Campus de São Carlos

"RECONHECIMENTO ÓPTICO DE CARACTERES ISOLADOS BASEADO
EM SUAS CARACTERÍSTICAS FUNDAMENTAIS"

Autora: EVELINA MARIA DE ALMEIDA NEVES

Orientador: Prof. Dr. Adilson Gonzaga

UNIVERSIDADE DE SÃO PAULO

**ESCOLA DE ENGENHARIA
DE SÃO CARLOS**

RECONHECIMENTO ÓPTICO DE CARACTERES
ISOLADOS BASEADO EM SUAS
CARACTERÍSTICAS FUNDAMENTAIS

DEDALUS - Acervo - EESC

31100016582

EVELINA M. ALMEIDA NEVES

Dissertação apresentada à Escola de Engenharia de São Carlos, da Universidade de São Paulo, como parte dos requisitos para obtenção do Título de Mestre em Engenharia Elétrica.

ORIENTADOR: Prof. Dr. Adilson Gonzaga

São Carlos

1995

Class.	Tese-EESC
Crtl.	3929
Tombo	079/95

Eletricidade

st 0742 096

**Ficha catalográfica preparada pela Seção de Tratamento da Informação do
Serviço de Biblioteca-EESC-USP**

N518r Neves, Evelina Maria de Almeida
Reconhecimento óptico de caracteres isolados
baseado em suas características fundamentais /
Evelina Maria de Almeida Neves. -- São Carlos,
1995.
191p.

Dissertação (Mestrado) -- Escola de Engenharia
de São Carlos-Universidade de São Paulo, 1995.
Orientador: Adilson Gonzaga

1. Reconhecimento de caracteres. I. Título

FOLHA DE APROVAÇÃO

Dissertação defendida e aprovada em 17/3/1995
perante a Comissão Julgadora:

Prof. Dr. ADILSON GONZAGA - Orientador
(Escola de Engenharia de São Carlos - USP)

Profa. Dra. AGMA JUCI MACHADO TRAINA
(Instituto de Ciências Matemáticas de São Carlos - USP)

Profa. Dra. MARIA STELA VELUDO DE PAIVA
(Escola de Engenharia de São Carlos - USP)

Presidente da Comissão de Pós-Graduação
Prof. Dr. EDUARDO CLETO PIRES

Coordenador da área - Engenharia Elétrica
Prof. Dr. AMILCAR CARELI CESAR

Ao Márcio e Yasmim, com amor.

AGRADECIMENTOS

Ao prof. Adilson Gonzaga pelo incentivo, amizade e excelente orientação.

Aos professores e funcionários do departamento de Engenharia Elétrica, em especial à Rosane, pela consideração em todos os momentos.

Ao amigo Celso pela inestimável ajuda, relativa a programação em linguagem C, meus sinceros agradecimentos.

Aos amigos: Anette, Belém, Dulce, Edson, Ernany, Evandro, Glauco, Jerson, Kity, Marcos, Sérgio, Tainha e Tião, pelos momentos que passamos juntos.

À minha família, pelo apoio.

Em especial, meus agradecimentos ao Durval, Beatriz, Eduardo e Maurício, pelos momentos de reflexão, frutos da harmonia, equilíbrio e beleza da Fazenda Paraizo.

SUMÁRIO

LISTA DE FIGURAS	i
LISTA DE TABELAS	x
RESUMO	xi
ABSTRACT	xii
CAPITULO 1 - RECONHECIMENTO ÓPTICO DE CARACTERES MULTI-FONTE	1
1 - INTRODUÇÃO	1
2 - OBJETIVOS	2
3 - FORMATO DA DISSERTAÇÃO	2
CAPITULO 2 - SISTEMAS DE RECONHECIMENTO DE CARACTERES	4
1 - INTRODUÇÃO	4
2 - HISTÓRICO	5
3 - CLASSIFICAÇÃO DOS SISTEMAS DE RECONHECIMENTO DE CARACTERES.....	8

4 - PRINCIPAIS PARÂMETROS DETERMINANTES DA QUALIDADE DE UM SISTEMA DE OCR	12
4.1 - Taxa de erros permissível	12
4.2 - Volume de dados envolvidos	14
4.3 - Formato do documento a ser lido	15
4.4 - Qualidade de impressão e da imagem	16
4.5 - Tipos de caracteres a serem reconhecidos	18
4.5.1 - Tipos estilizados, base de dados e restrições impostas (<i>constraints</i>)	19
4.5.2 -Textos datilografados	23
4.5.3 - Textos impressos	24
4.5.4 - Escrita à mão	25
4.5.4.1 - Caracteres manuscritos isolados	27
4.5.4.2 - Escrita cursiva	29
4.5.5 - Alfabetos especiais	30

5 - COMPARAÇÃO ENTRE SISTEMAS DE RECONHECIMENTO ON-LINE E OFF-LINE	30
6 - APLICAÇÕES DE RECONHECIMENTO DE CARACTERES	34
7 - SISTEMAS COMERCIAIS DISPONÍVEIS NO MERCADO	37
CAPÍTULO 3 - AQUISIÇÃO E PRÉ-PROCESSAMENTO DA IMAGEM....	41
1 - SCANNERS UTILIZADOS NA AQUISIÇÃO DA IMAGEM	41
2 - IMPLICAÇÕES NO USO DE SCANNERS	43
3 - TÉCNICAS DE ARMAZENAMENTO DAS IMAGENS	46
4 - TAMANHO TÍPICO DAS MATRIZES DOS CARACTERES BINARIZADOS	49
5 - PRÉ-PROCESSAMENTO DA IMAGEM	50
5.1 - Segmentação de textos	52
5.2 - Segmentação vertical	54
5.3 - Segmentação horizontal	55
5.4 - Técnicas de suavização (<i>smoothing</i>)	58

5.5 - Técnicas de afinamento	59
5.6 - Técnicas de normalização	62
CAPÍTULO 4 - METODOLOGIAS DE RECONHECIMENTO.....	64
1 - EXTRAÇÃO DE CARACTERÍSTICAS	64
2 - TÉCNICAS DE EXTRAÇÃO DE CARACTERÍSTICAS	65
2.1 - Características Globais	66
2.1.1 - <i>Template matching</i> e correlações	66
2.1.2 - Transformações e expansões em séries	69
2.2 - Distribuição de pontos	69
2.2.1 - Zonificação (<i>zoning</i>)	70
2.2.2 - Distância	71
2.2.3 - <i>N-tuples</i>	71
2.2.4 - Características locais (<i>characteristic loci</i>) ...	72
2.2.5 - Cruzamentos	73

2.2.6 - Outras técnicas	74
2.3 - Características geométricas e topológicas	74
2.3.1 - Obtenção de segmentos de linhas e bordas	75
2.3.2 - Traçado do contorno do caracter (<i>contour tracing</i>)..	75
2.3.3 - Determinação do eixo médio (<i>centre-line</i>)	77
3 - ANÁLISE DAS TÉCNICAS DE EXTRAÇÃO DE CARACTERÍSTICAS..	78
4 - RECONHECIMENTO DE CARACTERES	81
4.1 - Métodos matemáticos para Reconhecimento de Padrões..	84
4.1.1 - Métodos determinísticos	84
4.1.2 - Métodos estatísticos	84
4.2 - Reconhecimento de Padrões Sintáticos	87
4.3 - Métodos heurísticos para Reconhecimento de Padrões.	89
5 - PÓS-PROCESSAMENTO	90

CAPITULO 5 - ALGORITMO PARA RECONHECIMENTO DE CARACTERES MULTI-FONTE.....	94
1 - INTRODUÇÃO	94
2 - DESENVOLVIMENTO DO ALGORITMO PROPOSTO	97
2.1 - Especificação das imagens de entrada	98
2.2 - Aquisição da imagem	99
2.3 - Pré-processamento	100
2.4 - Definição das características a serem utilizadas..	103
2.5 - Extração de características e metodologia de reconhecimento	108
2.6 - Determinação das classes.....	110
2.6.1 - Metodologia de reconhecimento para a classe 1...	115
2.6.2 - Metodologia de reconhecimento para a classe 2...	117
2.6.3 - Metodologia de reconhecimento para a classe 3...	119
2.6.4 - Metodologia de reconhecimento para a classe 4 ..	119

2.6.5 - Metodologia de reconhecimento para a classe 5...122

2.6.6 - Metodologia de reconhecimento para a classe 6 ..124

2.6.7 - Metodologia de reconhecimento para a classe 7 ..128

2.6.8 - Metodologia de reconhecimento para a classe 8 ..130

2.6.9 - Metodologia de reconhecimento para a classe 9 ..136

2.6.10 - Metodologia de reconhecimento para a classe 10 142

2.6.11 - Metodologia de reconhecimento para a classe 11 150

CAPITULO 6 - RESULTADOS E CONCLUSÕES161

1 - AVALIAÇÃO DO SISTEMA161

1.1 - Avaliação através de outras fontes163

1.2 - Avaliação em diferentes resoluções166

1.3 - Avaliação com caracteres rotacionados167

1.4 - Avaliação sob-ruído168

2 - CONCLUSÕES E DISCUSSÃO PARA TRABALHOS FUTUROS.....168

BIBLIOGRAFIA	172
--------------------	-----

LISTA DE FIGURAS

Figura 2.1 - Classificação dos sistemas de reconhecimento de caracteres	9
Figura 2.2 - Estilos da escrita à mão	10
Figura 2.3 - Diagrama em blocos de um sistema de reconhecimento de caracteres	11
Figura 2.4 - Exemplos de tolerância na qualidade de impressão	16
Figura 2.5 - Caracteres em letra de forma propostos pela ANSI em 1969	20
Figura 2.6 - Exemplos de tipos estilizados para OCR	21
Figura 2.7 - Exemplos de variações no traçado do caracter "E"	25
Figura 2.8 - Estilos de escrita	26
Figura 2.9 - Múltiplas formas de letras como uma função da (a) posição da palavra, (b) contexto da letra	26
Figura 3.1 - Variação no número de padrões em função de	

velocidade de deslizamento do <i>scanner</i>	45
Figura 3.2 - Quantização do sinal do scanner dependente do contraste	46
Figura 3.3 - Representações esquemáticas do espectro de Fourier obtidas por diferentes tipos de imagens.....	53
Figura 3.4 - Exemplos de técnicas de segmentação	56
Figura 3.5 - Exemplos de problemas encontrados na segmentação de caracteres na escrita cursiva	58
Figura 3.6 - (a) Imagem binária, (b) Imagem em níveis de cinza, (c) Esqueleto em <i>bitmap</i>	60
Figura 4.1 - OCR por comparação de matrizes	68
Figura 4.2 - Método de detecção da característica "densidade" determinada para o caracter "5"	71
Figura 4.3 - Exemplos de códigos gerados por algoritmos de "características locais"	73
Figura 4.4 - Ilustração do método de traçado de contorno	76
Figura 4.5 - Imagem, esqueleto (S) e código.....	77

Figura 4.6 - Exemplos de características estruturais típicas	79
Figura 4.7 - Diagrama em blocos de um sistema de Reconhecimento estatístico de Padrões	85
Figura 4.8 - Diagrama em blocos de um sistema de Reconhecimento sintático de Padrões	87
Figura 4.9 - Ambigüidade de caracteres, resolvida apenas pelo conhecimento do assunto	93
Figura 5.1 - Exemplos de características que identificam cada caracter	95
Figura 5.2 - Esquema geral do sistema de reconhecimento de caracteres	98
Figura 5.3 - Determinação da largura dos caracteres	102
Figura 5.4 - Verificação da conectividade do pixel (i,j) em relação a sua vizinhança	103
Figura 5.5 - Diferentes representações para a letra "A"	104
Figura 5.6 - Visualização da classificação determinada em função do comportamento binário	110

Figura 5.7 - Diferentes comportamentos produzidos para as letras "M", "N" e "W"	113
Figura 5.8 - Visualização dos caracteres pertencentes à classe 1	115
Figura 5.9 - Teste realizado para distinguir entre as letras "M" e "W"	116
Figura 5.10 - Diagrama em árvore do reconhecimento para a classe 1	117
Figura 5.11 - Visualização dos caracteres pertencentes à classe 2	118
Figura 5.12 - Diagrama em árvore do reconhecimento para classe 2	118
Figura 5.13 - Diagrama em árvore do reconhecimento para a classe 3	119
Figura 5.14 - Comportamento (3/3) das letras "E", "M", "N" e "W" pertencentes à classe 4	120
Figura 5.15 - Distinção realizada para separar a letra "E", de "M", "N" e "W"	121

Figura 5.16 - Diagrama em árvore do reconhecimento para a classe 4	122
Figura 5.17 - Comportamento (3/2) das letras "E", "F", "M", "N" e "W" pertencentes à classe 5	123
Figura 5.18 - Diagrama em árvore do reconhecimento para a classe 5	124
Figura 5.19 - Comportamento (2/3) das letras "R", "Q", "M", "N" e "W" pertencentes à classe 6	125
Figura 5.20 - Distinção entre as letras "Q" e "R"	126
Figura 5.21 - Teste realizado para distinguir entre as letras "M" e "N"	127
Figura 5.22 - Diagrama em árvore do reconhecimento para a classe 6	128
Figura 5.23 - Comportamento (3/1) das letras "E", "F", "T" pertencentes à classe 7	128
Figura 5.24 - Teste realizado para distinguir entre as letras "F" e "T" e as letras "E" e "F"	129
Figura 5.25 - Diagrama em árvore da hierarquia de	

reconhecimento determinada para a classe 7	130
Figura 5.26 - Comportamento (2/1) das letras "A", "C", "E", "F", "P", "Q", "V" e "Y"	131
Figura 5.27 - Determinação das sub-classes da classe 8..	132
Figura 5.28 - Teste realizado para discriminar entre as letras "V" e "Y"	132
Figura 5.29 - Teste realizado para separar o conjunto de letras pertencentes à classe 8	133
Figura 5.30 - Distinção de letra "Q" de "C"	134
Figura 5.31 - Distinção entre as letras "F" e "P"	135
Figura 5.32 - Diagrama em árvore da hierarquia de reconhecimento para classe 8.....	136
Figura 5.33 - Comportamento (1/2) das letras "A", "C", "E", "F", "G", "J", "L", "S" e "Z"	137
Figura 5.34 - Formação das sub-classes, determinadas na classe 9	137
Figura 5.35 - Teste realizado para distinguir entre as	

letras "J" e "L"	138
Figura 5.36 - Teste realizado para distinguir a letra "A" das letras "J", "C", "L" e "G"	139
Figura 5.37 - Teste realizado para discriminar as letras "L", "J", "C", e "G"	140
Figura 5.38 - Testes realizados para separar a letra "Z" das letras "E", "G", "S"	141
Figura 5.39 - Diagrama em árvore da hierarquia de reconhecimento para classe 9	142
Figura 5.40 - Comportamento (1/1) das letras "A", "C", "E", "F", "I", "J", "L", "S", "T" e "Z"	143
Figura 5.41 - Formação das sub-classes, determinadas na classe 9	143
Figura 5.42 - Determinação das sub-classes 1.1 e 1.2 relativas às letras "J", "I", "J" e "T"	144
Figura 5.43 - Teste realizado para separar a letra "T" das letras "J", "I", "L"	145
Figura 5.44 - Teste realizado para separar a letra "J" das	

letras "L" e "I"	146
Figura 5.45 - Teste realizado para separar a letra "T" das letras "I" e "J"	147
Figura 5.46 - Distinção entre a letra "C" e as letras "A" e "F"	148
Figura 5.47 - Sequência de testes realizada para discriminar as letras "E", "F", "S" e "Z"	149
Figura 5.48 - Diagrama em árvore da hierarquia de reconhecimento determinada para a classe 10	150
Figura 5.49 - Comportamento (2/2) das letras "A", "B", "C", "D", "E", "G", "H", "K", "M", "N", "O", "Q", "R", "S", "U", "V", "W", "X" e "Z"	151
Figura 5.50 - Formação da sub-classe 1, determinada na classe 9	152
Figura 5.51 - Distinção entre as letras "H", "K", e "X"	153
Figura 5.52 - Distinção entre as letras "H", "K", "V" e "U"	154
Figura 5.53 - Separação das letras "C", "D", "G", "O", e "Q"	

de "R" e "X"	155
Figura 5.54 - Testes realizados para distinguir entre as letras "C", "D", "G", "O" e "Q"	156
Figura 5.55 - Sequência de testes utilizada na discriminação das letras "A", "X", "R" e "K"	157
Figura 5.56 - Separação entre as letras "B", "E" e "Q" das letras "G", "S" e "Z"	158
Figura 5.57 - Sequência de testes realizada para discriminar as letras "E", "Q", "B"	159
Figura 5.58 - Diagrama em árvore da hierarquia de reconhecimento determinada para a classe 11	160
Figura 6.1 - Exemplo de caracter com ruído, reconhecido pelo sistema	168

LISTA DE TABELAS

TABELA 1.1 - Taxas de substituição e rejeição típicas ...	14
TABELA 3.1 - Tamanhos típicos das matrizes dos caracteres binarizados.....	50
TABELA 4.1 - Comparação de várias técnicas de extração de características.....	81
TABELA 6.1 - Taxas de reconhecimento para as fontes observadas no treinamento.....	163
TABELA 6.2 - Taxas de reconhecimento para as fontes não previstas no treinamento.....	164
TABELA 6.3 - Taxas de reconhecimento em função da resolução do <i>scanner</i>	166
TABELA 6.4 - Taxa de reconhecimento obtida com caracteres rotacionados.....	167

RESUMO

Foi projetado um sistema para reconhecimento multi-fonte de caracteres alfabéticos isolados, impressos em letras maiúsculas, usando extração de características topológicas. Através da especificação de um conjunto apropriado de características, tais como traços verticais, horizontais e inclinados, áreas abertas e fechadas, denominadas aqui de características fundamentais, o reconhecimento foi determinado baseado na comparação de vetores de características. Os resultados de reconhecimento com protótipos de fontes impressas mostraram que o sistema proposto é capaz de fazer a leitura de diferentes tipos de caracteres impressos a uma taxa de precisão de 98.83%. Este trabalho também apresenta uma revisão descritiva dos aspectos básicos da aquisição, pré-processamento e extração de características da imagem. São apresentados ainda alguns aspectos históricos e mencionados alguns sistemas de OCR disponíveis comercialmente.

ABSTRACT

A multifont character recognition system has been designed by making use of topological feature extraction to recognize capital isolated letters. By properly specifying a set of features such as vertical, horizontal and slant strokes, curvatures, open and closed areas, named here fundamental features, the recognition has been performed based on feature vector matching. Recognition experiments with a prototype system for a variety of printed fonts show that the proposed system is capable of reading different types of printed characters at an accuracy rate of 98,83%. This work also presents a descriptive review about the basic aspects of image acquisition, preprocessing, feature extraction, historical aspects and commercially available OCR systems.

CAPITULO 1

RECONHECIMENTO ÓPTICO DE CARACTERES MULTI-FONTE

1 - INTRODUÇÃO

A necessidade de se integrar eficientemente diversos meios de comunicação em um processo convencionalmente denominado "Multimídia", tem suscitado o avanço tecnológico das diversas técnicas de digitalização, processamento de sinais, reconhecimento, compreensão e decisão. Dentro desse contexto, digitalizar e reconhecer caracteres e textos impressos para posterior processamento tem concentrado esforços de diversos fabricantes de equipamentos de digitalização gráfica e de pesquisadores de todo o mundo. Entretanto, os sistemas comerciais de reconhecimento de caracteres atuais ainda não apresentam um índice satisfatório de acertos e são muito sensíveis a ruídos. Se o equipamento de digitalização é manual, uma simples inclinação no texto de entrada inviabiliza por completo o reconhecimento.

Reconhecer caracteres ou textos com a capacidade e tolerância de um sistema visual humano está ainda longe de ser alcançado apesar de todo o avanço da tecnologia atual.

2 - OBJETIVOS

Este trabalho prevê o desenvolvimento de um algoritmo para o reconhecimento multi-fonte de caracteres alfabéticos isolados, impressos em letras maiúsculas, com aplicabilidade em qualquer sistema integrado de *scanner* com microcomputador. O método foi avaliado levando-se em consideração pequenas rotações, e também alguns tipos de distorções e estilizações, a fim de se aumentar a performance de reconhecimento. Para efeito de comparação e verificação da eficiência utilizou-se outros tipos de fontes fixas, para as quais não foi realizada a etapa de treinamento. O trabalho objetiva testar a taxa de reconhecimento de caracteres, relativamente à variação na escala dos mesmos, dentro de uma mesma fonte. Baseado na forma como os seres humanos explicariam a formação de cada caracter isolado, constituindo-o com retas, inclinações, arcos, etc, desenvolveu-se este método, denominado aqui de "Características Fundamentais" formantes de cada caracter.

3 - FORMATO DA DISSERTAÇÃO

O capítulo 2 apresenta os principais conceitos relacionados aos sistemas de reconhecimento de caracteres, tais como definições, aspectos históricos e aplicações. São estabelecidos ainda os principais parâmetros responsáveis pela qualidade de um sistema de OCR e mencionados alguns

sistemas comerciais mais conhecidos.

O capítulo 3 apresenta os aspectos básicos de aquisição da imagem e técnicas de pré-processamento, enquanto que no capítulo 4 são revistas as principais técnicas de extração de características, métodos de reconhecimento e pós-processamento em sistemas de reconhecimento de caracteres.

O capítulo 5 apresenta o desenvolvimento do algoritmo de reconhecimento e no capítulo 6 são apresentadas as análises dos resultados, conclusões e discussões para trabalhos futuros.

CAPITULO 2

SISTEMAS DE RECONHECIMENTO DE CARACTERES

1. INTRODUÇÃO

As questões relacionadas ao funcionamento do próprio corpo e às replicações das funções humanas sempre atraíram pesquisadores que almejam a construção de máquinas que reproduzam não apenas os mecanismos biológicos, mas também seu comportamento psicológico. Entretanto, apesar das diversas pesquisas dispensadas principalmente nas últimas quatro décadas e dos avanços teóricos alcançados no modelamento matemático sobre a percepção humana, o objetivo de se construir uma máquina realmente inteligente ainda não foi alcançado. O nome da disciplina que abriga a teoria da percepção, aplicada às diversas áreas da ciência é o Reconhecimento de Padrões e a sua sub-área que se refere à simulação da leitura humana é denominada Reconhecimento de Caracteres.

O Reconhecimento Óptico de Caracteres (OCR) tem sido um assunto de grande interesse devido às suas habilidades em aplicações tais como leitura automática de textos e de grandes volumes de dados, processamento de linguagens, em máquinas tradutoras e nos processos que envolvem comunicação homem-máquina.

Importantes contribuições, envolvendo o estado da

arte, resumos e revisões são apresentados em LINDGREN (1965), STEVENS (1968), NAGY (1968), HARMON (1972), FREEDMAN (1974), STALLINGS (1976), SUEN et al. (1980), DAVIS & LYALL (1986), MANTAS (1986), TAPPERT et al. (1990), NOUBOUD & PLAMANDON (1990) e GOVINDAN & SHIVAPRASAD (1990). Um importante tutorial envolvendo os principais fundamentos, aplicações, tendências e perspectivas em OCR pode ser visto em MARQUES (1990c)

2 - HISTÓRICO

Segundo RABINOW (1969), as raízes de OCR tiveram origem em 1870, quando Carey, de Boston, patenteou um sistema de transmissão de imagens usando um *scanner* de retina, que consistia em um mosaico de fotocélulas. Mais tarde, em 1890, o polonês Nipkow inventou o *scanner* seqüencial, onde a imagem era analisada linha por linha, em um sistema semelhante ao utilizado na televisão moderna e em algumas máquinas leitoras.

O reconhecimento de caracteres apareceu primeiramente como uma forma de auxílio a deficientes visuais, e os primeiros esforços bem sucedidos foram realizados pelo cientista russo Tyurin em 1900. Esforços subseqüentes relatados nesse sentido foram o telefone óptico de Fourier em 1912 e o dispositivo de auxílio tátil de Thomas, em 1926. Outras motivações foram marcadas pela primeira conferência de reconhecimento de padrões, em 1954:

"*Technical Meeting on Reading Machines*".

A primeira referência encontrada por RABINOW (1969) que pode ser considerada como um sistema leitor foi a máquina que convertia padrões impressos em códigos telegráficos, sem a intervenção humana, patenteada por Goldenberg de Chicago, em 1912. Máquinas que convertiam mensagens codificadas em caracteres já existiam.

Através de toda a história da computação, perfuradoras largamente desenvolvidas nas décadas de 20 e 30 dominaram o mercado de entrada de dados. A perfuração não era apenas lenta e incômoda, mas também propensa a erros, causados pelo chaveamento.

A versão moderna de OCR apareceu na metade dos anos 40, com o desenvolvimento de dispositivos de processamento de dados, principalmente do computador digital que facilitou a solução dos problemas relacionados com a entrada de dados em sistemas comerciais. Nesse sentido, o grande pioneiro na construção de equipamentos de OCR comerciais foi David Shepared (1954), da *Intelligent Machine Research Co.*, que abordou técnicas de *template matching*, semelhante à patente publicada por Goldemberg, em 1927. Outros trabalhos sucederam-se a esse e os principais conceitos básicos dessa tecnologia foram fundamentados na década de 50. Historicamente, abordagens alternativas, onde a identidade do caracter é determinada através da extração de características e comparação das mesmas com protótipos armazenados, foram publicados primeiramente em 1955, na

"Western Joint Computer Conference".

Sistemas foram desenvolvidos em grande escala comercial a partir da década de 60¹ até o presente momento.

A partir do início da década de 70 houve uma ênfase de questões envolvendo interação homem-máquina, padronização e controle de qualidade, incluindo problemas de densidades das tintas, qualidade do papel, e *design* de formatos. A praticabilidade das técnicas de OCR utilizadas foi enfatizada em termo das vantagens econômicas obtidas no *re-typing* de materiais impressos ou datilografados, quanto ao controle de erros de saída. Começaram a se desenvolver métodos mais sofisticados, incluindo fontes estilizadas para escrita cursiva, sistema multi-fontes, escrita em letra de forma dentro de certas restrições e alfabetos especiais, incluindo-se também técnicas de análise contextual.

Segundo MARQUES (1990c), a evolução histórica da vem sendo acompanhada sob os seguintes pontos de vista:

- das inovações tecnológicas em dispositivos de aquisição, monitoramento e impressão de imagens (digitalizado, monitores de alta resolução, impressoras a *laser*, etc.);
- dos trabalhos acadêmicos publicados, principalmente relatando aprimoramentos nos algoritmos e apresentação novas técnicas de reconhecimento;
- do desenvolvimento do *hardware* necessário à implementação

1 - Em 1969, Rabinow estimou a existência de 600 máquinas leitoras.

desses algoritmos, de forma a dotar os sistemas de reconhecimento de uma boa relação custo/eficiência.

3 - CLASSIFICAÇÃO DOS SISTEMAS DE RECONHECIMENTO DE CARACTERES

O Reconhecimento de Caracteres é uma sub-área do Reconhecimento de Padrões e pode ser realizado por meios mecânicos, ópticos ou magnéticos².

Reconhecimento de caracteres foi definido em BALM (1970) como "a transcrição de alfabetos e números em outra forma humanamente legível necessária para comunicação e transmissão dos mesmos".

"Reconhecimento de caracteres é a identificação dos caracteres através de meios automáticos" (International Standard Organization - ISO).

"Reconhecimento de caracteres é a identificação gráfica ou fônica, através de diversos meios" (CAMARÃO, 1989).

"Técnicas de reconhecimento de caracteres associam uma identidade simbólica à imagem de um caracter" (GOVINDAN & SHIVAPRASAD, 1990).

Na figura 2.1 pode ser vista uma classificação dos principais sistemas de reconhecimento de caracteres gráficos.

2 - O reconhecimento magnético de caracteres geralmente opera sobre caracteres impressos em tinta magnética feita com partículas de óxido de ferro, através de um mecanismo sensível à magnetização. Caracteres magnetizados do mesmo tamanho e estilo são facilmente identificados por dispositivos de leitura adequados (cabeça magnética).

FIGURA 2.1 - Classificação dos sistemas de Reconhecimento de Caracteres.

O termo OCR (*Optical Character Recognition*) refere-se a caracteres processados opticamente, ou seja, analisados por qualquer sistema óptico.

CAMARÃO, (1989) define OCR como sendo a coleção de técnicas e dispositivos que permitem a obtenção de dados para processamento eletrônico, sem a necessidade de uma transcrição intermediária.

No terminal de entrada de qualquer sistema de OCR tem-se documentos que podem conter caracteres impressos, datilografados ou escritos à mão e que serão digitalizados. Os sistemas que reconhecem caracteres datilografados ou impressos em fontes específicas (OCRA, OCRB, Pica, Elite,

Courier, etc.) são denominados sistemas de reconhecimento de fonte fixa. Os sistemas que podem reconhecer mais de um tipo de fonte são denominados multi-fontes e os que podem reconhecer um grande número de fontes são denominados omni-fontes (*omnifont*). Os caracteres manuscritos, mostrados na figura 2.2, podem ser caligráficos (*handwritting* ou *manuscript*) ou escritos em letra de forma (*handprinted* ou *block*) e podem estar conectados como na escrita cursiva (*cursive script*) ou isolados.

BLOCK PRINT

manuscript

cursive writing

FIGURA 2.2 - Estilos da escrita à mão (SUEN, 1983).

O reconhecimento da escrita à mão é classificado em duas categorias, de acordo com o modo de aquisição de dados utilizado.

Sistemas *on-line* geralmente utilizam como superfície de escrita as chamadas mesas digitalizadoras³ que digitalizam os dados à medida que a escrita é efetuada. Sistemas *off-line* utilizam dispositivos ópticos

3 - O termo *on-line* encontrado nas referências significa que o sistema realiza o reconhecimento quase em tempo real e está associado geralmente ao reconhecimento de caracteres manuscritos e ao uso de mesas digitalizadoras. Em oposição, a maioria dos sistemas de OCR comerciais são *off-line*, reconhecem caracteres impressos e utilizam scanners ópticos. As mesas digitalizadoras geralmente não utilizam transdutores ópticos portanto tais sistemas não devem ser considerados como uma sub-área de OCR.

digitalizadores, geralmente *scanners*, reconhecendo a escrita depois da mesma ter sido representada. No reconhecimento *on-line*, não é fornecida apenas a imagem do caracter, mas também a informação temporal de cada traço.

Em geral os sistemas de Reconhecimento de Caracteres, assim como os sistemas de Reconhecimento de Padrões consistem de seis seções funcionais principais, conforme ilustra a figura 2.3.

FIGURA 2.3 - Diagrama em blocos de um sistema de Reconhecimento de Caracteres.

Na aquisição da imagem, um transdutor obtém uma representação do padrão de entrada; o pré-processador melhora a representação e o extrator de características extrai características do padrão detectado. Através de uma função de decisão é atribuído ao padrão uma das possíveis classes. Além disso, alguns sistemas incluem uma etapa de pós-processamento, a fim de melhorar a taxa de reconhecimento.

4 - PRINCIPAIS PARÂMETROS DETERMINANTES DA QUALIDADE DE UM SISTEMA DE OCR

Segundo NAGY (1982) apud MARQUES (1990c), entre os diversos elementos a serem considerados na conversão de operações manuais de entrada de dados para OCR pode-se citar a taxa de erros permissível, o volume de dados envolvido, as características dos documentos, o estilo e a qualidade dos caracteres impressos.

APSEY(1978), TERSOFF (1978) e YACIK (1978) abordam diversos aspectos envolvendo capacidades, limitações e preferências de seres humanos e do *hardware*, envolvidos nos sistemas ópticos leitores, bem como suas influências na performance dos sistemas.

4.1 - Taxa de erros permissível

Os erros cometidos por um sistema de OCR podem ser avaliados tomando-se como base a taxa de substituições não detectadas ou a taxa de rejeição do sistema.

Em sistemas que envolvem um pequeno volume de dados, os caracteres rejeitados podem ser exibidos na tela do operador para que sejam corrigidos imediatamente. Em sistemas de grande porte, entretanto, a perda de eficiência resultante seria intolerável e os documentos com caracteres rejeitados seriam separados para subsequente entrada manual.

As principais fontes de erros em um sistema de OCR residem no estágio de aquisição da imagem, causados por não linearidades geométricas ou fotométricas dos transdutores ópticos utilizados na digitalização da imagem e no estágio de pré-processamento, onde o processo de segmentação dos caracteres pode fornecer em sua saída caracteres mutilados ou incompletos. Outra fonte de erros é a presença de manchas indesejáveis no documento, próximas aos caracteres. No estágio de classificação, os erros podem resultar de uma má escolha do algoritmo classificador ou devido ao conjunto de treinamento, utilizado como parâmetro do sistema, ser pouco representativo dos dados utilizados em aplicações práticas.

Existem inúmeras dificuldades em se determinar as taxas de erros dos sistemas, devido a fatores tais como diferenças de qualidade de impressão, tamanho do conjunto utilizado para computar as taxas e o trabalho exigido na verificação dos erros. A tabela 1.1 ilustra alguns valores típicos:

Muitos sistemas atuais podem ler dados com uma razão de substituição menor que 0,01%. Além disso, em sistemas mais elaborados, o *software* gera uma correção do erro no texto explorado, com uma descrição do número do documento, do campo e dos caracteres não legíveis. Os caracteres rejeitados podem ser corrigidos e fundidos dentro dos arquivos de dados corretos.

TABELA 1.1 - Taxas de substituição e rejeição típicas
(NAGY, 1982 apud MARQUES, 1990c).

Tipo de Caracter	Taxa de Substituição	Taxa de rejeição
Estilizado	1 / 200.000	1 / 20.000
Datilografado (qualidade ocr)	1 / 100.000	1 / 10.000
Datilografado (normal)	1 / 2.000	1 / 200
Escrito em letra de forma (alfa numérico)	1 / 200	1 / 20
Book face	1 / 1.000	1 / 100

4.2 - Volume de dados envolvidos

Os primeiros sistemas de OCR custavam milhares ou até mesmo milhões de dólares e eram considerados economicamente viáveis apenas se pudessem substituir centenas de operadores manuais, enquanto que os sistemas atuais que utilizam *scanners* manuais e um *software* de OCR de baixo custo, podem implementar aplicações que requeiram apenas um conjunto restrito de caracteres.

4.3 - Formato do documento a ser lido

Como o transporte do documento influencia com uma parcela significativa o custo de sistemas de maior porte, a densidade de dados em um documento é importante.

As variações dos tipos de entradas dos sistemas comerciais dependem do grau de sofisticação dos mesmos, entretanto os sistemas comerciais podem ser classificados como leitores de fita de caixa registradora, leitores de documentos e leitores de página inteira. Sistemas que combinem esses três tipos de entrada também são possíveis de serem implementados.

Documentos como cartões de crédito e fichas de depósito bancário são pequenos e possuem formato e qualidade de papel bem definidos. Devido ao fato de possuírem poucas linhas legíveis por documento, o mecanismo de transporte deve ser capaz de deslocar várias dezenas de documentos sob a cabeça leitora. O formato do campo de leitura pode ser programável para cada lote de documentos.

Os leitores de páginas movem apenas 1 a 2 documentos por segundo e cada página datilografada pode conter até 2000 caracteres. O movimento vertical da página é guiado pelo algoritmo de localização das linhas.

Um dos problemas mais críticos envolvidos com o transporte, encontra-se em aplicações de ordenação e separação de correspondências, onde o tamanho, espessura, qualidade do papel e localização do endereço no envelope

são variáveis.

4.4 - Qualidade de impressão e da imagem

A especificação do transdutor utilizado na aquisição da imagem e na etapa de classificação dos caracteres, depende principalmente da qualidade do caracter, expressa por:

- formato e espaçamento dos caracteres⁴;
- tipo de tinta usada;
- contraste entre o documento e a tinta utilizada;
- faixa de qualidade de impressão, conforme ilustra a figura 2.4;

FIGURA 2.4 - Exemplo de tolerância na qualidade de impressão BALM (1970).

4 - Os caracteres devem estar suficientemente espaçados dentro de uma fronteira específica a fim de se alcançar uma boa resolução.

- propriedades reflectivas do papel e/ou material a ser lido;
- presença de materiais indesejáveis no documento;
- as características do mecanismo de impressão, tais como definição de bordas e variações de densidade entre símbolos;
- disposição dos símbolos na página (espaçamento e alinhamento de linhas e caracteres);
- as possíveis variações no tamanho dos caracteres;
- a razão entre a altura do caracter e a largura média dos traços;
- as variações de largura entre os símbolos;
- o número de classes (tipos de escrita) e
- o grau de diferenciação entre os pares de caracteres mais parecidos entre si.

THROSSEL & FRYER (1974) abordam o problema de se medir a qualidade de impressão e a maneira com que esta se correlaciona com a performance obtida nos sistemas de reconhecimento de caracteres, com referência às especificações existentes e aos procedimentos de medida. Alguns resultados são citados e discutidos.

Os efeitos de se ignorar a qualidade de impressão podem implicar no uso de máquinas leitoras mais complexas e onerosas.

Muitos esforços têm sido feitos para se definir e especificar níveis de qualidade de impressão. Um dos maiores trabalhos nesse sentido foram as recomendações da

ISO (*ISO Draft Recommendation*)⁵, desenvolvidas em 1968, e aplicadas a dois conjuntos de caracteres padronizados, projetados especificamente para OCR⁶: OCRA, que possui uma aparência estilizada e OCRB que constitui uma fonte sem serifas⁷, dentro de uma estética simples e de exigências tipográficas e onde ambos os estilos possuem larguras dos traços constantes. Além disso, o documento da ISO define um sinal de contraste de impressão como uma função da reflectância em qualquer ponto e a reflectância do fundo do papel. A especificação esforça-se para englobar todas as categorias de impressão, determinando dois conjuntos de parâmetros, um específico para boa qualidade e o outro mais flexível.

4.5 - Tipos de caracteres a serem reconhecidos

As seis principais classes de entrada de dados, utilizando-se técnicas disponíveis atualmente, e em grau decrescente de adequação à automação são:

- tipos estilizados, próprios para OCR,
- textos datilografados,
- textos impressos,
- caracteres impressos,

5 - International Standards Organization (1968). Draft Recommendation 1831 apud THROSSELL & FRYER, (1974).

6 - International Standards Organization (1967). Recomendação N.1073, Alphanumeric Sets for Character Recognition, apud THROSSELL & FRYER, (1974).

7 - Serifas são traços transversais finos que arrematam as extremidades das hastes principais de um caracter.

- caracteres manuscritos isolados em posições (retângulos) bem definidas (*preprinted*),
- escrita cursiva,
- alfabetos especiais.

Além dessas variações, as entradas dos sistemas podem conter informações (*marks*), tamanho do alfabeto variável (numérico ou alfa-numérico), fontes com símbolos especiais (FORTRAN, e fontes utilizadas em cheques e cartões de crédito por exemplo), letras maiúsculas e minúsculas, com serifas (*serifs*) e sem serifas (*sans-serif*s), com diferentes pesos (negrito ou itálico), bem como provenientes de diversas fontes impressoras, como por exemplo máquinas calculadoras. Além disso, símbolos de uma mesma classe podem variar em estilo, detalhes geométricos e de construção, como por exemplo, espessura da caneta.

4.5.1 - Tipos estilizados, base de dados e restrições impostas (*constraints*)

Tipos estilizados são tipos de impressão projetados especialmente para facilitar o reconhecimento através do computador, como por exemplo o tipo americano OCRA, contendo 64 símbolos em 3 tamanhos, e o tipo europeu OCRB, contendo 113 símbolos, inclusive letras minúsculas.

Além de fontes impressas de OCR, padrões para a escrita à mão também foram desenvolvidos. O conjunto de

letras maiúsculas para o alfabeto inglês proposto pela ANSI⁸ (USASI) com seqüência dos traços padronizada pode ser visto na figura 2.5. Diversos outros aspectos foram padronizados, como por exemplo o grau de inclinação de linhas verticais permitido não deve ultrapassar 11 graus (YACYK, 1978).

FIGURA 2.5 - Caracteres em letra de forma propostos pela ANSI em 1969 (ITO & CHUI, 1978).

Devido ao limitado conjunto de caracteres, alguns pesquisadores estabeleceram base de dados padronizada que contém muitas variações de diferentes caracteres, os quais podem ser usados como conjuntos de teste e treinamento. As bases de dados mais comumente usadas foram criadas por MUNSON (1968) e HIGHLEYMAN (1961) apud DAVIES & LYALL

8 - (ANSI) American National Standard Institute, 1974, Standard X3.45.

(1986) e possuem matrizes binárias com resolução de 24x24 e 12x12 respectivamente.

Em paralelo às pesquisas desenvolvidas em técnicas de reconhecimento, os comitês de OCR de alguns países desenvolveram conjuntos de caracteres padronizados a fim de reduzir as variações nos formatos, resultando em uma melhor identificação e taxas de reconhecimento mais altas.

Em 1974, foi desenvolvido um conjunto de caracteres alfa-numéricos para letras de forma pelo Comitê de OCR da ANSI e um conjunto similar a este foi desenvolvido para a indústria japonesa. Em 1978, o Comitê canadense desenvolveu um conjunto numérico padronizado.

FIGURA 2.6 - Exemplos de tipos estilizados para OCR: (a) OCRA (americano), (b) OCRB (europeu), (c) padrão da indústria japonesa, (d) padrão canadense, (MARQUES, 1990c).

Examinando-se o formato dos caracteres de fontes OCRA e OCRB na figura 2.6, é fácil notar que cada caracter possui uma única forma, a fim de se evitarem ambigüidades com outros caracteres de formato similar. Usando-se esses conjuntos de caracteres é comum se alcançar uma precisão de 99,99% em máquinas comerciais com alta velocidade de leitura (acima de 100 caracteres/segundo).

Além dessas fontes padronizadas, outras fontes têm sido designadas para cartões de crédito e caixas registradoras.

A maioria dos sistemas de OCR desenvolvidos para a indústria reconhecem apenas caracteres padronizados, escritos em letra de forma, colocados em quadrados, com ou sem linhas de referências. Os usuários são instruídos ainda a representar os caracteres o mais próximo dos modelos especificados, evitando descontinuidades e floreios. Uma outra restrição comum é que os caracteres devem ser escritos no maior tamanho possível, dentro de fronteiras específicas, a fim de se obter uma boa resolução. São geralmente recomendados lápis HB.

Segundo SHAPIRO et al. (1987) apud GOVINDAN & SHIVAPRASAD (1990), algumas das restrições impostas mais comumente na maioria das máquinas são:

- caracteres individuais não devem tocar-se e o texto deve ser claramente impresso em tinta escura em um fundo colorido.

- a localização de caracteres individuais deve cair dentro de limites especificados.

- a capacidade de reconhecimento multi-fonte é determinada apenas se o operador treina a máquina para novas fontes.

Apesar das restrições terem sido projetadas originalmente para se reduzirem os esforços de reconhecimento realizados pela máquina, ignorá-las pode resultar na deterioração da performance, além de diminuir a velocidade de reconhecimento e até mesmo resultar em uma rejeição humana do processo.

SUEN (1982) analisa os principais problemas de interação homem-máquina, relacionados com o reconhecimento automático de numerais manuscritos, a fim de otimizar a performance dos leitores ópticos quanto ao uso de restrições, considerando fatores humanos e da máquina.

4.5.2 - Textos datilografados

A fim de se facilitar o reconhecimento de materiais datilografados utilizando um único tipo de impressão, a máquina de escrever deve estar bem alinhada, utilizar fita de filme de carbono e as convenções de formato devem ser cuidadosamente especificadas. Tipos sem serifas geralmente conduzem a menores taxas de erros porque as serifas tendem a conectar caracteres adjacentes.

4.5.3 - Textos impressos

O interesse no reconhecimento automático de textos impressos está ligado ao processo de recuperação automática (computadorizada) da informação.

Os principais problemas encontrados são: o grande número de estilos encontrados em um único documento⁹ (multi-fonte), sem contar as variações de tamanho, espaçamento proporcional entre os caracteres, espaçamento variável entre linhas, documentos com textos e figuras, variações na densidade das linhas como nas fontes em negrito, fontes em itálico, onde os caracteres são inclinados para a direita e as dificuldades de se segmentarem caracteres com larguras variáveis ou caracteres espaçosos que encostam nos caracteres adjacentes.

Diversas técnicas vêm sendo desenvolvidas para o reconhecimento de caracteres impressos, entre elas pode-se citar: BRADSHAW (1963), CASEY & NAGY (1968), SHEIBERG (1970), TROXEL (1976), COX et al. (1978), SRIHARI et al. (1983), CASEY & JIH (1983), MARQUES (1990a), MARQUES (1990b), SILVA (1990), GONZAGA & RODA, (1991) e LOPES & MARQUES (1992).

Em ULLMANN & KIDD (1968) é descrito um sistema multi-fonte para leitura de numerais impressos em envelopes. ASCHER et al. (1971), SCHÜRMANN (1978), HULL (1986), KAHAN

9 - Estima-se que existem no mínimo 300000 símbolos de uso comum na indústria de impressão americana. Essa variabilidade é determinada em termo de tamanho, mudanças na proporção dos caracteres e famílias de caracteres distintas, sem contar os diversos símbolos provenientes de disciplinas científicas e alfabetos estrangeiros (ASCHER et al., 1971).

(1987) e LEBOURGEOIS et al. (1992) também propõem sistemas para reconhecimento de caracteres impressos para várias fontes e tamanhos de caracteres.

Um interessante trabalho, quando o reconhecimento de multi-fontes está envolvido, pode ser visto em COX et al. (1982) que busca extrair características essenciais dos caracteres, independentes do tipo de fonte utilizada.

4.5.4 - Escrita à mão

A variabilidade nos caracteres manuscritos é ainda maior do que em sistemas multi-fontes pois cada pessoa possui seu próprio estilo de escrita e amostras de caracteres escritas pela mesma pessoa nunca são idênticas na forma, tamanho ou ordem de seus componentes, conforme ilustra a figura 2.7.

FIGURA 2.7 - Exemplos de variações no traçado do caracter "E"

Segundo SUEN (1986), a grande variabilidade na escrita à mão pode ser atribuída aos hábitos de escrita, estilo, cuidados, educação, região de origem (exemplificada na figura 2.8), estados de ânimo, saúde e outras condições do

escritor.

FIGURA 2.8 - Estilos de escrita (a) Norte-americano, (b) Europeu (NOUBOUD & PLAMONDON, 1990).

Em WING (1979) são analisados aspectos da variabilidade da escrita à mão, quanto a legibilidade e quanto ao tempo levado para executar a escrita. Segundo WING (1983), a variabilidade da escrita cursiva também é decorrente da posição da letra na palavra. Algumas formas de letras são escritas de maneira diferente no início e fim da palavra e também são dependentes das necessidades requeridas pelas letras antecedentes e precedentes que unem os caracteres na escrita cursiva, conforme exemplifica a figura 2.9.

FIGURA 2.9 - Múltiplas formas de letras como uma função da (a) posição da palavra, (b) contexto da letra, (c) variação aleatória (WING, 1983).

No terceiro exemplo da figura 2.9 (b) o conjunto "ing" (determinado pelo contexto) deve ser tratado como um único caracter (esses aspectos devem ser considerados em verificação de assinaturas). Na figura 2.9 (c) as mudanças podem ser atribuídas à inerente dificuldade do sistema neuromuscular em realizar movimentos motores finos, às limitações de precisão de repetição e às variações de velocidade da escrita.

SUEN et al. (1980) sugerem que o ensino elementar da escrita deva ser revisto a fim de providenciar melhores modelos e melhorar a padronização da escrita à mão.

SUEN (1983) descreve uma série de experiências realizadas, considerando os diversos estilos da escrita manual (*cursive*, *handprinted* e *manuscript*). Foram abordados diversos aspectos relacionados ao ensino da escrita, geração, percepção, reconhecimento e análise de várias propriedades morfológicas e estruturais dos caracteres manuscritos.

Além desses fatores humanos, os instrumentos e superfície de escrita, equipamentos de digitalização e os algoritmos utilizados afetam o reconhecimento. WARD & BLESSER (1985) e WARD & PHILLIPS (1988) abordam alguns desses aspectos.

4.5.4.1 - Caracteres manuscritos isolados

São encontradas muitas aplicações para documentos

escritos em letras de forma, escritas em espaços bem definidos, como por exemplo em formulários de declaração de imposto de renda e cartão-resposta comercial.

O principal problema encontrado nesse tipo de sistema consiste em treiná-lo para interpretar caracteres escritos por pessoas diferentes, embora seja facilitada a segmentação horizontal (entre caracteres) e identificação do campo de interesse dentro do documento.

Trabalhos relacionados com o reconhecimento de caracteres *handprinted* podem ser vistos em: KNOLL (1969), SHEINBERG (1970), TOUSSAINT & DONALDSON (1970), NAYLOR (1971), GRANLUND (1972), HUSSAIN et al. (1972), BLESSER et al. (1976), SUEN & SHILLMAN (1977), e BROWN et al. (1988). Alguns sistemas de reconhecimento *on-line* são descritos em: SIMEK & TUNIS (1967), ITO & CHUI (1978), LOY & LANDAU (1982) e KERRICK & BOVIK (1988).

Diversos trabalhos desenvolvidos para o reconhecimento de numerais manuscritos são relatados em: GREANIAS et al. (1963), PERROTO (1963), GENCHI et al. (1968), MORI et al. (1970), BEUN (1973a), BEUN (1973b), SIY & CHEN (1974), ALI & PAVLIDIS (1977) e CAI (1991).

Trabalhos relacionados com caracteres caligráficos (*handwritten*) são descritos em: GRIMSDALE (1958), SESAKI & KATARGIRI (1965), SESAKI et al. (1965), TOU & GONZALEZ (1972), BEUN (1973a), BEUN (1973b), e sistemas *on-line* são abordados em IKEDA et al. (1981) e WARD & KUKLINSK (1988).

WARD & KUKLINSKI (1988) desenvolveram um modelo que prevê muitas das variabilidades ocorridas na escrita em letra de forma, baseado na observação dos efeitos físicos (neuromotores e mecânicos) envolvidos no processo da escrita.

4.5.4.2 - Escrita cursiva

As máquinas comerciais capazes de reconhecer a escrita cursiva estão emergindo na forma de mesas digitalizadoras, capazes de prover o reconhecimento *on-line* de caracteres escritos à mão livre, através da detecção dos movimentos e seqüência dos traços, podendo inclusive servir para verificação automática de assinaturas.

Segundo MARQUES (1990c) essas máquinas apresentam taxas de erro típicas de 3% para taxa de rejeição e 0,2% de substituição, obtida com uma população de 300 escritores.

Algumas técnicas utilizadas para reconhecimento *off-line* e *on-line* da escrita cursiva podem ser vistas em: EDEN (1962), SAYRE (1973), DUTTA (1974) e BOZINOVIC & SRIHARI (1989).

DOOIJES (1983) realiza uma análise da dinâmica envolvida na escrita cursiva, considerando os movimentos realizados como resposta de um modelo de mecanismos psicológicos à uma determinada classe de sinais de entrada.

4.5.5 - Alfabetos especiais

Com o desenvolvimento do campo de OCR, uma considerável pesquisa vem sendo realizada no *software* e *hardware* para reconhecer não apenas numerais e letras romanas, mas também símbolos especiais como aqueles usados em comandos e linguagens de computação, símbolos matemáticos, circuitos eletrônicos e taquigrafia bem como ideogramas japoneses, chineses e diversos outros alfabetos como árabe, indiano e russo.

O grande desafio encontrado pelos pesquisadores atualmente é o reconhecimento de ideogramas devido à complexidade da forma e grande vocabulário envolvido. Muitos trabalhos vêm sendo desenvolvidos para o reconhecimento de ideogramas chineses e japoneses como em AGUI & NAGAHASHI (1979), IKEDA et al. (1981), KIMURA et al. (1987), AKIYAMA & HAGITA (1990) e TAPPERT et al. (1990). AGUI et al. (1979) descreve um sistema de reconhecimento para caracteres coreanos.

5 - COMPARAÇÃO ENTRE SISTEMAS DE RECONHECIMENTO ON-LINE E OFF-LINE

As pranchetas eletrônicas, desenvolvidas no fim dos anos 50, geralmente indicam as coordenadas dos pontos descritos pelos movimentos da ponta de uma caneta, obtidas em intervalos regulares, e representam uma ferramenta

poderosa na digitalização tanto da escrita, quanto na representação de desenhos e gráficos.

No início da década de 80, houve um grande interesse e retomada de pesquisas no reconhecimento em tempo real de caracteres manuscritos. Isso devido ao desenvolvimento de mesas eletrônicas mais precisas, computadores mais poderosos e melhores algoritmos de reconhecimento. Também contribuíram fatores tais como o interesse em métodos mais naturais de entrada de dados, maior conhecimento no *design* de interfaces com o usuário, e ainda avanços no *hardware* que permitiram que mesas digitalizadoras (entrada) e *displays* (saída) ocupassem a mesma superfície, habilitando o uso da tinta eletrônica¹¹ e permitindo uma maior interação do escritor com a escrita digitalizada.

Existem duas grandes tecnologias empregadas no desenvolvimento das mesas digitalizadoras: mesas eletromagnéticas/eletrostáticas que enviam as coordenadas da ponta da caneta para um computador em intervalos regulares e mesas sensíveis à pressão.

A medida de precisão das mesas digitalizadoras é caracterizada pela resolução, precisão e taxa de amostragem. As mesas digitalizadoras operam com uma taxa de amostragem constante ou com incrementos espaciais, através da obtenção dos pontos sucessivos de cruzamento da escrita com uma grade fixa, e possuem tipicamente uma resolução de

¹¹ - Tinta eletrônica é a visualização instantânea do traçado descrito pelo movimento da ponta de um estilete (*stylus*) (SUEN & TAPPERT, 1990).

200 pontos/polegada e uma taxa de amostragem de 100 pontos/segundo.

O ruído na representação após a amostragem ocorre devido ao processo de digitalização onde a resolução do desenho é limitada pela precisão das mesas, e pela distância existente entre a ponta da caneta e as coordenadas dos pontos obtidos.

O reconhecimento de caracteres *on-line* possui aplicações em CAD (*computer-aided design*), entrada de dados *on-line*, programas escritos à mão e aprendizagem por computador.

Uma vantagem dos dispositivos que digitalizam a imagem em tempo real é a captura de informações dinâmicas ou temporais da escrita (quantidade de traços, ordenação, direção e velocidade de escrita dos traços¹²) como uma seqüência de coordenadas de pontos, enquanto que sistemas *off-line* geralmente necessitam de técnicas de pré-processamento de alto custo computacional, para extrair os contornos e afiná-los.

A informação temporal dos traços melhora a precisão no reconhecimento, por outro lado pode dificultar o reconhecimento de imagens com variações no traçado, não perceptíveis em imagens estáticas. O tempo de reconhecimento de sistemas *on-line* pode então ser maior do que nos leitores ópticos de caracteres (OCR), onde é

12 - Um traço é definido como a porção escrita entre um estado de *pen down* e um *pen up* (SUEN & TAPPERT, (1990)).

necessário reconhecer os caracteres mais rápido do que foram escritos e existem menos restrições aos caracteres, permitindo maiores variações de orientação e tamanho.

A velocidade de digitalização nos dispositivos *on-line* geralmente ocorre a uma taxa de 1,5 à 2,5 caracteres/segundo para caracteres alfanuméricos ingleses e 0,2 à 2,5 caracteres/segundo para caracteres chineses, segundo SUEN & TAPPERT (1990).

A grande vantagem dos sistemas *off-line* é a sua alta velocidade de leitura. Para fontes simples, tal como OCRA, a velocidade é maior que 3000 numerais/segundo, conforme SUEN (1986). As maiores desvantagens é que tais sistemas não refletem características como *retraces* ou informação tempo-seqüência e a sua resolução (tipicamente 12x12 ou 24x24) é relativamente pequena para cobrir todos os efeitos causados pela variabilidade da escrita.

Outras vantagens dos sistemas *on-line* são determinadas em termo da grande interatividade e capacidade de adaptação proporcionadas pelo sistema quanto à correção de erros. Há possibilidade de correção de erros tão logo eles ocorram e alguns sistemas são capazes de armazenar os caracteres rejeitados a fim de melhorar o reconhecimento futuro, caso haja uma retro-alimentação no sistema.

As maiores desvantagens que os sistemas *on-line* apresentam são a necessidade do uso de equipamentos especiais e a não aplicabilidade a documentos impressos.

DAVIES & LYALL (1986) citam ainda alguns problemas dos

sistemas *on-line* comparado com sistemas de OCR: o conjunto de caracteres utilizados em CAD, por exemplo, é muito grande (mais do que 85 caracteres) em comparação aos 36 caracteres alfa-numéricos; em sistemas mais complexos há necessidade de se introduzir um vocabulário mais extenso e capacidade de aprendizagem; escrita borrada ou dispositivo de escrita inadequado podem reduzir a taxa de reconhecimento; os ruídos representam um problema principalmente se os caracteres forem muito pequenos, devido à precisão das mesas gráficas (0,1 à 0,5 mm). Devido ao custo envolvido no aumento da precisão de mesas digitalizadoras, geralmente é difícil reconhecer caracteres pequenos (3 à 5 mm); como resultado, muitos sistemas ficam restritos a caracteres com 5 à 10 mm de altura, o que é insatisfatório para aplicações típicas como em máquinas de faturamento.

Enquanto a maioria dos sistemas *off-line* de reconhecimento baseia-se em algoritmos de treinamento que rotulam os nomes dos caracteres antes da aquisição de dados, sistemas *on-line* possuem a capacidade de aprendizagem utilizando vantagens interativas com o usuário, por exemplo através da rotulagem do caracter após ter ocorrido um erro ou rejeição.

6 - APLICAÇÕES DE RECONHECIMENTO DE CARACTERES

O Reconhecimento Óptico de Caracteres possui muitas

aplicações práticas sugeridas por pesquisadores. Como exemplos, pode-se citar:

- em sistemas de automação de escritório (HONDA et al., 1984);
- como forma de auxílio à pessoas cegas (BLISS, 1969) e surdas (BROOKS & NEWELL, 1985);
- uso em departamento postal, para leitura de endereço e códigos postais (GENCHI et al., 1968; ULLMANN & KIDD, 1969; MORI et al., 1970 e CAI, 1991);
- leitura e classificação de documentos, como em reservas aéreas;
- leitura automática de placas de veículos (LOTUFO & OLIVEIRA, 1990) e gravação para controle de tráfego;
- uso em indústria de publicação e comunicação em terminais de dados;
- processamento direto de documentos, para entrada de dados com alta velocidade, como leitores de micro-filme;
- uso em sistemas de faturamento, como em sistemas de cobrança de contas de telefone e água;
- uso em leitura de código de barra e em sistemas de inspeção automática;
- em sistemas de identificação pessoal, como em cartões de crédito (POITEVENT, 1969);
- aplicações financeiras, como leitura e classificação de cheques (WHITE & ROHRER, 1983);
- Verificação de assinaturas (EDEN, 1962 e TAPPERT et al., 1990);

- em sistemas comerciais, como leitores de preços, códigos de produtos, livro-mestre, etc.;
- cartografia automática (BROWN et al., 1988);
- tradução simultânea;
- em sistemas de gerenciamento automático da informação (SCHEIN, 1989);
- uso em facsímile (WHITE & ROHRER, 1983);
- tradução de estenografia (LEEDHAM & DOWNTOM, 1986 e LEEDHAM & DOWNTOM, 1987);
- leitura de caracteres estampados em estruturas metálicas (NAKAMURA et al., 1986);
- aplicações judiciais (ANSELL, 1979);
- leitura de linguagens de computação (BORNAT & BRADY, 1974);
- em projetos assistidos por computador (CAD);

DVORAK et al. (1992) citaram algumas aplicações práticas de OCR como solução para as tarefas do mundo real, como por exemplo:

- No começo de 1992, o *Australia Post* começou a utilizar equipamentos de classificação com OCR em suas principais agências, com capacidade para classificar correspondências datilografadas a uma taxa de 10 cartas por segundo.
- A *PhoneDisc* criou um banco de dados com todos os números telefônicos residenciais dos Estados Unidos, usando o *Gigaread*, um pacote de OCR desenvolvido na Alemanha para digitalizar todas as listas telefônicas dos Estados Unidos e a empresa comercializa essas informações em dois CD's.

- O IRS (*Internal Revenue Service*), equivalente à Receita Federal no Brasil, utiliza sistemas de OCR para digitalizar certos tipos de documentos de impostos e declarações, capaz de digitalizar mais de 13000 documentos de vários tipos por hora.

- O *American Express Travel Related Services* utiliza o OCR para acelerar o processamento das despesas de clientes a uma taxa de acertos de 80%. Apesar de não ser uma taxa de grande sucesso, auxilia a *American Express* a enviar mais rapidamente as notas de cobrança aos clientes, recebendo seu dinheiro mais rapidamente.

7 - SISTEMAS COMERCIAIS DISPONÍVEIS NO MERCADO

Existem atualmente muitas companhias norte americanas e japonesas que possuem sistemas de reconhecimento *on-line* disponíveis no mercado e que utilizam geralmente mesas eletromagnéticas ou sensíveis à pressão e trabalham com alfabetos que vão desde 50 até mais de 3000 caracteres (caracteres ASCII, alfanumérico ou de alfabetos especiais, como FORTRAN e japonês ou chinês respectivamente), e são vendidos por preço variando entre US\$ 800.00 e US\$ 3,335.00, conforme MARQUES (1990c).

Os sistemas de OCR *off-line* consistem basicamente de um pacote de *software*, eventualmente acompanhado de uma placa de expansão para computadores do tipo IBM-PC e Macintosh, e podem ler inúmeros tipos de impressão, por

vezes combinados em uma mesma página, em tamanhos que variam desde corpo-8 até corpo 72. A maioria dos sistemas permite ao usuário o treinamento de diferentes tipos de fontes e a faixa de preços desses sistemas varia entre US\$ 269.00 e US\$ 3,995.00 (MARQUES, 1990c). A maior parte dos pacotes custa entre US\$ 500.00 e US\$ 1,000.00. Os únicos programas que custam menos são os incluídos em pacotes de *scanners*. Evidentemente ao se avaliarem programas de OCR deve-se pesar a relação custo\benefício em termos de tempo e esforços poupados.

Os pacotes comerciais de OCR reduzem o processo de entrada de documento a uma tarefa rápida e fácil através da habilidade de explorar, reconhecer e carregar textos direta e automaticamente dentro de aplicações. Muitos deles reproduzem não apenas o texto da página impressa, mas também a estrutura da página e até mesmo a fonte na qual foi impressa. Os *softwares* de OCR necessitam de um *scanner* para criar uma imagem da página em mapa de bits e os resultados de reconhecimento são gravados em formato ASCII ou em outro formato específico da aplicação. O pacote processa o arquivo da imagem, tentando encontrar caracteres nos padrões de bits gerados durante a digitalização. Na outra extremidade, os pacotes de OCR atuam como programas de conversão. Enquanto programas de digitalização convertem uma imagem do mundo real (ou mesmo uma imagem de texto) em uma imagem gráfica, programas de OCR convertem uma imagem gráfica em texto. Os programas *WordScan Plus* e o *ReadRight*

são dois dos programas de OCR mais conhecidos.

Os pacotes de OCR podem estar na forma de *hardware* e *software* ou apenas *software*. Os primeiros pacotes possuíam dispositivos bastante caros de *hardware*. Em geral, os produtos para OCR ainda são dispendiosos, pois o mercado não chegou a crescer até um tamanho crítico. De fato, os principais participantes do mercado de OCR para PCs como a *Calera* e a *Xerox Imaging Systems* ainda vendem equipamentos acima de US\$ 20,000.00 (*desk top*). Entretanto, a situação do mercado deve mudar nos próximos quatro anos (DVORAK et al., 1992).

A maior parte dos programas atuais utiliza a tecnologia *omnifont*. Um programa *omnifont* possui uma grande biblioteca de informações sobre fontes, que o programa pode consultar ao executar suas tarefas e que possui as características das fontes ao invés dos padrões de bits reais. Alguns pacotes utilizam exclusivamente a tecnologia *omnifont*; outros a utilizam em conjunto com algoritmos de aprendizado.

Apesar de todas as principais empresas que trabalham com OCR destinado aos PCs utilizarem a tecnologia *omnifont*, não usam porém os mesmos algoritmos, podendo haver grandes diferenças de desempenho entre os pacotes de OCR. Além das capacidades de reconhecimento das fontes e tecnologia de reconhecimento, a qualidade da imagem digitalizada faz uma grande diferença no sucesso do programa.

Em JONES (1993) são analisados nove tipos de *softwares* comerciais existentes: *OmniPage Direct*, *OmniPage Professional*, *OmniPage 386*, *Perceive*, *ReadRight for Windows*, *Recognition Plus*, *TypeReader*, *WordScan* e *WordScan Plus*, escolhidos para análise por apresentarem características *omnifont*, técnicas de extração de características (podendo incluir análise de contexto através do uso de um dicionário), resolução da imagem de 400 pontos por polegada, capacidade de decomposição automática da página eliminando-se elementos gráficos, capacidade de leitura de documentos com múltiplas colunas e por apresentarem preços inferiores a US\$ 1,000.00.

Os detalhes de tecnologia e operações de vários OCRs e *scanners*, bem como a análise de alguns *softwares* podem ser vistos em DVORAK et al. (1992).

Apesar dos sistemas comerciais atuais apresentarem relativamente baixo custo e alta taxa de reconhecimento, ainda é necessário o desenvolvimento de novas técnicas que solucionem os diversos problemas ainda encontrados nos sistemas e que permitam por exemplo, o reconhecimento de caracteres rotacionados.

CAPÍTULO 3

AQUISIÇÃO E PRÉ-PROCESSAMENTO DA IMAGEM

1 - SCANNERS UTILIZADOS NA AQUISIÇÃO DA IMAGEM

O primeiro estágio de qualquer sistema típico de OCR consiste na aquisição e digitalização dos caracteres, geralmente realizadas por *scanners* ópticos, convertendo-os a um formato adequado para processamento eletrônico subsequente.

Uma das principais fontes de erros dos sistemas de OCR ocorrem no estágio de aquisição da imagem, devido à não-linearidade geométrica ou fotométrica dos transdutores ópticos. A qualidade global de um sistema de OCR depende diretamente da qualidade do dispositivo utilizado na digitalização da imagem. Esse dispositivo deve ser extremamente fiel à qualidade e aos detalhes da imagem original, não omitindo detalhes significativos ao processo de discriminação dos padrões similares nem deve introduzir distorções que dificultem o processo de reconhecimento (MARQUES, 1990c).

Se um feixe explorador de um CRT (*Cathode Ray Tube*), *laser* ou arranjo de LEDs é focalizado nos caracteres e sua reflexão é sensoriada por um fotodetector fixo, a exploração é dita ativa. A exploração é dita passiva se todo o campo é iluminado uniformemente, com o sensor

detectando a reflexão de apenas um único ponto por vez.

De acordo com BALM (1970), FREEDMAN (1974) e MANTAS (1986) os *scanners* mais popularmente utilizados são:

- *scanner* de varredura, onde um feixe de luz controlado varre as direções acima e abaixo do caracter e a luz refletida é medida por algum dispositivo sensor de luz como uma função do tempo. Esse tipo de varredura ocorre por exemplo nos *scanners* de ponto de luz móvel, *scanner* vidicon e *scanner* de varredura mecânica.
- *scanner* mecânico, também chamado disco de Nipkov, consiste de um disco contendo aberturas por onde passa a luz refletida do caracter para uma fotocélula de medida.
- *scanner* de fenda dividida, onde uma coluna vertical de dispositivos foto-elétricos pode ser usada como uma fenda dividida e a luz refletida através da fenda pode ser medida por fotocélulas.
- *scanner* em arranjo que utiliza uma matriz bi-dimensional de fotocélulas, LEDs ou dispositivos similares, e provê uma imagem bi-dimensional do caracter.
- *scanner* seguidor de curva, onde um feixe de luz controlado segue o contorno externo de cada padrão através das mudanças de contraste e relaciona a distância e a informação direcional.

Com a evolução da tecnologia dos circuitos integrados, foi possível integrar em um mesmo *chip* os circuitos de varredura com elementos foto-sensíveis, e produzir um sinal seqüencial similar aos gerados no tubo de TV. Esses

dispositivos são conhecidos como arranjos de estado-sólido ou de auto-varredura. A partir da década de 70 novas tecnologias em semicondutores permitiram o aparecimento de dispositivos de auto-varredura como os acoplados por carga CCD (*Charge Coupled Devices*) ou os dispositivos por injeção de cargas CID (*Charge Injection Devices*). Os sensores CCD acumulam carga em suas capacitâncias internas de acordo com a intensidade e tempo de exposição da luz aplicada.

A complexidade dos *scanners* aumenta com a inclusão de funções tais como normalização do tamanho, *threshold* dinâmico ajustável, capacidade de ler marcas e capacidade de realizar a digitalização de um mesmo caracter várias vezes a fim de superar as variações de entrada.

EVANGELISTI (1982) desenvolveu um método para avaliar e auxiliar o desenvolvimento de *scanners* utilizados no reconhecimento de caracteres. Utilizando-se um grande número de padrões coletados de um *scanner*, o método, baseado em um conjunto de parâmetros, determina a cada padrão uma figura de mérito, que expressa a dificuldade de reconhecimento do mesmo.

2 - IMPLICAÇÕES NO USO DE SCANNERS

A imagem a ser analisada pelo *scanner* usualmente está impressa em papel e pode ser constituída de texto, linha, desenhos, ilustrações em retícula (*halftone*) em cores ou monocromáticas.

Em OCR, a imagem de entrada geralmente é constituída apenas de texto e a saída do *scanner* é um sinal analógico (níveis de cinza) correspondente à quantidade de luz refletida de cada pixel no documento de entrada. Ao invés de manter a imagem em níveis de cinza, é comum convertê-la em matrizes binárias para reduzir o espaço de armazenamento e diminuir o esforço computacional. Para caracteres pequenos e detalhes finos em imagens, o número de sensores deve ser suficiente para explorar a estrutura dos detalhes. Os fabricantes de *scanners* e OCRs determinam 300 dpi (*dots per inch*) como ótima a relação custo/resolução e às vezes a utilização de *scanners* de 400 dpi pode tornar o sistema lento dependendo da sua quantidade de RAMs, e pode haver um decréscimo na precisão pelo aumento de ruído na imagem explorada.¹³

Um dos problemas mais significativos em OCR é a conversão de um sinal analógico não ideal em uma imagem binária ideal. Os documentos originais são geralmente sujos, multi-coloridos, e os caracteres são produzidos por uma variedade de tipos de canetas, lápis ou outros mecanismos de impressão. Além disso, segundo WHITE & ROHRER (1983), devido à tecnologia e às limitações no custo do *hardware* dos *scanners*, podem ocorrer ainda: não-uniformidades na iluminação, sensibilidade e variações na corrente de escuro entre elementos no arranjo sensor,

13 - Por outro lado, o software "OmniPage Professional" cometeu 59 erros com a fonte "Times Roman" de 6 pontos, à 300 dpi, entretanto cometeu apenas 12 erros com a mesma amostragem à 400 dp (JONES, 1992).

resolução não ideal devido às lentes e interferências no arranjo.

Um outro problema decorrente do uso de *scanners* é a velocidade de exploração manual que altera a quantidade de padrões pretos e brancos, conforme exemplifica a figura 3.1. Se o *scanner* é movido rápido ou lentamente na direção horizontal, os padrões de pixels diminuem ou aumentam respectivamente nessa direção.

FIGURA 3.1 - Variação no número de padrões em função de velocidade de deslizamento do *scanner* (SATO et al., 1983).

A operação de binarização determina um valor onde o pixel deve ser preto ou branco. No reconhecimento de caracteres, *thresholding* descreve a supressão dos padrões de fundo não desejáveis a fim de que apenas caracteres impressos ou manuscritos sejam capturados como imagens eletrônicas.

Um simples valor pré-fixado de *threshold* é adequado para papéis brancos e de alta qualidade. Entretanto,

valores dinâmicos de *threshold*, conforme ilustra a figura 3.2, são utilizados em sistemas mais avançados de OCR, capazes de gerar uma imagem binária confiável independente da qualidade do papel (CASEY & JIH, 1980).

Figura 3.2 - Quantização do sinal do scanner dependente do contraste (FREEDMAN, 1974)

Em WHITE & ROHRER (1983) é descrito detalhadamente o desenvolvimento de um algoritmo de *thresholding* dinâmico, onde o cálculo básico do *threshold* é representado como uma equação diferencial de primeira ordem com coeficientes não lineares, determinados heurísticamente.

3 - TÉCNICAS DE ARMAZENAMENTO DAS IMAGENS

Dois métodos destacam-se na representação de imagens gráficas: a representação vetorial¹⁴ que descreve uma imagem

¹⁴ - Os gráficos vetoriais, também denominados gráficos baseados em objetos, armazenam imagens gráficas na forma de representações matemáticas que podem ser dimensionadas sem distorções. São adequados à aplicações que requeiram precisão e mudanças de escala (programas de desenho geométrico). (PFAFFENBERGER, 1992)

em função de linhas e figuras e a representação em *bitmap*¹⁵.

A maioria dos dispositivos de entrada, como *scanners*, câmeras de vídeo e fac-símiles, gera imagens em *bitmap*, com exceção do *mouse* e das mesas digitalizadoras que geram dados vetoriais.

O mapeamento de bits é o mais utilizado devido às facilidades de implementação e capacidade de ser aplicado a qualquer imagem. Nesse método a imagem é quebrada em uma grade de pontos, onde a cor ou nível de cinza de cada elemento da imagem é armazenado individualmente e cuja coordenada corresponde ao bit de dado do ponto da imagem a ser representada.

O formato de arquivos em *bitmap* é adequado a imagens que possuem grande variedade de cores, sombras e formas, tais como em fotografias, pinturas e imagens digitalizadas em telas de vídeo. No formato vetorial as representações são baseadas em linhas (como em CAD), imagens com sombreamento, formas e colorações mais simples. Alguns tipos de gráficos e desenhos à mão-livre são tipicamente gravados em arquivos vetoriais.

Apesar da simplicidade na programação, a representação de imagens em *bitmap* pode sofrer vários problemas práticos. Um deles é o problema do tamanho da imagem que em imagens coloridas de alta resolução necessita de vários megabytes para armazenamento e mais megabytes de memória para seu

15 - Os gráficos de mapas de bits, também denominados gráficos de varredura, armazenam a imagem na forma de padrões de pixels. Em programas de desenho livre permite criar padrões de sombreamento mas qualquer redimensionamento da escala provoca distorções desagradáveis. (PFAFFENBERGER, 1992)

processamento e visualização. Por isso existe a necessidade de se compactarem as imagens que serão armazenadas (KAY & LEVINE, 1984 e WELCH, 1984).

Um outro problema apresentado em imagens *bitmap* é a falta de relacionamento intrínseco entre os pixels, dificultando a identificação de imagens isoladas e havendo necessidade de programas mais complexos que determinem entre os pixels adjacentes alguma relação de similaridade. A representação em *bitmap* apresenta ainda problemas devidos à resolução fixa da grade de pixels na qual a imagem é dividida, podendo causar um efeito de serrilhado (*alising*) quando a imagem é escalonada. Alguns programas gráficos compensam esse problema através da interpolação linear entre os pixels, mas isto envolve maior tempo de processamento. Analogamente, quando se compacta a imagem, perde-se em resolução. Quando a imagem é recuperada ao tamanho natural, sofre distorções e apresenta efeitos de *alising*.

Embora imagens *bitmap* possam ser manipuladas de muitas maneiras, mapa de bits de texto não podem ser usados em aplicações tais como processamento da palavra ou gerenciamento de base de dados. A fim de se acessar e manipular imagens de texto, o mesmo deve ser convertido em ASCII.

A representação vetorial é mais limitada do que a *bitmap* em termos de capacidade de representação, mas em alguns aspectos pode ser mais eficiente. Por exemplo, uma

linha reta pode ser representada por apenas dois pontos extremos; uma linha curva pode ser aproximada por vários segmentos de retas unidos; dessa maneira, imagens isoladas podem ser tratadas de maneira independente e escalonadas sem afetar a resolução. Além disso, os gráficos vetoriais gerados ficam restritos à resolução máxima da impressora (até 300 dpi para a impressora à laser), enquanto que os gráficos em mapa de bits limitam a resolução da impressão à resolução do vídeo utilizado.

Independente das variações nos formatos de arquivos gráficos, existem procedimentos comuns no modo em que os dados são codificados e algumas maneiras convencionais de se comprimir dados gráficos ocupando menos espaço de armazenamento nos arquivos.

Existe uma grande quantidade de arquivos gráficos *bitmap* desenvolvidos para diversas aplicações; entre os mais populares pode-se citar: GEM/IMG, PCX, GIF e TIFF.

4 - TAMANHO TÍPICO DAS MATRIZES DOS CARACTERES BINARIZADOS

Dependendo da complexidade do formato envolvido, o tamanho da matriz, o qual reflete a resolução de um caracter digitalizado, varia para melhorar a velocidade e precisão. Tipicamente o tamanho 8 (largura) x 10 (altura) pixels é suficiente para o reconhecimento de fontes estilizadas como OCRA e OCRB. Tamanhos maiores de matrizes são necessários para tipos de fontes regulares como Elite,

Courier e Gótica. Para escrita à mão e caracteres chineses as dimensões das matrizes tem que ser muito maiores a fim de que características detalhadas possam ser extraídas de padrões complexos. A tabela 3.1 relaciona o tamanho da matriz ao tipo de caracter digitalizado.

TABELA 3.1 - Tamanhos típicos das matrizes dos caracteres binarizados (SUEN et al., 1986)

FONTES ESTILIZADAS	FONTES REGULARES	FONTES ALFANUMÉRICAS MANUSCRITAS	CARACTERES CHINESES IMPRESSOS	CARACTERES CHINESES MANUSCRITOS
8 x 10 ↓ 12 x 15	12 x 15 ↓ 20 x 25	15 x 20 ↓ 30 x 40	25 x 32 ↓ 35 x 45	30 x 40 ↓ 60 x 80

5 - PRÉ-PROCESSAMENTO DA IMAGEM

Dá-se o nome de pré-processamento à conversão do padrão preto e branco de uma página de um documento em um conjunto de padrões menores que servirão como entrada para o algoritmo de classificação (MARQUES, 1990c). O pré-processamento necessário para a classificação deve localizar cada caracter na página, numa seqüência que faça sentido no arquivo codificado e apresentar cada caracter isolado ao algoritmo de reconhecimento de uma forma adequada.

Dependendo do processo de digitalização utilizado e da resolução do equipamento digitalizador, os dados podem apresentar ruídos que distorçam severamente os padrões, dificultam a extração de suas características e afetam a precisão do processo final de classificação. O objetivo do pré-processamento é, portanto, aumentar a qualidade dos caracteres de entrada de modo que os mesmos sejam mais similares à classe a que pertençam.

As principais técnicas de pré-processamento necessárias ao processo de OCR incluem: técnicas de suavização e filtragem para remoção de ruídos na imagem do caracter; algoritmos de localização de linhas e de identificação do formato do documento em questão, através da identificação das áreas de interesse; técnicas de segmentação dos caracteres, através das quais separa-se o caracter que se deseja reconhecer dos demais; e técnicas de normalização dos caracteres que objetivam adaptar o tamanho e a localização dos caracteres aos parâmetros do sistema, tornando-os tanto quanto possível invariáveis ao tamanho, orientação e às eventuais rotações e inclinações.

Segundo GÜDESEN (1976) as técnicas de pré-processamento podem ser divididas em operações locais e em transformações que modificam toda a estrutura do caracter.

As operações locais apenas modificam o estado de um único ponto de uma grade, através da obtenção de informação dos pontos vizinhos. Isso é geralmente feito através da utilização de matrizes para se extrair a informação

desejada da vizinhança de uma grade de pontos. Essas matrizes são deslocadas sobre todos os pontos da grade seqüencialmente. Alguns exemplos de operações locais de interesse ao reconhecimento de caracteres são o preenchimento de buracos, remoção de ruídos (pontos isolados) e afinamento de caracteres.

As transformações na estrutura completa do caracter são essencialmente geométricas e produzem não-linearidades, muitas vezes essenciais para o processo de reconhecimento. A centralização de caracteres, rotação e normalização são alguns exemplos de pré-processamento geométrico.

Técnicas de pré-processamento podem eliminar muitas das variabilidades iniciais encontradas na escrita à mão. Tipicamente, um pré-processador para a escrita cursiva se preocupa com translação, rotação, escalonamento, suavização, eliminação de ruídos, determinação da linha de referência e correção da inclinação da escrita. Técnicas de pré-processamento específicas para a escrita cursiva são discutidas em DUTTA (1974), BROWN & GANAPATHY (1983), DAVIES & LYALL (1986) e BOZINOVIC & SRIHARI (1989).

5.1 - Segmentação de textos

Um documento genérico pode ser constituído por diferentes tipos de imagens, tais como texto, diagramas e fotografias. Antes que o texto possa ser reconhecido ao nível do caracter ou da palavra, há necessidade de se

identificar as regiões de interesse no documento. As técnicas de segmentação de texto têm sido desenvolvidas para aplicações tais como processamento de documentos para transmissão em fac-símiles e para localização de códigos postais em envelopes.

Existem diversos métodos utilizados para realizar a segmentação de textos, como por exemplo os mostrados em SCHERL & HUNDT (1984), AKIYAMA & HAGITA (1990) e ELLIMAN & LANCASTER (1990).

FIGURA 3.3 - Representações esquemáticas do espectro de Fourier obtidas por diferentes tipos de imagens, utilizadas em segmentação de textos (ELLIMAN & LANCASTER, 1990).

SCHERL et al. (1980) apud ELLIMAN & LANCASTER (1990) perceberam que os espectros de Fourier para tipos

diferentes de imagens em níveis de cinza são distintos, conforme pode ser visto na figura 3.3, e que para diferentes tipos de fontes, o espectro de Fourier traz ainda informações de orientação do texto, tamanho, espaçamento entre os caracteres e entre as linhas, entretanto é necessário um grande tempo de processamento. A segmentação ao nível de texto não pode ser geralmente realizada através da análise dos histogramas, devido ao fato que diferentes regiões podem apresentar histogramas similares.

5.2 - Segmentação vertical

Muitas máquinas sofisticadas de OCR têm sido construídas para reconhecer vários tipos de fontes, com diferentes espaçamentos (*pitch*)¹⁶ entre caracteres e entre as linhas. Nesse tipo de máquina um sofisticado algoritmo de determinação de linhas é necessário antes de ser realizada a segmentação.

No *design* de uma determinada fonte, cada caracter possui uma posição vertical específica, relativa a uma linha horizontal imaginária, denominada linha de base¹⁷.

16 - Uma medida horizontal do número de caracteres por polegada linear impressos ou mostrados na tela com uma fonte de espaçamento simples, como as utilizadas nas máquinas de datilografia, impressoras matriciais e impressoras de margarida. O *pitch* elite por exemplo equivale a 10 caracteres por polegada (PFAFFENBERGER, 1992).

17 - Em cada tipo de fonte os caracteres possuem uma específica posição vertical relativa a uma imaginária linha horizontal denominada linha de base, ignorando-se as extremidades inferiores de alguns caracteres (como por exemplo "j", "g", e "p"). Na verdade essa "linha" representa uma média de todas as linhas de base da sequência total dos caracteres.

A segmentação vertical elimina os ruídos causados pelos caracteres das linhas adjacentes. Esse tipo de segmentação é necessária principalmente em aplicações onde haja a presença de caracteres minúsculos, símbolos, sublinhados, e caracteres subscritos ou sobrescritos em um documento impresso no padrão de seis linhas por polegada.

A informação sobre a linha de base é necessária tanto para a segmentação vertical (entre linhas) quanto para a segmentação horizontal (entre caracteres), além de determinar o desvio das linhas e a posição das linhas seguintes.

Caracteres desconexos tais como "j" e "i" apresentam problemas na segmentação onde há necessidade de se assegurar que a parte isolada pertence ao caracter e não corresponde aos ruídos da imagem.

5.3 - Segmentação horizontal

Muitos sistemas comerciais de OCR realizam a segmentação de caracteres determinando que os usuários escrevam os caracteres no centro de blocos bem definidos ou em localizações conhecidas, geralmente guiadas por marcas impressas anteriormente no documento de entrada em uma cor invisível para o *scanner*. Uma vez encontradas as regiões, os blocos de dados são então segmentados dentro de imagens de caracteres. Uma outra técnica faz uso de marcas ou barras, conforme pode ser visto na figura 3.4. A área acima

das barras contém exatamente cinco caracteres, com altura, largura e espaçamento conhecidos.

A segmentação é um passo necessário e crucial para OCR e qualquer erro no processo afeta a performance de reconhecimento. A utilização de um só tipo de fonte com espaçamento fixo¹⁸ entre caracteres facilita a segmentação, entretanto ainda assim podem haver distorções causadas por desalinhamentos nos mecanismos de impressão e não uniformidades dos dispositivos digitalizadores que dificultam o processo. Além disso, descontinuidades, inclinações, contato ou aproximação entre os caracteres, bem como caracteres sublinhados, sobrescritos ou subscritos podem tornar a segmentação mais difícil.

FIGURA 3.4 - Exemplos de técnicas de segmentação que providenciam uma localização única para os caracteres (FREEDMAN, 1974).

¹⁸ - Diz-se de uma família, como a Courier, na qual a largura de todos os caracteres é igual, e cujo resultado impresso lembra os textos datilografados. No espaçamento proporcional a alocação de espaço horizontal dos caracteres é feita de modo que um caracter estreito ocupe um espaço menor do que um caracter largo (PFAFFENBERGER, 1992).

As barras, mostradas na figura 3.4 podem ser usadas também para detectar desvios nos caracteres, calculados através da diferença angular entre as barras e linhas verticais e horizontais verdadeiras. Os dados dos caracteres são rotacionados a fim de se corrigir os desvios.

Caracteres próximos podem ser resultantes da baixa qualidade de impressão ou de caracteres com espessura maior que o *pitch* padrão. A fusão de caracteres pode ocorrer ainda devido à baixa resolução do dispositivo de entrada ou de valores altos de *threshold* escolhidos para se evitar caracteres descontínuos (KAHAN et al., 1987).

Segundo NAGY (1968) a aproximação dos caracteres é responsável pela maioria dos erros ocorrido no reconhecimento de caracteres impressos manuscritos. Alguns pares ou trincas de caracteres, quando fundidos, podem se assemelhar a outros caracteres, como por exemplo: lo -> b, nn -> m, cl -> d, etc.

Alguns métodos desenvolvidos para a segmentação de caracteres impressos são abordados em ASCHER et al. (1971), HOFFMAN & McCULLOUGH (1971), CASEY & JIH (1983) e em KAHAN et al. (1987).

Na verdade, o grande desafio encontrado pelos pesquisadores consiste na segmentação de caracteres conectados na escrita cursiva. O problema é análogo à separação de fonemas no reconhecimento da fala. A figura 3.5 exemplifica o problema.

FIGURA 3.5 - Exemplos de problemas encontrados na segmentação de caracteres na escrita cursiva (WING, 1979).

Técnicas de segmentação para isolar os caracteres na escrita cursiva podem ser vistas em SAYRE (1973) e BALESTRI (1988).

5.4 - Técnicas de suavização (*smoothing*)

Imagens formadas por *scanners* podem carregar uma certa quantidade de ruídos, principalmente em equipamentos de resolução mais baixa. A fim de eliminar ruídos indesejáveis que causam distorções, características ambíguas e conseqüentes baixas taxas de reconhecimento, são aplicadas técnicas de suavização nos caracteres digitalizados. Além da remoção de ruídos, as técnicas de suavização realizam também funções de preenchimento (*filling*) para eliminação de descontinuidades nos caracteres. A maioria dos algoritmos de suavização baseia-se no deslocamento de uma janela 3x3 sobre a matriz binária do caracter.

O método de suavização utilizado por BROWN et al. (1988) consiste em se aplicar sobre a imagem um filtro

passa-baixa (média) não recursivo. Se mais do que a metade dos pixels em uma janela (w_k+1) por $(2k+1)$ centrada no pixel em questão for preto, então o mesmo é determinado preto, caso contrário é determinado branco. O problema desse método consiste em se determinar o tamanho adequado da janela que influencia diretamente a quantidade de suavização da imagem. O valor mais adequado, determinado nas experiências foi $k=1$ para caracteres (50×50) pontos e $k=2$ para caracteres maiores.

Na técnica de preenchimento de buracos descrita em GÜEDESSEN (1976) a vizinhança de cada "vazio" é expressa em uma palavra código de oito bits e combinada com uma condição de "AND" com as palavras códigos das máscaras apropriadas. As máscaras foram determinadas intuitivamente e o processo de preenchimento em pontos desordenados não apresentou uma redução considerável na taxa de erros do sistema em teste.

5.5 - Técnicas de afinamento

Um importante conceito no processamento de linhas de imagens é o de *thinning* ou afinamento. Afinamento, também conhecido como esqueletização ou transformação do eixo médio, é o processo de reduzir a largura de uma linha do objeto constituída por muitos pixels, para uma linha com apenas um pixel de largura, conforme ilustra a figura 3.6.

FIGURA 3.6 - (a) Imagem binária, (b) Imagem em níveis de cinza, (c) Esqueleto em *bitmap* (d) Esqueleto vetorizado. (SMITH, 1982)

As técnicas de afinamento objetivam principalmente a redução do volume de dados, através da remoção de informações redundantes e conseqüente aumento de eficiência dos sistemas de processamento de imagens.

No reconhecimento óptico de caracteres o afinamento simplifica a extração de características fundamentais, diminui o tempo envolvido no reconhecimento e auxilia o desenvolvimento de sistemas que envolvam o reconhecimento de vários tipos de fontes.

A tarefa de se determinar uma boa representação do esqueleto não é fácil pois depende de diversos fatores como por exemplo do padrão inicial a ser reconhecido, do dispositivo digitalizador, do valor de *thresholding* utilizado na binarização da imagem e do procedimento utilizado.

BEUN (1973) empregou uma técnica iterativa aplicada sobre uma imagem binária, onde durante o primeiro passo são marcados todos os pontos das bordas e no passo subsequente, são removidos todos os pontos que não causem desconexões ou que representem pontos finais das linhas. Esses passos são repetidos iterativamente até que nenhum ponto possa mais ser removido. Após isso, um último passo reduz o esqueleto a uma linha "8-conectada". Beun realiza ainda testes e impõe condições que evitam erosão excessiva, descontinuidades e presença de extremidades nos esqueletos não pertencentes à figura original.

Os métodos iterativos comumente usados são muito lentos para a maioria das aplicações e novas técnicas estão sendo ainda desenvolvidas. Uma vez que uma camada de pixels é extraída iterativamente, o número de iterações é governado pela parte mais larga do objeto e o tempo envolvido em cada iteração é governado pelo número de pixels existentes em cada célula do caracter.

A melhor maneira de se otimizar o grande tempo gasto no processo de afinamento é associá-lo a um *hardware* especializado. A nível de *software*, essa otimização vem sendo realizada através do desenvolvimento de algoritmos paralelizados que permitem que todos os pontos do caracter sejam analisados e transformados simultaneamente. Alguns trabalhos com essa abordagem podem ser vistos em TRIENDEL (1969), ZHANG & SUEN (1984), WANG & LU (1986) e em LI & BASU (1991).

Em BROWN et al. (1988) é descrito um eficiente algoritmo de esqueletização de caracteres, levando-se em consideração os conceitos de traço, modelo da caneta e relação largura-comprimento da linha. O algoritmo prevê ainda a detecção de certos artefatos ou falhas nos esqueletos em forma de raízes e cruzamentos, que podem prejudicar a extração de características.

Em SMITH (1987) é apresentado um resumo das principais técnicas de afinamento e análise de linhas de imagens digitalizadas com ênfase no reconhecimento de caracteres.

5.6 - Técnicas de normalização

Em alguns casos, são realizadas operações de normalização de tamanho, orientação, posição e espessura das linhas, visando facilidade no manuseio dos dados, extração de características e melhoramento da taxa de reconhecimento.

Em NAGY & TUONG (1970) são aplicadas técnicas de normalização de padrões baseadas na projeção geométrica, com o objetivo de reduzir cada padrão a uma forma canônica e aumentar a similaridade entre padrões de uma mesma classe.

Em NAYLOR (1971) é descrita uma técnica para normalização da inclinação de caracteres manuscritos, através da aplicação de transformadas lineares.

No *scanner* leitor desenvolvido por SATO et al. (1983),

a unidade de pré-processamento determina o comprimento horizontal do caracter através do número de pixels pretos em cada célula. O valor máximo de ocorrência de pretos é armazenado e dividido pelo número de pixels pretos encontrados em cada célula. Essa normalização permite que o valor da ocorrência de pixels pretos seja independente da velocidade com que o *scanner* manual é deslocado sobre os caracteres.

GÜDESEN (1976) devotou-se à análise quantitativa das principais técnicas de processamento, e concluiu que as técnicas de normalização foram as que apresentaram melhores resultados, reduzindo a taxa de erros a um fator 6.

CAPÍTULO 4

METODOLOGIAS DE RECONHECIMENTO

1 - EXTRAÇÃO DE CARACTERÍSTICAS

Após terem sido realizadas operações de pré-processamento a fim de se isolar e melhorar a qualidade da imagem dos caracteres, no estágio subsequente são extraídas características próprias dos mesmos, utilizadas no processo de reconhecimento.

Um sistema de reconhecimento de padrões em si é composto de duas partes principais. Primeiro, o padrão é tratado e características pré-determinadas são extraídas através de alguns parâmetros de medidas a fim de se obter máxima informação sobre o padrão. Essas medidas podem ser discretas, contínuas ou ambas. Em seguida as informações são direcionadas a um categorizador que realiza a classificação das amostras desconhecidas.

A seleção de características a ser usada na solução de um problema particular do Reconhecimento de Padrões é complexa e depende de muitos fatores tais como a taxa de erros desejada, tipos de erros que podem ser tolerados, grau de segurança requerido para a resposta, custo de implementação, velocidade de classificação e necessidades de armazenamento, entre outros (TROXEL, 1976).

Idealmente, as características extraídas dos

caracteres devem habilitar o sistema a distinguir corretamente uma classe de caracter das outras. Em qualquer processo de reconhecimento, a tarefa é extrair características do padrão de entrada e utilizar dados armazenados do conjunto de aprendizado ou treinamento em conjunção com essas características a fim de habilitar o reconhecimento. Como não existe uma maneira ótima de se extrair características, é usual selecionar algumas bem conhecidas ao invés de se selecionar aleatoriamente. Um conjunto muito grande de características envolve uma vasta quantidade de armazenamento e processamento de dados, além da necessidade de muitas amostragens para a construção do conjunto de treinamento. Isso é uma das razões que tornam o pré-processamento de padrões tão importante no reconhecimento de caracteres, como uma maneira de se reduzir a complexidade do processo de extração de características.

Especialmente no que diz respeito ao reconhecimento de caracteres, é importante determinar características que sejam invariantes aos diferentes tipos de estilos, tamanho, largura, inclinação do caracter, etc. Essas características devem fornecer informações suficientes de maneira a não permitir uma classificação ambígua.

2 - TÉCNICAS DE EXTRAÇÃO DE CARACTERÍSTICAS

Em termos das características utilizadas, as técnicas

de reconhecimento de caracteres podem ser classificadas dentro de três principais categorias (SUEN, 1982): características globais, distribuição de pontos e características geométricas e topológicas.

2.1 - Características Globais

As técnicas de globais extraem as características de todos os pontos que pertençam ao retângulo (*frame*) que circunscreve o caracter e tais características não refletem nenhuma propriedade local, geométrica ou topológica. Algumas dessas técnicas foram inicialmente desenvolvidas para o reconhecimento apenas de caracteres impressos à máquina.

2.1.1 - *Template matching* e correlações

Essa técnica toma como característica os estados (branco ou preto) de todos os pontos dentro da imagem do caracter e realiza uma comparação pixel a pixel com o padrão de referência. Conhecendo-se a posição e a estatística dos estados dos pontos na matriz do caracter podem ser criadas máscaras denominadas máscaras ao nível do pixel.

As técnicas de *template matching* comparam diretamente o caracter de entrada a um conjunto padronizado de protótipos armazenados em uma seqüência pré-estabelecida,

e o reconhecimento é determinado através de medidas de similaridade. Esse método é dependente do tamanho, portanto todos os caracteres de entrada devem ser normalizados antes de serem comparados com os padrões de referência. A "distância de Hamming¹⁹" é uma medida geralmente utilizada para medir a similaridade entre os padrões de entrada e os padrões armazenados (*templates*). O padrão que apresentar a máxima correlação é escolhido como sendo a identidade do caracter, conforme ilustra a figura 4.1.

Os métodos de comparação podem ser tão simples como a comparação um a um, ou tão complexos como análise de árvores de decisão onde apenas pixels selecionados são testados. Esse tipo de técnica é mais sensível a ruídos e não é adaptável às diferenças de estilo da escrita. O grande problema envolvido nesse tipo de técnica é a alta dimensionalidade do vetor de características resultante.

Com o aumento do número de tipos de fontes que aparecem em documentos, há necessidade do sistema possuir uma grande biblioteca com padrões armazenados, porém um grande tempo de processamento é gasto na procura e seleção de padrões, reduzindo a eficiência e precisão. A necessidade de intervenção humana para iniciar uma rotina de treinamento a cada vez que o sistema encontra uma nova fonte reduz os benefícios da produtividade do reconhecimento automático.

19 - A distância de Hamming foi definida em EVANGELIST (1982) como sendo o número de elementos da matriz no qual dois padrões diferem.

FIGURA 4.1 - OCR por comparação de matrizes (*matrix matching*) (SCHEIN, 1989).

Em ULLMANN & KIDD (1969), a fim de se compensarem os efeitos da variabilidade da escrita, são utilizadas várias máscaras, correspondentes a diferentes padrões pertencentes a uma mesma classe de caracter. São descritas ainda técnicas para se determinarem as máscaras, tais como independência estatística, ponderação não-logarítmica e não normalizada e correlação cruzada com máscaras proporcionais.

Em LEBOURGEOIS & EMPTOZ (1992), o estágio de casamento de matrizes procura pela melhor máscara que caracteriza a principal característica, definida pela maior área

retângular formada na imagem "ou-exclusivo" do caracter.

2.1.2 - Transformações e expansões em séries

Ao invés de utilizar todos os elementos da matriz como uma característica, a matriz inteira do caracter pode ser transformada em uma série, espectro ou vetor. Esse método ajuda a reduzir dimensionalmente o vetor de características e providencia características invariantes a algumas transformações globais como translação e rotação. Foram encontrados trabalhos que utilizam transformada de Fourier (GRANLUND, 1972), Walsh (ANDREWS, 1971), Haar e Hadamard (WENDLING et al., 1978), Karhunen-Loeve (GÜDESEN, 1976) e Hough (DAINESE & SAITO, 1990), entre outras.

As características e construções de máscaras são facilmente realizadas nesses tipos de técnicas, entretanto até mesmo um pequeno ponto de sujeira ou ruído pode desordenar o *frame* e afetar as posições das características e conseqüentemente o reconhecimento. Assim, as principais desvantagens de todas as técnicas globais são a dependência no alinhamento das posições, alta sensibilidade a distorções e variações de estilo e a alta demanda computacional requerida.

2.2 - Distribuição de pontos

As características derivadas das distribuições

estatísticas dos pontos representam uma outra maneira de se reduzir dimensionalmente o conjunto de características. Diferentes técnicas de reconhecimento que utilizam distribuição de pontos foram experimentadas:

2.2.1 - Zonificação (*zoning*)

Nessa técnica, o *frame* contendo o caracter é dividido em várias zonas sobrepostas e não sobrepostas e as densidades de pontos nessas diferentes regiões constituem as características.

Em HUSSAIN et al. (1972) é descrito um método de extração de características baseado simplesmente na contagem de pixels pretos em regiões bem definidas, utilizando-se como classificador a regra de Bayes.

Uma outro método que se baseia na densidade de pixels é ilustrado na figura 4.2, onde a área que contém o caracter é dividida em um conjunto de fileiras e o número de pixels em cada fileira é determinado (0 ou 1) de acordo com um limiar. Os limiares estabelecidos na figura 4.2 foram 3 e 4 para cada linha e coluna respectivamente. O código de saída gerado não tem relação direta com a geometria do caracter, por isso apesar da simplicidade, essa técnica só pode ser aplicada para uma única fonte altamente estilizada. BOWMAN & McVEY (1970) utilizaram características análogas, provenientes da combinação apropriada de medidas de densidades tomadas em linhas e

colunas, para determinar o reconhecimento.

FIGURA 4.2 Método de detecção da característica "densidade" determinada para o caractere "5" (FREEDMAN, 1974).

2.2.2 - Distância

O método conhecido como momento utiliza como características a soma das distâncias de todos os elementos a um determinado ponto de referência (por exemplo o *frame*). Os momentos de pontos pretos, por exemplo do centro de gravidade do caractere ou do sistema de coordenadas escolhido, são usados como características.

2.2.3 - N-tuples

Nessa técnica, a ocorrência de n elementos pretos ou

brancos escolhidos aleatoriamente ou ocorrências combinadas desses elementos são usadas como características e comparadas com as referências correspondentes. Por exemplo, seja pxq a matriz do caracter, N o subconjunto de 50 pontos escolhidos aleatoriamente de pxq , número de bits igual a dois e C o conjunto de caracteres. Os 2^2 estados de dois bits dos 25 pares de 2-tuples serão comparados com sua contraparte em cada caracter de C . O que apresentar o maior valor é dado como reconhecido.

Em CASEY & NAGY (1968) é desenvolvido um método onde cada ponto no espaço amostrado é um vetor binário obtido através da aplicação de n -tuples quase randômicas em todas as direções da imagem binária do caracter. As amostras similares são agrupadas em classes (*clustering*), baseadas em uma medida de distância de similaridade.

Em ULLMANN & KIDD (1969) também podem ser vistas algumas experiências com o método n -tuple.

2.2.4 - Características locais (*characteristic loci*)

Nessa técnica, para cada ponto branco no *frame* do caracter, são gerados códigos de quatro dígitos, onde cada dígito representa o número de vezes (no máximo duas a fim de limitar a dimensionalidade do conjunto de características) que os segmentos de retas são interceptados no ponto em uma das quatro direções perpendiculares: esquerda, direita, acima e abaixo do ponto

respectivamente. Códigos com três ou quatro zeros não são usados pois correspondem a pontos fora do *frame* da imagem do caracter. Como resultado, apenas 72 códigos são usados como características e os caracteres são reconhecidos comparando esses códigos com as referências armazenadas. A figura 4.3 exemplifica códigos gerados por algoritmos de características locais.

FIGURA 4.3 - Exemplos de códigos gerados por algoritmos de "características locais" (SUEN, 1982).

Em KNOLL (1969) podem ser vistas algumas experiências, utilizando-se características locais para reconhecimento de caracteres manuscritos isolados.

2.2.5 - Cruzamentos

Nesse método, calcula-se o número de vezes que um vetor transita de uma região de pixels brancos para uma região de pixels pretos. Essa informação indica o número de

segmentos de reta que o caracter possui ao longo do caminho do vetor. Características são medidas através do número de vezes que segmentos de reta são interceptados por vetores em direções específicas, como em AKIYAMA (1990) e GONZAGA & RODA (1990).

2.2.6 - Outras técnicas

Além dessas técnicas, podem ser usadas como características medidas físicas como altura e largura do caracter, determinadas através de varreduras horizontais e verticais. Pode-se utilizar tais medidas para computar informações como razão de aspecto (SUEN & SHILLMAN, 1977) e áreas cobertas por pixels brancos e pretos, entre outras. Várias medidas físicas tem sido usadas juntamente com outros grupos de características.

2.3 - Características geométricas e topológicas

Constituem as técnicas mais utilizadas por pesquisadores. Nessa abordagem, o reconhecimento diz respeito à disposição física (análise estrutural) do caracter e as características de interesse representam propriedades globais ou locais. Utiliza a lógica mais óbvia e sofisticada para a extração de características, e reduzem a saída quantizada do *scanner* a um conjunto de linhas verticais, horizontais, diagonais, cantos, arcos e

terminações de linhas. Esse conjunto de características, mais a posição relativa de cada característica, é processado utilizando-se uma lógica de reconhecimento. São usadas três metodologias principais:

2.3.1 - Obtenção de segmentos de linhas e bordas

Linhas e bordas na imagem digital podem ser detectadas deslocando-se, por exemplo, pequenas máscaras na imagem inteira do caracter, como ocorre em GUENCHI et al. (1969) e MORI et al. (1970). MARQUES (1990a) desenvolveu um sistema de reconhecimento onde foram incorporadas à biblioteca máscaras de tamanho fixo, destinadas à identificação de cantos, arcos e orifícios, e máscaras de tamanho variável, proporcionais à altura média dos caracteres presentes na cena, para detecção de bordas horizontais, verticais ou diagonais.

2.3.2 - Traçado do contorno do caracter (*contour tracing*)

Essa técnica é especialmente conveniente quando se utiliza *scanner* de ponto móvel. Os contornos podem ser extraídos explorando-se os pontos pretos na imagem do caracter no sentido horário ou anti-horário; por exemplo, conforme ilustra a figura 4.4, a partir do primeiro pixel preto encontrado, o feixe explorador vira à esquerda e avança uma unidade no sentido horário e faz o oposto quando

pontos brancos são encontrados. O traçado é completado quando o primeiro ponto é encontrado novamente. Depois de se registrarem as coordenadas dos pontos periféricos, podem ser extraídas características tais como extremidades, perímetro, orientação, posição, inclinação dos seguimentos de linha, ângulos agudos, *spikes* (pulsos), arcos e curvas, fendas, *loops* e círculos, pontos de inflexão, concavidades e convexidades, pontas e ganchos.

FIGURA 4.4 - Ilustração do método de traçado de contorno (TOUSSAINT & DONALDSON, 1970).

Um dos primeiros algoritmos de reconhecimento baseados na análise do contorno pode ser visto em GREANIAS et al. (1963). TOUSSAINT & DONALDSON (1970) também abordaram essa técnica de extração, utilizando diferentes algoritmos de reconhecimento.

A técnica mais utilizada para reconhecimento de caracteres manuscritos consiste no traçado de contorno (DUTTA, 1974). A principal desvantagem desse método é que sua eficiência depende da análise da sequência do traçado e apresenta dificuldades consideráveis quando se trabalha

com imagens gravadas previamente. Os mecanismos de traçado de curva se confundem em pontos de múltiplos cruzamentos ou quando os traços são total ou parcialmente coincidentes.

2.3.3 - Determinação do eixo médio (*centre-line*)

A determinação do eixo médio do caracter é obtido através do processo de esqueletização. Geralmente é utilizado o código da cadeia de 8 direções uma vez que os elementos da matriz possuem apenas vizinhos na direção horizontal, vertical e diagonal, conforme exemplifica a figura 4.5.

Muitas características podem ser extraídas dessa seqüência de códigos tais como extremidades, segmentos de linha reta vertical e horizontal, linha diagonal, desvio, concavidade, *loops*, interseções, garfos, ramos e nós.

FIGURA 4.5 - Imagem, esqueleto (S) e código (SUEN, 1982).

Exemplos de algoritmos que utilizam o código da cadeia para extração de características podem ser vistos em AGUI & NAGAHASHI (1979), CAI (1991) e LOPES & MARQUES (1992).

3 - ANÁLISE DAS TÉCNICAS DE EXTRAÇÃO DE CARACTERÍSTICAS

Todas as técnicas de distribuição de pontos apresentadas nas seções anteriores, apresentam de uma certa maneira tolerância a distorções a pequenas variações no estilo da escrita. Isso ocorre porque algumas características levam em consideração algumas informações topológicas, sendo particularmente claro no caso de características locais, cruzamentos e distâncias.

Nesse grupo de técnicas, é relativamente difícil a construção de máscaras nas técnicas de zonificação, *n-tuples*, cruzamentos e distâncias, principalmente devido ao grande número de combinações possíveis de características e às variações dinâmicas de tamanho e formato dos caracteres manuscritos. Por outro lado, tais técnicas proporcionam alta velocidade e baixa complexidade de implementação.

Todas as três técnicas de extração de características geométricas e topológicas apresentadas produzem uma representação do caracter em linhas, através de informações de extremidades, comprimento, orientação e interseções e ainda de características derivadas tais como extremidades livres, mudanças de inclinações, descontinuidades, tamanho

e perímetro. Características típicas incluem: segmentos de reta em várias direções (*strokes*), curvas (*bays, loops*) e ramos (*branches*), conforme exemplifica a figura 4.6.

FIGURA 4.6 - Exemplos de características estruturais típicas (SUEN et al., 1980 e SUEN, 1982) .

A principal vantagem das características geométricas e topológicas é sua alta tolerância às distorções e variações de estilo quando comparadas às outras técnicas. Isso explica o porquê de tais técnicas serem exploradas amplamente. Elas apresentam ainda tolerância a um certo grau de translação e rotação, entretanto o processo de extração de características é muito complexo, o que dificulta o desenvolvimento de máscaras. Uma vez implementadas, essas técnicas podem processar os caracteres a uma alta velocidade e independentemente de outras técnicas ou características suplementares.

Devido a essas propriedades, muitos trabalhos com abordagens geométrica ou topológica para extração de

características foram dedicados ao reconhecimento da escrita à mão, como em EDEN (1962), SHEINBERG (1970), TOU & GONZALES (1972), BROWN et al. (1988) e KERRICK & BOVIK (1988) e em STALLINGS (1976) e SUEN et al. (1980) são revistas várias técnicas de reconhecimento de caracteres manuscritos.

Em SUEN et al. (1980) são analisadas várias técnicas de reconhecimento, considerando-se velocidade de reconhecimento, (fortemente influenciada pelo *hardware* utilizado), sensibilidade às deformações da imagem causadas pelos ruídos, buracos, segmentos desconexos, distorções, variações de estilo, translação e rotação do caracter, e a implementação prática da técnica definida pela independência de técnicas suplementares e a facilidade de construção de máscaras. A tabela 4.1 sumariza a análise realizada em função dos critérios considerados.

Considerações da taxa de reconhecimento sozinha pode dar uma visão distorcida da performance de duas famílias diferentes de características devido a diversos fatores tais como: diferenças nos dados usados, resolução e subjetividade do escritor, diferenças entre rotinas e procedimentos, protótipos usados, regras e restrições impostas, diferenças no material usado, papel, instrumento e superfície de escrita; conseqüentemente é impossível afirmar com exatidão as vantagens e desvantagens dos vários tipos de características usados, e cada abordagem deve ser analisada em termos do tamanho do dispositivo, velocidade

de processamento e porcentagem de reconhecimento correto.

TABELA 4.1 - Comparação de várias técnicas de extração de características (SUEN et al., 1980).

Critério Técnicas de Reconhecimento	I. Tolerância para Deformação de Imagens					II. Facilidade de Implementação			
	Ruído	Distorção	Varição de Estilo	Translação	Rotação	Criação de máscaras	Velocidade	Complexidade	Independência
<u>1. Características Globais</u>									
"Template Matching"	+	0	-	-	-	+	-	0	-
Transformações	-	+	+	+	+	0	-	-	0
<u>2. Distribuição de Pontos</u>									
Zonificação	-	0	-	-	0	-	+	+	-
Momentos	0	0	-	+	+	0	-	0	-
"n-tuple"	0	-	0	-	0	-	+	+	0
Características Locais	-	+	+	+	0	0	+	+	-
Cruzamentos e Distâncias	-	+	+	+	0	-	+	+	-
<u>3. Características Geométricas e Topológicas</u>	-	+	+	+	0	-	+	0	+

4 - RECONHECIMENTO DE CARACTERES

Muitas técnicas de análise de características têm sido desenvolvidas e aplicadas ao reconhecimento de caracteres; muitas delas são exemplos de métodos de Reconhecimento de Padrões tradicionais e são usualmente adequadas para aplicações em domínios restritos.

O problema envolvido no Reconhecimento de Padrões consiste essencialmente em se selecionar um conjunto de padrões que concorde suficientemente com as características relativas ao padrão desconhecido. A eficiência do sistema de reconhecimento irá depender de quão significantes forem as diferenças entre as classes de padrões caracterizadas pelo conjunto de medidas e da precisão com a qual o critério de decisão classifica as medidas do padrão.

No processo de decisão são realizados testes a fim de serem obtidas as informações necessárias para a escolha entre alternativas. Se o processo é seqüencial, realiza-se apenas um teste de cada vez e o próximo teste é selecionado de acordo com os resultados de testes anteriores.

Os sistemas comerciais típicos de OCR são não-seqüenciais, empregando um número de subunidades operando em paralelo a fim de extrair informações de cada padrão de entrada. O uso de tais subunidades reduzem a informação redundante mas não influencia a performance de classificação.

As características são geralmente comparadas a uma lista de referências com uma base de conhecimento construída durante o processo de aprendizado, através da estatística de ocorrência das características obtidas na amostragem do conjunto de aprendizado, para classificação dos caracteres. Além disso, podem ser usadas como características medidas de distância, derivação do formato, comparação dos formatos (*shape matching*) e comparação

hierárquica de características na forma de árvores de decisão binárias.

A escolha da regra de decisão é fortemente influenciada pelos tipos de características detectadas. Um sistema de reconhecimento eficiente deve utilizar um detector de características associado a um classificador (SUEN, 1986).

O projeto de reconhecimento de Padrões depende da forma como as classes de padrões são caracterizadas (GONZAGA, 1991):

- por uma "lista de seus membros"
- pelas "propriedades comuns" compartilhadas por todos os membros, ou
- através de "propriedades de agrupamento" dentro do espaço de padrões.

A caracterização de uma classe de padrões pelo conceito de "lista de membros" sugere a técnica de comparações de padrões (*template matching*), onde o padrão de entrada é comparado um a um com padrões armazenados, enquanto que a caracterização pelo conceito de "propriedade comum" sugere a detecção de características que serão comparadas com características armazenadas no sistema. Uma classe de padrões pode ser caracterizada por suas "propriedades de agrupamento" dentro do espaço de padrões, quando seus elementos são vetores formados por números reais. Se as classes são definidas por grupos, podem ser empregados esquemas de classificação por distância mínima.

Segundo GONZAGA (1991) as diferentes técnicas usadas para solucionar os problemas de Reconhecimento de Padrões podem ser agrupadas em três metodologias: abordagem matemática, abordagem sintática, linguística ou estrutural e métodos heurísticos.

4.1 - Métodos matemáticos para Reconhecimento de Padrões

A abordagem matemática baseia-se em regras de classificação e utiliza conceitos de propriedades comuns e agrupamento e pode ser subdividida em duas categorias: determinística e estatística.

4.1.1 - Métodos determinísticos

O método determinístico não emprega explicitamente as propriedades estatísticas das classes de padrões, como por exemplo os classificadores por treinamento de padrões através de redes neurais como em SIMEK & TUNIS (1967), CARRIJO (1990), OSÓRIO (1991) e HAMMERSTROM (1993).

4.1.2 - Métodos estatísticos

No reconhecimento estatístico, esquematizado na figura 4.7, um padrão é representado por N características em um vetor de dimensão N , cujas coordenadas correspondem ao local onde foram tomadas as medidas. As características são

mapeadas em um espaço multi-dimensional, onde as classes de padrões individuais são agrupadas em *clusters*. Se as características foram adequadamente escolhidas, os diferentes *clusters* irão cair suficientemente longe uns dos outros. Os processos de decisão baseiam-se em uma medida de similaridade, expressa em termos de distância, funções de semelhança e funções discriminantes.

Segundo MANTAS (1987) a abordagem estatística pode ser subdividida em classificações não-paramétrica, paramétrica e seqüencial, embora existam outras abordagens baseadas em análise de *clusters* (CASEY & NAGY, 1968; ASCHER, 1971 e CUNHA, 1992) e lógica nebulosa (SIY & CHEN, 1974).

FIGURA 4.7 - Diagrama em blocos de um sistema de Reconhecimento estatístico de Padrões (GONZAGA, 1991).

A classificação baseada na análise de agrupamentos pode ser bastante atrativa em situações não supervisionadas, quando o número de classes não é conhecido precisamente. A lógica *fuzzy* fornece resultados mais realísticos quando não existe um conhecimento prévio e a

probabilidade não pode ser calculada.

Os métodos de classificação não-paramétricos utilizam funções discriminantes a fim de separar diferentes classes de padrões ao longo de hiperplanos definidos em um dado hiperespaço. O método de vizinhança mais próxima (*Nearest Neighbour*) cria mais de um hiperplano para separar duas classes de padrões e seleciona o hiperplano mais próximo de ambos, como em NAYLOR (1971). Outros métodos que utilizam funções discriminantes podem ser vistos em KNOLL (1969), GRANLUND (1972), TOU & GONZALES (1972), SCHÜRMANN (1978), LOY & LANDAU (1982) e KIMURA et al. (1987).

O método de classificação paramétrica mais conhecido é o baseado na regra de Bayes, onde um determinado vetor de características de entrada pertence a uma classe particular, denominada j , se a razão de semelhança entre duas classes de padrões, i e j , for maior que a razão das probabilidades de ocorrência da classe j sobre a i . A regra de Bayes evidentemente baseia-se em um conhecimento anterior.

Na prática, quando é utilizado um grande número de padrões, as funções de densidade das classes podem ser estimadas ou aprendidas das amostras e então uma regra de classificação ótima pode ser obtida. Se a forma paramétrica das funções de densidade para cada classe é conhecida, então apenas parâmetros precisam ser aprendidos dos padrões de entrada.

Sistemas que utilizam a regra de Bayes como

classificador podem ser vistos em ULLMANN & KIDD (1969), TOUSSAINT & DONALDSON (1970), HUSSAIN et al. (1972), CASEY & JIH (1983), BERNHARDT (1984), BOZINOVIC & SRIHARI (1989) e KAHAN et al. (1987).

4.2 - Reconhecimento de Padrões Sintático

No reconhecimento sintático, esquematizado na figura 4.8, um padrão é representado por uma seqüência (*string*), árvore ou em grafos relacionais que relacionam as primitivas dos padrões. Essa abordagem realiza uma analogia entre a estrutura dos padrões e a sintaxe de uma linguagem descritiva (geralmente denominada *Picture Description Language*), e o reconhecimento de cada padrão é geralmente feito através de um procedimento de análise gramatical.

FIGURA 4.8 - Diagrama em blocos de um sistema de Reconhecimento sintático de Padrões (GONZAGA, 1991).

Após a seleção das primitivas, o passo seguinte consiste na criação de uma gramática que gera uma linguagem que descreve os padrões em consideração. Aumentando-se o poder descritivo da linguagem, aumenta-se também a complexidade da análise sintática realizada. A seleção de uma determinada gramática é afetada pelas primitivas selecionadas e pela eficiência da análise esperada.

Em AGUI et al. (1979) é aplicado um método sintático para o reconhecimento de caracteres coreanos, nos quais a adição de linhas determinam caracteres fundamentais e a adição seqüencial de caracteres determinam caracteres combinados. Além disso, caracteres coreanos são determinados através de caracteres fundamentais combinados de acordo com regras de posicionamento. Devido a essas características, a análise estrutural apresentando um método sintático é eficiente para o reconhecimento do alfabeto coreano. Para a obtenção dos grafos dos padrões, foram extraídos quatro tipos de características: extremidade, curva, ramo e cruz. As características representam os nós que são numerados de acordo com a ordem de extração enquanto que os traços que conectam os nós são aproximados por retas com direção. O grafo que representa as relações estruturais de um caracter é analisado por uma gramática em árvore, através de uma extração seqüencial de caracteres básicos.

Outros exemplos de sistemas de reconhecimento sintático de caracteres são abordados em EDEN (1962),

PERROTO (1963), ALI & PAVLIDES (1977), HUNDT & SCHERL (1984), BROOKS & NEWELL (1985) e BROWN et al. (1988).

A fim de solucionar os problemas de ruídos e distorções em padrões ambíguos, têm sido utilizadas linguagens estocásticas (Thomason, 1986) e técnicas de relaxação probabilística. Uma gramática estocástica, com probabilidades associadas às regras gramaticais, geram sentenças com distribuição de probabilidade.

4.3 - Métodos heurísticos para Reconhecimento de Padrões

A abordagem heurística é baseada na intuição e experiência humana, fazendo uso dos conceitos de lista de membros e de propriedades comuns. As técnicas de Inteligência Artificial são amplamente empregadas nesta abordagem por permitirem inferências lógicas em bancos de conhecimento estabelecidos previamente.

Algoritmos que combinam técnicas heurísticas para reconhecimento de caracteres podem ser vistos em GREANIAS (1963), GENCHI et al. (1968), MORI et al. (1970), SAYRE (1973), BORNAT & BRADY (1976), KERRICK & BOVIK (1988), BOZINOVIC & SRIHARI (1989), AKIYAMA & HAGITA (1990), GONZAGA & RODA (1990) e LOPES & MARQUES, (1992).

Apesar das subdivisões apresentadas, um modelo híbrido é a única solução em problemas de reconhecimento práticos, empregando-se cada abordagem conforme necessário. ITO & CHUI (1978) e LEBOURGEOIS & EMPTOZ (1992) combinaram uma

abordagem sintática e uma abordagem estatística para completar o reconhecimento.

5 - PÓS-PROCESSAMENTO

Define-se pós processamento como o conjunto de operações processadas sobre o dado obtido na saída da etapa de reconhecimento.

Se a saída do classificador não apresenta uma correlação exata do caracter de entrada com os protótipos armazenados, ou se essa correlação for aproximadamente igual a dois ou mais protótipos (ambigüidades)²⁷, é muito provável que acontecerá um erro de classificação. Como esse erro muitas vezes pode ser dispendioso para o sistema, é preferível colocar o caracter na categoria de rejeitado. A partir desse ponto, várias estratégias alternativas podem ser seguidas (FREEDMAN, 1974):

- O documento completo pode ser rejeitado e mantido para processamento manual por um operador humano;
- O caracter rejeitado pode ser visualizado em um monitor, permitindo que um operador insira sua interpretação via teclado manual;
- Se existe uma redundância suficiente, é possível aplicar uma correção baseada no contexto nos caracteres rejeitados, entretanto a análise de contexto não pode ser aplicada em

²⁷ — No campo do reconhecimento da fala, em particular, a entrada não será determinada até que o sistema use contexto semântico para auxiliar o processo de reconhecimento. Um caso extremo pode ser exemplificado com as palavras "too", "two" e "to" que possuem a mesma pronuncia e só podem ser diferenciados apenas pelo contexto da sentença (WARD & BLESSER, 1985).

caracteres não relacionados a seus vizinhos.

Um exemplo típico das operações de pós-processamento consiste no estágio classificador apresentar em sua saída as melhores opções, em ordem decrescente de probabilidade, para que o pós-processador realize uma análise contextual que determine qual o candidato que melhor se enquadra dentro da palavra ou sentença em questão. O estágio pós-processador pode conter apenas um dicionário de consulta ou incluir técnicas sofisticadas de análise sintática e semântica (MARQUES, 1990c).

Em KAHAN et al. (1987) por exemplo, é construído um arranjo onde a cada escolha proveniente dos estágios de classificação é determinada uma figura de mérito. Nesse ponto, podem ser usados dois tipos de informação contextual: contexto da disposição, onde são tomadas informações da linha de base e da localização de cada caracter em relação à seus vizinhos e contexto lingüístico, que inclui regras ortográficas, gramaticais e de pontuação.

Outras técnicas de análise do contexto são abordadas em SAYRE (1973), SCHÜRMANN (1978), SRIHARI et al. (1983), BERNHARDT (1984), BOZINOVIC & SRIHARI (1989) e LEBOURGEOIS et al. (1992).

Em BLESSER et al. (1974) foi definido o conceito de contexto gráfico (não semântico) onde ambigüidades são resolvidas por regras inferidas através do exame de caracteres no contexto em que aparecem, ou examinando-se as relações entre partes do caracter sob consideração.

Em HARMON (1972) são revistas e analisadas várias técnicas de detecção e correção de erros baseada no contexto semântico e sintático e em ELLIMAN & LANCASTER (1990) é apresentada uma revisão de técnicas de análise contextual, como os métodos baseados em Markov e busca em dicionários. Várias técnicas de armazenamento de informação em dicionários são analisadas e é discutida a importância da escolha do contexto adequado a cada aplicação.

A importância desse estágio para resolver ambigüidades e melhorar a performance global do sistema é indiscutível. Segundo SUEN (1986) até mesmo seres humanos, que possuem os melhores leitores ópticos treinados, cometem 4% de erros na ausência de contexto. Os fatores contextuais conhecidos pelos seres humanos são: sequência de letras, dependência da palavra, estrutura da sentença e fraseologia, estilo, assunto, bem como compreensão, inferência, associação, adivinhação, previsão e imaginação, empregados naturalmente durante o processo de leitura (SRIHARI et al., 1983).

O conhecimento sobre o tipo de cena representada é uma fonte rica para avaliar a consistência das ambigüidades. Considerando-se por exemplo a figura 4.9, a quarta palavra pode ser "CLIP" ou "CUP". O conhecimento sobre a parte do texto no qual a frase ocorre ou sobre o assunto em discussão é a única maneira de resolver a ambigüidade.

A correção de erros em textos pelo computador é importante para a flexibilidade na comunicação homem-computador. A capacidade dos sistemas atuais em corrigir

textos através do reconhecimento de palavras impressas e manuscritas ainda é muito limitada.

HAND ME THE CLIP

FIGURA 4.9 - Ambigüidade de caracteres, resolvida apenas pelo conhecimento do assunto (BORNAT & BRADY, 1976).

CAPITULO 5

ALGORITMO PARA RECONHECIMENTO DE CARACTERES MULTI-FONTE

1 - INTRODUÇÃO

Os primeiros *softwares* de OCR freqüentemente incluíam um *hardware* dedicado e executavam uma comparação de padrões (*template matching*). Esses primeiros programas podiam reconhecer apenas os caracteres para os quais o programa possuísse o modelo. Atualmente ainda existem alguns programas desse tipo no mercado (DVORAK, 1993).

A partir da década de 70, contribuições considerando diversos aspectos psicológicos e neuro-motores, relativos ao reconhecimento da escrita e aos aspectos visuais envolvidos no processo de leitura, ajudaram a compreender um pouco mais os complicados aspectos da percepção humana e da formação dos caracteres. Alguns trabalhos nesse sentido podem ser vistos em BOUMA (1971), BLESSER et al. (1974), BLESSER et al. (1976), NAUS & SHILLMAN (1976), SUEN & SHILLMAN (1977), BRADY (1981), HABER & HABER (1981) e COX et al. (1982).

Com os desenvolvimentos ocorridos na Inteligência Artificial e na psicologia cognitiva na década de 80, houve o aparecimento de uma nova geração de *softwares*, o que possibilitou um grande avanço em relação aos primeiros esforços. A estratégia encontrada foi a utilização de

softwares mais inteligentes que utilizam comparações de características, baseadas em vetores. Tais algoritmos de comparação são mais flexíveis e podem lidar com maior número de distorções e ambigüidades. Ao invés de se comparar fisicamente cada caracter com padrões armazenados na memória, os caracteres são identificados pela presença ou ausência de um conjunto particular de características. Tais características são independentes da fonte ou do tamanho do caracter, conforme ilustra a figura 5.1.

Figura 5.1 - Exemplos de características que identificam cada caracter pela presença ou ausência de um conjunto particular (SCHEIN, 1989).

A grande dificuldade no reconhecimento multi-fonte é o grande número de formas de caracteres, geradas por

diferentes fontes, utilizadas para representar uma letra. É interessante notar que amostras de caracteres representando a mesma letra nem sempre são idênticas em forma ou tamanho, entretanto seres humanos podem reconhecer a maioria delas, em parte devido à experiência e em parte devido à detecção de pistas adicionais (características) que auxiliam o reconhecimento (SUEN, 1982).

A propriedade fundamental da escrita que faz com que a comunicação seja possível é que as diferenças entre caracteres distintos são mais significativas que as diferenças entre diferentes representações para o mesmo caracter. Há quem argumente que isso nem sempre é verdade, porque certos símbolos diferentes ("O e 0" por exemplo) podem ser distingüidos por seres humanos graças à capacidade de análise de contexto (MARQUES, 1990c).

O reconhecimento por computador, em contraste com os mecanismos de percepção humana, apresenta uma boa performance apenas para as fontes utilizadas no treinamento. A fim de se construírem máquinas que possam reconhecer caracteres com a precisão humana, há a necessidade de se estabelecer primeiramente quais os conhecimentos específicos utilizados pelos seres humanos na classificação, o que ainda não é totalmente conhecido.

Com o aparecimento de sistemas para reconhecimento automático de caracteres provenientes de diferentes fontes nos processos de geração e *design* de fontes utilizando computadores, surgiu a necessidade de se determinar uma

descrição mais genérica dos caracteres a fim de identificá-los. Tal descrição deve ser capaz de distinguir claramente todos os aspectos que contribuem para a definição das letras em si, mas também todos os aspectos que são utilizados como adornos e que contribuem pouco ou nada para a identificação das mesmas.

A extração de características exerce um importante papel no reconhecimento de padrões. Indubitavelmente o ponto crucial no reconhecimento de caracteres consiste na detecção de características apropriadas. Quanto mais consistentes forem as características utilizadas, maior será a eficiência do algoritmo para reconhecer novas fontes

Para o desenvolvimento deste trabalho, procurou-se estabelecer características tais como a curvatura ou inclinação de um segmento de reta, espaço (*gaps*) entre duas retas, interligações entre as retas, distâncias relativas, proporcionalidade entre os traços e outras características geométricas e topológicas do caracter. Através da transferência desses conhecimentos para o computador é possível reconhecer os caracteres automaticamente, embora a capacidade humana de generalizar, isto é, agrupar consistentemente objetos fisicamente diferentes em uma classe comum, seja uma tarefa difícil de ser imitada.

2 - DESENVOLVIMENTO DO ALGORITMO PROPOSTO

O trabalho objeto desta dissertação, consiste na

implementação de um algoritmo de reconhecimento de caracteres alfabéticos isolados, escrito em linguagem C, e desenvolvido em um microcomputador do tipo PC-386 SX com placa gráfica padrão VGA. A figura 5.2 mostra o esquema geral do sistema de reconhecimento utilizado. Ao programa desenvolvido cabe as tarefas de pré-processamento da imagem digitalizada, extração de características e classificação dos caracteres de entrada.

Figura 5.2 - Esquema geral do sistema de reconhecimento de caracteres.

2.1 - Especificação das imagens de entrada

O primeiro passo para a execução deste trabalho foi a definição da fonte que seria utilizada como base para classificar os demais caracteres. A fonte escolhida foi a *Univers*, uma fonte de estética simples, sem serifas,

extraída do editor de textos *Word Perfect*.

As características utilizadas na etapa de classificação foram determinadas examinando-se os traços fundamentais que constituíam seus caracteres. Entretanto, como o programa visava ser aplicado a outras fontes, procurou-se determinar o maior número possível de características comuns que pudessem expressar cada caracter. Para tanto, foram incluídas no conjunto de treinamento fontes com um certo grau de estilização como a *CG Times*, *Courier*, *Helvética*, *ITC Avant Gard Gothic Book*, *ITC Bookman Demi*, *New Century School Book*, *Palatino* e *Times Roman*, que sugeriram possíveis variações na disposição física dos traçados dos caracteres.

A fim de se analisar a eficiência do algoritmo na presença de novas fontes, foram incluídas na etapa de teste caracteres provenientes das seguintes fontes: *Modern*, *Arial*, *Arial Rounded*, *Bookman Old Style*, *Britannic Bold*, *Century Gothic*, *Footlight Mt Light*, *Impact* e *MS Line Draw*.

Todas as fontes utilizadas podem ser vistas no apêndice 1.

2.2 - Aquisição da imagem

Na etapa de aquisição da imagem utilizou-se um *scanner* manual DFI HS-3000 PLUS de 100 dpi que digitaliza a imagem do caracter com os ruídos e distorções inerentes ao processo.

Devido à confiabilidade do sistema relacionar-se diretamente com o tamanho do caracter a ser reconhecido, procurou-se estabelecer também a altura e largura mínimas que permitissem o reconhecimento. Assim a escolha da resolução da imagem deveria satisfazer o requisito de tamanho mínimo do caracter a ser digitalizado.

Como a saída gerada pelo *scanner* no formato .IMG é compactada em meios-tons, houve necessidade de descompactá-la através de um algoritmo de *dithering*, desenvolvido por GONZAGA & FRANÇA (1993), obtendo-se uma imagem pronta para ser processada e visualizada em um *display* VGA com resolução de 640 x 480 pontos. A matriz binária obtida é composta por pixels pretos (valor 0), relativos aos pontos do caracter, e pixels brancos (valor 15), relativos ao fundo da imagem.

2.3 - Pré-processamento

A etapa de pré-processamento consiste primeiramente na procura de início e fim do caracter onde o algoritmo desenvolvido procura as coordenadas do primeiro ponto de topo (XT,YT) e da base do caracter (XB,YB), através de uma varredura das colunas, da esquerda para a direita, realizada pelas rotinas *readcolorT()* e *readcolorB()*, e certifica se os pontos encontrados são pontos de ruídos ou pertencem ao caracter através das rotinas *pesquisaTE()* e *pesquisaBE()*. Todas as rotinas desenvolvidas podem ser

vistas no algoritmo de reconhecimento localizado no apêndice 2.

Através da obtenção desses pontos (YT,YB), pode-se determinar a altura (h) do caracter, que será utilizada como medida relativa, necessária ao processo de extração de características, conforme pode ser visto na figura 5.6.

Um outro parâmetro de medida necessário é a determinação da largura (l) dos caracteres, obtida através do histograma relativo às colunas da imagem, e realizada pela rotina `histogramming()`. A soma de todos os pixels pretos em cada coluna (y_i) é colocada em um vetor-linha que é então binarizado de acordo com um *threshold* (a fim de serem desconsiderados os possíveis pontos de ruído existentes no fundo da imagem do caracter). Realiza-se então a derivada do vetor, expressa pela fórmula:

$$dv = \frac{dx}{di} = \frac{X_{i+1} - X_i}{(i+1) - i} \quad (1)$$

Com isso, é possível a obtenção do ponto de descontinuidade positiva (XA) que indica uma transição de preto para branco, e do ponto de descontinuidade negativa (XZ), que indica uma transição de branco para preto. A distância entre esses pontos determina a largura (l) do caracter, conforme ilustra a figura 5.3.

Figura 5.3 - Determinação da largura dos caracteres, necessária à etapa de extração de características

Como algumas imagens geradas por *scanners* carregam uma grande quantidade de ruídos, foi desenvolvida uma rotina (*clean()*) de suavização para eliminação dos mesmos. A técnica empregada não permite a existência de linhas de ruídos horizontais, verticais ou diagonais com mais de dois pixels de largura, através de uma análise da vizinhança do

pixel em questão. Isso é equivalente a considerar um certo pixel preto (i,j) como caracter (nível lógico 0), apenas se os pixels $(i+1,j)$, $(i,j+1)$, $(i+1,j+1)$ apresentarem simultaneamente nível lógico 0 e pelo menos $(i+2,j)$ ou $(i,j+2)$ também possuir um nível lógico 0, conforme pode ser visto na figura 5.4. Caso essa condição não seja satisfeita, o valor 0 do pixel (i,j) é trocado por 15 (condição de fundo da imagem), eliminando-se dessa maneira o ruído.

Figura 5.4 - Verificação da conectividade do pixel (i,j) em relação a sua vizinhança, necessária à rotina de eliminação de ruídos.

2.4 - Definição das características a serem utilizadas:

No desenvolvimento deste trabalho, procurou-se estabelecer quais as linhas fundamentais que descrevem cada caracter, através da observação do comportamento de tais linhas em diferentes representações para uma mesma letra.

Observando-se as representações para a letra "A" na figura 5.5, pode-se perceber que independentemente do

número de retas que a formam, existe sempre a idéia de duas áreas, uma aberta e outra fechada, na região interior às bordas do caracter, mesmo que essa delimitação física não ocorra totalmente.

Figura 5.5 - Diferentes representações para a letra "A" (BLESSER et al., 1974).

KAHAN et al (1987) perceberam que o reconhecimento omni-fonte é melhor determinado, aplicando-se regras similares às descrições relatadas por pessoas. Se questionada a definir a letra "A", por exemplo, uma pessoa geralmente descreverá o relacionamento entre três traços e não a definição dada acima. Isso ocorre talvez devido à maneira como primeiramente somos ensinados a escrever, ou seja através de traços e não descrevendo contornos ou áreas.

Apesar da descrição dos caracteres em função de suas linhas fundamentais não ser apenas o único tipo de característica usada no reconhecimento humano (SUEN, 1982) e também de não resolver definitivamente o problema para o reconhecimento de fontes mais estilizadas sem um

treinamento prévio, pode-se conseguir uma boa taxa de reconhecimento, prevendo-se muitas das variações possíveis entre caracteres, inclusive rotação e alguns tipos de distorções.

Analisando-se os traços fundamentais de alguns tipos de fontes utilizados no treinamento, sem levar em consideração as características estéticas, como serifas, proporções entre as retas, e simetria, de uma maneira genérica, pôde-se estabelecer um conjunto de características para cada letra maiúscula do alfabeto romano:

- Letra "A": todos os "A" observados são compostos por duas retas oblíquas, unidas na região superior e separadas por um traço horizontal localizado aproximadamente em regiões iguais ou maiores que 50% da altura, tomando-se como referência o topo do caracter.
- Letra "B": formada por uma reta vertical à esquerda, e dois arcos fechados em relação à mesma.
- Letra "C": formada por uma curva quase totalmente fechada, com abertura à direita.
- Letra "D": Uma reta vertical à esquerda e um semi-círculo fechado em relação à mesma.
- Letra "E": Uma reta vertical à esquerda unida a três retas horizontais localizadas aproximadamente nas regiões superior, média e inferior da mesma.
- Letra "F": Uma reta vertical à esquerda, unida à duas retas horizontais, dispostas na região média e superior da

mesma.

- Letra "G": Um arco quase totalmente fechado, com uma abertura à direita, cuja extremidade inferior está unida à um traço horizontal.

- Letra "H": formada por duas retas verticais paralelas separadas por uma reta horizontal, localizada aproximadamente na região média das mesmas.

- Letra "I": formada por uma reta vertical.

- Letra "J": formada por uma reta vertical à direita, unida na extremidade inferior à um semi-círculo com abertura para cima.

- Letra "K": formada por uma reta vertical, unida na região média superior a uma reta oblíqua crescente, e na região média inferior a uma reta oblíqua decrescente.

- Letra "L": formada por uma reta vertical à esquerda, unida na extremidade inferior à uma reta horizontal.

- Letra "M": formada por duas retas aproximadamente verticais e paralelas, unidas entre si por duas retas oblíquas decrescente e crescente, respectivamente.

- Letra "N": formada por duas retas verticais paralelas, separadas entre si por uma reta oblíqua decrescente.

- Letra "O": formada por um arco totalmente fechado.

- Letra "P": formada por uma reta vertical, unida na região média superior a um semi-círculo fechado à esquerda.

- Letra "Q": formada por uma curva totalmente fechada, em cuja parte inferior existe uma "cauda" que pode possuir diferentes formatos (curva, reta horizontal, oblíqua,

etc.).

- Letra "R": formada por uma reta vertical à esquerda, unida por um arco na região média superior e por uma reta oblíqua decrescente na região média inferior.

- Letra "S": formada por um semi-círculo com abertura para baixo na extremidade superior, unido através de uma reta oblíqua decrescente a um semi-círculo com abertura para cima, na extremidade inferior.

- Letra "T": formada por uma reta horizontal na região superior, unida em sua parte central a uma reta vertical.

- Letra "U": formada por duas retas verticais paralelas, unidas em suas extremidades inferiores a um arco com abertura para cima.

- Letra "V": formada por uma reta oblíqua decrescente mais à esquerda, unida à uma reta oblíqua crescente.

- Letra "W": formada por quatro retas oblíquas, decrescente, crescente, decrescente e crescente, da esquerda para a direita respectivamente, unidas em suas extremidades.

- Letra "X": formada por duas retas oblíquas decrescente e crescente, que se cruzam na região média.

- Letra "Y": formada por uma reta oblíqua decrescente à esquerda, unida por uma reta oblíqua crescente a direita, ambas unidas a uma reta vertical localizada aproximadamente na região média da largura do caracter.

- Letra "Z": formada por uma reta horizontal superior cuja extremidade à direita está unida a extremidade mais à esquerda de uma reta horizontal inferior, por uma reta

oblíqua decrescente.

2.5 - Extração de características e metodologia de reconhecimento

Após a localização do caracter na imagem, conforme descrito no item 2.3, realiza-se a extração de características. As características geométricas e topológicas são determinadas diretamente da imagem em termos de linhas fundamentais (horizontais, verticais, diagonais e curvas) do traçado dos caracteres, determinando uma primeira classificação, e separando os caracteres que apresentarem certas propriedades comuns. Tais características devem descrever suficientemente os caracteres e permitir a identificação dos mesmos na fase de classificação.

As possíveis classes foram determinadas previamente e especificadas através de características locais definidas, examinando-se cuidadosamente o formato das fontes de caracteres utilizadas na fase de treinamento, com o objetivo de se identificar a presença de determinadas propriedades e as relações estruturais significativas entre elas.

O reconhecimento é então determinado, analisando-se os comportamentos binários dos caracteres em relação à linhas de corte determinados na imagem e comparando-os com protótipos armazenados, em uma técnica heurística

semelhante à utilizada em GONZAGA & RODA (1990). Apesar da desvantagem de não se analisar os caracteres globalmente, e sim em regiões fixas, essa técnica foi escolhida devido à grande facilidade de extração de características em relação às outras técnicas de extração de características topológicas.

Inicialmente, o algoritmo desenvolvido baseou-se nas relações estruturais das letras maiúsculas da fonte Universal. Decidiu-se então dividir os 26 caracteres em sub-grupos, cada um deles contendo letras com propriedades comuns definidas pela leitura do comportamento binário produzido pelas linhas de cortes horizontais V1 e V2, realizadas em regiões fixas de 30% e 80% da altura do caracter, conforme mostra a figura 5.6. A fim de manter o algoritmo independente do tamanho dos caracteres, as linhas de corte horizontais e verticais são definidos relativamente à altura e largura dos caracteres, respectivamente.

Armazenando-se o comportamento de cada linha na forma de um vetor e derivando-o, de maneira análoga à determinação da largura dos caracteres, é possível analisar o número de transições entre o fundo e a imagem do caracter, isto é, o número de segmentos de retas que cada linha de corte intercepta. Através de uma medida de distância entre essas transições (positivas ou negativas) dentro de uma mesma linha de varredura, pode-se determinar por exemplo, a espessura dos caracteres e através de

medidas de distância entre as transições das duas linhas de corte pode-se extrair informações tais como inclinação e curvatura do segmento de reta. Na figura 5.6, as ordenadas x_1 , x_3 , x_5 e x_7 constituem pontos de transições negativas, enquanto que as ordenadas x_2 , x_4 , x_6 e x_8 constituem pontos de transições positivas.

Figura 5.6 - Visualização da classificação determinada em função do comportamento binário das linhas de cortes para o caracter "A".

2.6 - Determinação das classes

Devido ao fato que diferentes fontes possuem características estéticas diferentes, analisar o comportamento do traçado de uma única fonte constitui um conjunto muito pobre de características a serem tomadas como modelo, o que pode resultar em uma baixa taxa de

reconhecimento. Dessa maneira, procurou-se incorporar ao protótipo desenvolvido para a fonte *Univers*, o comportamento de outras fontes em relação aos planos de corte, bem como a inclusão de novos testes que pudessem distinguir entre um maior número de características ambíguas.

Para tanto, realizou-se um estudo que permitiu avaliar todas as possibilidades quanto ao número de retas (transições positivas ou negativas) possíveis de serem interceptadas nas duas linhas de corte fixadas em 30% e 80% da altura do carácter. Como os caracteres romanos maiúsculos são formados por no máximo 4 retas fundamentais, referentes às letras "M" e "W", e no mínimo 1 reta, referente à letra "I", pôde-se estabelecer 16 combinações (classes) na forma (nV1/nV2), relativas ao número (n) de transições determinadas em V1 e V2 respectivamente: (1/1), (1/2), (1/3), (1/4), (2/1), (2/2), (2/3), (2/4), (3/1), (3/2), (3/3), (3/4), (4/1), (4/2), (4/3) e (4/4).

Considerando-se características tipográficas, como simetria, proporcionalidade, espessura dos traços, presença de serifas e também alguns tipos de distorções e pequenas rotações, realizou-se um outro estudo visando determinar qual ou quais situações (combinações) poderiam ocorrer para cada letra. Assim, as combinações (1/4), (4/1) e (1/3) foram desconsideradas por constituírem caracteres alfabéticos improváveis. Algumas letras como "W" e "M" foram agrupadas em 7 classes distintas, necessitando de

diferentes testes para serem discriminadas. Enquanto as simples definições determinadas no item 2.4 deveriam ser suficientes para prover o reconhecimento, o mesmo não ocorre devido à limitada análise realizada pelas linhas de cortes fixas e também devido às combinações de propriedades estéticas diferentes que podem mascarar essas características no ponto onde estão sendo tomadas. A complexidade do problema pode ser visualizada na figura 5.7. Seguindo raciocínio análogo para as outras letras, decidiu-se agrupá-las então em 11 classes, de acordo com o comportamento (nV1/nV2) estabelecido:

- Classe 1: (2/4) e (3/4)
- Classe 2: (4/2) e (4/3)
- Classe 3: (4/4)
- Classe 4: (3/3)
- Classe 5: (3/2)
- Classe 6: (2/3)
- Classe 7: (3/1)
- Classe 8: (2/1)
- Classe 9: (1/2)
- Classe 10: (1/1)
- Classe 11: (2/2)

Algumas possibilidades para determinadas letras foram desprezadas por representarem características incomuns em fontes tipográficas, embora para o reconhecimento de caracteres manuscritos deveriam ser previstas.

Figura 5.7 - Diferentes comportamentos produzidos para as letras "M", "N" e "W", relativos ao número de transições interceptadas nas linhas de varredura V1 e V2.

As combinações (2/4) e (3/4), bem como (4/2) e (4/3) foram agrupadas por possuírem características análogas, que permitem suas discriminações. A tabela 5.1 sumariza as possibilidades de ocorrência dos caracteres dentro das diversas classes. A determinação das classes é realizada no algoritmo de reconhecimento através da rotina findclasses().

TABELA 5.1 - Possibilidade de ocorrência das letras dentro das diversas classes

	C1 2/4 3/4	C2 4/2 4/3	C3 4/4	C4 3/3	C5 3/2	C6 2/3	C7 3/1	C8 2/1	C9 1/2	C10 1/1	C11 2/2
A								X	X	X	X
B											X
C								X	X	X	X
D											X
E				X	X		X	X	X	X	X
F					X		X	X	X	X	
G									X		X
H											X
I										X	
J									X	X	
K											X
L									X	X	
M	X	X	X	X	X	X					X
N				X	X	X					X
O											X
P								X			
Q						X		X			X
R						X					X
S									X	X	X
T							X			X	
U											X
V								X			X
W	X	X	X	X	X	X					X
X											X
Y								X			
Z									X	X	X

2.6.1 - Metodologia de reconhecimento para a classe 1

Foram previstas nesta classe as letras "M" e "W" por possuírem 2 ou 3 transições (retas) na linha de corte 1 (V1) e 4 transições na linha de corte 2 (V2). Isso ocorre para a letra "M" quando há uma "junção" da primeira reta com a segunda ou da terceira reta com a quarta ou em ambas, com relação a linha de corte 1, devido a espessura ou rotação dos caracteres. No caso da letra "W", ocorre uma "junção" da segunda reta com a terceira, ou quando as duas retas internas não atingem a região superior (30% da altura). Essas condições podem ser visualizadas na figura 5.8.

Figura 5.8 - Visualização dos caracteres pertencentes à classe 1, que apresentam 2 ou 3 transições na linha de corte 1 (V1) e 4 transições na linha de corte 2 (V2).

Como é difícil estimar, no caso da letra "M", em quais retas ocorreu uma fusão, a distinção é feita na região mais inferior do caracter, através de uma medida de inclinação da segunda reta (crescente ou decrescente). Essa inclinação é determinada através de uma medida de distância entre a primeira e segunda reta, em linhas de varredura horizontal em 70% e 80% da altura do caracter, conforme explica a figura 5.9.

Figura 5.9 - Teste realizado para distinguir entre as letras "M" e "W" pertencentes à classe 1 em função da inclinação.

Se a distância $d1$ entre as duas primeiras retas na varredura em 70% for menor que a distância $d2$ entre as duas primeiras retas, tomada na varredura em 80%, pode-se afirmar que a reta é decrescente e portanto reconhece-se o caracter como letra "M", caso contrário a reta é crescente e determina-se a letra "W".

O diagrama em árvore do reconhecimento para a classe 1 pode ser visto na figura 5.10.

Figura 5.10 - Diagrama em árvore do reconhecimento para a classe 1.

2.6.2 - Metodologia de reconhecimento para a classe 2

Foram previstas nesta classe as letras "M" e "W" por apresentarem quatro transições na linha de corte 1 e 2 ou 3 transições na linha de varredura 2. Esta situação ocorre para a letra "W" quando há uma fusão da primeira com a segunda reta ou da terceira com a quarta reta ou em ambas, com relação a linha de corte 2, e para a letra "M" quando ocorre uma fusão da segunda com a terceira reta ou quando as retas internas não atingem a região inferior (80% da altura), conforme pode ser visto na figura 5.11.

Como é difícil prever qual das situações citadas acima realmente ocorre, a distinção é feita na parte superior do caracter, determinando-se a inclinação da segunda reta em 30% e 40% da altura, de maneira análoga à descrita na figura 5.9.

Figura 5.11 - Visualização dos caracteres pertencentes à classe 2, que apresentam 4 transições na linha de corte 1 (V1) e 2 ou 3 transições na linha de corte 2 (V2).

O diagrama em árvore para o reconhecimento da classe 2 é mostrado na figura 5.12.

Figura 5.12 - Diagrama em árvore do reconhecimento para classe 2.

2.6.3 - Metodologia de reconhecimento para a classe 3

Foram previstas nesta classe as letras "M" e "W" por apresentarem quatro transições na linha de corte 1 (V1) e quatro transições na linha de corte 2 (V2). A discriminação entre as letras foi determinada através da medida da inclinação tomada nas linhas de corte 1 e 2, em função da distância entre elas, de maneira análoga à mostrada na figura 5.9. Na figura 5.13 pode ser vista a representação em árvore do reconhecimento para a classe 3.

Figura 5.13 - Diagrama em árvore do reconhecimento para classe 3.

2.6.4 - Metodologia de reconhecimento para a classe 4

Pertencem a esta classe as letras "E", "M", "N" e "W" por apresentarem três transições na linha de varredura 1 e três transições na linha de varredura 2.

Isto ocorre para a letra "E" devido à presença de serifa em seus traços horizontais.

Para a letra "M" esta situação acontece quando ocorre simultaneamente a "junção" da primeira reta com a segunda

na linha de corte 1 e a "junção" da segunda reta com a terceira na linha de corte 2 ou simultaneamente a "junção" da terceira reta com a quarta na linha de corte 1 e a "junção" da segunda com a terceira reta na linha de corte 2.

No caso da letra "W" isto acontece quando ocorre simultaneamente uma fusão da segunda reta com a terceira, na linha de corte 1 e uma fusão da terceira reta e a quarta com relação à linha de corte 2 ou quando ocorre simultaneamente a fusão da segunda com a terceira reta na linha de corte 1 e a fusão da primeira reta com a segunda, na linha de corte 2. Estes comportamentos podem ser visualizados na figura 5.14.

Figura 5.14 - Comportamento (3/3) das letras "E", "M", "N" e "W" pertencentes à classe 4, com relação às linhas de corte V1 e V2.

A distinção é feita inicialmente separando-se a letra "E" das outras, através de uma varredura vertical (VV1) a 50% da largura, conforme ilustra a figura 5.15. Se o número de transições for igual a 3 (três retas), determina-se a letra "E", caso contrário, realiza-se uma nova seqüência de testes que distingue entre as letras "M", "N" e "W".

Figura 5.15 - Distinção realizada para separar a letra "E" de "M", "N" e "W", em função do número de transições determinado na varredura vertical (VV1) realizada a 50% da largura.

Para distinguir "N" das letras "M" e "W" é feita uma varredura horizontal a 50% da altura. Se o número de transições for igual a 3 determina-se a letra "N", caso contrário, distingue-se entre as letras "W" e "M". Como não se sabe qual a situação que realmente ocorre para tais letras, a distinção é feita determinando-se a inclinação da segunda reta em uma região mais central, através de uma

medida de distância entre as transições tomadas em 50% e 60% da altura, de maneira análoga à determinada na figura 5.9.

A figura 5.16 mostra o diagrama em árvore da classificação para a classe 4.

Figura 5.16 - Diagrama em árvore do reconhecimento para a classe 4.

2.6.5 - Metodologia de reconhecimento para a classe 5

Pertencem a esta classe as letras "E", "F", "M", "N" e "W" por possuírem três transições na linha de corte 1 e duas transições na linha de corte 2, conforme pode ser visto na figura 5.17. Isto ocorre para as letras "E" e "F" devido à presença de serifas.

Para a letra "M" esta situação ocorre devido à fusão da primeira reta com a segunda ou da terceira reta com a quarta com relação a linha de corte 1 e para a letra "W" quando ocorre uma fusão entre a segunda e terceira reta na

linha de corte 1 e simultaneamente uma fusão da primeira reta com a segunda e da terceira reta com a quarta com relação à linha de corte 2. Já para a letra "N" isso ocorre quando há uma fusão da segunda com a terceira reta com relação à linha de corte 2.

Figura 5.17 - Comportamento (3/2) das letras "E", "F", "M", "N" e "W" pertencentes à classe 5, com relação às linhas de corte V1 e V2.

A distinção é feita inicialmente contando-se o número de transições determinadas na varredura vertical realizada a 50% da largura, conforme figura 5.15. Se o número de transições for igual a 3, determina-se a letra "E", se for igual a 2 determina-se a letra "F" e se for igual a 1 distingue-se entre as letras "M", "N" e "W".

A distinção entre as letras "M", "N" e "W" é feita contando-se o número de retas na varredura horizontal a 50% da altura. Se o número de transições for igual a 3, distingue-se entre as letras "W" e "N" através da inclinação da segunda reta tomada a 30% e 50% da altura. Se o número de transições for igual a 4 distingue-se entre as letras "W" e "M" através de uma medida da inclinação tomada a 40% e 50% da altura, de maneira análoga à mostrada na figura 5.9.

O diagrama em árvore mostrando a hierarquia do reconhecimento para a classe 5 pode ser visto na figura 5.18.

Figura 5.18 - Diagrama em árvore do reconhecimento para a classe 5.

2.6.6 - Metodologia de reconhecimento para a classe 6

Pertencem a esta classe as letras "Q", "R", "M", "N" e "W" por apresentarem duas transições na linha de

varredura 1 e três transições na linha de varredura 2. Essa situação acontece para a letra "M" quando ocorre uma fusão da primeira com a segunda reta e da terceira reta com a quarta com relação à linha de corte 1, e da segunda reta com a terceira com relação à linha de corte 2. Para a letra "W" isto ocorre quando acontece uma fusão da primeira reta com a segunda ou da terceira reta com a quarta com relação à linha de corte 2 e quando suas retas internas não atingem a região superior, em 30% da altura. Já para a letra "N" isto acontece quando ocorre uma fusão da primeira com a segunda reta, conforme pode ser visto na figura 5.19.

Figura 5.19 - Comportamento (2/3) das letras "R", "Q", "M", "N" e "W" pertencentes à classe 6, com relação ao número de transições determinado nas linhas de corte 1 e 2.

A distinção é feita inicialmente através do cálculo do número de transições determinadas em uma varredura vertical (VV1) realizada a 50% da largura. Se o número de transições for maior ou igual a 2, distingue-se entre as letras "R" e "Q" através da distância entre as duas primeiras transições da varredura vertical, conforme mostrado na figura 5.20. Caso esta distância seja maior que 60% da altura, reconhece-se a letra "Q", caso contrário determina-se a letra "R".

Figura 5.20 - Distinção entre as letras "Q" e "R", em função da distância entre as duas primeiras transições, determinadas na varredura vertical (VV1) a 50% da largura.

Se o número de retas encontradas na varredura vertical for igual a 1, distingue-se entre as letras "M", "N" e "W"

através do cálculo do número de transições determinadas na varredura horizontal a 50% da altura. Se esse número for igual a 3, distingue-se entre "M" e "N", calculando-se a inclinação da segunda reta da letra "N" ou da terceira reta da letra "M", determinada através das varreduras horizontais a 50% e 80% da altura, conforme mostra a figura 5.21.

Figura 5.21 - Teste realizado para distinguir entre as letras "M" e "N" pertencentes à classe 6 em função da inclinação de suas retas.

Se o número de retas encontradas na varredura horizontal for igual a 4, distingue-se entre "W" e "M" em função da medida de inclinação tomada a 50% e 60% da altura, de maneira análoga.

O diagrama em árvore do reconhecimento para a classe 6 pode ser visto na figura 5.22.

Figura 5.22 - Diagrama em árvore do reconhecimento para a classe 6.

2.6.7 - Metodologia de reconhecimento para a classe 7

Pertencem a esta classe as letras "E", "F" e "T" por possuírem três transições na linha de corte 1 e uma transição na linha de corte 2, conforme mostra a figura 5.23.

Figura 5.23 - Comportamento (3/1) das letras "E", "F" e "T" pertencentes à classe 7 em relação às linhas de corte 1 e 2.

A distinção é feita calculando-se o número de retas que interceptam a varredura vertical (VV1) feita a 50% da largura do caracter. Se o número for igual a 1, reconhece-se a letra "T". Se o número for igual a 2, distingue-se entre "F" e tipos de "T" que possuam serifa longa horizontal na extremidade inferior através de uma medida de distância entre as retas horizontais interceptadas, conforme mostra a figura 5.24.

Figura 5.24 - Testes realizados para distinguir entre as letras "F" e "T" e as letras "E" e "F", pertencentes à classe 7, quando as mesmas possuírem respectivamente 2 e 3 transições na linha de varredura determinada a 50% da largura

Se o número de transições for igual a 3 distingue-se entre "E" e tipos de "F" que possuem serifas horizontais na extremidade inferior do caracter. Para discriminar essas duas letras é realizada uma varredura vertical a 90% da largura, se o número de transições encontrado for igual a 3 determina-se a letra "E", caso contrário determina-se a letra "F", conforme mostrado na figura 5.24.

O diagrama em árvore da classificação pode ser visto na figura 5.25.

Figura 5.25 - Diagrama em árvore da hierarquia de reconhecimento determinada para a classe 7.

2.6.8 - Metodologia de reconhecimento para a classe 8

Pertencem a essa classe as letras "A", "C", "E", "F", "P", "Q", "S", "V", "Y", e "Z", por possuírem duas transições na linha de corte 1 e uma transição na linha de corte 2, conforme ilustra a figura 5.26.

Figura 5.26 - Comportamento (2/1) das letras "A", "C", "E", "F", "P", "Q", "V" e "Y", pertencentes à classe 8, com relação às linhas de corte V1 e V2.

A distinção é feita inicialmente através de uma varredura vertical (VV1) a 50% da largura, agrupando-se os caracteres em três sub-classes, dependendo se o número de transições determinado for igual a 1, 2 ou 3 respectivamente, conforme mostrado na figura 5.27.

Na sub-classe 1, distingue-se entre "Y" e "V" através de uma medida da inclinação tomada através da distância entre as primeiras transições, determinadas nas varreduras horizontais V3 e V4, a 50% e 90% da altura respectivamente. Se a reta inferior for vertical, determina-se a letra "Y", se for inclinada determina-se a letra "V". O teste realizado pode ser visto na figura 5.28.

Figura 5.27 - Determinação das sub-classes da classe 8, em função do número de transições encontradas na varredura vertical (VV1) a 50% da largura do caracter.

Figura 5.28 - Teste realizado para discriminar entre as letras "V" e "Y" pertencentes à classe 8, na sub-classe 1.

Na sub-classe 2, distingue-se entre as letras "A", "C", "F", "P", "Q" e "Y", em função da medida de distância entre as duas transições positivas determinadas na varredura vertical (VV1), conforme figura 5.28.

Figura 5.29 - Teste realizado para separar o conjunto de letras pertencentes à classe 8, na sub-classe 2, em função da distância entre as transições determinadas na varredura vertical VV1.

Se a distância estiver entre 60% e 80% da altura do caracter, reconhece-se a letra "A".

Para distâncias iguais ou maiores a 80% da altura, separa-se a letra "Q" de "C" e "Y" se o número de transições determinado na varredura horizontal V3 a 50% da altura for igual a 2, conforme exemplifica a figura 5.30.

Distingue-se entre "C" e "Y", calculando-se o número de transições na varredura vertical (VV2), realizada a 90%

da largura. Se for igual a 2 determina-se "C", caso contrário reconhece-se a letra "Y".

Figura 5.30 - Distinção de letra "Q" de "C" e "Y" em função do número de retas interceptadas na varredura horizontal a 50% da altura.

Se a distância medida entre as transições verticais for menor que 60% da altura discrimina-se "P" de "F" em função do número de transições determinado na varredura horizontal em 40% da altura ser igual a 2.

Na sub-classe 3 distingue-se "Q" das letras "E", "F" e "P" se o número de transições determinado na varredura horizontal em 70% da altura for maior ou igual a 2.

Para separar a letra "E" de "F" e "P", realiza-se uma varredura vertical (VV2) a 90% da largura. Se o número encontrado for igual a 3, determina-se "E".

Nessa classe é difícil distinguir entre "F" e "P" tomando-se como referência a região superior como sendo

aberta ou fechada, pois a presença de serifa na letra "F" pode mascarar essa característica. Para resolver essa ambigüidade, estima-se, em função de observação nas fontes utilizadas no treinamento, que o segundo traço horizontal da letra "F" é menor que o primeiro, portanto a discriminação entre os dois caracteres é determinada calculando-se o número de transições na varredura vertical (VV3) realizada em 70% da largura. Se o número for igual a 2, determina-se "P", caso contrário determina-se "F", conforme ilustra a figura 5.31.

Figura 5.31 - Distinção entre as letras "F" e "P" pertencentes à classe 8, na sub-classe 3.

O diagrama em árvore do reconhecimento para a classe 8 pode ser visualizado na figura 5.32.

Figura 5.32 - Diagrama em árvore da hierarquia de reconhecimento determinada para a classe 8.

2.6.9 - Metodologia de reconhecimento para a classe 9

Pertencem a esta classe as letras "A", "C", "E", "F", "G", "J", "L", "S" e "Z", quando possuírem uma transição na linha de corte 1 e duas transições na linha de corte 2, conforme mostrado na figura 5.33.

A distinção é feita inicialmente através de uma varredura vertical (VV1) em 50% da largura, agrupando-se os caracteres em três sub-classes, dependendo se o número de transições determinado for igual a 1, 2 ou 3, respectivamente, conforme exemplifica a figura 5.34.

Figura 5.33 - Comportamento ($1/2$) das letras "A", "C", "E", "F", "G", "J", "L", "S" e "Z", pertencentes à classe 9, com relação às linhas de corte V1 e V2.

Figura 5.34 - Formação das sub-classes, determinadas na classe 9 em função do número de transições determinado na varredura vertical (VV1) em 50% da largura do caracter.

Na sub-classe 1, a distinção entre "J" e "L" é feita através de uma medida de distância entre a primeira transição negativa determinada na linha de corte 1 e a segunda transição negativa determinada na linha de corte 2, conforme pode ser visto na figura 5.35. Se a distância for menor que 20% da largura, determina-se a letra "J", caso contrário determina-se a letra "L".

Figura 5.35 - Teste realizado para distinguir entre as letras "J" e "L" pertencentes à classe 9, na sub-classe 1, em relação às linhas de corte 1 e 2.

Na sub-classe 2 distingue-se "J", "C", "L" e "G" da letra "A", em função da distância entre as transições na varredura vertical VV1 em 50% da largura ser superior a 80% da altura, conforme figura 5.36.

Figura 5.36 - Teste realizado para distinguir a letra "A" das letras "J", "C", "L" e "G", pertencentes à classe 9, na sub-classe 2, em função da distância entre as transições determinadas na varredura vertical (VV1) a 50% da largura.

Para distinguir entre "J", "C", "L" e "G" calcula-se o número de transições determinado na varredura vertical (VV2) em 80% da largura, conforme pode ser visualizado na figura 5.37. Se o número de transições for igual a 1, distingue-se entre "J" e "L", em um procedimento análogo ao demonstrado na figura 5.35. Se o número for igual a 3, determina-se "G". Se for igual a 2 discrimina-se "C" de "G" se o número de transições determinado na varredura horizontal em 60% da altura for igual a 1.

Figura 5.37 - Teste realizado para discriminar as letras "L", "J", "C" e "G", pertencentes à classe 9, sub-classe 2, em função do número de transições determinados na varredura vertical VV2 em 80% da largura do caracter.

Na sub-classe 3 discrimina-se "Z" de "E", "G" e "S" através da distância entre a transição da linha de corte 1 e a segunda transição na linha de corte 2 ser menor que 50% da largura, conforme mostra a figura 5.38.

Para distinguir entre "E", "G" e "S" calcula-se o número de transições na varredura vertical VV2, a 30% da largura. Se o número de transições for igual a 2 discrimina-se a letra "G", das letras "E" e "S". Nessa

classe, é difícil separar a letra "E" de "S" devido à presença de serifas. A distinção neste caso é determinada para a letra "E" por uma característica estética, em função do número de transições detectadas na varredura horizontal a 60% da altura ser igual a 2.

Figura 5.38 - Testes realizados para separar a letra "Z" das letras "E", "G" e "S", pertencentes à classe 9, na sub-classe 3, através de uma medida de distância entre a ordenada da transição determinada na linha de corte 1 e a ordenada da segunda transição determinada na linha de corte 2.

O diagrama em árvore da classificação realizada para a classe 9 é mostrado na figura 5.39.

Figura 5.39 - Diagrama em árvore da hierarquia de reconhecimento realizada para classe 9.

2.6.10 - Metodologia de reconhecimento para a classe 10

Pertencem a esta classe as letras "A", "C", "E", "F", "I", "J", "L", "S", "T" e "Z" por apresentarem uma transição na linha de varredura 1 e uma transição na linha de varredura 2, conforme figura 5.40.

A distinção é feita inicialmente através de uma varredura vertical (VV1) a 50% da largura, agrupando-se os caracteres em três sub-classes, dependendo se o número de transições detectadas for igual a 1, 2 ou 3. As sub-classes determinadas são exemplificadas na figura 5.41.

Figura 5.40 - Comportamento (1/1) das letras "A", "C", "E", "F", "I", "J", "L", "S", "T" e "Z", pertencentes à classe 10, com relação às linhas de corte V1 e V2.

Figura 5.41 - Formação das sub-classes, determinadas na classe 9, em função do número de transições determinado na varredura vertical (VV1), em 50% da largura do caracter.

Na sub-classe 1 a distinção é feita entre as letras "I", "J", "L" e "T"; calculando-se o número de transições encontrado na varredura vertical (VV2) a 30% da largura, pôde-se agrupar as letras em sub-classe 1.1 e sub-classe 1.2, mostradas na figura 5.42, de acordo com o número de transições detectado ser 1 ou 2, respectivamente:

Figura 5.42 - Determinação das sub-classes 1.1 e 1.2 relativas às letras "J", "I", "L" e "T", pertencentes à classe 10, em função do número de transições determinado na varredura vertical VV2, a 30% de largura do caractere.

- subclasse 1.1: quando o número de transições em VV2 for igual a 1, separa-se "T" de "J", "I" e "L", em função da distância entre a transição positiva determinada em VV2 e a ordenada de topo Y_T , conforme mostra a figura 5.43.

Figura 5.43 - Teste realizado para separar a letra "T" das letras "J", "I" e "L", pertencentes à sub-classe 1.1 da classe 10, em função da distância entre a transição positiva determinada na varredura vertical VV2 e a ordenada de topo Y_T

Para discriminar "J" de "I" e "L", calcula-se o número de transições que ocorrem na varredura horizontal V3 a 90% da altura. Se o número for igual a 2, determina-se "J", caso contrário distingue-se entre "I" e "L" em função da distância entre a transição negativa e a ordenada de topo Y_T , conforme mostra a figura 5.44.

Figura 5.44 - Teste realizado para separar a letra "J" das "L" e "I" pertencentes à sub-classe 1.1 da classe 10, em função do número de transições detectadas na varredura horizontal a 90% da altura. A distinção entre "L" e "I" é feita em função da distância entre a ordenada de topo e a ordenada da transição negativa.

- sub-classe 1.2: quando o número de transições em VV2 for igual a 2, discrimina-se "T" de "J" e "I", em função do número de transições na varredura horizontal V3 realizada em 20% da altura ser igual a 3, prevendo-se nesse caso a presença de serifas. Para separar a letra "I" de "J", realiza-se uma varredura vertical VV3 em 90% da altura e se o número de transições determinado for 2 reconhece-se "I", caso contrário, reconhece-se a letra "J". A seqüência de testes pode ser visualizada na figura 5.45.

Figura 5.45 - Teste realizado para separar a letra "T" das letras "I" e "J", pertencentes à sub-classe 1.2 da classe 10, em função do número de transições detectado na varredura a 20% da altura. A distinção entre as letras "I" e "J" é determinada pelo número de transições detectadas na varredura vertical (VV3) a 90% da largura.

Na sub-classe 2 discrimina-se "C" de "A" e "F" em função da distância entre as transições determinadas na varredura vertical VV1, conforme mostra a figura 5.46.

Para distinguir "A" de "F", calcula-se o número de transições obtidas pela varredura horizontal realizada em 90% da altura. Se esse número for igual a 1, determina-se "F", caso contrário a letra é "A".

Figura 5.46 - Distinção entre a letra "C" e as letras "A" e "F" pertencentes a sub-classe 2, da classe 10, em função do número de transições detectadas em VV1. A separação entre as letras "A" e "F" é obtida através do número de transições determinadas na varredura V3, a 90% da altura.

Na sub-classe 3 a distinção é feita através de uma medida de distância entre a transição determinada pela linha de corte 1 no ponto x_1 e a da linha de corte 2 no ponto x_2 . Se essa distância for menor que 20% da largura, isto é, se a reta for vertical ou estiver pouco inclinada, agrupa-se as letras "E" e "F", caso contrário distingue-se entre as letras podem ser "S" ou "Z", conforme mostrado na figura 5.47.

Se a abscissa x_1 for menor que x_2 , determina-se a

letra "S", caso contrário, a letra é "Z".

Se o número de transições determinadas na varredura vertical tomada a 70% da largura for igual a 3, determina-se a letra "E", e se for igual a 2, a letra é "F".

Figura 5.47 - Seqüência de testes realizada para discriminar as letras "E", "F", "S" e "Z", pertencentes à sub-classe 3 da classe 10, em função das ordenadas determinadas nas linhas de corte 1 e 2.

O diagrama em árvore da hierarquia do reconhecimento determinado para a classe 10 é mostrado na figura 5.48.

Figura 5.48 - Diagrama em árvore da hierarquia de reconhecimento determinada para a classe 10.

2.6.11 - Metodologia de reconhecimento para a classe 11

Pertencem a esta classe as letras "A", "B", "C", "D", "E", "G", "H", "K", "M", "N", "O", "Q", "R", "S", "U", "V", "W", "X" e "Z" por apresentarem duas transições na linha de corte 1 e duas transições na linha de corte 2. Exemplos de letras que possuem esse comportamento podem ser vistos na figura 5.49.

Figura 5.49 - Comportamento (2/2) das letras "A", "B", "C", "D", "E", "G", "H", "K", "M", "N", "O", "Q", "R", "S", "U", "V", "W", "X" e "Z", pertencentes à classe 11, com relação às linhas de corte V1 e V2.

A distinção é feita inicialmente através de uma varredura vertical a 50% da largura, agrupando-se os caracteres em três sub-classes, dependendo se o número de transições for igual a 1, 2 ou 3 respectivamente.

Na sub-classe 1, exemplificada na figura 5.50, a distinção entre as letras "H", "K", "U", "V", "N", "M" e "W" é feita através de uma varredura vertical em 50% da largura do caracter, determinando-se assim outras sub-classes, em função do número de transições ser igual a 1, 2, 3 e 4 respectivamente.

Figura 5.50 - Formação da sub-classe 1, determinada na classe 9, em função do número de transições detectado na varredura vertical (VV1) em 50% da largura do caracter.

Na sub-classe 1.1 distingue-se "X" de "H" e "K" percebendo-se a inclinação de suas retas mais à esquerda, através de uma medida de distância entre a abcissa da primeira transição detectada na linha de varredura horizontal a 50% da altura e a da linha de corte 2, conforme pode ser visto na figura 5.51. A distinção entre a letra "H" e "K" é feita de maneira análoga, percebendo-se entretanto a inclinação de suas retas mais à direita.

Figura 5.51 - Distinção entre as letra "H", "K", e "X", pertencentes à sub-classe 1.1 da classe 10, em função das inclinações de suas retas.

Na sub-classe 1.2 distingue-se entre as letras "U", "V", "H", e "K" em função da distância entre as transições determinadas na varredura vertical:

- Se a distância for menor que 50% da altura, reconhece-se a letra "M" se o número de transições na varredura horizontal a 40% da altura for maior ou igual a 3; caso contrário separa-se "H" de "K" em função da inclinação de suas retas à direita.
- Se a distância estiver entre 50% e 80% da altura, distingue-se entre "H" e "K" de acordo com a medida da inclinação de sua reta à direita.
- Para distâncias maiores que 80% da altura discrimina-se entre "U" e "V" através da medida da inclinação de suas retas, tomadas a 20% e 80% da altura.

A sequência de testes realizada para a sub-classe 1.2

é mostrada na figura 5.52.

Na sub-classe 1.3 determina-se a letra "N" (número de transições igual a 3)

Na sub-classe 1.4 distingue-se entre as letras "M" e "W" em função da inclinação da segunda reta, tomada em 40% e 50% da altura, de maneira análoga à realizada na figura 5.9.

Figura 5.52 - Distinção entre as letras "H", "K", "U" e "V", pertencentes à sub-classe 1.3 da classe 11, em função da inclinação de suas retas.

Na sub-classe 2 a separação entre as letras "C", "D", "G", "O", e "Q" de "R" e "X" é feita através da medida da distância entre as transições detectadas na varredura vertical (VV1), conforme mostra a figura 5.54, agrupando-as em duas sub-classes.

Figura 5.53 - Separação das letras "C", "D", "G", "O", e "Q" de "R" e "X", em função da distância entre as transições determinadas em VV1.

Na sub-classe 2.1, se o número de transições na varredura vertical VV2 a 65% da largura do carácter for igual a 3, determina-se a letra "G", caso contrário distingue-se "C" de "D", "O" e "Q" por possuir uma transição na varredura horizontal V3 tomada a 50% da altura. A separação da letra "D" de "O" e "Q" é feita através de sua reta vertical à direita, enquanto que a discriminação entre "Q" e "O" é feita percebendo-se sua "cauda" através da distância entre a transição na varredura vertical (VV3) a 95% da largura e a ordenada de base YB. Os procedimentos utilizados são mostrados na figura 5.54.

Figura 5.54 - Testes realizados para distinguir entre as letras "C", "D", "G", "O", e "Q" pertencentes à sub-classe 2.1 da classe 11.

Na sub-classe 2.2 separa-se "A" e "X" de "R" e "K" em função da inclinação de suas retas à esquerda. A distinção entre "R" e "K" é feita percebendo-se a sua abertura superior, através de uma medida de distância entre a primeira transição negativa e a ordenada de topo (YT). Distingue-se a letra "X" de "A" em função de sua reta superior à esquerda possuir inclinação decrescente. A seqüência dos procedimentos utilizados pode ser vista na figura 5.55.

Figura 5.55 - Sequência de testes utilizada na discriminação das letras "A", "X", "R" e "K", pertencentes à sub-classe 2.2 da classe 11.

Na sub-classe 3 separa-se as letras "E", "Q" e "B", de "G", "S" e "Z" por possuírem 2 transições na varredura horizontal V4 tomada a 40% da altura, conforme figura 5.56. Como nessa classe é difícil distinguir a letra "E" de "B" e "Q" em função de suas aberturas à direita devido a presença de serifas, essa distinção é feita em função de uma característica estética da fonte utilizada no treinamento, através de uma medida do comprimento do traço médio em relação à largura do caracter.

Figura 5.56 - Separação entre as letras "B", "E" e "Q" das letras "G", "S" e "Z", pertencentes à sub-classe 3, da classe 11, em função do número de transições detectadas na varredura horizontal a 40% da altura do caracter.

Para distinguir entre a letra "B" e "Q" calcula-se a distância da segunda transição em relação à metade da altura. A separação da letra "G" de "S" e "Z" é feita pelo fato da mesma apresentar uma pequena variação em módulo (com relação à largura do caracter) entre as abcissas das transições negativas x_1 e x_2 tomadas a 40% e 60% da altura respectivamente. Se x_1 for menor que x_2 reconhece-se a letra "S", caso contrário determina-se "Z".

Os procedimentos realizados para a sub-classe 3 são mostrados na figura 5.57.

Figura 5.57 - Seqüência de testes realizada para discriminar as letras "E", "Q", "B",

Na figura 5.58 pode ser visto o diagrama em árvore da hierarquia de reconhecimento para a classe 11.

Figura 5.58 - Diagrama em árvore da hierarquia de reconhecimento determinada para a classe 11.

CAPÍTULO 6

RESULTADOS E CONCLUSÕES

1 - AVALIAÇÃO DO SISTEMA

A fim de analisar o desempenho do sistema quantitativamente, decidiu-se testá-lo usando imagens constituídas de caracteres impressos no tamanho 100 pontos, provenientes das fontes *CG Times*, *Courier*, *Helvética*, *ITC Avant Gard Gothic Book*, *ITC Bookman Demi*, *New Century School Book*, *Palatino* e *Times Roman*. As imagens foram digitalizadas por um *scanner* de baixa resolução (100 dpi), com os ruídos inerentes ao processo. Com o objetivo de avaliar a independência do algoritmo desenvolvido quanto à variabilidade da escala, criou-se conjuntos de testes constituídos de caracteres reduzidos a um fator de 0.75, 0.5 e 0.25 em escala, para as fontes citadas acima. A performance do sistema foi avaliada individualmente para cada fonte e globalmente, considerando o conjunto de caracteres provenientes das nove fontes observadas no treinamento. Procurou-se estabelecer também o tamanho crítico e a taxa de erros apresentada para cada caracter isoladamente.

A taxa de confiabilidade variou entre 92,31% e 100%. As fontes mais críticas foram a *Palatino* que apresentou 1

erro nos tamanhos 100, 75 e 25 pontos para o mesmo caracter ("J"), a *Times Roman* que apresentou três erros distintos ("U" para o tamanho 50 e "J" e "S" no tamanho 25) e a fonte *Courier* que apresentou dois erros no tamanho 25 ("Q" e "U"). As fontes *Helvética* e *ITC Bookman Demi* apresentaram 1 erro no tamanho 25 ("Q") e a *New Century school Book* 1 erro no tamanho 100 ("D"). As demais fontes analisadas apresentaram uma confiabilidade de 100%.

Os resultados das experiências estão resumidos na tabela 6.1.

Considerando o conjunto de 936 caracteres utilizados como amostras, provenientes das 9 fontes em quatro tamanhos, a taxa de confiabilidade do sistema foi 98,83% (11 caracteres errados).

Analisando-se os caracteres isoladamente nos quatro tamanhos (36 amostras) a taxa de erro individual foi:

- 11,11% para a letra "J" (4 erros)
- 8,33% para a letra "Q" (3 erros)
- 5,55% para a letra "U" (2 erros)
- 2,78% para a letra "D" (1 erro)
- 2,78% para a letra "S" (1 erro)

As demais letras não apresentaram erros, o que significa que foram utilizadas boas descrições para representá-las. O baixo índice de acerto determinado para a letra "J" e "Q", quando comparado às outras letras já era esperado, por possuírem atributos comuns aos caracteres "I" e "O" respectivamente.

TABELA 6.1 - Taxas de reconhecimento para as fontes observadas no treinamento

FONTES	TAMANHO 100	TAMANHO 75	TAMANHO 50	TAMANHO 25
<i>UNIVERS</i>	100%	100%	100%	100%
<i>COURIER</i>	100%	100%	100%	92,31%
<i>HELVETICA</i>	100%	100%	100%	96,15%
<i>ITC AVANT GARD GOTHIC BOOK</i>	100%	100%	100%	100%
<i>ITC BOOKMAN DEMI</i>	100%	100%	100%	96,15%
<i>NEW CENTURY SCHOOL BOOK</i>	96,15%	100%	100%	100%
<i>PALATINO</i>	96,15%	96,15%	100%	96,15%
<i>TIMES ROMAN</i>	100%	100%	96,15%	92,31%
<i>CG TIMES</i>	100%	100%	100%	100%

O tamanho crítico determinado foi 25 pontos que apresentou 7 erros em 234 amostras (2,99%). Os tamanhos 50 e 75 pontos apresentaram apenas um erro (0,427%), enquanto que o tamanho 100 apresentou 2 erros (0.855%).

1.1 - Avaliação através de outras fontes

Para testar a eficiência e a confiabilidade do algoritmo, resolveu-se testa-lo para outras fontes não observadas na etapa de treinamento, no tamanho 50 pontos.

As fontes analisadas foram as seguintes: *Modern*, *Arial*, *Arial Rounded*, *Bookman Old Style*, *Britannic Bold*, *Century Gothic*, *Footlight Mt Light*, *Impact* e *MS Line Draw* e os resultados obtidos são mostrados na tabela 6.2.

TABELA 6.2 - Taxa de reconhecimento determinadas para fontes não previstas no treinamento.

FONTES	TAXA DE RECONHECIMENTO
ARIAL	100%
ARIAL ROUNDED	100%
BOOKMAN OLD STYLE	86,44%
BRITANNIC BOLD	100%
CENTURY GOTHIC	96,15%
MODERN	57%
IMPACT	61,54%
FOOTLIGHT MT LIGHT	80.77%
MS LINE DRAW	100%

Apesar das fontes testadas apresentarem estilizações diferentes, a taxa de reconhecimento determinada foi razoável para a maioria delas. O pior índice de acerto foi determinado para a fonte *Modern* por ser uma fonte muito

fina e o processo de digitalização da imagem, com resolução de 100 dpi, gerar muitas descontinuidades. Por outro lado, a fonte *Impact* também apresentou um baixo índice de reconhecimento devido ao fato de possuir traços muito grossos, não adequados aos parâmetros do sistema.

Analisando-se a confiabilidade do sistemas globalmente para as fontes não-treinadas, constatou-se 30 erros em 234 amostras, o que representa uma taxa de reconhecimento de 87,18%.

Observando as letras individualmente pôde-se verificar que:

- "C", "K", "M", "O", "S" e "V" não apresentaram nenhum erro;
- "A", "D", "F", "I", "L", "P", "Q", "R", "U", "W", "X", "Y" e "Z" apresentaram 1 erro (11,11%);
- "B", "E", "H", "J", "N" e "T" apresentaram 2 erros (22,22%);
- "G" apresentou 3 erros (33,33%).

Considerando as 18 fontes envolvidas e os caracteres no tamanho 50, o índice de reconhecimento determinado foi 93,4%. Levando-se em consideração as 1170 amostras analisadas e os 41 erros obtidos, a taxa de confiabilidade total do sistema foi 96,5%.

1.2 - Avaliação em diferentes resoluções

A fim de estabelecer a largura e altura mínimas que permitem o reconhecimento, analisou-se também para a fonte *Univers* o reconhecimento para os tamanhos 15, 10 e 5 pontos, digitalizando as imagens com o *scanner* em diferentes resoluções (100, 200, 300 e 400 dpi). A tabela 6.3 sumariza os resultados.

Tabela 6.3 - Taxa de reconhecimento em função da resolução do *scanner*

<i>UNIVERS</i>	RESOLUÇÃO 100 DPI	RESOLUÇÃO 200 DPI	RESOLUÇÃO 300 DPI	RESOLUÇÃO 400 DPI
TAMANHO 5 pontos	11,54%	38,46%	69,23%	100,00%
TAMANHO 10 pontos	46,15%	84,62%	100,00%	100,00%
TAMANHO 15 pontos	73,10%	100,00%	100,00%	96,15%

Como era esperado, o índice de acertos cai drasticamente com a diminuição da resolução do *scanner*, principalmente para caracteres com tamanho pequenos. Assim, para reconhecimento de fontes constituídas de caracteres

muito pequeno, deve-se utilizar *scanners* com 400 dpi, embora isso resulte em maior custo computacional. Com 400 dpi, o único caracter reconhecido erroneamente foi o "Q".

1.3 - Avaliação com caracteres rotacionados

Uma outra etapa de testes foi realizada para a fonte *Univers*, a fim de se verificar o comportamento da taxa de reconhecimento em função da rotação. Para isso, rotacionou-se os caracteres de até 15°, nos sentidos horário e anti-horário. Os resultados obtidos são mostrados na tabela 6.4.

TABELA 6.4 - Taxa de reconhecimento obtida com caracteres rotacionados.

UNIVERS	TAMANHO 75
-15°	50,00%
-10°	73,10%
-5°	84,62%
0°	100,00%
+5°	84,60%
+10°	88,46%
+15°	69,23%

Determinou-se também a taxa de reconhecimento para a fonte Univers em itálico, e sua performance caiu de 100% para 69,23%, porque tais variações não foram previstas durante a etapa de treinamento.

1.4 - Avaliação sob-ruído

A fim de testar a rotina de eliminação de ruídos (`clean()`) foi gerada imagens ruidosas aleatoriamente e verificada a atuação do algoritmo. Para ruídos isolados e retas com espessura menor que três pixels, os mesmos foram eliminados pelo sistema, que determinou o reconhecimento correto do caracter.

Figura 6.1 - Exemplo de caracter com ruído, reconhecido pelo sistema.

2 - CONCLUSÕES E DISCUSSÃO PARA TRABALHOS FUTUROS

O objetivo de desenvolver um sistema para reconhecimento multi-fonte de caracteres proposto neste

trabalho foi alcançado, obtendo-se uma taxa de precisão de 98,83% para as 9 fontes observadas no treinamento e de 96,5% para 18 fontes testadas, em diversos tamanhos. A técnica de reconhecimento utilizada baseou-se na comparação de vetores de características, facilmente detectadas e expressas em termos de arcos, retas horizontais, verticais e inclinadas. Tais características são mais flexíveis e podem prever um maior número de variações para diferentes representações de uma mesma letra.

A taxa de erros (3,5%) aparentemente alta, deve-se principalmente ao grande número de variações de estilo das fontes tipográficas, entretanto mesmo os sistemas de alto desempenho não determinam 100% de índice de acertos para sistemas multi-fonte. Para algumas fontes, tais como: *Univers*, *ITC Avant Gard*, *CG Times*, *Arial*, *Arial Rounded*, e *Brittanic Bold*, entre outras, a taxa de reconhecimento alcançada foi 100%. Apenas para fontes mais estilizadas a taxa de reconhecimento foi baixa, o que pode ser resolvido realizando-se uma nova etapa de treinamento e incorporando suas características ao sistema.

O sistema pretende ainda ser aprimorado, a fim de determinar taxas de reconhecimento ainda mais altas e prever o reconhecimento para caracteres manuscritos e para novas fontes. Muitas das variabilidades em termos de espessura dos traços e presença de serifas poderiam ser evitadas através da inclusão de um algoritmo de afinamento dos caracteres na etapa de pré-processamento ou através da

utilização de redes neurais durante a etapa de treinamento.

O sistema pretende ainda estender o reconhecimento para leitura de documentos, incluindo uma etapa de segmentação de linhas e caracteres.

Apesar das muitas técnicas e idéias sofisticadas sugeridas pelos pesquisadores, muitos dos problemas referentes ao reconhecimento ainda não foram totalmente solucionados. Embora os sistemas comerciais apresentem relativamente baixo custo e alta taxa de reconhecimento, ainda é necessário a resolução de diversos problemas relacionados à variabilidade dos caracteres manuscritos e tipográficos, bem como o reconhecimento de caracteres rotacionados.

À medida que o campo de OCR avança, cada vez mais as pesquisas direcionam-se ao reconhecimento de caracteres com menos restrições, visando uma melhoria na relação custo/eficiência das técnicas de digitalização, pré-processamento, extração de características e métodos de classificação.

Através de pesquisas e experiências práticas anteriores, uma abordagem promissora é dividir o processo de reconhecimento em vários estágios:

- estágio de pré-reconhecimento, o qual utiliza características mais gerais, baseadas na distribuição de pontos, afim de se reduzir o número de escolhas possíveis.
- estágio intermediário de reconhecimento, o qual faz uso de características geométricas e topológicas para limitar

a identificação do carácter a poucas escolhas.

- estágio de discriminação, o qual focaliza a detecção de detalhes e discrimina entre grupos ou pares confusos.
- estágio final, o qual faz uso de informações de lingüística, análise de contexto ou estatística, levando a uma identificação de caracteres, palavras e até sentenças.

BIBLIOGRAFIA

AGUI, T.;NAGAHASHI,H. A description method of handprinted chinese characters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v.PAMI-1, n.1, January 1979.

AGUI, T.; NAKAJIMA, M.; KIM, T.K.; TAKAHASHI, E.T. A method of recognition and representation of Korean characters by tree grammars, *IEEE Transactions on Pattern analysis and machine intelligence*, v. Pami-1, n.3, July 1979.

AKIYAMA, T. ; HAGITA, N. Automated entry system for printed documents, *Pattern Recognition*, v.3, v.11, pp. 1141-1154, 1990.

ALI, F. ;PAVIDS, T. Syntatic Recognition of handwritten Numerals, *IEEE Transactions on Systems, Man, and Cybernetics*, v. SMC-7, n.7, pp. 537-541, July 1977.

ANSELL, M. Handwritting classification in forensic science, *Visible Language*, v.XIII, n.3, 1979.

APSEY, R.S., Human factors of constrained handprinted for OCR, *IEEE Transactions on Systems, Man, and Cybernetics*, V. SMC-8, N.4, pp. 292- 296, April 1978.

ASCHER, R.N.; KOPPELMAN, G.M; MILLER, M.J.; NAGY, G. ;
SHELTON, G.L. An interactive System for Reading Unformatted
Printed Text, *IEEE Transactions on Computers*, v. c-20,
n.12, pp. 1527-1542, December 1971.

BALESTRI, M. ; MASERA, L. A system for isolating characters
in cursive script, *CSELT Technical reports*, v.XXI, n.7,
pp. 599-600, December 1988.

BALM, G.J. An introduction to optical character
considerations, *Pattern Recognition*, v.2, pp. 151-166,
1970.

BERNHARDT, L. Three Classical character recognition
problems, Three new solutions, *Siemens Forsch*, v. 13, n.3,
pp. 114-117, 1984.

BEUN, M. - A flexible method for automatic reading of
handwritten numerals, *Phillips Technical Review*, v.33,
n.4, pp. 89-101, 1973.

BEUN, M. - A flexible method for automatic reading of
handwritten numerals, *Phillips Technical Review*, v.33,
n.4, pp. 130-137, 1973

BLESSER, B.; SHILLMAN, R.; KUKLINKI, T., Cox, C.; EDEN, M.;
VENTURA, J. A theoretical approach for character

recognition based on phenomenological attributes, *International Journal of Man-Machine Studies*, v.6, pp. 701-714, 1974.

BLESSER, B.A., KULKINSK, T.T. & SHILLMAN, R.J., Empirical tests for feature selection based on a psychological theory of character recognition, *Pattern Recognition*, v,8, pp.77-85, 1976.

BLISS, J.C. A relative high-resolution reading aid for the blind, *IEEE Transactions on Man, Machine Systems*, v.10, pp. 1-9, 1969.

BORNART, R. ; BRADY, J.M. Using knowledge in the computer interpretation of handwritten FORTRAN coding sheets, *International Journal of Man-Machine Studies*, v.8, pp.13-27, 1976.

BOUMA, H. Visual recognition of isolated lower-case letters, *Vision Reserch*, v.11, pp. 459-474, 1971.

BOWMAN, R.M., McVEY, E.S., A method for the optimal design of a class of pattern recognition systems, *Pattern Recognition*, V.2, pp. 187-197, 1970.

BOZINOV, R.M. ; SRIHARA, S.N. Off-line cursive script word recognition, *IEEE Transactions on pattern analysis and*

machine intelligence, v.11, n.1, pp. 68-83, January, 1989.

BRADSHAW, J.A. Letter recognition using a captive scan, *IEEE Transaction on Eletronic Computers*, v. EC-12, February, 1963.

BRADY, M.Toward a computacional Theory of early visual processing in reading, *Visible Language*, v.XV, n.2, pp. 183-214, 1981.

BROOKS, C.P. ; NEWELL, A.F., Computer transcription of handwritten shorthand as an aid for the deaf - a feasibility study, *Internation Journal of Man-Machine Studies*, v.23, n.1,pp.45-60, 1985.

BROWN, R.M.; FAY, T.H.; WALKER, C.L. Handprinted symbol recognition system, *Pattern Recognition*, v.21, n.2, pp. 91-118, 1988.

BROWN, M.K. ; GANAPATHY, S. Preprocessing Techniques for Cursive Script Word Recognition, *Pattern Recognition*, v.16, n.5, pp. 447-458, 1983.

CAI, Z. A handwritten numeral recognition system using a multi-microprocessor, *Pattern Recognition Letters*, v.12, pp. 503-509, August 1991.

CAMARÃO, P.C.B. *Glossário de Informática*, Livros Técnicos e Científicos Editora LTDA., 1989.

CARRIJO, G.A. Uso de Redes Artificiais de Neurônios Lineares para o treinamento e Reconhecimento de Caracteres com Rotação, *III Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens*, Gramado, pp. 101-110, 1990.

CASEY, R.G. ; JIH, C.R. A processor-based OCR system, *IBM Journal of Reserch and Development*, v.27, n.4, pp.386-399, July 1983.

CASEY, R.G. ; NAGY, G. An autonomous reading machine, *IEEE Transactions on Computers*, v. C-17, n.5, May 1968.

COX, C.H.; BLESSER, B.A. ; EDEN, M. Graphical context of printed characters, *Visible Language*, v. XII, n.4, pp. 428-447, 1978.

COX, C.H; COUEIGNOUX, P.; BLESSER, B. ; EDEN, M. Skeletons: a link between theoretical and physical letter descriptions, *Pattern Recognition*, v.15, n.1, pp. 11-22, 1982.

CUNHA, S.H.O; Reconhecimento de Algarismos, 9º CBA - UFES, Vitória, ES, pp. 401-407, 14-18 de setembro de 1992.

DAINESE, C.A. ; SAITO, J.H. Sistemas de reconhecimento de padrão de caracteres em formulários (RECOMPAC), *Jornada EPUSP IEEE em computação visual*, pp.341-345, 4-7 dezembro de 1990.

DAVIES, R.H. ; LYAL, J Recognition of handwritten characters - a review, *Image and Vision Computing*, pp.208-218, 1986.

DOOIJES, E.H, Analysis of handwriting movements, *Acta Psychologica*, v.54, pp. 99-114, 1983.

DUTTA, A.K. An experimental procedure for handwritten character recognition, *IEEE Transactions on Computers*, May 1974.

DVORAK, J.C.; ANIS, N. ; SEIBEL, W. *Dvorak PC: guia de conectividade*, Berkeley Brasil Editora, 1993.

EDEN, M. Handwritting and Pattern Recognition, *IRE Transactions on Information Theory*, v.8, February 1962.

ELLIMAN, D.G. ; LANCASTER, I.T. A review of segmentation and contextual analysis for text recognition, *Pattern Recognition*, v.23, n.3/4, pp. 337-346, 1990.

EVANGELIST, C.J., Some experiments in the evaluation of a character recognition scanner. *Pattern Recognition*, V.16, N.3, pp.273-287, 1983.

FREEDMAN, M.D. Optical Character Recognition, *IEEE Spectrum*, pp. 44-52, March 1974.

GENCHI, H.; MORI, K.; WATANABE, S.; KATSURAGI, S. Recognition of handwritten numerical characteres for automatic letter sorting, *Proceedings of the IEEE*, v.56, n.8, pp. 1292-1301, August 1968.

GONZAGA, A. ; FRANÇA, C.A. Um sistema de processamento de imagens dirigido a microcomputador compatível com PC. *Anais do SIBGRAPI V*, pp. 1-4, 1992.

GONZAGA, A. ; RODA, V.O., Método para reconhecimento de caracteres alfabéticos, *Jornada EPUSP IEEE em computação visual*, pp.239-247, 4-7 dezembro de 1990.

GONZAGA, A. *Tecnologia para o reconhecimento do formato de objetos tri-dimensionais*, São Carlos, 1991, 226p.

Tese (Doutorado) - Instituto de Física e Química de São Carlos, Universidade de São Paulo,

GOVINDA, V.K. ; SHIVAPRASAD, A.P. Character Recognition - a Review, *Pattern Recognition*, v.23, n.7, pp.671-683, 1990.

GRANLUND, G.H. Fourier preprocessing for handprint character recognition, *IEEE Transactions on Computers*, v. C-21, n.2, pp. 195-201, 1972.

GREANIAS, E.C.; MEAGHER, P.F.; NORMAN, R.J. ; ESSINGER, P., The recognition of handwritten numerals by countour analysis, *IBM Journal*, v.7, n.1, pp. 14-21, January 1963.

GRIMSDALE, R.L.; SUMNER, F.H.; TUNIS, C.J. ; KILBURN, M.A., A system for automatic recognition of patterns, *The Institution of Eletrical engineers*, n.2792 M, pp. 210-221, December 1958.

GÜDESEN, A., Quantitative Analysis of Prepro cessing Techiniques for the Recognition of Handprinted Characters, *Pattern Recognition*, v.8, pp. 219-227, 1976.

HABER, R.N. ; HABER, L. R. Visual components of the reading process, *Visible Language*, v. XV, n.2, pp. 147-182, 1981.

HAMMERSTROM, D. Neural networks at work, *IEEE Spectrum*, pp.26-32, June 1993.

HARMON, L.D. Automatic recognition of print and script, *Proceedings of the IEEE*, v.60, n.10, pp.1165-1176, October 1972.

HONDA, M.; ISHIGURO, H., ; MURAI, Y. Concept and technologies of personal OCR, *NEC Reserch Development*, n.75, pp. 16-22, October 1984.

HOFFMAN, R.L. ; McCULLOUGH, J.W. Segmentation Methods for Recognition of Machine-printed Characters, *IBM Journal of Reserch and Development*, pp. 153-165, March 1971.

HULL, J.J. Hypothesis generation in a computacional model for visual word recognition, *IEEE Expert*, pp. 63-70, Fall 1986.

HUSSAIN, A,B.S.; TOUSSAINT, G.T. ; DONALDSON, R.W. Results obtained using a simple character recognition procedure on Munson's handprinted data, *IEEE Transactions on Computers*, pp. 201-205, February 1972.

IKEDA, K.; YAMAMURA, T.; MITAMURA, Y.; FUJIWARA, S.; TOMINAGA, Y.; KIYONO, T. On-line recognition of hand-written characters utilizing positional and stroke vector sequences, *Pattern Recognition*, v.13, n.3, pp. 191-206, 1981.

ITO, M.R. ; CHUI, T.L. On-line computer recognition of proposed standard ANSI (USASI) handpried characters, *Pattern Recognition*, v.10, pp. 341-349, 1978.

JONES, M. OCR gets you from there to here, *PC Magazine*, pp. 267-319, July 1992.

KAHAN, S.; PAVLIDS, T.; BAIRD, H.S. On the recognition of printed characters of any font and size, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. PAMI-9, n.2, March 1987.

KAY, D.C.; LEVINE, J.R., "Graphics file formats", *Windcrest/McGraw-Hill*, 1992.

KERRICK, D.D. ; BOVIK, A.C. Microprocessor-based recognition of handprinted characters from a tablet input, *Pattern Recognition*, v.21, n.5, pp. 525-537, 1988.

KIMURA, F.; TAKASHINA, K.; TSURUOKA, S. ; MIYAKE, Y. Modified quadratic discriminant functions and the application to chinese characters recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. PAMI-9, n.1, pp. 149-153, January 1987.

KNOLL, A.L., Experiments with characteristic loci for recognition of handprinted characters, *IEEE Transactions on Computers*, v. C-18, pp. 366-372, April 1969.

LEBOURGEOIS, F.; HENRY, J.L.; EMPTOZ, H. An OCR System for Printed Documents, *IAPR Workshop on Machine Vision*

Application, pp.83-86, Dec.7-9, Tokyo, 1992.

LEEDHAM, C.G. ; DOWNTON, A.C., On-line recognition of Pitman's handwritten shorthand - an evaluation of potential, *International Journal of Man-Machine Studies*, v.24, pp. 375-393, 1986.

LEEDHAM, C.G. ; DOWNTON, A.C., Automatic recognition and Transcription of Pitman's handwritten shortforms - an approach to short forms, *Pattern Recognition*, v.20, n.3, pp. 341-348, 1987.

LI, X. ; BASU, A. Variable resolution character thinning, *Pattern Recognition Letters*, v.12, n.4, pp. 241-248, April 1991.

LINDGREN, N. Machine recognition of human language, part III - cursive script recognition, *IEEE Spectrum*, pp. 104-116, May 1965.

LOPES, M.A. ; MARQUES, O.F. Um sistema de reconhecimento ótico de caracteres usando características topológicas e algoritmo de aprendizado de máquina. *Anais do SIBGRAPI V* pp. 295-300, 1992.

LOTUFO, R.A. ; OLIVEIRA, J.R., Sistema de reconhecimento de placas de veículo, *Jornada EPUSP IEEE em computação visual*,

pp.249-255, 4-7 dezembro de 1990.

LOY, W.W. ;LANDAU, I.D. An on-line procedure for recognition of handprinted alphanumeric characters, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v.Pami-4, n.4, pp. 422-427, July 1982.

MANTAS, J. An overview of character recognition methodologies, *Pattern Recognition*, v.19, n.6, pp. 425-430, 1986.

MARQUES, O.F, Uma proposta de reconhecimento de caracteres utilizando correlação com máscaras binárias, *III Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens*, Gramado, pp. 80-89, 1990.

MARQUES, O.F, Sistema automático de reconhecimento óptico de caractere a partir de um conjunto de exemplos, *III Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens*, Gramado, pp. 90-100, 1990.

MARQUES, O.F, Sistemas de OCR: Fundamentos, aplicações, tendências e perspectivas, *Jornada EPUSP IEEE em computação visual*, pp.61- 80, 4-7 dezembro de 1990.

MORI, K.; GENCHI, H.; WATANABE, S. ; KATSURAGI, S. Microprogram controlled Pattern Processing in a handwritten

mail reader-sorter, *Pattern Recognition*, v.2, pp. 175-185, 1970.

NAGY, G. State of the art in Pattern Recognition, *Proceedings of the IEEE*, v.56, n.5, pp. 836-862, May 1968.

NAGY, G.; TUONG, N. Normalization techniques for handprinted numerals, *Communications of the ACM*, v.13, n.13, pp. 475-481, August, 1970.

NAKAMURA, Y.; SUDA, M.; SAKAI, K.; TAKEDA, Y; UDAKA, M.; Development of high performance stamped character reader, *IEEE Transactions on Industry Eletronics*, V.33, pp. 144-147, 1986.

NAUS, M.J.,; SHILLMAN, R.J., Why a y is not a V: A new look at the distinctive features of letters, *Journal of experimental Psychology: Human, Perception and Performance*, v.2, n.3, pp. 394-400, 1976.

NAYLOR, W.C., Some studies in the interactive design of character recognition systems, *IEEE Transactions on computers*, v.20, pp.1075-1086, 1971.

NOUBOUND, F. ; PLAMOND, R. On-line recognition of handprinted characteres: surveys and beta tests, *Pattern Recognition*, v.23, n.9, pp. 1031-1044, 1990.

OSÓRIO, F.S., Um sistema de reconhecimento de caracteres através de redes neurais, *Jornada EPUSP IEEE em computação visual*, pp.335-339, 4-7 dezembro de 1990.

OSÓRIO, F.S., Redes Neurais: Modelos de simulação e utilização no Reconhecimento de Padrões, *Jornada EPUSP IEEE em computação visual*, pp.39-58, 4-7 dezembro de 1990.

PEROTTO, P.G., A new method for automatic character recognition, *Transaction on Eletronic Computer*, v. EC-12, n.5, pp. 521-526, October 1963.

PFAFFENBERGER, B. QUE Dicionário dos usuários de micro computadores, *Editora Campus*, 1992.

POITEVENT, J.L., OCR for credit card processing, *Datamation*, pp. 49-52, July 1969.

RABINOW, J.C. Whiter OCR?, *Datamation*, pp. 38-42, July 1969.

SATO, k. et al. Hand-scan with a one dimensional image sensor, pp.459 - 467, 1983.

SAYRE, K.M. Machine recognition of handwritten words: a project report, *Pattern Recognition*, v.5, pp. 213-228, 1973.

SCHEIN, A. Optical storage and OCR - key components of automated information management systems, *Optical Information Systems*, pp.9-15, January/February 1989.

SCHERL, W.; HUNDT, E., A syntatic method of pattern recognition for document processing, *Siemens Forsch*, V.13 N.3, pp. 109- 113, 1984.

SEZAKI, N. ; KATAGIRI, H. Character recognition by follow method, *Proceedings of IEEE*, pp. 510, May 1965.

SEZAKI, N.; KATAGIRI, H. ; KANEBO, T. Pattern reproduction by follow method, *Proceedings of IEEE*, v.53, pp. 1656-1657, 1965.

SHEINBERG, I., The INPUT 2 document reader (a new optical character recognition system), v.2, pp.167-173, 1970.

SCHÜRMANN, J., A multifont word recognition system for postal address reading, *IEEE Transactions on computers*, v. C-27, n.8, pp. 721-732, August 1978.

SILVA, A.J..M., Reconhecimento Automático de Caracteres Impressos, *III Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens*, Gramado, pp. 111-120, 1990.

SIMEK. J.G. ; TUNIS, C.J., Handprinting input device for computer systems, *IEEE Spectrum*, pp. 72-81, July 1967.

SIY, P. ; CHEN, C.S. Fuzzy logic for handwritten numeral character recognition, *IEEE Transactions on System, Man and Cybernetics*, pp. 570-575, November 1974.

SMITH, R.W. Computer Processing of line Images: a survey, *Pattern Recognition*, V.20, pp. 7-15, 1978.

SRIHARI, S.N.; HULL, J.J.;CHOUDHARI, R., Integrating diverse knowledge sources in text recognition, *ACM Transactions on Office Information Systems*, v.1, n.1, pp.68-87, January 1983.

STALLINGS, W. Approaches to chinese character recognition, *Pattern Recognition*, v.8, pp.87-98, 1976.

STEVENS, M.E. Selected Pattern Recognition Projects in Europe, *Pattern Recognition*, v1, pp. 103-118, 1968.

SUEN, C.Y. ; SHILLMAN, R., Low error rate optical character recognition of unconstrained handprinted letters based on a model of human perception, *IEEE Transactions on System, Man and Cybernetics*, pp. 491-495, June 1977.

SUEN, C.Y., A study on man-machine interactions problems in character recognition, *IEEE Transaction on Systems, man, and Cybernetics*, V. SMC-9, N.11, pp. 732-737, 1979.

SUEN, C.H.; Berthod, M ; Mori, S. Automatic recognition of handprinted characters, - the state of the art, *Proceedings of the IEEE*, v.68, n.4, pp. 469 - 485, April 1980.

SUEN, C.Y., Distinctive features in automatic recognition of handprinted characters, *Signal Processing*, v.4, pp. 193-207, 1982.

SUEN, C.Y., Handwritting generation, perception and recognition, *Acta Psicológica*, v.54, pp. 295-312, 1983.

SUEN, C.H., Handbook of Pattern Recognition - Cap.23: Character Recognitions by computer and applications, Academic Press, inc., pp. 569-586, 1986.

TAPPERT, C.C.; SUEN, C.Y. ;WAKAHARA, T. The state of the art in on-line handwritting reconition, *IEEE Transaction on attern analysis and Machine Intelligence*, v.12, n.8, pp. 787 - 809, August, 1990.

TERSOFF, A.I., Man-machine consideration in automatic handprinted recognition, *IEEE Transactions on Systems, Man & Cybernetics*, v. SMC-8, n.4, pp. 279, April 1978.

THOMASON, M.G., Handbook of Pattern Recognition - Cap.5: Syntactic Pattern Recognition: Stochastic languages, Academic Press, inc., pp. 119-142, 1986.

THROSSELL, W.R. ; FRYER, P.R., The measurement of print quality for optical character recognition systems, *Pattern Recognition*, v.6, pp. 141-147, 1974.

TOU, J.T. ; Gonzalez, R.C. Recognition of handwritten characters by topological feature extraction and multilevel categorization, *IEEE Transactions in Computers*, pp. 776-785, July 1972.

TOUSSAINT, G.T. ; DONALDSON, R.W. Algorithms for recognizing counter-traced handprinted characters, *IEEE Transactions on Computers*, pp. 541-546, June 1970.

TRIENDL, E.E. Skeletization of noisy handdraw symbols using parallel operations, *Pattern Recognition*, v.2, pp. 241-248, 1969.

TROXEL, D.E. Feature selection for low error rate OCR, *Pattern Recognition*, v.8, pp. 73-76, 1976.

ULLMANN, J.R.; KIDD, P.A. Recognition Experiments with typed numerals from envelopes in the mail, *Pattern Recognition*, v.1, pp. 273-289, 1969.

WANG, P.S.P; LU, H.E., A comment on a fast parallel algorithm for thinning digital patterns, *Communications of the ACM*, V.29, N.3, pp. 239-242, 1986.

WARD, J.R.; BLESSER, B.; Interactive recognition of handprinted characters for computer input, *IEEE Computer Graphics and applications*, pp. 24- 37, September 1985.

WARD, J.R.; PHILLIPS, M.J., Digitizer technology: performance characteristics and the effects on the user interface, *IEEE Computer Graphics and applications*, pp.31-43, April 1987.

WARD, J.R.; KUKLINSK, T., A model for variability effects in handprinting with implications for design of handwriting character recognitions systems, *IEEE Transactions on Systems, man and Cybernetics*, V.18, N.3, pp. 438-451, May/June 1988.

WELCH, T.A., A technique for high-performance data compression, *Computer*, pp.9-19, June 1984.

WENDLING, S.; GAGNEUX, G. ; STAMON, G. A set of invariants within the power spectrum of unitary transformation, *IEEE Transactions on Computers*, v.27, 1978, pp. 1213-1216.

WHITE, J.M. ; ROHRER, G.D., Image thresholding and others applications requiring characters image extraction, *IBM Journal Reserch and Development*, v.27, n.4, pp. 400-410, July 1983.

WING, A.M. Variability in handwritten charaters, *Visible Language*, v.XIII, n.3, pp. 283-2298, 1979.

WING, A.M.; NIMMO-SMITH, I. ; ELDRIDGE, M.A., The consistency of a cursive letter formation as a function of position in the word, *Acta Psychologica*, v.54, pp. 197-204, 1983.

YACYK, J. Alphabetic handprint reading, *IEEE Transactions on Systems, man, and Cybernetics*, v. SMC-8, n.4, pp. 279-282, April 1978.

ZHANG, T.Y. ; SUEN, C.Y. A fast parallel algorithm for thinning digital patterns, *Communications of the ACM*, v.27, n.3, pp.236-239, March 1984.

APÊNDICE 1

UNIVERS

15

[illegible]

T10

[illegible]

T15

[illegible]

T25

A B C D E F G H I J K L
M N O P Q R S T U V W X
Y Z

T50

A B C D E F G
H I J K L M
N O P Q R S T
U V W X Y Z

T50

A B C D E F
G H I J K L M
N O P Q R S T
U V W X Y Z

T75

A B C D E F

G H I J K L

M N O P Q

R S T U V

W X Y Z

T100

A B C D E F

G H I J K L

M N O P Q

R S T U V W

X Y Z

ITC AVANT GARD GOTHIC BOOK

T25

A B C D E F G H I J K L

M N O P Q R S T U V W X

Y Z

T50

A B C D E F G H I

J K L M N O P Q R

S T U V W X Y Z

T75

A B C D E F

G H I J K L

M N O P Q

R S T U V W

X Y Z

T100 ABC

D E F G H

I J K L M

N O P Q

R S T U V

W X Y Z

ITC BOOKMAN DEMI

T25

A B C D E F G H I J K L

M N O P Q R S T U V W X

Y Z

T50

A B C D E F G H

I J K L M N O P

Q R S T U V W X

Y Z

T75

A B C D E

F G H I J K

L M N O P

Q R S T U

V W X Y Z

T100

A B C D E

G H I J K L

M N O P Q

R S T U V

W X Y Z

NEW CENTURY SCHOOL BOOK

T25

A B C D E F G H I J K L

M N O P Q R S T U V W X

Y Z

T50

A B C D E F G H

I J K L M N O P

Q R S T U V W X

Y Z

T 75

A	B	C	D	E	
F	G	H	I	J	K
L	M	N	O	P	
Q	R	S	T	U	
V	W	X	Y	Z	

T100

A B C D F G

H I J K L M

N O P Q R

S T U V W

X Y Z

PALATINO

T25

A B C D E F G H I J K L

M N O P Q R S T U V W Y

Z

T50

A B C D E F G H I

J K L M N O P Q

R S T U V W X Y

Z

TIMES ROMAN

T25

A B C D E F G H I J K L

M N O P Q R S T U V W X

Y Z

T50

A B C D E F G H I

J K L M N O P Q R

S T U V W X Y Z

T75

A B C D E F

G H I J K L

M N O P Q

R S T U V W

X Y Z

T100

A B C D E F

G H I J K L

M N O P Q

R S T U V

W X Z

COURIER

T25

A B C D E F G H I J K L

M N O P Q R S T U V W X

Y Z

T50

A B C D E F G

H I J K L M N

O P Q R S T U

V X Y Z

T75

A B C D E

F G H I J

K L M N O

P Q R S T

U V W X Y

Z

T100

A B C D E F G

H I J K L M N

O P Q R S T U

V W X Y Z

CG TIMES

T25

A B C D E F G H I J K L
M N O P Q R S T U V W X
Y Z

T50

A B C D E F G H I
J K L M N O P Q R
S T U V W X Y Z

T75

A B C D E F

G H I J K L

M N O P Q

R S T U V W

X Y Z

T100

A B C D E F

G H I J K L

M N O P Q

R S T U V W

X Y Z

MODERN

A B C D E F G H
I J K L M N O P
Q R S T U V W X
Y Z

ARIAL

A B C D E F G
H I J K L M N O
P Q R S T U V
W X Y Z

ARIAL ROUNDED - MT

A B C D E F G
H I J K L M N O
P Q R S T U V
W X Y Z

BOOKMAN OLD STYLE

A B C D E F
G H I J K L M
N O P Q R S
T U V W X Y
Z

BRITANNIC BOLD

A B C D E F G
H I J K L M N O
P Q R S T U V
W X Y Z

CENTURY GOTHIC

A B C D E F G
H I J K L M N O
P Q R S T U V
W X Y Z

FOOTLIGHT MT LIGHT

A B C D E F G H I
J K L M N O P Q
R S T U V W X Y
Z

IMPACT

A B C D E F G H I J K
L M N O P Q R S T U
V W X Y Z

MS LINE DRAW

A B C D E F

G H I J K L

M N O P Q R

S T U V W X

Y Z

APÊNDICE 2

**ALGORITMO PARA RECONHECIMENTO ÓPTICO DE CARACTERES MULTI-FONTE, BASEADO EM SUAS LINHAS
FUNDAMENTAIS**

Desenvolvido por Evelina Maria de Almeida Neves

```
/* Um programa para procurar uma imagem .IMG */
```

```
#include "mem.h"  
#include "conio.h"  
#include "stdio.h"  
#include "alloc.h"  
#include "dos.h"  
#include "graphics.h"  
#include "stdarg.h"
```

```
char linebuf[8192];
```

```
void main(), init(), deinit(), readcolorB(), readcolorT();  
int pesquisaBE(int, int), pesquisaTE (int, int);  
void ShowImgPicture(), TELA();  
void gprintxy();  
void findclasses ();  
void clean();  
int histogramming(int[], int[]);  
int XB=0,YB=0,XT=0,YT=0;  
int x1=0,x2=0;  
int l=0;  
int h=0;  
int v1,v2,v3,v4,v5,v6,v7,v8,v9;  
int cont1=0;  
int cont2=0;
```

```
void classe1();  
void classe2();  
void classe3();  
void classe4();  
void classe5();  
void classe6();  
void classe7();  
void classe8();  
void classe9();  
void classe10();  
void classe11();
```

```
int UnpackImgFile(),ReadImgLine();
```

```
unsigned int width,depth;  
unsigned int patternsize=2;  
unsigned int bytes,repcount;  
unsigned int pixels2bytes();  
char *farPtr();
```

```
void main(argc,argv)  
int argc;
```

```
char *argv[];  
{  
    char header[16];  
    FILE *fp;  
    char *p;
```

```

if(argc > 1) {
    /* atencao para a abertura de arquivo */
    if((fp = fopen(argv[1], "rb")) != NULL) {
        /* leitura do cabecalho */
        if (fread(header, 1, 16, fp) == 16) {
            /* certificar-se se e uma figura */
            if(!memcmp("\x00\x01\x00\x08\x00\x01", header, 6)) {
                /* aloca um buffer */
                patternsiz = (header[7] & 0xff) + ((header[6] & 0xff) < 8);
                width = (header[13] & 0xff) + ((header[12] & 0xff) < 8);
                depth = (header[15] & 0xff) + ((header[14] & 0xff) < 8);
                bytes = pixels2bytes(width);
                TELA();

                p = farmalloc(4L + (long)bytes*(long)depth);
                if (p != NULL) {
                    /* unpack o arquivo */
                    if(UnpackImgFile(farPtr(p + 4L, 4L), fp) == bytes)
                        /* Mostra a figura */
                        ShowImgPicture(farPtr(p, 4L));
                    // clean();
                    readcolorT();
                    readcolorB();
                    findclasses();
                    farfree(p);

                } else puts("Erro alocao memoria");
                } else printf("Nao e um arquivo IMG. \n");
            } else printf("Erro de leitura %s.\n", argv[1]);
            fclose(fp);
        } else printf("Erro de abertura %s.\n", argv[1]);
    }
    else printf("Digite o nome do arquivo:");
    closegraph();
}

```

```

void ShowImgPicture(p) /* Mostra o topo da figura */
char *p;
{
    register int i, rot;
    long j;
    char ch = 'c';
    union scan {
        int set;
        char ch[2];
    } sc;

    j = 0;
    rot = 4L;
    /* Copia o grafico para a tela */
    for(;;) {
        for(i = j; i < j + 480; i++)
            memcpy(MK_FP(0xa000, 80*(i-j)),
                farPtr(p, rot + ((long)i*(long)bytes)), bytes);
        sc.set = get_key();

        switch(sc.ch[1]) {
            case 72: j = j - 10;
                if(j < 0) j = 0;
                break;
            case 80: j = j + 10;
                if(j > depth-480) j = (depth-480);
                break;
            case 77: rot = ((long)rot + 2L);
                break;
            case 75: rot = ((long)rot - 2L);
                if(rot < 4L) rot = 4L;
                break;
            default: ch = 's';
        }
    }
}

```



```

    }
    if(ch == 's') break;
}
}

```

```

get_key(void)
{
    union REGS r;
    r.h.ah=0;
    return int86(0x16, &r, &r);
}

```

```

UnpackImgFile(p,fp) /* Abre e imprime a imagem GEM/IMG n */
char *p;
FILE *fp;
{
    register int i;
    unsigned int n;

```

```

    for(i=0;i<depth;) {
        if((n=ReadImgLine(linebuf,fp)) != bytes) break;
        while(repcount--) {
            memcpy(p,linebuf,bytes);
            p += i;
            p = farPtr(p,(long)bytes);
        }
        return(n);
    }
}

```

```

ReadImgLine(p,fp) /*Leitura e codificacao de uma linha GEM/IMG atraves de p*/

```

```

char *p;
FILE *fp;
{
    char *pr;
    int n=0,c,i;
    register int j, k;

    memset(p,0,bytes);

    /*ajuste do default do contador de replicacao */

    repcount = 1;

    /*fica em loop ate a codificacao de toda a linha*/
    do {
        c=fgetc(fp) & 0xff;
        if(c==0) {
            /*e o padrao ou a replicacao*/
            c=fgetc(fp) & 0xff;
            if(c==0) {
                /*e a mudanca do contador de replicacao*/
                fgetc(fp);
                /*joga o ff*/
                repcount = fgetc(fp) & 0xff;
            }
        }
        else {
            i=c & 0xff;
            pr=p+n;
            j=patternsize;
            while(j--) p[n++] = fgetc(fp);
            k=i-1;
            while(k--) {
                memcpy(p+n,pr,patternsize);
                n += patternsize;
            }
        }
    }
}

```

```

    }
    }
    else if(c == 0x80) {
        /* e uma cadeia de bytes */
        i = fgetc(fp) & 0xff;
        pr = p + n;
        j = i;
        while(j--) p[n++] = ~fgetc(fp);
    }
    else if(c & 0x80) {
        /* e um branco cheio-aqui ocorre uma inversao artificial */
        i = c & 0x7f;
        pr = p + n;
        j = i;
        while(j--) p[n++] = ~0xff;
    }
    else {
        /* e um preto cheio */
        i = c & 0x7f;
        pr = p + n;
        j = i;
        while(j--) p[n++] = ~0x00;
    }
} while(n < bytes);
return(n);
}

```

```

unsigned int pixels2bytes (n)
unsigned int n;
{
    if(n & 0x0007) return((n > 3) + 1);
    else return(n > 3);
}

```

```

char *farPtr(p,l)
char *p;
long l;
{
    unsigned int seg, off;

    seg = FP_SEG(p);
    off = FP_OFF(p);
    seg += (off / 16);
    off &= 0x000f;
    off += (unsigned int) (l & 0x000fL);
    seg += (l / 16L);
    p = MK_FP(seg,off);
    return(p);
}

```

```

void TELA()
{
    int driver, mode;
    char str[5];

    driver = DETECT;
    initgraph(&driver,&mode,"C:\TC\BGI");
    setfillstyle(SOLID_FILL,9);
    bar(0,0,639,259);
    setcolor(5);
    settextrstyle(GOTHIC_FONT, HORIZ_DIR, 8);
    outtextxy(0, 15, " O C R ");
    settextrstyle(GOTHIC_FONT, HORIZ_DIR, 5);
    outtextxy(10, 150, "Laboratorio Visao Computacional");
    outtextxy(29, 200, "Software by Evelina M. A. Neves");
    setfillstyle(SOLID_FILL,1);
    bar(0,260,639,479);
    settextrstyle(TRIPLEX_FONT, HORIZ_DIR, 5);
    outtextxy(10, 285, "Esta imagem contem");
}

```

```

gprintxy(520, 285, "%u", depth);
outtextxy(10, 320, "linhas");
settextstyle(TRIPLEX_FONT, HORIZ_DIR, 5);
outtextxy(10, 400, "com ");

gprintxy(100, 400, "%u", bytes);
outtextxy(250, 400, "bytes cada linha");
getch();
cleardevice();
}

void gprintxy(int xloc, int yloc, char *fmt)
{
    va_list argptr;
    char str[20];
    struct textsettingstype textinfo;
    va_start(argptr, fmt);
    vsprintf(str, fmt, argptr);
    gettextsettings(&textinfo);
    outtextxy(xloc, yloc, str);
    va_end( argptr );
}

void readcolorT() //DETERMINA PONTO DE TOPO DA IMAGEM
{
    int i,j;
    int corT, FT=0;

    for(j=0; j<depth; j++)
    {
        for(i=0; i<(8*bytes); i++)
        {
            if(FT==0)
            {
                corT = getpixel(i,j);

                if (corT==0)
                {
                    if (pesquisaTE(i,j)==1)
                    {
                        XT=i;
                        YT=j;
                        FT=1;
                    }
                }
            }
        }
    }
}

void readcolorB() //DETERMINA PONTO DE BASE DO CARACTER
{
    int i,j,corB,FB=0;

    for (j=depth; j>0; j--)
    {
        for(i=0; i<(8*bytes); i++)
        {
            if(FB==0)
            {
                corB = getpixel(i,j);

                if (corB==0)
                {
                    if (pesquisaBE(i,j)==1)

```



```

    {
        XB=i;
        YB=j;
        FB=1;
    }
}
}
}
getch();
}

```

```

pesquisaBE (int i, int j) //PESQUISA SE PONTO DE BASE DA IMAGEM E RUÍDO
{
    int cori, corj, cork, corl, corn;

    cori=getpixel((i+1),j);
    corj=getpixel((i+2),j);
    cork=getpixel(i,(j-1));
    corl=getpixel((i+1),(j-1));
    corn=getpixel(i,(j-2));
    if(((cori==0) && (cork==0) && (corl==0)) && ((corn==0) || (corj==0)))
        return (1); /* caracter */
    else
    {
        putpixel(i,j,15);
        return (0); /* ruído */
    }
}

```

```

pesquisaTE (int i, int j) //PESQUISA SE PONTO DE TOPO DA IMAGEM E RUÍDO
{
    int cora, corb, corc, cord, corf, a, b;

    cora=getpixel((i+1),j);
    corb=getpixel((i+2),j);
    corc=getpixel(i,(j+1));
    cord=getpixel((i+1),(j+1));
    corf=getpixel(i,(j+2));
    if(((cora==0) && (corc==0) && (cord==0)) && ((corb==0) || (corf==0)))
        return (1); /* caracter */
    else
    {
        putpixel(i,j,15);
        return (0); /* ruído */
    }
}

```

```

void findclasses()
{
    int i,k;
    int vetor1[250];
    int vetor2[250];
    int vetor3[250];
    int vetor4[250];

    vetor1[0]=0;
    vetor2[0]=0;
    vetor3[0]=0;
    h=abs(YB-YT);

    v1=0.3*h + YT;
}

```

```

v2=0.8*h + YT;
v3=0.4*h + YT;
v4=0.7*h + YT;
v5=0.5*h + YT;
v6=0.6*h + YT;
v7=0.9*h + YT;
v8=0.2*h + YT;
v9=0.1*h + YT;
for(i=1; i<(8*bytes); i++) //calculo do numero de transicoes na linha de varredura horizontal 1
{
    vetor1[i] = (getpixel((i+1),v1)) - (getpixel(i,v1));
    if (vetor1[i] == 15)
        cont1 = cont1 + 1;
}

for(k=1; k<(8*bytes); k++) //calculo do numero de transicoes na linha de varredura horizontal 2
{
    vetor2[k] = (getpixel((k+1),v2)) - (getpixel(k,v2));
    if (vetor2[k] == 15)
        cont2 = cont2 + 1;
}

I = histogramming(vetor3, vetor4);

//determinacao das classes

/* todas as variacoes de M, N e W (4/4, 4/3, 4/2,3/4, 3/3, 3/2,2/4,2/3) exceto 2/2: */

// CASO 1: (3/4) OU (2/4) "M,W"

if((cont1 == 3 || cont1 == 2) && cont2 == 4) //distincao entre M e W atraves da inclinacao de suas retas tomadas
em 70% e 80% de h
{
    classe1(vetor2,vetor3);
}

//CASO 2: (4/2) OU (4/3) M,W

if(cont1 == 4 && (cont2 == 2 || cont2 == 3)) // distincao entre W e M atraves da inclinacao de suas retas tomadas
em 30% e 40% de h
{
    classe2(vetor1,vetor3);
}

//CASO 3: M,W (4/4)

if (cont1 == 4 && cont2 == 4) //distingue entre W e M em funcao da inclinacao de suas retas tomadas em 30% e 80%
de h
{
    classe3(vetor1,vetor2);
}

//CASO 4: distincao entre E,M,N e W atraves do numero de transicoes em h/2 e da medida de inclinacao de suas retas
if(cont1 == 3 && cont2 == 3)
{
    classe4(vetor3);
}

//CASO 5: distincao entre E,F,M,N e W quando 3/2

if(cont1 == 3 && cont2 == 2)
{
    classe5(vetor2,vetor3);
}

```

```

//CASO 6: distincao entre Q,M,N e W quando 2/3
if(cont1 == 2 && cont2 == 3)
{
    classe6(vetor2, vetor3);
}

//CASO 7: distincao entre E,F,T (3/1) atraves de varredura vertical em 50% de l
if(cont1 == 3 && cont2 == 1)
{
    classe7(vetor3);
}

//CASO 8:Distingue entre A,C,E,F,P,Q,V,Y, quando 2/1
if(cont1 == 2 && cont2 == 1)
{
    classe8(vetor3);
}

} //fecha o caso 8

//CASO 9: distingue entre A,C,E,F,G,J,L,S,Z quando 1/2

if(cont1 == 1 && cont2 == 2)
{
    classe9(vetor1,vetor2,vetor3);
} // fecha o caso 9

//CASO 10: distingue entre A,C,E,F,I,J,L,S,T,Z quando 1/1

if(cont1 == 1 && cont2 == 1)
{
    classe10(vetor1,vetor2,vetor3);
}

//CASO 11: distingue entre A,B,C,D,E,G,H,K,M,N,O,Q,R,S,U,V,W,X,Z quando 2/2

if(cont1 == 2 && cont2 == 2)
{
    classe11(vetor1,vetor2,vetor3,vetor4);
}

getch();
} // fecha o programa

//FUNCAO LIMPA RUIDO DA IMAGEM:

void clean()
{
    int i,j;
    int cora, corb, corc, cord, corf, cor;

    for (j=0; j<(depth-3); j++)
    {
        for(i=0; i<(8*bytes-3); i++)
        {

            cor=getpixel(i,j);
            if(cor==0)
            {
                cora=getpixel((i+1),j);
                corb=getpixel((i+2),j);
                corc=getpixel(i,(j+1));
                cord=getpixel((i+1),(j+1));
                corf=getpixel(i,(j+2));
                if(!(((cora==0) && (corc==0) && (cord==0)) && ((corb==0) || (corf==0))))

                    putpixel(i,j,15);
            }
        }
    }
}

```



```

    }
}

```

```

//DETERMINACAO DA LARGURA DOS CARACTERES:

```

```

histogramming (int vetor4[], int vetor5[])

```

```

{
    int i,j,cor;

```

```

    int cont=0;
    int larg=0;
    vetor5[0]=1;
    for(i=1; i<(8*bytes-1); i++)
    {
        for(j=1; j<(depth-1); j++)
        {
            cor=getpixel(i,j);
            if (cor==0)
                cont=cont++;
        }
        vetor4[i]=cont;
        cont=0;
    }

```

```

    for(i=1; i<(8*bytes-1); i++)
    {
        if(vetor4[i]>=3)
            vetor5[i]=1;
        else
            vetor5[i]=0;
    }

```

```

    for(i=0; i<(8*bytes-1); i++)
    {
        vetor4[i+1]=(vetor5[i+1]-vetor5[i]);
        if(vetor4[i]==1)
            x1=i;
    }
    for(i=0; i<(8*bytes-1); i++)
    {
        vetor4[i+1]=(vetor5[i+1]-vetor5[i]);
        if(vetor4[i]==-1)
            x2=i;
    }

```

```

    larg=(x2-x1);
    return(larg);
}

```

```

//CLASSE 1 (3/4) OU (2/4) "M,W"

```

```

void classe1(int vetor2[], int vetor3[]) ////distincao entre M e W atraves da inclinacao de suas retas tomadas em 70%
e 80% de h
{

```

```

    int m,i,f2=0,d1=0,n,f1=0,d2=0;

```

```

    for(m=1; m<(8*bytes); m++)
    {
        vetor3[m]=((getpixel((m+1),v4))-getpixel(m,v4));
    }
    for(m=1; m<(8*bytes); m++)
    {
        if(vetor3[m]==15)
        {
            if(f2<=1)

```

```

{
d1 = abs(d1-m);
f2 = f2 + +;
}
}
}

```

```

for(n=1;n<(8*bytes); n+ +)
{
if(f1 <= 1)
{
if(vetor2[n] == 15)
{
d2 = abs(d2-n);
f1 = f1 + 1;
}
}
}
}

```

//SUBCLASSE 1.1 (NO) "W"

```

if(d2 < d1)
printf("W\la");

```

//SUBCLASSE 1.2 (NO) "M"

```

else
printf("M\la");

```

```

}

```

//CLASSE 2 (4/2) OU (4/3) "M,W"

void classe2(int vetor1[], int vetor3[]) // distincao entre W e M atraves da inclinacao de suas retas tomadas em 30% e 40% de h

```

{
int m,d2=0,f2=0,n=0,f1=0,d1=0;
for(m=1; m<(8*bytes); m+ +)
{
vetor3[m] = ((getpixel((m+1),v3)) - (getpixel(m,v3)));
}
for(m=1; m<(8*bytes); m+ +)
{
if(vetor3[m] == 15)
{
if(f2 <= 1)
{
d2 = abs(d2-m);
f2 = f2 + +;
}
}
}
}
}

```

```

for(n=1;n<(8*bytes); n+ +)
{
if(f1 <= 1)
{
if(vetor1[n] == 15)
{
d1 = abs(d1-n);
f1 = f1 + 1;
}
}
}
}

```

//SUBCLASSE 2.1 (NO) "W"

```

if(d2 < d1)

```

```

printf("W\\a");

//SUBCLASSE 2.2 (NO) "M"

else
printf("M\\a");

}

//CLASSE 3 (4/4) "M,W"

void classe3(int vetor1[], int vetor2[]) //distingue entre W e M em funcao da inclinacao de suas retas tomadas em 30%
e 80% de h
{
int i,f1=0,d1=0,d2=0,f2=0;

for(i=1;i<(8*bytes);i++)
{
if(f1<=1)
{
if(vetor1[i]==15)
{
d1=abs(d1-i);
f1=f1+1;
}
}
}
for(i=1;i<(8*bytes);i++)
{
if(f2<=1)
{
if(vetor2[i]==15)
{
d2=abs(d2-i);
f2=f2+1;
}
}
}

// SUBCLASSE 3.1 (NO) "W"

if(d2<d1)
printf("W\\a");

//SUBCLASSE 3.2 (NO) "M"
else
printf(" M\\a");

}

//CLASSE 4 (3/3) "E,M,N e W"

void classe4(int vetor3[])
{
int vv1=0,y,i,cont4=0,cont3=0,f1=0,d1=0,m,f2=0,d2=0;
vv1=x1+l*0.5;
for(y=0;y<depth;y++) //distingue E de M,N,W atraves de varredura vertical em 50% de larg
{
vetor3[y+1]=(getpixel(vv1,(y+1))-getpixel(vv1,y));
if(vetor3[y+1]==-15)
cont4=cont4++;
}

//SUBCLASSE 4.1 (NO) "E"

if(cont4==3)
printf("E\\a");

//SUBCLASSE 4.2 "M,N,W"

```



```

else
{
    for(i=1; i<(8*bytes); i++) //calculo do numero de transicoes na linha de varredura horizontal 5
    {
        vetor3[i] = ((getpixel((i+1),v5)) - (getpixel(i,v5)));
        if (vetor3[i] == 15)
            cont3 = cont3 + 1;
    }

    //SUB 4.2.1 (NO) "N"

    if (cont3 == 3)
        printf("N\\a");

    //SUB 4.2.2 "M,W"

    else
    {
        for(i=1; i<(8*bytes); i++)
        {
            if(f1 <= 1)
            {
                if(vetor3[i] == 15)
                {
                    d1 = abs(d1-i);
                    f1 = f1 + 1;
                }
            }
        }

        for(m=1; m<(8*bytes); m++)
        {
            vetor3[m] = ((getpixel((m+1),v6)) - (getpixel(m,v6)));
        }
        for(m=1; m<(8*bytes); m++)
        {
            if(vetor3[m] == 15)
            {
                if(f2 <= 1)
                {
                    d2 = abs(d2-m);
                    f2 = f2 + 1;
                }
            }
        }

        //SUB 4.2.2.1 (NO) "W"

        if(d2 < d1)
            printf("W\\a");

        //SUB 4.2.2.2 (NO) "M"
        else
            printf("M\\a");
    }

    //CLASSE 5 (3/2) "E,F,M,N e W"

    void classe5(int vetor2[], int vetor3[])
    {
        int vv1=0,y,cont4=0,cont3=0,i,f1=0,d2=0,m,d1=0,f2=0;
        vv1 = x1 + l*0.5;
        for(y=0; y<depth; y++) //distincao de E,F de M,N,W atraves de varredura vertical em 50% de l
        {
            vetor3[y+1] = (getpixel(vv1,(y+1))-getpixel(vv1,y));
            if(vetor3[y+1] == -15)

```

```

cont4 = cont4 + +;
}

//SUBCLASSE 5.1 (NO) "E"

if(cont4 == 3)
printf("E\ta");

//SUBCLASSE 5.2 (NO) "F"
if(cont4 == 2)
printf("F\ta");

//SUBCLASSE 5.3 "M,N,W"

if(cont4 == 1)
{
for(i = 1; i < (8*bytes); i + +) //calculo do numero de transicoes na linha de varredura horizontal 5
{
vetor3[i] = ((getpixel((i + 1),v5)) - (getpixel(i,v5)));
if (vetor3[i] == 15)
cont3 = cont3 + +;
}

//SUB 5.3.1 "M,N"

if (cont3 == 3) //distincao entre W e N em 30% e 50% de h
{
for(i = 1; i < (8*bytes); i + +)
{
if(f1 <= 1)
{
if(vetor3[i] == -15)
{
d2 = abs(d2-i);
f1 = f1 + 1;
}
}
}

for(m = 1; m < (8*bytes); m + +)
{
vetor2[m] = ((getpixel((m + 1),v1)) - (getpixel(m,v1)));
}

for(m = 1; m < (8*bytes); m + +)
{
if(vetor2[m] == -15)
{
if(f2 <= 1)
{
d1 = abs(d1-m);
f2 = f2 + 1;
}
}
}
}

//SUB 5.3.1.1 (NO) "W"

if(d2 < d1)
printf("W\ta");

//SUB 5.3.1.2 (NO) "N"

else
printf("N\ta");
}

//SUB 5.3.2 "M,W"

if(cont3 == 4) // distincao entre W e M em 40% e 50% de h

```

```

{
for(i=1;i<(8*bytes); i++)
{
if(f1 <= 1)
{
if(vetor3[i] == 15)
{
d2=abs(d2-i);
f1=f1+1;
}
}
}

for(m=1; m<(8*bytes); m++)
{
vetor3[m]=((getpixel((m+1),v3)) - (getpixel(m,v3)));
}
for(m=1; m<(8*bytes); m++)
{
if(vetor3[m] == 15)
{
if(f2 <= 1)
{
d1=abs(d1-m);
f2=f2++;
}
}
}
//SUB 5.3.2.1 (NO) "W"

if(d2<d1)
printf("W\\a");

//SUB 5.3.2.2 (NO) "M"
else
printf("M\\a");
}
}

//CLASSE 6 (2/3) "Q,R,M,N,W "

void classe6(int vetor2[], int vetor3[])
{
int vv1=0,y,cont4=0,cont3=0,i,f3=0, f1=0,d2=0,m,d1=0,f2=0,d3=0;
vv1=x1+l*0.5;
for(y=0; y<(depth-1); y++) //distincao entre Q e de M,N,W atraves de varredura vertical em 50% de l
{
vetor3[y+1]=(getpixel(vv1,(y+1))-getpixel(vv1,y));
if(vetor3[y+1]==-15)
cont4=cont4++;
}

//SUBCLASSE 6.1 "Q,R"

if(cont4==3)
{
for(y=1; y<(depth-1); y++)
{
if(vetor3[y]==-15)
{
if(f3 <= 1)
{
d3=abs(d3-y);
f3=f3++;
}
}
}
}

```



```

}

//SUB 6.1.1 (NO) "Q"

if(d3>0.6*h)
printf("Q\la");

//SUB 6.1.2 (NO) "R"
else
printf("R\la");
}

//SUBCLASSE 6.2 "M,N,W"

else
{
for(i=0; i<(8*bytes); i++) //calculo do numero de transicoes na linha de varredura horizontal 5
{
vetor3[i+1] = ((getpixel((i+1),v5)) - (getpixel(i,v5)));
if (vetor3[i] == 15)
cont3=cont3++;
}

//SUB 6.2.1 "M,N"

if (cont3 == 3) //distincao entre M e N atraves da inclinacao em 50% e 80% da h
{

for(i=1; i<(8*bytes); i++)
{
if(vetor3[i] == 15)
{
f1=f1+1;
if(f1>0)
{
d1=abs(d1-i);
}
}
}

for(i=1; i<(8*bytes); i++)
{
if(vetor2[i] == 15)
{
f2=f2+1;
if(f2>1)
{
d2=abs(d2-i);
}
}
}

//SUB 6.2.1.1 (NO) "N"

if(d1>d2)
printf("N/a");

//SUB 6.2.1.2 (NO) "M"

else
printf("M/a");

}

//SUB 6.2.2 "M,W"

else // distincao entre M e W em 50% e 60% de h
{
for(i=1; i<(8*bytes); i++)
{

```

```

if(f1 <= 1)
{
if(vetor3[i] == 15)
{
d1 = abs(d1-i);
f1 = f1 + 1;
}
}
}

for(m=1; m<(8*bytes); m++)
{
vetor3[m] = ((getpixel((m+1),v6)) - (getpixel(m,v6)));
}
for(m=1; m<(8*bytes); m++)
{
if(vetor3[m] == 15)
{
if(f2 <= 1)
{
d2 = abs(d2-m);
f2 = f2 + 1;
}
}
}

//SUB 6.2.2.1 (NO) "W"

if(d2 < d1)
printf("W\\a");

//SUB 6.2.2.2 (NO) "M"

else
printf("M\\a");
}
}

//CLASSE 7 (3/1) "E,F,T"

void classe7(int vetor3[])
{
int vv1=0,y,cont4=0,d3=0,vv2=0;

vv1 = x1 + l*0.5;
for(y=0; y<depth; y++)
{
vetor3[y+1] = (getpixel(vv1,(y+1))-getpixel(vv1,y));
if(vetor3[y+1] == -15)
cont4 = cont4 + 1;
}

//SUBCLASSE 7.1 (NO) "T"

if(cont4 == 1)
printf("T\\a");

//SUBCLASSE 7.2 "F,T"

if(cont4 == 2) //distingue entre F e T atraves da distancia entre as transicoes verticais
{
for(y=1; y<depth; y++)
{
if(vetor3[y] == 15)
d3 = abs(d3-y);
}
}

```

```

//SUB 7.2.1 (NO) "T"

if(d3 >= 0.8*h)
printf("T\0a");

//SUB 7.2.2 (NO) "F"
else
printf("F\0a");
}

//SUBCLASSE 7.3 "E,F"

if(cont4 == 3) //distingue entre E e F atraves de outra varredura vertical em 90% de l
{
vv2 = x1 + l*0.9;
vetor3[0] = 15;
cont4 = 0;
for(y=0; y < depth; y++)
{
vetor3[y+1] = (getpixel(vv2,(y+1)) - getpixel(vv2,y));
if(vetor3[y+1] == -15)
cont4 = cont4 + 1;
}

//SUB 7.3.1 (NO) "F"

if(cont4 == 1)
printf("F\0a");

//SUB 7.3.2 (NO) "E"

else
printf("E\0a");
}
}

//CLASSE 8 (2/1) "A,C,E,F,P,Q,V,Y,"

void classe8(int vetor3[])
{
int vv1 = 0, y, cont4 = 0, f1 = 0, f2 = 0, d1 = 0, d2 = 0, m, m1 = 0, m2 = 0, cont10 = 0, vv2 = 0, cont9 = 0, cont5 = 0, cont6 = 0, cont7 = 0, cont8 = 0;

vv1 = x1 + l*0.5;

for(y=0; y < depth; y++)
{
vetor3[y+1] = (getpixel(vv1,(y+1)) - getpixel(vv1,y));
if(vetor3[y+1] == -15)
cont4 = cont4 + 1;
}

//SUBCLASSE 8.1 "Y,V"

if(cont4 == 1) //distingue entre Y e V atraves de distancia entre a transicao e YT
{
for(y=1; y < depth; y++)
{
if(vetor3[y] == -15)
d1 = y;
}

for(y=1; y < depth; y++)
{
if(vetor3[y] == 15)
d2 = y;
}

//SUB 8.1.1 (NO) "V"

```



```

if((d2-d1)<=0.45*h) //grau de tolerancia de 45% de espessura p/ V
printf("V\ra");

```

```

//SUB 8.1.2 "V,Y"

```

```

else //distingue ambiguidades entre Y e V atraves da inclinacao inferior
{

```

```

for(m=1; m<(8*bytes); m++)
{
vetor3[m]=((getpixel((m+1),v5)) - (getpixel(m,v5)));
if(vetor3[m]==-15)
{
if(f1==0)
{
m1=m;
f1=1;
}
}
}

```

```

for(m=1; m<(8*bytes); m++)
{
vetor3[m]=((getpixel((m+1),v7)) - (getpixel(m,v7)));
if(vetor3[m]==-15)
{
if(f2==0)
{
m2=m;
f2=1;
}
}
}

```

```

//SUB 8.1.2.1 (NO) "Y"

```

```

if(abs(m2-m1)<0.1*h)
printf("Y\ra");

```

```

//SUB 8.1.2.2 (NO) "V"

```

```

else
printf("V\ra");
}
}

```

```

//SUBCLASSE 8.2 "A,C,F,P,Q,Y"

```

```

if(cont4==2) //distingue entre A,C,F,P,Q,Y
{

```

```

for(y=1; y<depth; y++)
{
if(vetor3[y]==15)
d1=abs(d1-y);
}

```

```

//SUB 8.2.1 (NO) "A"

```

```

if(d1>0.6*h && d1<0.8*h) //determina A
printf("A\ra");

```

```

// SUB 8.2.2 "C,Q,Y"

```

```

if(d1>=0.8*h) //determina C,Q ou Y
{
for(m=1; m<(8*bytes); m++)
{

```

```

vetor3[m] = ((getpixel((m + 1),v5)) - (getpixel(m,v5)));
}

for(m=1; m < (8*bytes); m++)
{
if(vetor3[m] == 15)
cont10 = cont10 + +;
}
//SUB 8.2.2.1 (NO) "Q"

if(cont10 == 2) // determinaQ
printf("Q");

//SUB 8.2.2.2 "C,Y"

else //distingue entre C e Y
{
vv2 = x1 + l*0.9;

cont4 = 0;
for(y=0; y < depth; y++)
{
vetor3[y+1] = (getpixel(vv2,(y+1)) - getpixel(vv2,y));
if(vetor3[y+1] == -15)
cont4 = cont4 + +;
}

//SUB 8.2.2.2.1 (NO) "C"

if(cont4 == 2)
printf("C\ta");

//SUB 8.2.2.2.2 (NO) "Y"
else
printf("Y\ta");
}

//SUB 8.2.3 "P,F"

if(d1 <= 0.6*h)
{
for(m=1; m < (8*bytes); m++)
{
vetor3[m] = ((getpixel((m + 1),v3)) - (getpixel(m,v3)));
}

for(m=1; m < (8*bytes); m++)
{
if(vetor3[m] == 15)
cont9 = cont9 + +;
}

//SUB 8.2.3.1 (NO) "P"

if(cont9 == 2)
printf("P\ta");

//SUB 8.2.3.2 (NO) "F"
else
printf("F\ta");
}

}

//SUBCLASSE 8.3 "E,F,P,Q"

if(cont4 == 3) //distingue entre E,F,P e Q
{

```

```

for(m = 1; m < (8*bytes); m++)
{
    vetor3[m] = ((getpixel((m+1),v4)) - (getpixel(m,v4)));
}

for(m = 1; m < (8*bytes); m++)
{
    if(vetor3[m] == 15)
        cont5 = cont5++;
}

//SUB 8.3.1 (NO) "Q"

if(cont5 > 2) //letra Q: determina possivel ambiguidade entre E F P e Q
    printf("Q\\a");

else
{
    //SUB 8.3.2 "E,F,P"

    if(cont5 == 1) //resolve ambiguidade entre E F,P
    {
        vv1 = x1 + l*0.9;

        for(y = 0; y < depth; y++)
        {
            vetor3[y+1] = (getpixel(vv1,(y+1)) - getpixel(vv1,y));
        }
        for(y = 1; y < depth; y++)
        {
            if(vetor3[y] == 15)
                cont6 = cont6++;
        }
        //SUB 8.3.2.1 "F,P"
        if(cont6 == 1) //distingue entre F e P
        {

            vv1 = x1 + l*0.7;

            for(y = 0; y < depth; y++)
            {
                vetor3[y+1] = (getpixel(vv1,(y+1)) - getpixel(vv1,y));
            }
            for(y = 1; y < depth; y++)
            {
                if(vetor3[y] == 15)
                    cont8 = cont8++;
            }

            //SUB 8.3.2.1.1 (NO) "P"

            if(cont8 == 2)
                printf("P\\a");

            //SUB 8.3.2.1.2 (NO) "F"

            else
                printf("F\\a");
        }

        //SUB 8.3.2.2 (NO) "E"

        else
            printf("E\\a");
    }

    //SUB 8.3.3 "F,Q"

```



```

if(cont5 == 2) // resolve ambiguidade entre F e Q
{
    vv1 = x1 + l * 0.9;

    for(y = 0; y < depth; y++)
    {
        vetor3[y + 1] = (getpixel(vv1, (y + 1)) - getpixel(vv1, y));
        if(vetor3[y + 1] == -15)
            cont7 = cont7++;
    }

    // SUB 8.3.3.1 (NO) "F"

    if(cont7 == 1)
        printf("F\n");

    // SUB 8.3.3.2 (NO) "Q"
    else
        printf("Q\n");
    }

} // c4 = 3

}

// CLASSE9 (1/2) "A,C,E,F,G,J,L,S,Z"

void classe9(int vetor1[], int vetor2[], int vetor3[])
{
    int vv1 = 0, y, cont4 = 0, d1 = 0, cont7 = 0, cont8 = 0, k, d, i, x3, x4;

    vv1 = x1 + l * 0.5;

    for(y = 0; y < depth; y++)
    {
        vetor3[y + 1] = (getpixel(vv1, (y + 1)) - getpixel(vv1, y));
    }
    for(y = 1; y < depth; y++)
    {
        if(vetor3[y] == -15)
            cont4 = cont4++;
    }

    // SUBCLASSE 9.1 "J,L"

    if(cont4 == 1) // distingue entre J e L através da dist entre x1V1 e x2V2
    {
        for(i = 1; i < (8 * bytes - 1); i++)
        {
            if(vetor1[i] == -15)
                x3 = i;
        }

        for(i = 1; i < (8 * bytes - 1); i++)
        {
            if(vetor2[i] == -15)
                x4 = i;
        }

        d = abs(x4 - x3);

        // SUB 9.1.1 (NO) "J"

        if(d < 0.2 * l)
            printf("J\n");
    }
}

```

```

//SUB 9.1.2 (NO) "L"

else
printf("L\a");

} //fim de c4 = 1

//SUBCLASSE 9.2 "A,C,G,J,L"

if(cont4 == 2) //distingue entre A,C,G,J e L
{
for(y = 1; y < depth; y++)
{
if(vetor3[y] == -15)
d1 = abs(d1 - y);
}

// SUB 9.2.1 "J,C,L,G"

if(d1 > 0.8*h) //distingue entre J,C,L ou G
{
vv1 = 0;
vv1 = x1 + l*0.8;

for(y = 0; y < depth; y++)
{
vetor3[y + 1] = (getpixel(vv1, (y + 1)) - getpixel(vv1, y));
if(vetor3[y + 1] == -15)
cont7 = cont7 + +;
}

//SUB 9.2.1.1 (NO) "G"
if(cont7 == 3)
printf("G\a"); //determina G

//SUB 9.2.1.2 "C,G"

if(cont7 == 2) //determina C ou G
{
for(k = 1; k < (8*bytes); k++)
{
vetor3[k] = (getpixel((k + 1), v3) - getpixel(k, v3));
if (vetor3[k] == 15)
cont8 = cont8 + +;
}

//SUB 9.2.1.2.1 (NO) "C"
if(cont8 == 1)
printf("C\a");

//SUB 9.2.1.2.2 (NO) "G"
else
printf("G\a");
}

//SUB 9.2.1.3 "J,L"

if(cont7 == 1) //distingue entre J ou L atraves da dist entre x1V1 e x2V2
{
for(i = 1; i < (8*bytes-1); i++)
{
if(vetor1[i] == -15)
x3 = i;
}

for(i = 1; i < (8*bytes-1); i++)

```

```

{
    if(vetor2[i] == -15)
        x4 = i;
    }

    d = abs(x4 - x3);

    //SUB 9.2.1.3.1 (NO) "J"
    if(d < 0.2 * l)
        printf("J\\a");

    //SUB 9.2.1.3.2 (NO) "L"
    else
        printf("L\\a");
}

}

//SUB 9.2.2 (NO) "A"

else //determina A

    printf("A\\a");

} //fim de c4 = 2

//SUBCLASSE 9.3 "E,G,S,Z"

if(cont4 == 3) //distingue entre E,G,S e Z atraves da distancia entre as trans
{
    for(k = 0; k < (8 * bytes); k++)
    {
        if(vetor1[k] == 15)
            x3 = k;
    }
    for(k = 0; k < (8 * bytes); k++)
    {
        if(vetor2[k] == 15)
            x4 = k;
    }
    d = abs(x4 - x3);

    //SUB 9.3.1 "E,G,S"

    if(d > 0.5 * l) //determina E,G ou S
    {
        vv1 = x1 + l * 0.3;
        cont7 = 0;
        for(y = 0; y < depth; y++)
        {
            vetor3[y + 1] = (getpixel(vv1, (y + 1)) - getpixel(vv1, y));
            if(vetor3[y + 1] == -15)
                cont7 = cont7 + 1;
        }

        //SUB 9.3.1.1 (NO) "G"

        if(cont7 == 2)
            printf("G\\a");

        //SUB 9.3.1.2 "E,S"

        else //distingue entre E ou S
        {
            cont8 = 0;
            for(k = 0; k < (8 * bytes); k++)

```



```

{
veto3[k]=(getpixel((k+1),v6)- getpixel(k,v6));
if (veto3[k] == 15)
cont8=cont8++;
}

//SUB 9.3.1.2.1 (NO) "S"

if(cont8 == 1)
printf("S\\a");

//SUB 9.3.1.2.2 (NO) "E"

else
printf("E\\a");
}
}

//SUB 9.3.2 (NO) "Z"

else
printf("Z\\a"); //determina Z

} // fim de c4=3
}

//CLASSE 10 (1/1) "A,C,E,F,I,J,L,S,T,Z"

void classe10 (int veto1[], int veto2[], int veto3[])
{
int vv1,y,cont4=0,d1=0,d2=0,cont5=0,cont6=0,cont3=0,m;
vv1=x1+l*0.5;
for(y=0; y<depth; y++)
{
veto3[y+1]=(getpixel(vv1,(y+1))-getpixel(vv1,y));
if(veto3[y+1] == 15)
cont4=cont4++;
}

//SUBCLASSE 10.1 "I,J,L,T"

if(cont4 == 1) //distingue entre I,J,L e T
{
vv1=x1+l*0.3;

for(y=0; y<depth; y++)
{
veto3[y+1]=(getpixel(vv1,(y+1))-getpixel(vv1,y));
if(veto3[y+1] == 15)
cont3=cont3++;
}

//SUB 10.1.1

if(cont3 == 2)//distingue entre I,J e T
{
for(m=1; m<(8*bytes); m++)
{
veto3[m]=((getpixel((m+1),v8)) - (getpixel(m,v8)));
if(veto3[m] == 15)
cont5=cont5++;
}

//SUB 10.1.1.1 (NO)

if(cont5 == 3)
printf("T\\a");

//SUB 10.1.1.2 "J,I"

```

```

else
{
cont5=0;
for(m=1; m<(8*bytes); m++)
{
vetor3[m]=((getpixel((m+1),v7)) - (getpixel(m,v7)));
if(vetor3[m]==15)
cont5=cont5++;
}

// SUB 10.1.1.2.1 (NO) "J"

if(cont5==2)
printf("J\\a");

// SUB 10.1.1.2.2 (NO) "I"
else
printf("I\\a");
}
}

//SUB 10.1.2 "I,J,L,T"

if(cont3==1) //distingue entre I,J,L,T
{
for(y=1; y<(depth-1); y++)
{
if(vetor3[y]==15)
d1=y;
}

// SUB 10.1.2.1 (NO) "T"

if(abs(YT-d1)<0.2*h)
printf("T\\a");

//SUB 10.1.2.2 "I,J,L"

else//distingue entre I,J,L
{
cont5=0;
for(m=1; m<(8*bytes); m++)
{
vetor3[m]=((getpixel((m+1),v7)) - (getpixel(m,v7)));
if(vetor3[m]==15)
cont5=cont5++;
}

// SUB 10.1.2.2.1 (NO) "J"
if(cont5==2) // distingue J de I e L
printf("J\\a");

// SUB 10.1.2.2.2 "I,L"

else //distingue entre I e L
{
d1=0;
cont5=0;
vv1=x1+l*0.9;
for(y=0; y<(depth-1); y++)
{
vetor3[y+1]=(getpixel(vv1,(y+1))-getpixel(vv1,y));
if(vetor3[y+1]==-15)
cont5=cont5++;
}

// SUB 10.1.2.2.2.1 (NO) "I"

```



```
//SUBCLASSE 10.3 "E,F,S,Z"
```

```
if(cont4 == 3)//distingue entre E,F,S e Z  
{
```

```
for(m = 1; m < (8*bytes); m++)  
{  
if(vetor1[m] == 15)  
d1 = m;  
}
```

```
for(m = 1; m < (8*bytes); m++)  
{  
if(vetor2[m] == 15)  
d2 = m;  
}
```

```
//SUB 10.3.1 "E,F"
```

```
if(abs(d2-d1) < 0.2*1)// distingue entre E e F  
{  
vv1 = x1 + 1*0.7;
```

```
for(y = 0; y < depth; y++)  
{  
vetor3[y+1] = (getpixel(vv1,(y+1))-getpixel(vv1,y));  
if(vetor3[y+1] == 15)  
cont6 = cont6 + +;  
}
```

```
//SUB 10.3.1.1 (NO) "E"  
if(cont6 == 3)  
printf("E\n");
```

```
//SUB 10.3.1.2 (NO) "F"
```

```
else  
printf("F\n");  
}
```

```
//SUB 10.3.2 "S,Z"
```

```
else //distingue entre S e Z  
{  
//SUB 10.3.2.1 (NO) "S"
```

```
if(d2 > d1)  
printf("S\n");
```

```
//SUB 10.3.2.2 (NO) "Z"
```

```
else  
printf("Z\n");  
}
```

```
//c4 = 3
```

```
}
```

```
//CLASSE 11 (2/2) "A,B,C,D,E,G,H,K,M,N,O,Q,R,S,U,V,W,X,Z"
```

```
void classe11 (int vetor1[], int vetor2[], int vetor3[], int vetor4[])  
{
```

```
int vv1,y,y1,y3=0,y4,cont3=0,d1=0,m2,m1,m3,d,f1=0,f2=0,cont4=0,m,cont5=0, cont6=0,cont7=0;  
vv1 = x1 + 1*0.5;
```

```

for(y=0; y<(depth); y++) //varredura vertical em 50% de l
{
    vetor3[y] = (getpixel(vv1,(y+1))-getpixel(vv1,y));
    if(vetor3[y] == 15)
    {
        cont3 = cont3 + +;
        y4 = y;
    }
}

//SUBCLASSE 11.1 "H,K,V,U,X,N,M,W"

if(cont3 == 1) //distingue entre H,K,V,U,X,N,M,W por possuirem uma trans em 50% de l
{
    for(m=1; m<(8*bytes-1); m++) //calcula num de trans em 50% de h
    {
        vetor3[m] = ((getpixel((m+1),v5)) - (getpixel(m,v5)));
        if(vetor3[m] == -15)
        cont5 = cont5 + +;
    }

    //SUB 11.1.1 (NO') "N"

    if(cont5 == 3) //determina N por possuir tres trans em 50% de h
    printf("N\n"); //OBS: dificil resolver ambiguidade entre M e W nesta classe
    // devido ao numero de possibilidades que continuam ocorrendo!!!

    //SUB 11.1.2 ("NO'") "W OU M"

    if(cont5 == 4) //determina possivel M e W por possuirem 4 trans em 50% de h
    printf("W ou M"); //

    //SUB 11.1.3 "U,V,H,K"

    if(cont5 == 2) //determina U,V,H,K por possuirem 2 transicoes em 50% de h
    {

        // SUB 11.1.3.1 "H,K"

        if(abs(YT-y4) <= 0.5*h) //determina ,H e K quando trans ocorre abaixo de 50% de h
        {

            for(m=1; m<(8*bytes-1); m++) //calcula num de trans em 50% de h
            {
                vetor4[m] = ((getpixel((m+1),v3)) - (getpixel(m,v3)));
                if(vetor4[m] == -15)
                cont4 = cont4 + +;
            }

            // SUB 11.1.3.1.2 "H,K"
            else //distingue H de K em funcao da inclinacao de sua reta a direita
            {
                m1 = 0;
                m2 = 0;
                f1 = 0;
                f2 = 0;
                for(m=1; m<(8*bytes-1); m++)
                {
                    if(vetor3[m] == 15)
                    {
                        if(f1 <= 1)
                        {
                            m1 = m;
                            f1 = f1 + +;
                        }
                    }
                }

                for(m=1; m<(8*bytes-1); m++)

```

```

{
if(vetor2[m] == 15)
{
if(f2 <= 1)
{
m2 = m;
f2 = f2 + +;
}
}
}

// SUB 11.1.3.1.2.1 (NO') "H"

if(abs(m2-m1)<0.1*I)//distingue H de K em funcao da inclinacao a direita em 50% e 80% de h
printf("H\la");

// SUB 11.1.3.1.2.2 (NO') "K"
else
printf("K\la");
} //c4

} //c5

//SUB 11.1.3.2 "K,H"

else

h if((abs(YT-y4)<0.8*h) && (abs(YT-y4)>0.5*h)) //determina possivel K,H quando trans ocorre entre 50% e 80% de
{
m1=0;
m2=0;
f1=0;
f2=0;
for(m=1; m<(8*bytes-1); m++)
{
if(vetor3[m] == 15)
{
if(f1 <= 1)
{
m1 = m;
f1 = f1 + +;
}
}
}

for(m=1; m<(8*bytes-1); m++)
{
if(vetor2[m] == 15)
{
if(f2 <= 1)
{
m2 = m;
f2 = f2 + +;
}
}
}

//SUB 11.1.3.2.1 (NO') "H"

if(abs(m2-m1)<0.1*I)//distingue H de K em funcao da inclinacao a direita em 50% e 80% de h
printf("H\la");

// SUB 11.1.3.2.2 (NO') "K"
else
printf("K\la");
}

//SUB 11.1.3.3 "U,V"

```



```

else
if(abs(YT-y4)>=0.8*h) //determina U e V quando a transicao pb ocorre acima de 80% de h
{

m1=0;
m2=0;
f1=0;
f2=0;
for(m=1; m<(8*bytes-1); m++)
{
if(vetor1[m]==-15)
{
if(f1==0)
{
m1=m;
f1=1;
}
}
}

for(m=1; m<(8*bytes-1); m++)
{
if(vetor2[m]==-15)
{
if(f2==0)
{
m2=m;
f2=1;
}
}
}

//SUB 11.1.3.3.1 (NO') "U"

if(abs(m1-m2)<=0.1*I) //distingue entre U e V atraves da inclinacao
printf("U\");

//SUB 11.1.3.3.2 (NO') "V"
else
printf("V\");
} //>0.8*h
} //c5

// SUB 11.1.4 "H,K,X"

if(cont5==1)//determina H,K e X
{
m1=0;
m2=0;
f1=0;
f2=0;
for(m=1; m<(8*bytes-1); m++)
{
if(vetor3[m]==-15)
{
if(f1==0)
{
m1=m;
f1=1;
}
}
}

for(m=1; m<(8*bytes-1); m++)
{
if(vetor2[m]==-15)
{
if(f2==0)
{

```

```

m2=m;
f2=1;
}
}
}

// SUB 11.1.4.1 "H,K"

if(abs(m2-m1)<0.1*l)//distingue (H e K) de X
{
m1=0;
m2=0;
f1=0;
f2=0;
for(m=1; m<(8*bytes-1); m++)
{
if(vetor3[m]==15)
{
if(f1<=1)
{
m1=m;
f1=f1++;
}
}
}

for(m=1; m<(8*bytes-1); m++)
{
if(vetor2[m]==15)
{
if(f2<=1)
{
m2=m;
f2=f2++;
}
}
}

// SUB 11.1.4.1.1 (NO') "H"
if(abs(m2-m1)<0.1*l)//distingue H de K em funcao da inclinacao a direita
printf("H\n");

// SUB 11.1.4.1.2 (NO') "K"
else
printf("K\n");
}

// SUB 11.1.4.2 (NO') "X"
else //distingue X de H e K em funcao da inclinacao de sua reta a esquerda
printf("X\n");

} //c5
} //c3

//SUBCLASSE 11.2 "A,C,D,G,K,O,Q,R,X"

if(cont3==2) //distingue entre A,C,D,G,K,O,Q,R,X
{
for(y=1; y<depth; y++)
{
if(vetor3[y]==15)
d1={abs(d1-y)};
}
// SUB 11.2.1 "C,D,G,O,Q"

if(d1>=0.7*h) //distingue entre C,D,G,O e Q
{
vv1=x1+l*0.65;
for(y=0; y<(depth); y++)

```

```

{
veto3[y+1]=(getpixel(vv1,(y+1))-getpixel(vv1,y));
if(veto3[y+1] == 15)
cont4=cont4++;
}
// SUB 11.2.1.1 (NO') "G"
if(cont4 == 3)//determina G
printf("G\\a");

// SUB 11.2.1.2 "C,D,O,Q"

else//distingue entre C,D,O,Q
{
for(m=1; m<(8*bytes-1); m++)
{
veto3[m]=((getpixel((m+1),v5)) - (getpixel(m,v5)));
if(veto3[m] == -15)
cont5=cont5++;
}

//SUB 11.2.1.2.1 (NO') "C"

if(cont5 == 1)
printf("C\\a");

//SUB 11.2.1.2.2 "D,O,Q"

else//distingue entre D,O e Q
{
// SUB 11.2.1.2.2.1 (NO') "D"
if(abs(XT-x1)<0.25*I) //determina D atraves da inclinacao do XT em rela a x1
printf("D\\a");

//SUB 11.2.1.2.2.2 "O,Q"
else
{
vv1=x1+I*0.95;
for(y=0; y<(depth-1); y++)
{
veto3[y+1]=(getpixel(vv1,(y+1))-getpixel(vv1,y));
if(veto3[y+1] == -15)
y1=y;
}

//SUB 11.2.1.2.2.2.1 (NO') "Q"
if(abs(YB-y1)<=(0.15*h))
printf("Q\\a");
//SUB 11.2.1.2.2.2.2 (NO') "O"
else
printf("O\\a");
}
}
}

//SUB 11.2.2 "A,K,R,X"
else //distingue entre A,K,R,X
{

m1=0;
m2=0;
f1=0;
f2=0;
for(m=1; m<(8*bytes-1); m++)
{
if(veto1[m] == -15)
{
if(f1 == 0)
{

```



```

m1 = m;
f1 = 1;
}
}
}

for(m = 1; m < (8*bytes-1); m + +)
{
if(vetor2[m] == -15)
{
if(f2 == 0)
{
m2 = m;
f2 = 1;
}
}
}

//SUB 11.2.2.1 "K,R"

if(abs(m2-m1) <= 0.1*I) //determina K ou R em funcao da inclinacao
{
f1 = 0;
d = 0;
vv1 = x1 + I*0.5;
for(y = 1; y < (depth-1); y + +)
{
vetor3[y] = (getpixel(vv1, (y + 1)) - getpixel(vv1, y));
if(vetor3[y] == 15)
{
if(f1 == 0)
{
d = y;
f1 = 1;
}
}
}

//SUB 11.2.2.1.1 (NO) "R"

if(abs(d-YT) < (0.25*h))
printf("R\la");

//SUB 11.2.2.1.2 (NO) "K"

else
printf("K\la");
}

//SUB 11.2.2.2 "X,A"
else //distingue X e A de K e R em funcao da inclinacao
{
f1 = 0;
for(m = 1; m < (8*bytes-1); m + +)
{
vetor3[m] = ((getpixel((m + 1), v5)) - (getpixel(m, v5)));
if(vetor3[m] == -15)
{
if(f1 == 0)
{
m3 = m;
f1 = 1;
}
}
}

//SUB 11.2.2.2.1 (NO) "X"
if(m1 < m3) //distingue X de A em funcao da inclinacao de sua reta esquerda superior
printf("X\la");
//SUB 11.2.2.2.2 (NO) "A"
else //determina A

```

```

printf("A\la");
}
}
} //c3

//SUBCLASSE 11.3 "B,E,G,Q,S,Z"

if(cont3 == 3) //distingue entre B,E,G,Q,S,Z
{
f2 = 0;
m1 = 0;
for(y = 1; y < depth; y++)
{
if(vetor3[y] == -15)
{
if(f2 <= 1)
{
y3 = abs(y3 - y);
f2 = f2++;
}
}
}
for(m = 0; m < (8*bytes-1); m++)
{
vetor3[m] = ((getpixel((m+1),v3)) - (getpixel(m,v3)));
if(vetor3[m] == -15)
{
cont6 = cont6++;
m1 = abs(m1-m);
}
}

//SUB 11.3.1 "E,Q,B"

if(cont6 == 2) //determina E,Q e B
{

//SUB 11.3.1.1 (NO') "E"

if(m1 <= 0.45*i) //distingue E entre Q e B em funcao da dist entre trans em 40% de h
printf("E\la");

//SUB 11.3.1.2 "Q,B"

else //difere entre Q ou B em funcao da posicao da trans em 50% da h
{

//SUB 11.3.1.2.1 (NO') "B"

if(y3 < 0.5*h)
printf("B\la");

//SUB 11.3.1.2.2 (NO') "Q"
else
printf("Q\la");

}

} //c6
//SUB 11.3.2 "G,S,Z"
else // distingue entre G,S e Z
{
m2 = 0;
for(m = 0; m < (8*bytes-1); m++)
{
vetor3[m] = ((getpixel((m+1),v6)) - (getpixel(m,v6)));
if(vetor3[m] == -15)
{
cont7 = cont7++;
m2 = abs(m2-m);
}
}
}

```

```

    }
    }

//SUB 11.3.2.1 (NO') "G"

if(abs(m2-m1)<0.1*I)
printf("G\a"); //determina G em funcao da dist entre      trans em linhas simetricas

// SUB 11.3.2.2 "S,Z"

else //distingue entre Z e S em funcao da inclinacao      (cresc ou decres) de sua linha constituinte
{

//SUB 11.3.2.2.1(NO') "S"

if(m2>m1)
printf("S\a");

//SUB 11.3.2.2.2 (NO') "Z"

else
printf("Z\a");
}
} //c6

} //c3

} //11

```