objects and background. The four methods referred above will be compared using these measures.

3.1 Uniformity Measure

The uniformity measure was introduced by Levine & Nazif [4] to evaluate the performance of a segmentation method. The uniformity of a feature (e.g., gray value) over a region is inversely proportional to the variance of the values of that feature evaluated at every pixel belonging to that region. Suppose that an image is segmented into two regions by thresholding at gray level t. Then the uniformity measure is:

$$U(t)=1-\frac{\sigma_1^{\,2}+\sigma_2^{\,2}}{C_u\,(\text{normalization})}$$

$$\sigma_i^{\,2}=\sum_{(x,y)\in R_i}\left(f(x,y)-\mu_i\right)^2 \qquad \mu_i=\frac{\sum_{(x,y)\in R_i} f(x,y)}{A_i}$$

R_i - Region i A_i - number of pixels' in R_i $i\in\{1,2\}$

As normalization factor we used the maximum ratio that the variance can take in an image with N_T pixels, so it is

$$C_u=\sigma^2{}_{máx}=N_T\cdot\left(L_1-L_0\right)^2\cdot 0{,}25=N_T\cdot 256^2\cdot 0{,}25=128^2\cdot N_T.$$

L_1 is the maximum gray value, and L_0 the minimum gray value.

3.2 Shape Measure

The shape of an object in an image is also an important factor. In fact, the problem of finding optimal threshold for an image is a dual to the problem of finding edges or features related to the shape of an object in that image.

A measure that provides information regarding the shape (e.g., edges) of an object can be constructed as follows:

(i) assign a generalized gradient value to every pixel

$$\Delta(x,y)=\sqrt{\sum_{k=1}^{4} D_k^{\,2}+\sqrt{2}\cdot D_1\cdot\left(D_3+D_4\right)-\sqrt{2}\cdot D_2\cdot\left(D_3-D_4\right)}$$

$$D_1=f(x+1,y)-f(x-1,y); \qquad D_2=f(x,y-1)-f(x,y+1);$$

$$D_3=f(x+1,y+1)-f(x-1,y-1); \quad D_4=f(x+1,y-1)-f(x-1,y+1).$$

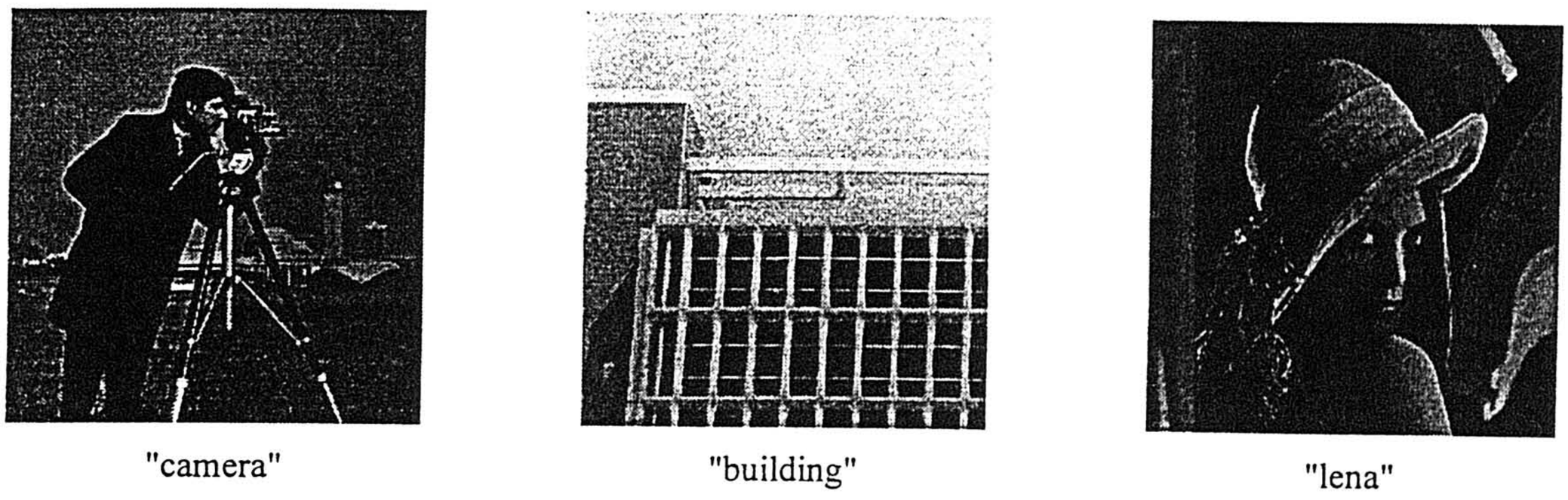
(ii) compute the shape measure using the formula

$$S(t) = \frac{\sum_{(x,y)} sgn\left(f(x,y) - \overline{f_{N(x,y)}}\right) \cdot \Delta(x,y) \cdot sgn\left(f(x,y) - t\right)}{C_s(\text{normalization})} \qquad sgn(x) = \begin{cases} 1 & ,x \geq 0 \\ -1 & ,x < 0 \end{cases}$$

$\overline{f_{N(x,y)}}$ - average gray value in the neighborhood $N(x,y)$ As normalization factor we used the number of pixels of an image N_T, i.e, $C_s = N_T$.

4. Evaluation Results

The four methods described above were evaluated, based on the results obtained for the two measures described. In table 1 the optimal gray level values for both measures are shown. Three images were used all with 256 gray levels, two of them with a resolution of 256 by 256 pixels ("camera", "building"), and the other with 512 by 512 pixels ("lena") (figure 1). The results are normalized in such a way that the maximum measure is one (Table 2), for the optimal conditions represented in table 1. The thresholded images are shown in figures 2, 3 and 4.



"camera" "building" "lena"

Figure 1: Images used to evaluate the threshold selection methods.

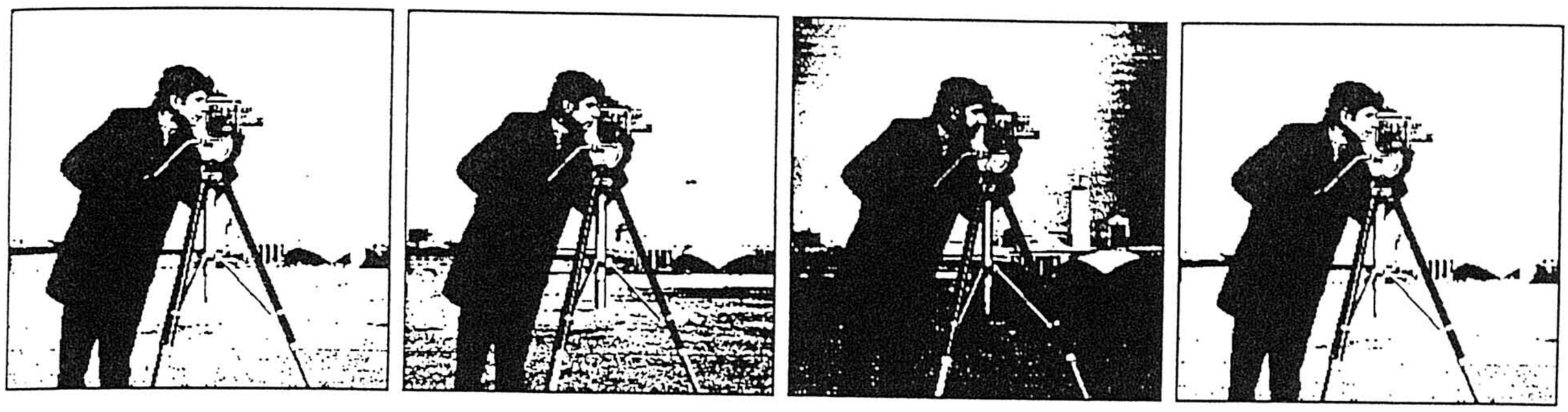
Table 1. Optimal gray values for both measures to all images.

	"camera"	"building"	"lena"
Optimal Uniformity	88	156	101
Optimal Shape	120	137	88

Table 2. Computed threshold values and measures values.

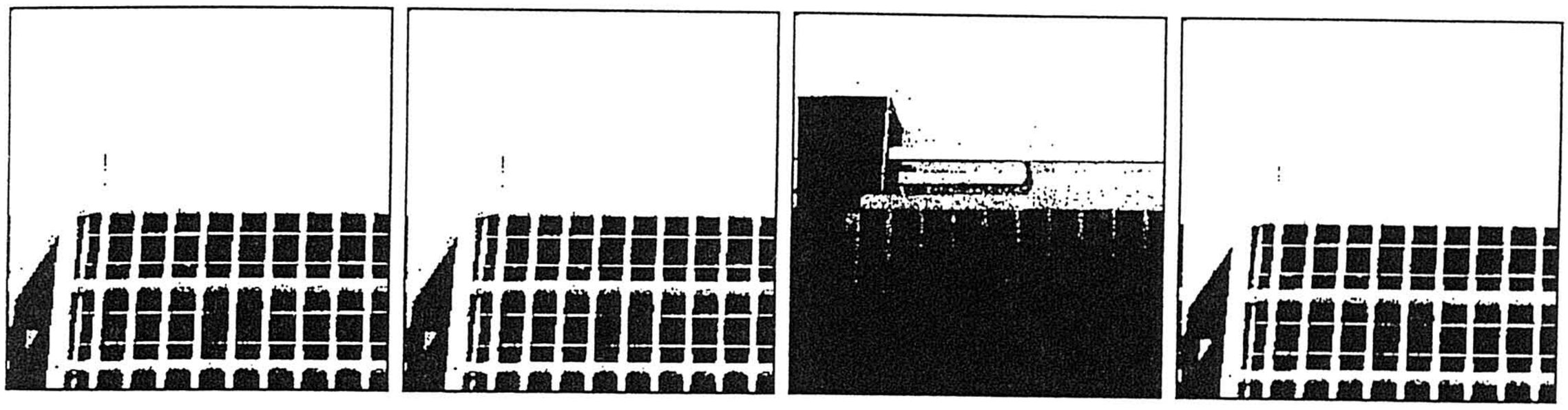
	"camera"			"building"			"lena"		
Method	Threshold	U(t)	S(t)	Threshold	U(t)	S(t)	Threshold	U(t)	S(t)
1	88	1.000000	0.860881	156	1.000000	0.932140	101	1.000000	0.967398
2	120	0.982241	1.000000	156	1.000000	0.932140	97	0.999744	0.987298
3	158	0.904799	0.635747	224	0.919914	0.127419	116	0.999577	0.873626
4	83	0.999824	0.851945	152	0.999735	0.976167	105	0.999760	0.976290

We may notice that the Otsu method (minimization of inter-classes variance) provides a threshold that corresponds to the maximum uniformity, and that the Maximum Entropy method because of uniformity and shape's measures, will provide in most of the images such a threshold that corresponds to the maximum uniformity or maximum shape. This method will be in most of the cases one of the best methods.



Otsu | Maximum Entropy | Uniform Error | Minimum Error

Figure 2: Binary image "camera", after thresholding



Otsu | Maximum Entropy | Uniform Error | Minimum Error

Figure 3: Binary image "building", after thresholding

Otsu | Maximum Entropy | Uniform Error | Minimum Error

Figure 4: Binary image "lena", after thresholding.

5. Conclusion

Relatively to the uniformity measure the best method is the Otsu method for all the three images. Relatively to the shape measure the best methods are the Minimum Error method to the image "building", and the Maximum Entropy

method to the images "camera" and "lena". The Uniform Error method doesn't give satisfactory results with these images, being adequate only to those images where the objects are big relatively to the background.

This study confirmed, to these test images, that the Otsu method is a good threshold selection method, and that the Minimum Error method give reasonable binary images, and also good results relatively to the measures used. Another method with good results is the Maximum Entropy method, being however the one with higher computer burden.

It is still an open question if the methods evaluated, will perform well or not, in general, with every kind of images, or even, if some methods eventually considered worst, will be better with other images.

References

[1] P. K. Sahoo, S. Soltani, A. K. C. Wong, and Y. Chen, "A survey of thresholding techniques", Computer, Vision, Graphics & Image Processing, vol. 41, pp. 233-260, 1988.

[2] A. K. C. Wong, P. K. Sahoo, "A gray-level threshold selection method based on maximum entropy principle", IEEE Transactions on Systems, Man, and Cybernetics, vol. 19, n° 4, pp. 866-871, 1989.

[3] S. M. Dunn, D. Harwood, and L. S. Davis, "Local estimation of the uniform error threshold", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-6, n° 6, pp. 742-747, 1984.

[4] M. D. Levine and A. M. Nazif, "Dynamic measurement of computer generated image segmentations", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-7, n° 6, pp. 155-164, 1985.

[5] F. Deravi and S. K. Pal, "Gray-level thresholding using second-order statistics", Pattern Recognition Letters, vol. 1, pp. 417-422, 1983.

[6] G. Johannsen and J. Bille, "A thresholding selection method using information measures", in Proceedings, 6th Int. Conf. on Pattern Recognition, Munich, Germany, pp. 140-143, 1982.

[7] J. N. Kapur, P. K. Sahoo and A. K. C. Wong, "A new method for gray-level picture thresholding using the entropy of the histogram", Computer, Vision, Graphics & Image Processing, vol. 29, pp. 273-285, 1985.

[8] N. Otsu, "A threshold selection method from gray level histogram", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. SMC-8, pp. 62-66, 1978.

[9] A. Tsai, "Moment-preserving thresholding: a new approach", Computer, Vision, Graphics & Image Processing, vol. 29, pp. 377-393, 1985.

[10] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison Wesley, 1992.

[11] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*, Academic Press, 1982.

A System for Image Analysis of Documents

Rui M. Lima[1,2,3], Aurélio J. C. Campilho[2,3]

[2]*Departamento de Engenharia Electrotécnica e de Computadores*
Faculdade de Engenharia da Universidade do Porto
Rua dos Bragas 4099 Porto Codex, Portugal
email: gpai@obelix.fe.up.pt
[3]*INEB - Instituto de Engenharia Biomédica, Praça Coronel Pacheco, 1, 4050 Porto Codex*
Email: campilho@garfield.fe.up.pt

Abstract- **For the analysis of document images the use of pre-processing procedures is mandatory. Examples of pre-processing techniques frequently use are: thresholding, morphologic operations, thinning, component labelling, identification and recognition of different types of blocks (e.g. text, lines, and graphics. In this paper, a software system is presented, for document image analysis. It can be used as a general purpose image processing system or as an engineering image document analysis system. In order to illustrate the application of this system for document analysis, a text identification procedure is presented and illustrated. This procedure is composed of two main steps: region labelling, and characterisation of each connected component, allowing the separation of the image into different classes, such as, text and lines, specially vertical and horizontal lines.**

I. Introduction

Most of the systems developed for image analysis are driven by a specific application. As a consequence, they have several development restrictions in their organisation. The system described in this paper is intended to be a general purpose system, in spite of having in mind a specific application for document analysis. For this purpose, a special option was created, where the main procedures for image document analysis was introduced. This system is based on Microsoft Windows graphical environment. It is a menu driven system with several general purpose options for image manipulation, image visualisation in a multi-window environment, image processing and analysis, and a document analysis oriented option.

The image document option includes several procedures to process engineering drawings, namely: text identification; line classification using line segments, arcs and splines. Finally, a global line description is derived and saved as a text file.

In this extended abstract, the general system organisation and a special procedure for text identification in engineering documents are briefly presented.

II. System Organisation

The application has been developed under *Microsoft Visual C++* using the four main classes: the *mainframe*, the *document*, the *window*, and the *view* classes. The first manages all elements of the global application window, as the borders, title,

[1] Work supported by the grant BM/3537/92-IA from Programa CIÊNCIA

and menus. The *document* class, controlling the information needed by the application, allows the use of multiple documents, because it is a MDI ('Multiple Document Interface') application [15, 3, 11]. All documents are of the same type, a bitmap image, that can be processed independently, or combined with other, depending on the processing operation. The *window* class manages the child windows associated with each document. Finally, the view *class* controls the way the document is viewed; it is attached to the child window workspace, controlling the image appearance, all or only a part, and allowing several operations namely: image scroll, zooming, palette selection, printing control and preview.

Basically an image analysis system must load images to memory, show them on a monitor, process them under operator control, and finally store the image results. To implement these operations, it is necessary to define a suitable image format, which, in sake of portability, must be independent of video resources, with their resolution and colour differences. Windows 3.1 offers an image format with those capabilities, which is the DIB ("Device Independent Bitmap") format [6], and a memory representation of BMP format [16]. This image format allows the manipulation of monochrome bitmaps (1 bit per pixel), colour bitmaps ranging from representations with 4 bits and 8 bits per pixel to the true colour bitmaps with 24 bits per pixel (8 per colour channel).

This system is a menu driven system (figure 1) having the following options, developed so far:

'File' : under the 'File' menu option, images in BMP (or DIB) format can be loaded, each one to a child window, and can be saved in the same format. A text file can be saved describing line segments and three point arcs. The image can be printed. Image

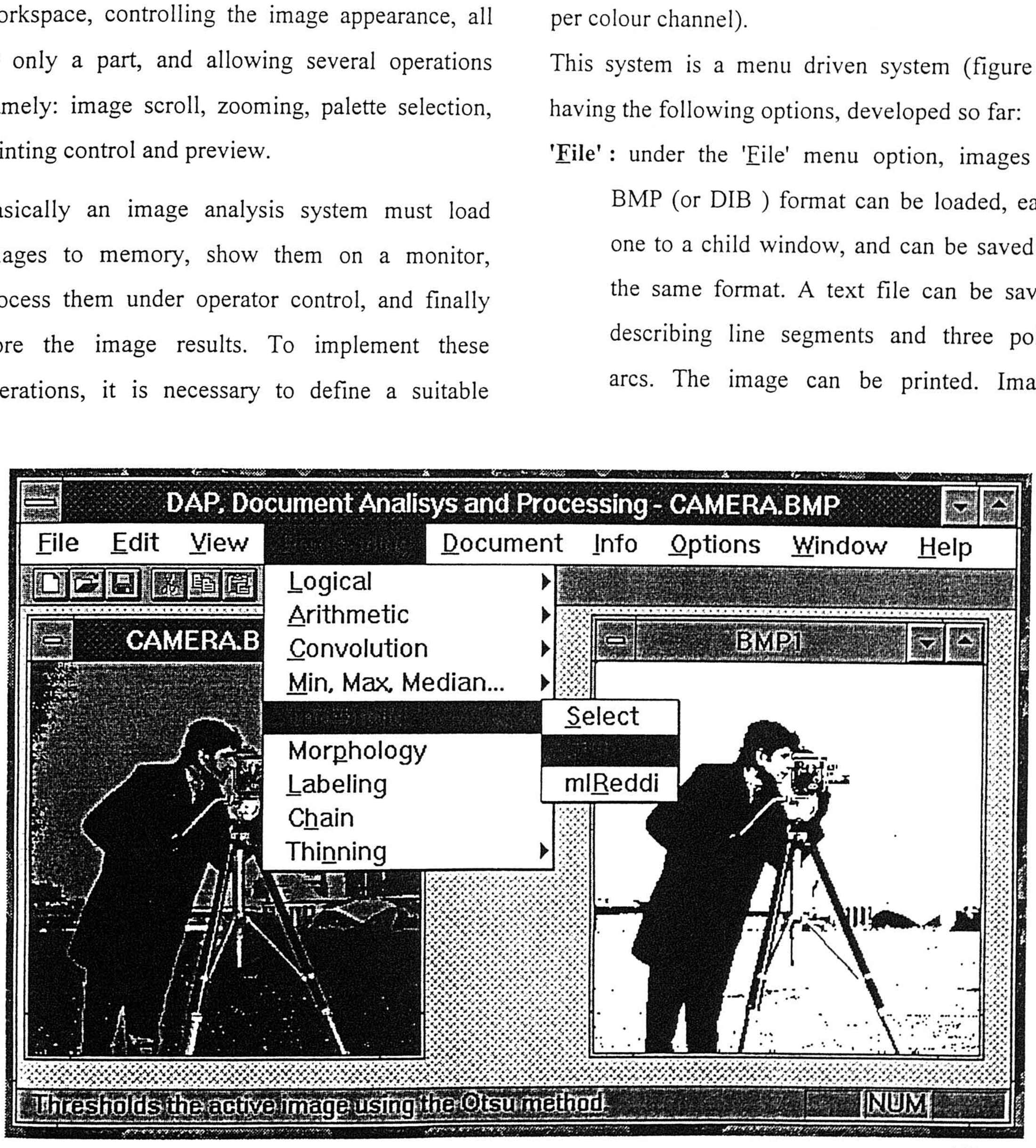


Figure 1: The system interface window.

preview and printer set-up are also available.

'Edit': the 'Edit' menu option allows, through the clipboard, the standard copy and paste operations between child windows, or between different applications. The definition of a rectangular area of interest, using a mouse is also available for controlling the copy area.

'View': the 'View' menu option controls the visualisation of the tool bar, and the status bar, with help indications for the current operation.

'Info': using the 'Info' menu option the user can obtain information about the image, like the file name, image resolution, number of colours, and compression type. It is also possible to obtain a bitmap representation of image's histogram, with indication of minimum and maximum of the gray levels present in the image.

'Options': the 'Options' menu allows the definition of convolution kernels, and the definition of the active child's image, in two-operand processing operations. Parameters needed to polygonal, arc and spline approximation of line drawings can be set in this option.

'Processing': in this option several standard image analysis and processing operations are available, namely: logical and arithmetic operations, convolution with an user defined kernel or a pre-defined kernel (e.g. Sobel operator, laplacian operator, LoG filters); non-linear operations (e.g. maximum, minimum, and median filters); morphological operators (dilation, erosion, open, close, hit-and-miss both on gray level and binary images, and image thinning and pruning); thresholding (using Otsu and Reddi methods).

'Document': this is the application oriented option for document analysis; several procedures are implemented, such as: identification of text regions on engineering drawings; corner detection by "*cos*" [21] and by a "*b-spline*" [12] method; fast polygonal method [19] to polygonal approximation; arc approximation based on a three arc drawing algorithm [10].

III. Identification of Text on Engineering Drawings

Engineering documents are usually composed by several types of elements, such as lines, arcs, different kinds of symbols, and text regions. As referred in the previous section an option for detection and delineation of text regions was developed. For this purpose, several methods have been proposed. Casey & Wong [8], Bow & Kasturi [7] and Ejiri et al. [9] have used block segmentation techniques based on component labelling [14]; 'Run-Length Smearing' (RLS) is a top-down method described by Wong, Casey & Wahl [17] and applied on images with pictures and large blocks of text. Methods based on classification techniques using several features, like width, height, and area, were referred by Wong, Casey & Wahl [20] and Bow & Kasturi [7]. A method based on a shape measure was referenced by Casey & Wong [8]. A method used by Bow & Kasturi [7] is based on the Hough Transform [14].

In order to produce a vectorial description of engineering documents, we have considered the following main steps:

1. Binarization using Otsu method;

2. Region labelling identifying each connected component, and to each one an area ratio parameter is computed:

$$R_a = a_{comp} / a_{rect} ,$$

where a_{comp} is the number of black pixels, and a_{rect} is the area of the involving rectangle;

3. Text identification;

Figure 2 illustrates the text identification method, showing the value of individual R_a for each connected component of real image (fig. 4a). We can clearly observe that three regions are identifiable: line drawings region, text region, and horizontal and vertical line segments region.

The overall result is illustrated on synthetic image (fig. 3a-c) and on real image (fig 4a-c), showing that a good separation between regions is obtained.

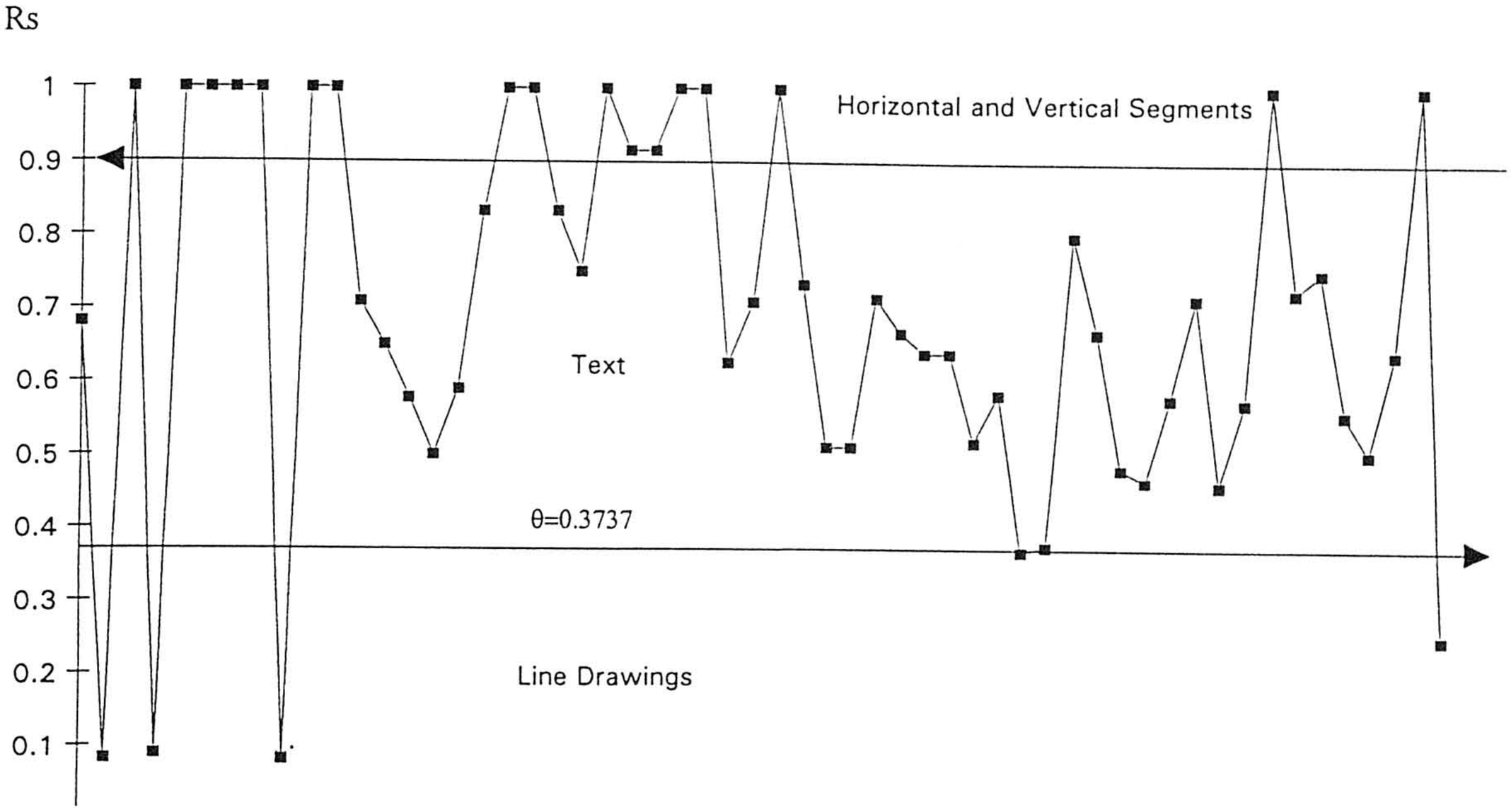


Figure 2: Area ratio parameter value associated to each connected component

4. Line description and vectorization.

We may notice that the R_a parameter is usually greater for text components, than for line drawings, with the exception of horizontal or vertical lines, and symbols that can be treated as characters for recognition purposes. So, a region is considered as text if the parameter R_a, satisfies the condition, $\theta < R_a < 0.9$, and θ is the threshold computed by the Otsu method [14], when applied to the R_a histogram; and is considered as a region of horizontal, vertical lines, or isolated black regions for $R_a > 0.9$.

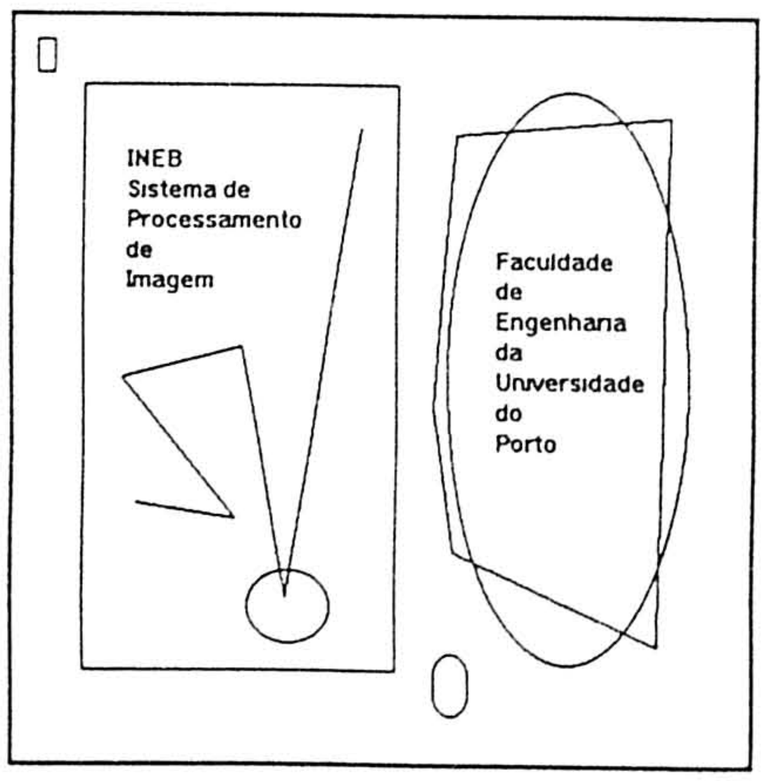


Figure 3a: Synthetic image.

Figure 3b: Synthetic image - Text extraction.

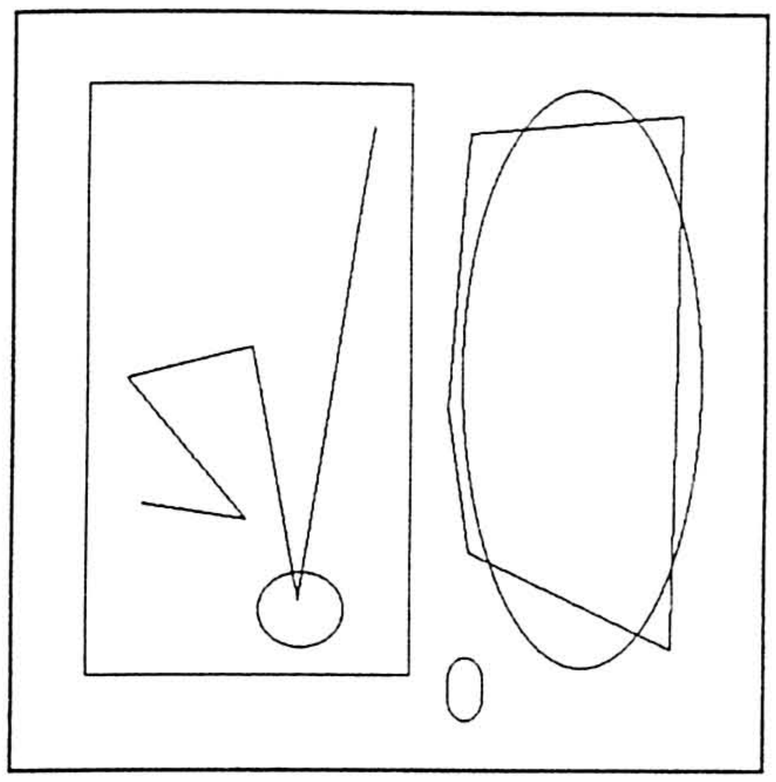


Figure 3c: Synthetic image. Line drawing extraction.

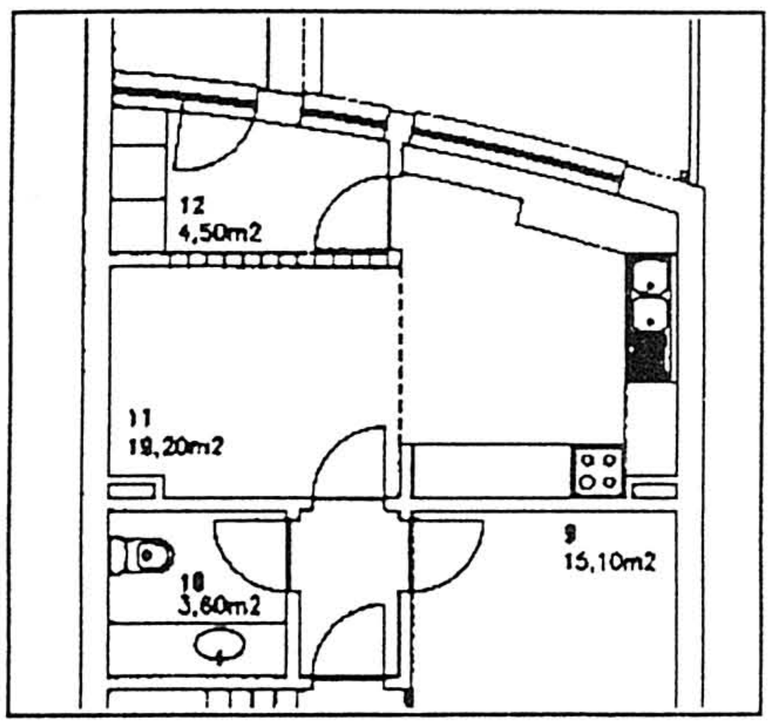


Figure 4a: "Real" image.

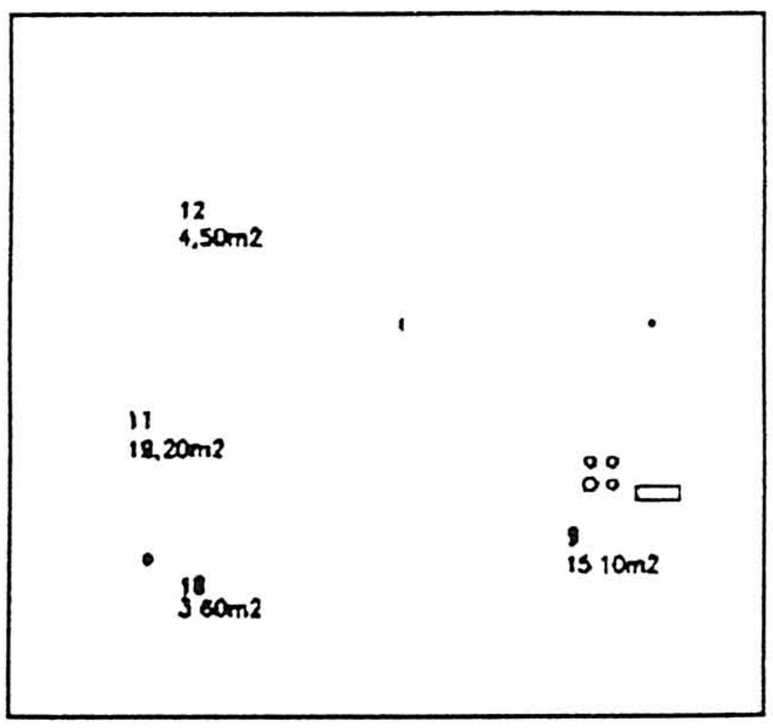


Figure 4b: Real image - Text extraction.

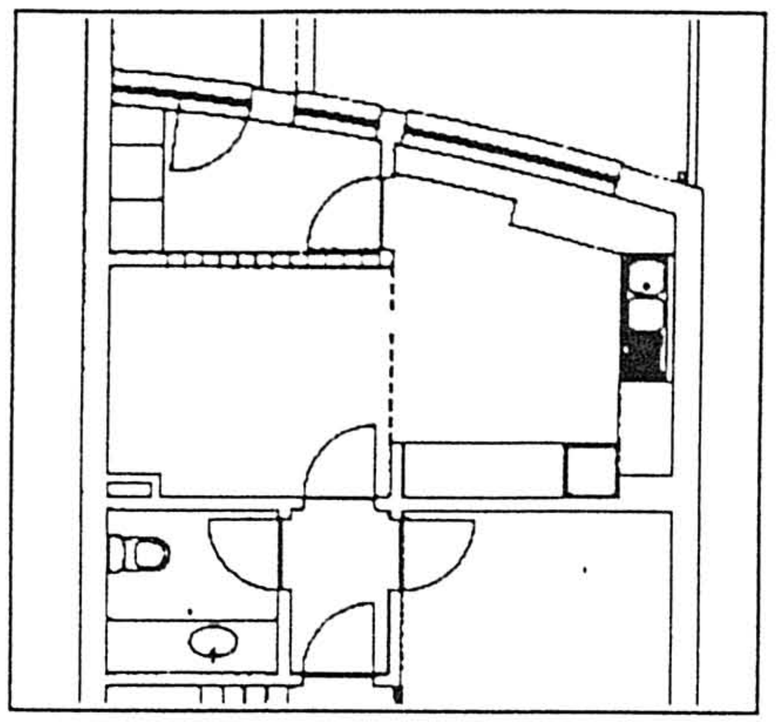


Figure 4c: Real image - Line drawing extraction.

IV. Conclusion

Keeping in mind that our ultimate goal was to process graphic lines of engineering drawings, in order to export a description to a CAD system, we developed a procedure to separate text blocks from line drawings blocks. This procedure as was illustrated, and analysed, showed good results. It seems to be a well behaved procedure even when using other images.

Besides, the software system presented is of great interest, both to newcomers to the processing image field, or to more advanced researchers in the field.

References

[1] Adler, Marc; "Learning Windows Part I: The Message Based Paradigm"; Microsoft Systems Journal; July/1990; pp. 85-91.

[2] Adler, Marc; "Learning Windows Part II: Resources and Menuing System"; Microsoft Systems Journal; September/1990; pp. 75-90.

[3] Adler, Marc; "Learning Windows Part III: Control Windows and MDI Support"; Microsoft Systems Journal; November/1990; pp. 67-85.

[4] Adler, Marc; "Learning Windows Part IV: Integrating Controls and Dialog Boxes"; Microsoft Systems Journal; January/1991; pp. 97-118.

[5] Adler, Marc; "Learning Windows Part V: Exploring the Graphics Device Interface"; Microsoft Systems Journal; March/1991; pp. 75-88.

[6] Adler, Marc; "Learning Windows Part VI: Fonts, Bitmaps and Printing"; Microsoft Systems Journal; May/1991; pp. 115-131.

[7] Bow, Sing-tze & Kasturi, Rangachar; "2. A Graphics-Recognition System for Interpretation of Line Drawings" from *Image Analysis Applications*, edited by Rangachar Kasturi and Mohan M. Trivedi; Marcel Dekker, Inc.; 1991; pp. 37-71.

[8] Casey, Richard G. & Wong, Kwan Y.; "1. Document-Analysis Systems and Techniques" from *Image Analysis Applications*, edited by Rangachar Kasturi and Mohan M. Trivedi; Marcel Dekker, Inc.; 1991; pp. 1-34.

[9] Ejiri, Masakasu & Kakumoto, Shigeru & Miyatake, Takafumi & Shimada, Shigeru & Iwamura, Kasuaki; "3. Automatic Recognition of Engineering Drawings and Maps" from *Image Analysis Applications*, edited by Rangachar Kasturi and Mohan M. Trivedi; Marcel Dekker, Inc.; 1991; pp. 73-126.

[10] Galton, Ian; "An Efficient Three-Point Arc Algorithm"; CGA; November 1989; pp.44-49.

[11] Geary, Michael; "An Introduction to Microsoftâ Windows Version 3.0: A Developer's Viewpoint"; Microsoft Systems Journal; July/1990; pp. 01-28.

[12] Medioni, Gerard and Yasumoto, Yoshio; "Corner Detection and Curve Representation Using Cubic B-Splines"; CVGIP vol. 39, n. 3; March 1987; pp. 267-278.

[13] Nagasamy, Vijay & Langrana, Noshir A.; "Engineering Drawing Processing and Vectorization System"; CVGIP 49; 1990; pp. 379-397.

[14] Otsu, N.; "A threshold selection method from gray level histogram", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. SMC-8, pp. 62-66, 1978.

[15] Petzold, Charles; "A New Multiple Document Interface Simplifies MDI Application Development"; Microsoft Systems Journal; July/1990; pp. 53-63.

[16] Rimmer, Steve; *Supercharged Bitmapped Graphics*; Windcrest/McGraw-Hill; Chapter 5; 1992; pp.163-211.

[17] Rogerson, Dale; "Allocating Memory the Old-Fashioned Way: _fmalloc and Applications for Windows"; CDRom; July/1992.

[18] Rosenfeld, Azriel & Kak, Avinash C.; *Digital Picture Processing*; Academic Press; 1982.

[19] Wall, Danielsson; "A Fast Sequential Method for Polygonal Approximation of Digitized Curves"; CVGIP 28; 1984; pp. 220-227.

[20] Wong, K. Y. & Casey, G. R. & Wahl, F. M.; "Document Analysis System"; IBM JRD 26 (6); 1982; pp. 647-656.

[21] Wu, Wang; "Detecting the Dominant Points by the Curvature-Based Polygonal Approximation"; CVGIP 55 (2); 3/1993; pp. 79-88.

[22] Yao, Paul; "Windows 3.0 Memory Menagement: Supporting Disparate 80x86 Architectures"; Microsoft Systems Journal; November/1990; pp. 35-43.

REFERÊNCIAS BIBLIOGRÁFICAS

REFERÊNCIAS BIBLIOGRÁFICAS

Neste capítulo, as referências bibliográficas estão agrupadas por livros e por publicações, estando as publicações agrupadas por assunto. Em cada grupo de referências foi efectuada uma ordenação pelo nome do autor. Todas as publicações, efectivamente referidas no texto ou que serviram de inspiração (marcadas com o símbolo †) para o desenvolvimento do trabalho são apresentadas.

LIVROS

[1] Gonzalez, Rafael C. and Woods, Richard E.; *Digital Image Processing*; Addison-Wesley Publishing Company; 1987.
[2] Hearn, Donald and Baker, M. Pauline; *Computer Graphics*; Prentice-Hall; 1986.
[3] Plastock, Ray A. and Kalley, Gordon; *Computação Gráfica*; Editora McGraw-Hill de Portugal; 1991.
[4] Rimmer, Steve; *Supercharged Bitmapped Graphics*; Windcrest / McGraw-Hill; 1992.
[5] Rosenfeld, Azriel and Kak, Avinash C.; *Digital Picture Processing*; Academic Press; 1982.
[6] Schalkoff, Robert J.; *Digital Image Processing and Computer Vision*; John Wiley, Son's; 1989.

PUBLICAÇÕES

C&G - Computer Graphics
CGA - IEEE Computer Graphics and Applications
CVGIP - Computer, Vision, Graphics and Image Processing
GMIP - Graphical Models and Image Processing
PR - Pattern Recognition
T-PAMI - IEEE Transactions on Pattern Analysis and Machine Intelligence
IBM JRD - IBM Journal of Research and Development

WINDOWS

[7] Adler, Marc; "Learning Windowsä Part I: The Message Based Paradigm"; Microsoft Systems Journal; July/1990; pp. 85-91.
[8] Adler, Marc; "Learning Windows™ Part II: Resources and Menuing System"; Microsoft Systems Journal; September/1990; pp. 75-90.
[9] Adler, Marc; "Learning Windows™ Part III: Control Windows and MDI Support"; Microsoft Systems Journal; November/1990; pp. 67-85.
[10] Adler, Marc; "Learning Windows™ Part IV: Integrating Controls and Dialog Boxes"; Microsoft Systems Journal; January/1991; pp. 97-118.
[11] Adler, Marc; "Learning Windows™ Part V: Exploring the Graphics Device Interface"; Microsoft Systems Journal; March/1991; pp. 75-88.
[12] Adler, Marc; "Learning Windows™ Part VI: Fonts, Bitmaps and Printing"; Microsoft Systems Journal; May/1991; pp. 115-131.
[13] Geary, Michael; "An Introduction to Microsoft® Windows™ Version 3.0: A Developer's Viewpoint"; Microsoft Systems Journal; July/1990; pp. 01-28.
[14] Petzold, Charles; "A New Multiple Document Interface Simplifies MDI Application Development"; Microsoft Systems Journal; July/1990; pp. 53-63.
[15] Rogerson, Dale; "Allocating Memory the Old-Fashioned Way: _fmalloc and Applications for Windows™"; CDRom; July/1992.
[16] Yao, Paul; "Windows™ 3.0 Memory Menagement: Supporting Disparate 80x86 Architectures"; Microsoft Systems Journal; November/1990; pp. 35-43.

ANÁLISE DE DOCUMENTOS

[17] Antonacopoulos, A. and Ritchings, R. T.; "Flexible Page Segmentation Using the Background"; Proceedings of the 12th IAPR International Conference on Pattern Recognition; 1994; pp. 334-344.

[18] Bow, Sing-tze and Kasturi, Rangachar; "A Graphics-Recognition System for Interpretation of Line Drawings" from *Image Analysis Applications*, edited by Rangachar Kasturi and Mohan M. Trivedi; Marcel Dekker, Inc.; 1991; pp. 37-71.

[19] Casey, Richard G. and Wong, Kwan Y.; "Document-Analysis Systems and Techniques" from *Image Analysis Applications*, edited by Rangachar Kasturi and Mohan M. Trivedi; Marcel Dekker, Inc.; 1991; pp. 1-34.

[20] Ejiri, M., Kakumoto, S., Miyatake, T., Shimada, S. and Iwamura, K.; "Automatic Recognition of Engineering Drawings and Maps" from *Image Analysis Applications*, edited by Rangachar Kasturi and Mohan M. Trivedi; Marcel Dekker, Inc.; 1991; pp. 73-126.

[21] Krile, Thomas F. and Walkup, John F.; "Enhancement of Fingerprints Using Digital and Optical Techniques" from *Image Analysis Applications*, edited by Rangachar Kasturi and Mohan M. Trivedi; Marcel Dekker, Inc.; 1991; pp. 343-371.

[22]† Lima, R. and Campilho, A.; "A System for Image Analysis"; Proceedings of the 7th Portuguese Conference on Pattern Recognition; Aveiro, Portugal; 1995; pp. .

[23] Lima, R. and Campilho, A.; "Thresholding Selection Methods: a Comparative Study"; Proceedings of the 6th Portuguese Conference on Pattern Recognition; Lisboa, Portugal; 1994; pp. 235-242.

[24] Nagasamy, Vijay and Langrana, Noshir A.; "Engineering Drawing Processing and Vectorization System"; CVGIP 49; 1990; pp. 379-397.

[25]† Tanigawa, S., Hori, O. and Shimotsuji, S.; "Precise line detection from an engineering drawing using a figure fitting method based on contours and skeletons"; Proceedings of the 12th IAPR International Conference on Pattern Recognition; 1994; pp. 356-360.

[26]† Wang, D. and Srihari, Sargur N.; "Classification of Newspaper Image Blocks Using Texture Analysis"; CVGIP vol. 47, n. 3; September 1989; pp. 327-352.

[27] Wong, K. Y., Casey, G. R. and Wahl, F. M.; "Document Analysis System"; IBM JRD 26 (6); 1982; pp. 647-656.

SURVEYS

[28] Nadler, Morton; "Survey, Document Segmentation and Coding Techniques"; CVGIP 28; 1984; pp. 240-262.

[29]† Rosenfeld, Azriel; "Survey, Image Analisys and Computer Vision: 1992"; CVGIP 55 (3); 5/1992; pp. 349-380.

FILTROS 'RANKING'

[30]† Gil, Werman; "Computing 2D Min, Median and Max Filters"; PAMI 15 (5); 5/1993; pp. 504-505

[31]† Nie, Unbehanen; "Edge Preserving Filtering by Combining Nonlinear Mean and Median Filters"; T-SP 39; 1991; pp. 2552-2554.

LIMIARIZAÇÃO

[32] Otsu, N.; "A threshold selection method from gray level histogram", IEEE T-PAMI, vol. 8; 1978; pp. 62-66.

[33] Sahoo, P. K., Soltani, S., Wong, Andrew K. C. and Chen, Y. C.; "A Survey of Thresholding Techniques"; CVGIP vol. 41; 1988; pp. 233-260.

[34] Wong, Andrew K. C. and Sahoo, P. K. ; "A Gray-Level Threshold Selection Method Based on Maximum Entropy Principle"; IEEE Transactions on Systems, MAN, and Cybernetics vol. 19, n. 4; July/August 1989; pp. 866-871.

OPERAÇÕES MORFOLÓGICAS / ÁLGEBRA DA IMAGEM

[35] Boomgaard, R. and Balen, R.; "Methods for Fast Morphological Image Transforms Using Bitmapped Binary Images"; CVGIP 54 (3); 5/1992; pp. 252-258

[36]† Crabtree, Yuan and Ehrlich; "A Fast and Accurate Erosion - Dilation Method Suitable for Microcomputers"; GMIP 53; 1991; pp. 283-290.

[37]† Schonfeld, G.; "Optimal Morphological Pattern Restoration from Noisy Binary Images"; T-PAMI 13; 1991; pp. 14-29.

VECTORIZAÇÃO / INTERPOLAÇÃO

[38] Albano, A.; "Representation of Digital Contours in terms of conic arcs and straight line segments"; CGIP 2; 1974; pp. 23-33.

[39] Alia, G., Barsi, F., Martinelli, E. and Tani, N.; "Angular Spline: A New Approach to the Interpolation Problem in Computer Graphics"; CVGIP vol. 39, n. 1; January 1987; pp. 56-72.

[40]† Barry, G.; "Interpolation and Aproximation of Curves and Surfaces Using Pólya Polynomials"; CVGIP 53 (2); 3/1991; pp. 137-148.

[41]† Barsky, B. and DeRose, T.; "Geometric Continuity of Parametric Curves: Three Equivalent Characterizations"; CGA; 11/1989; pp. 60-68.

[42]† Barsky, B. and DeRose, T.; "Geometric Continuity of Parametric Curves: Construction of Geometrically Continuous Splines"; CGA; 1/1990; pp. 60-68.

[43] Galton, Ian; "An Efficient Three-Point Arc Algorithm"; CGA; November 1989; pp.44-49.

[44]† Goshtasby, Cheng and Barsky; "B-Splines Viewed as Digital Filters"; CVGIP 52; 1990; pp. 264-275.

[45]† Hakimi, S.; "Fitting Polygonal Functions to a Set of Points in the Plane"; CVGIP 53 (2); 3/1991; pp. 132-136.

[46] Leung, M. and Yang, Yee-Hong; "Dinamic Strip Algorithm in Curve Fitting"; CVGIP 51; 1990; pp. 146-165.

[47]† Lin, Schunck and Meyer; "Optimal Contour Approximation by Deformable Piecewise Cubic Splines"; CVPR ; 1991; pp. 638-643.

[48]† Mamrak, S., O'Connell, C. and Parent, R.; "The Automatic Generation of Translation Software for Graphic Objects"; CGA; 9/1989; pp. 34-42.

[49] Medioni, G. and Yasumoto, Y.; "Corner Detection and Curve Representation Using Cubic B-Splines"; CVGIP vol. 39, n. 3; March 1987; pp. 267-278.

[50]† Pham, Binh; "Conic B-Splines for Curve Fitting: A Unifying Approach"; CVGIP vol. 45, n. 1; January 1989 pp. 117-125.

[51]† Piegl, L. and Tiller, W.; "A Menagerie of Rational B-Spline Circles"; CGA; 9/1989; pp. 48-56.

[52]† Rabert ; "Even Degree B-Spline Curves and Surfaces"; CVGIP 54 (4); 6/1992; pp. 351-356.

[53] Thomas, Samuel M.; "A Simple Approach for the Estimation of Circular Arc Center and Its Radius"; CVGIP vol. 45, n. 3; March 1989; pp. 362-370.

[54] Wall, Danielsson; "A Fast Sequential Method for Polygonal Approximation of Digitized Curves"; CVGIP 28; 1984; pp. 220-227.

[55]† Wang, Xu; "The Termination Criterion for Subdivision of the Rational Bézier Curves"; CVGIP 53 (1); 1/1991; pp. 93-96.

[56]† Yates Fletcher, David F. McAllister; "Automatic Tension Adjustment for Interpolatory Splines"; CGA; 1/1990; pp. 10-17.

ADELGAÇAMENTO

[57] Brandt, J. and Algazi, V.; "Continuous Skeleton Computation by Voronoi Diagram"; CVGIP 55 (3); 5/1992; 329-338.

[58] O'Gorman, Lawrence; "KxK Thinning"; CVGIP 51; 1990; pp. pp. 195-215.

CURVAS, REPRESENTAÇÃO E TOPOLOGIA

[59] Ansari, Delp; "On Detecting Dominant Points"; PR 24; 1991; pp. 441-451.

[60]† Chandler; "A Recursive Tecnique for Rendering Parametric Curves"; C&G 14; 1990; pp. 477-479.

[61]† Dudek, Tsotsos; "Shape Representation and Recognition from Curvature "; CVPR ; 1991; pp. 28-34.
[62]† Giraudon, Deriche; "On Corner and Vertex Detection"; CVPR; 1991; pp. 650-655.
[63]† Lee, Poston and Rosenfeld; "Representation of Orthogonal Regions by Vertices"; CVGIP 53 (2); 3/1991; pp. 149-156.
[64]† Subirana-Vilanova; "On Contour Texture"; CVPR; 1991; pp. 753-754.
[65]† West, Rosin; "Tecniques for Segmenting Image Curves into Meaningful Descriptions"; PR 24; 1991; pp. 643-652.
[66] Wu, Wen-Yen and Wang, Mao-Jiun; "Detecting the Dominant Points by the Curvature-Based Polygonal Approximation"; CVGIP 55 (2); 3/1993; pp. 79-88.
[67]† Wuescher, Boyer; "Robust Contour Decomposition Using a Constant Curvature Criterion"; T-PAMI 13; 1991; pp. 41-51.

CAD
[68] Manual de utilizador do AutoCAD, versão 11, da AutoDesk; Anexo C; pp. 521-559.