Binarizacao Por Otsu e Outras técnicas Usadas na Detecção de Placas

(by Leonardo Hiss Monteiro)

1.1 Introdução

Para reconhecer a placa de automóvel é necessário submetê-la primeiramente a um pré–processamento. Nesse capítulo vamos apresentar todos os métodos que foram utilizados nessa fase de pré–processamento, e que é composta pelas seguintes técnicas: Binarização ou limiarização; Erosão; Segmentação de objetos conectados; e esqueletização ou "thinning".

A primeira fase de processamento é composta pelas técnicas de momento invariantes e a segunda fase por técnicas de detecção das extremidades e cavidades dos caracteres. A figura 1 apresenta as fases do processamento.

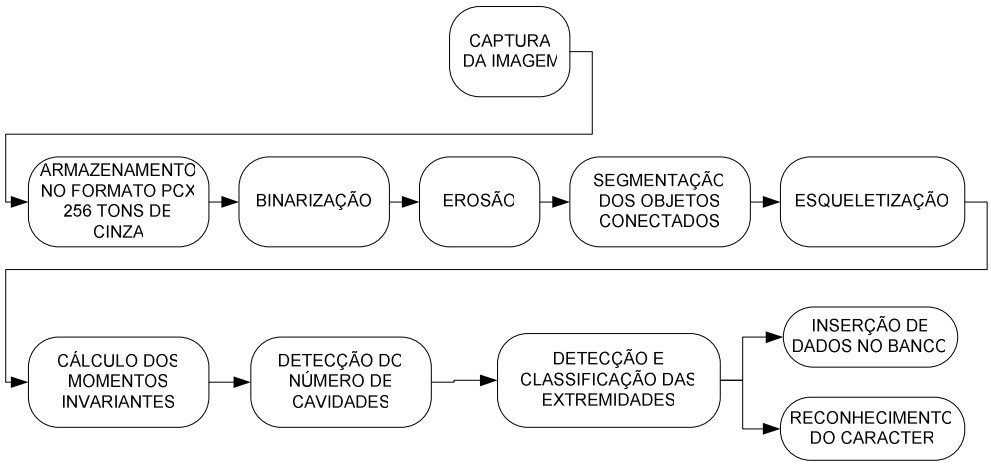

Figura 1.1 – Fases do processamento

1.2 Binarização

Para separar os objetos que desejamos analisar da imagem inicial, utilizamos técnicas de binarização ou limiarização. A binarização é o método mais simples de segmentação de imagens. Resumidamente consiste em separar uma imagem (figura 1.1), em regiões de interesse e não interesse através da escolha de um ponto de

corte [12]. Essas regiões podem ser representadas por *pixels* pretos e brancos. Os métodos mais simples de limiarização utilizam um único ponto de corte também conhecido por *threshold*. Em alguns casos, no entanto, não se consegue apenas um limiar que resulte em uma boa segmentação para toda a imagem. Para esses casos existem técnicas de limiarização variáveis e multiníveis baseadas em várias medidas estatísticas [17]. No programa implementado estamos fazendo uso de uma técnica de limiarização global simples, ou seja, utilizando apenas um ponto de threshold já que o conteúdo das imagens processadas nesse trabalho (figura 1.2) são basicamente bimodais.

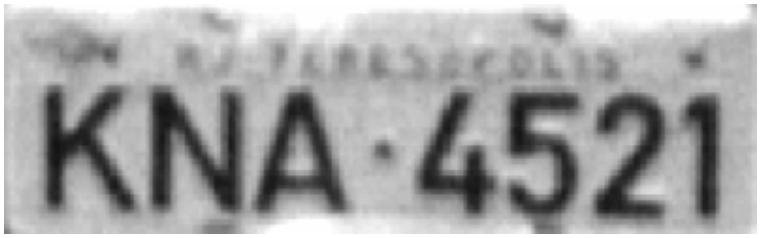

Figura 1.2 – Placa utilizada na binarização

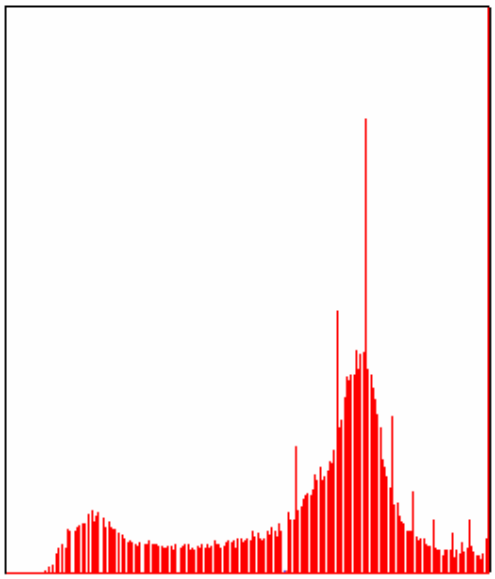

Figura 1.3 – Histograma da placa

Observamos então que decidir se o *pixel* pertence ou não a região de interesse depende do ponto de corte. A representação de área de interesse e área de não interesse pode ser feita utilizando duas cores distintas. No caso do programa utilizamos preto e branco. O valor da nova cor que terá o *pixel* é realizada de acordo com o ponto de corte ou *threshold.* Qualquer *pixel* com intensidade menor ou igual ao ponto de corte passa a ser preto. Se o *pixel* tiver intensidade maior que o ponto de corte passa a ter a cor branca. Como sabemos que uma imagem digitalizada pode ser escrita como uma função f(x,y) (figura 1.3), a função binarização pode ser escrita como:

$$g(x,y) = \begin{cases} R_1 \ se \ f(x,y,) \leq T \\ R_2 \ se \ f(x,y) > T \end{cases}$$

Onde R_1 e R_2 são os valores estipulados para os níveis de cinza da imagem binarizada, no caso utiliza-se 0 (preto) e 255 (branco) (figura 1.4).

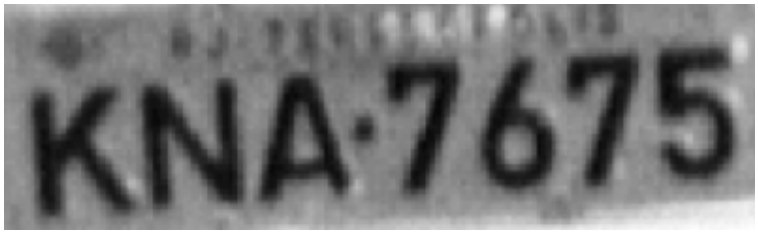

Figura 1.4 – Placa antes da binarização

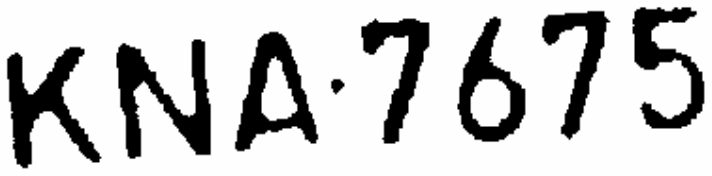

Figura 1.5 – Placa depois da binarização

Alguns autores se referem ao método de binarização como um método para separar os objetos do fundo (*background*) da imagem (*foregroud*). Por exemplo na figura 1.4 observemos a imagem da placa de um veículo que é composto

por vários objetos (caracteres). O *threshold* nesse caso (figura 1.5) deve ser escolhido com o cuidado para que não capture imagens que não façam parte do objeto de interesse. O *threshold* escolhido nesse caso foi 50. Observamos que sendo a imagem bimodal, qualquer valor entre os dois picos conseguirá separar os objetos do fundo branco.

Concluímos que é fundamental a escolha adequada do ponto de corte T. Desse modo se torna fundamental utilizar o histograma da imagem. A observação do histograma permite a localização do melhor valor de T para que seja separados os objetos do fundo. Esta localização é tão mais fácil quanto mais bimodal for o histograma. Por exemplo, se a imagem for composta por um objeto e um fundo podemos estabelecer o ponto de corte através de um isolamento das regiões que representem o fundo e o objeto. Para imagem compostas por mais de um objeto de intensidade diferente sob um fundo, podemos utilizar uma técnica de binarização multi-nível (*multilevel thresholding*). Quando não é possível estabelecer as regiões de cada objeto no histograma se torna mais difícil encontrar o ponto de corte ideal. Atualmente existem métodos automáticos para a escolha e ajuste do ponto de corte que usam o histograma da imagem ou a imagem propriamente dita como referência. Métodos que comparam conhecimentos prévios sobre parâmetros da imagem e obtêm vários pontos de corte são também conhecidos [17].

No programa implementado utilizamos o método de limiarização bimodal de Otsu. Esse método de limiarização baseia-se na análise discriminante. A operação de limiarização é considerada como sendo o particionamento dos *pixels* de uma imagem com l níveis de cinza em duas classes C_0 e C_1, que representam o objeto e o fundo, ou vice-versa, sendo que esta partição se dará no nível de cinza *t*. Desta forma teremos:

$C_0 = \{0,1,2,3,...,t\}$

e

$C_1 = \{t+1, t+2, t+3,...,l\}$

Seja:

σ_w^2 variância dentro da classe

σ_b^2 variância entre as classes

σ_t^2 variância total

Um limiar ótimo pode ser obtido através da maximização de umas das seguintes funções critérios:

$$\lambda = \frac{\sigma_b^2}{\sigma_w^2} \tag{1.1}$$

$$\eta = \frac{\sigma_b^2}{\sigma_t^2} \tag{1.2}$$

$$\mathrm{K} = \frac{\sigma_t^2}{\sigma_w^2} \tag{1.3}$$

Das três funções critérios acima, η é a mais simples, logo, o limiar ótimo pode ser obtido encontrando o valor de t que maximiza a função η .

$$\sigma_t^2 = \sum_{i=0}^{l-1} (i - \mu t)^2 P_i \tag{1.4}$$

$$\mu t = \sum_{i=0}^{l-1} i P_i \tag{1.5}$$

$$\sigma_b^2 = \omega_0 \omega_1 (\mu_0 - \mu_1)^2 \tag{1.6}$$

$$\omega_0 = \sum_{i=0}^{t} P_i \tag{1.7}$$

$$\omega_1 = 1 - \omega_0 \tag{1.8}$$

$$\mu_0 = \frac{\mu_t}{\omega_0} \tag{1.9}$$

$$\mu_1 = \frac{\mu t - \mu_t}{1 - \omega_0} \tag{1.10}$$

$$\mu_t = \sum_{i=0}^{t} iP_i \tag{1.11}$$

$$P_i = \frac{n_i}{n} \tag{1.12}$$

Onde n_i é o número de pixels como o nível de cinza *i* e *n* é o numero total de pixels de uma dada imagem definida como:

$$n = \sum_{i=0}^{l-1} n_i \tag{1.13}$$

Além disso, P_i é a probabilidade de ocorrência do nível de cinza *i*.

O método de Otsu como proposto disponibiliza meios para se analisar outros aspectos além da seleção de um limiar ótimo para uma dada imagem. Para a seleção do limiar *t* de uma dada imagem, a classe de probabilidades ω_0 e ω_1 indicam as porções das áreas ocupadas pelas classes C_0 e C_1. As médias de classes μ_0 e μ_1 servem como estimativa dos níveis médios das classes na imagem original em níveis de cinza. Além disso, o valor máximo de η, denotado por $\eta_{\max}$, pode ser utilizado como medida de separabilidade das classes C_0 e C_1 na imagem original ou na bimodalidade do histograma. Esta é uma medida bastante significante pois é invariante para transformações afins da escala de níveis de cinza e unicamente determinada dentro do intervalo:

$$0 \le \eta \le 1$$

O limite inferior é obtido quando e somente quando uma dada imagem tenha um único e constante nível de cinza, e o limite superior é obtido quando e somente quando a imagem apresenta apenas dois valores.

Em imagens digitais, a uniformidade dos objetos tem papel importante na separação destes objetos do fundo. De fato, a abordagem de Otsu para a limiarização de imagens em nível de cinza é eficiente e tem como base a medida entre as duas classes C_0 e C_1 a serem segmentadas.

O método de Otsu se caracteriza por sua natureza não paramétrica e não supervisionada de seleção de limiar e tem as seguintes vantagens desejáveis: (1) o processo como um todo é muito simples; (2) são utilizados somente os momentos cumulativos zero e de primeira ordem do histograma de níveis de cinza; e (3) viabiliza a análise de outros aspectos importantes, tais como estimativa dos níveis médios das classes e, separabilidade das classes.

Uma extensão direta para problemas que exijam vários limiares é viabilizada face ao critério no qual o método está baseado, qual seja, a análise de discriminantes. Um limiar ótimo (ou conjunto de limiares) é selecionado de forma automática e estável, não baseado na diferenciação (uma propriedade local como um vale) mas sim na integração (propriedade global) do histograma.

É importante ressaltar que o limiar calculado pelo método proposto não é indicado para imagens com as variâncias dos níveis de cinza de objetos e fundo ou populações de *pixels* correspondentes sejam extremamente diferentes.

1.3 Erosão

Após a binarização da imagem alguns caracteres apresentaram alguns ruídos na imagem que atrapalhavam no processamento seguinte. Esses ruídos na maioria dos casos era causado pelo fato dos caracteres nas placas do veículos se

apresentarem em alto relevo. Esse fato gerava um fino contorno em volta dos caracteres nas placas de veículos, como é mostrado na figura 1.6.

Figura 1.6 – Contorno dos caracteres

Percebemos então a necessidade de utilizar alguma técnica de processamento de imagem para remover esses ruídos. A solução adotada para resolver esse problema foi um processo conhecido como erosão [7]. Esse processamento faz com que uma imagem sofra diminuição em seu tamanho original sem que perca ou altere as suas características geométricas que são de suma importância para as demais fases de reconhecimento.

No processo de erosão primeiramente escolhemos o elemento que vai ser utilizado para realizar a erosão da imagem. A escolha do elemento que vai fazer a erosão é muito importante pois esse elemento é que vai determinar o quanto a imagem vai reduzir de tamanho cada vez que é submetida a uma iteração no processo de erosão. Uma má escolha do elemento utilizado para realizar a erosão da imagem pode acarretar na perda ou alteração das características geométricas da imagem. Isso comprometeria as demais fases do reconhecimento do caracter. No programa que foi implementado foi escolhido um elemento simples que garanta que a imagem original não perca suas características. A figura 1.7 representa o elemento que foi utilizado.

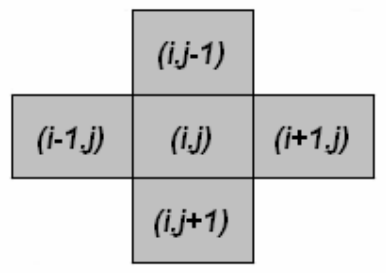

Figura 1.7 – Elemento escolhido para erosão

Depois da escolha do elemento a ser utilizado devemos analisar a vizinhança de cada um dos *pixels* da imagem em relação ao elemento. Essa análise é feita da seguinte forma:

1. Coloca-se o elemento escolhido para fazer a erosão na coordenada (i,j)
2. Verifica-se se os vizinhos do elemento são *pixels* do objeto (com valor 1)
3. Se todos os vizinhos possuírem valor 1 mantêm-se o *pixel* central com valor 1.
4. Se algum dos pixels vizinho não possuir valor 1, muda-se o valor do pixel central para 0 (pixel de fundo).

Apresentamos na figura 1.8 uma imagem antes e depois de duas iterações do algoritmo de erosão.

Figura 1.8 – Imagem antes e depois da erosão

Observando as imagens da figura 1.8 antes e depois da erosão verificamos que os finos contornos gerados pelo altos relevos dos caracteres foram removidos. Os outros elementos de espessura pequena também desapareceram. Porém as características geométricas da imagem não foram alteradas. Como os caracteres reduzem um pouco de tamanho após o procedimento de erosão, utilizamos sempre o mesmo número de erosões na análise de todas as placas de veículos. Verificamos que duas iterações são suficientes para remover os finos contornos apresentados nos caracteres. É importante ressaltar que o número de iterações não pode ser muito grande para não haver alteração das características dos caracteres que desejamos analizar.

De uma maneira geral o processo de erosão além de eliminar esse contorno indesejável nos caracteres também apresenta outra característica interessante. O processo faz uma "limpeza" na imagem original eliminando pequenos objetos que poderiam interferir no processo de segmentação dos elementos conectados.

1.4 Segmentação de Objetos Conectados

Depois da etapa de separação de objetos do fundo se torna necessário identificar e separar cada caracter para que possa ser analisado separadamente. Desenvolvemos então uma técnica de separação dos objetos que estão conectados. No caso de placas de veículos necessitamos analisar os 7 caracteres relevantes independentemente. Observamos dessa forma a necessidade de separar essas imagens por meio de um processo de segmentação.

O processo de segmentação desenvolvido utiliza a imagem da placa do veículo já binarizada e realiza a segmentação e contagem do número de objetos conectados. Para otimizar o processo de segmentação foi desenvolvido um único

algoritmo que faz a contagem e a segmentação das imagens. O algoritmo funciona da seguinte forma: Utilizamos o algoritmo de reconhecimento dos objetos conectados sendo que toda a vez que for terminada a segmentação de um objeto será contabilizado o número de objetos conectados, ou seja, incrementando em uma variável o número de objetos.

O algoritmo de segmentação funciona da seguinte forma:

1- Num primeiro momento é feita uma varredura na imagem buscando o primeiro *pixel* do objeto (*pixel* com valor 1).

2- O valor desse *pixel* é alterado para o valor de um índice I.

3- O valor desse índice I é incrementado (I=I+1).

4- É feita uma varredura nos *pixels* vizinhos a esse, de modo, que toda a vez que um *pixel* vizinho é encontrado o seu valor é alterado para o valor do índice e o índice I é incrementado.

5- Esse processo se repete até que não se encontrem mais *pixels* vizinhos. Quando isso ocorre o valor do ultimo índice é armazenado em um vetor e volta-se ao passo 1 enquanto houver *pixel* não analisado na imagem.

A figura 1.9 representa o funcionamento do algoritmo de segmentação de objetos conectados. A imagem após a contagem se apresenta como mostrado na figura 1.8.

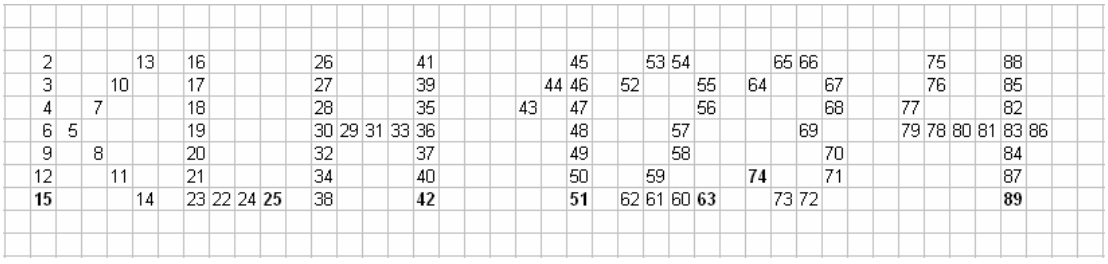

Figura 1.9 – Placa após o processamento do algoritmo de contagem

Com a imagem dessa maneira e com o vetor de armazenamento dos últimos *pixels* de cada elemento se torna possível a segmentação. Para o caso apresentado na figura 1.8 o vetor possuirá os seguintes valores :

MAT [15,25,42,51,63,74,89]

Então para separar o primeiro elemento vamos considerar apenas os *pixels* com os valores entre 2 e 14, já para separar o segundo vamos considerar os *pixels* com valor entre 15 e 24 e assim por diante. O restante dos *pixels* vamos atribuir o valor zero. Assim conseguiremos sete imagens oriundas da primeira, sendo que cada uma delas representa um dos objetos conectados da imagem original, ou seja, os caracteres conforme apresentado na figura 1.10.

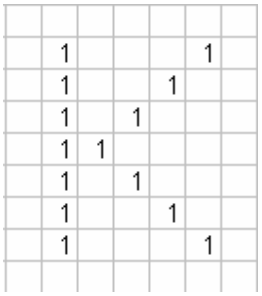

Figura 1.10 – Caracter segmentado

Figura 1.11

Finalmente conseguimos que os objetos sejam devidamente segmentados e separados e imagens distintas. Dessa forma conseguimos obter imagens em condições de serem submetidas ao processo posterior de reconhecimento de imagens. Cada imagem vai ser submetida a última etapa do processamento deve ser composta apenas pelo objeto e pelo fundo.

1.5 Eliminação de Ruídos Conectados aos Caracteres

Mesmo depois dos filtros utilizados para eliminar os elementos que não faziam parte dos caracteres, observamos que alguns caracteres após a binarização eram constituídos pelo caracter mais o ruído. Esse problema acontecia geralmente no segundo e no sexto caracter. Acontecia em alguns casos a união do furo para a fixação da placa do veículo com o caracter. A figura 1.11 apresenta uma placa com esse problema.

Figura 1.12 – União do caracter com o furo de fixação

Para eliminar esse problema utilizamos um filtro que elimina todos os ruídos acima e abaixo dos caracteres. Constatamos que esse problema de união do caracter com o furo de fixação nunca acontecia no primeiro e no último caracter da placa. Os caracteres das placas apresentam também um alinhamento superior e inferior. Com essas características foi desenvolvido um processo que calcula duas retas que tangenciam os caracteres na parte superior e inferior. Essa reta é traçada através do *pixel* de coordenada mais alta do primeiro caracter até o *pixel* de coordenada mais alta do último caracter. A segunda reta é traçada do *pixel* de coordenada mais baixa do primeiro caracter até o *pixel* de coordenada mais baixa do último caracter. Com as retas calculadas são eliminados todos os pixels que se encontram na parte superior da primeira reta e inferior da segunda reta (figura 1.12).

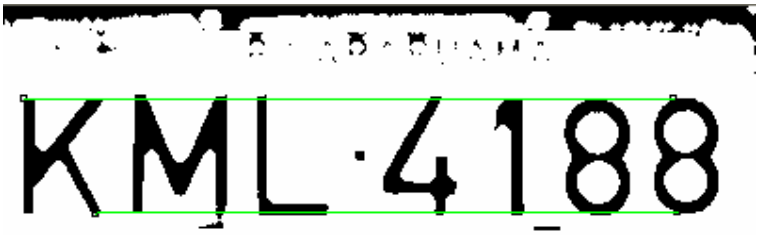

Figura 1.13 – Retas superior e inferior

Figura 1.14 – Caracteres após o filtro superior e inferior

Após o processamento, a imagem se encontra sem a união do segundo caracter com o furo de fixação inferior da placa do veículo. Observamos também que a utilização do filtro não remove nenhuma característica relevante dos caracteres (figura 1.13).

1.6 Filtragem dos Elementos Relevantes

Após a limiarização dos objetos, observamos que alguns que não são partes dos caracteres são detectados. Dessa forma o processo de filtragem deve eliminar esses objetos que não fazem parte dos caracteres.

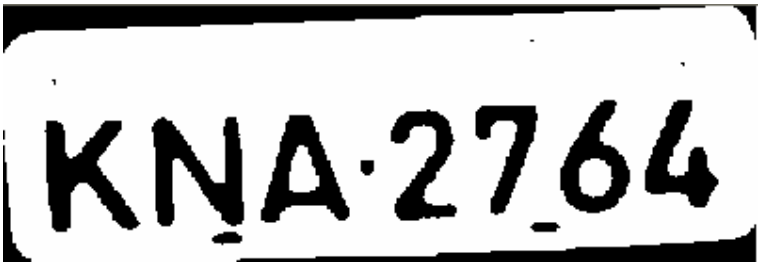

Figura 1.15 – Placa binarizada antes da filtragem

Na figura 1.14 observamos que vários objetos que não fazem parte dos caracteres foram reconhecidos. Para remover esses objetos utilizamos alguns critérios para considerar ou não os objetos como relevantes. Esses critérios são o seguinte:

- Remoção dos objetos que estão situados na parte superior e inferior da imagem, de acordo com o limite.
- Remoção dos objetos que estão na extremidade direita ou esquerda da imagem, de acordo com o limite.
- Remoção dos objetos muito pequenos
- Remoção dos objetos que contem dimensões horizontais muito grandes.

Para a remoção dos objetos que estão situados na parte superior e inferior da imagem utilizamos o seguinte algoritmo (figura 1.15).

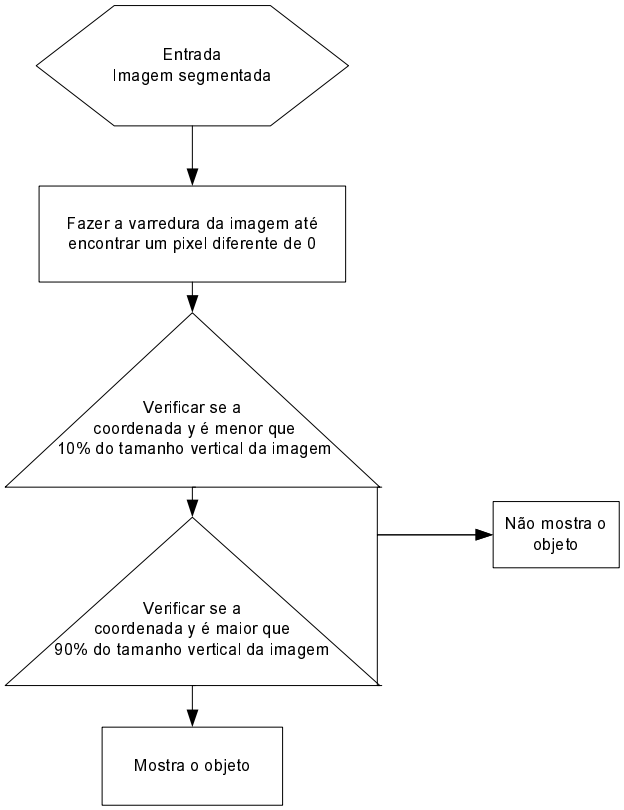

Figura 1.16 – Algoritmo para remoção de objetos na parte superior e inferior

Para a remoção dos objetos que se encontram na extremidade direita e esquerda utilizamos um algoritmo semelhante ao anterior apenas verificando se a coordenada x é maior que 2,5% da coordenada vertical e menor que 97,5%.

Para a remoção de objetos muito pequenos podemos utilizar o vetor que contem o valor do último *pixel* encontrado em cada objeto. Desse modo encontramos o valor da área de cada objeto através da seguinte expressão:

$$A_i = MAT_i - MAT_{i+1}$$

onde MAT_{i} é o valor que o pixel divisor dos elementos segmentados possui.

Dessa forma é necessário apenas verificar o valor de A para cada objeto e descartar os valores menores que 0,6% da área da imagem original.

Para a remoção de objetos que possuem dimensões horizontais muito grandes utilizamos um algoritmo desenvolvido que é mostrado na figura 1.16.

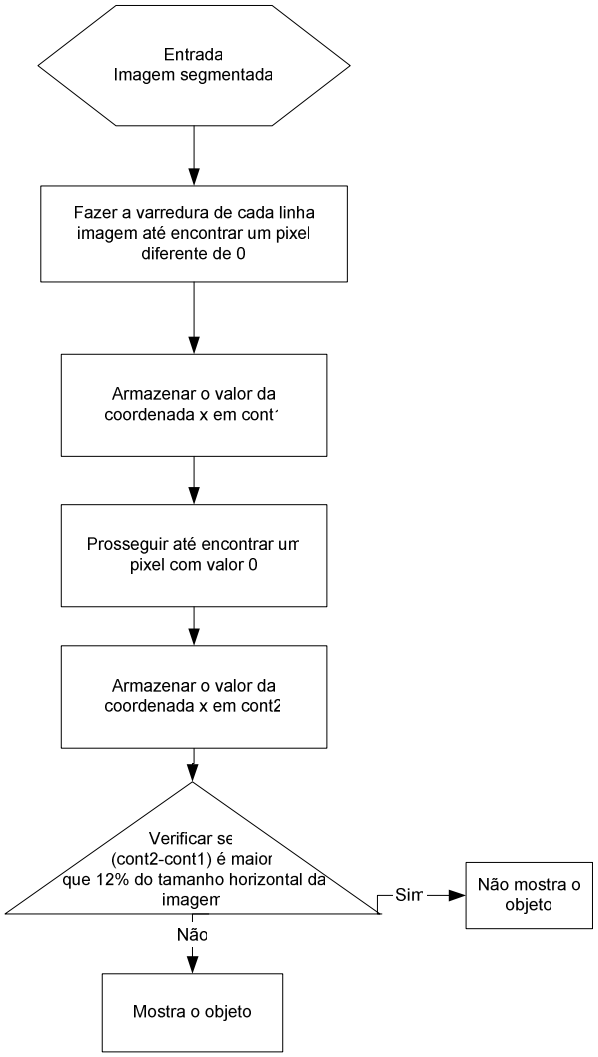

Figura 1.17 – Remoção de objetos com dimensões horizontais grande

Esse algoritmo faz a varredura da imagem linha a linha. Quando um pixel que representa um objeto é encontrado o valor da coordenada x dele é armazenado. A varredura continua até o último pixel do objeto na linha em análise. Quando o último

pixel do objeto em análise é encontrado o valor da coordenada x dele é subtraído do valor encontrado no início da análise do objeto. Esse valor encontrado é comparado com o valor máximo permitido. Se for maior o objeto não será considerado (é tratado como sombra).

Os valores utilizados nos filtros aplicados nessa etapa do processamento foram obtidos de forma empírica de acordo com experiências que consideram as características geométricas apresentadas nas placas utilizadas nos veículos brasileiros.

1.7 Esqueletização

Após o processo de limiarização e segmentação dos objetos é realizado o processo de esqueletização também conhecido como *thinning*. A esqueletização consiste na obtenção do esqueleto dos objetos submetidos. Ou seja reduzir as partes de um objeto a uma linha fina que a representa. O processo de esqueletização tem uma grande variedade de aplicações, pois representa um resultado que facilita a análise dos dados. No caso de caracteres o processo de esqueletização ideal se aproxima dos padrões humanos de escrita dos caracteres.

De uma forma geral a esqueletização favorece uma análise estrutural simples e um projeto mais intuitivo de algoritmos de reconhecimento. Adicionalmente, a redução de uma imagem a sua essência pode eliminar algumas distorções no contorno enquanto mantém as propriedades geométricas e topológicas. Em termos mais práticos a esqueletização pode fornecer uma representação mais fácil de extrairem-se propriedades críticas como pontos extremos, junções, e conexões ao longo dos objetos em uma imagem.

Naturalmente, para um algoritmo de esqueletização ser eficiente, ele precisa apresentar algumas características como: (1) compactar dados; (2) manter propriedades significativas dos padrões; e (3) eliminar ruídos locais sem introduzir distorções.

Os algoritmos de esqueletização pertencem a dois grupos sendo eles divididos em sequenciais e paralelos.

Os algoritmos sequenciais geram esqueletos melhores do ponto de vista de conectividade e da conservação topológica. Contudo, exigem muito tempo de processamento. Os algoritmos paralelos se caracterizam pela velocidade de processamento, mas muitas vezes geram esqueletos com falhas. Além desses dois grupos há um terceiro grupo que se caracteriza por tentar gerar um esqueleto em um único passo. Esses algoritmos são conhecidos como algoritmos não-interativos.

Utilizamos nesse trabalho um algoritmo MAT (Medial Axis Tranformation) para imagens binárias. O algoritmo MAT faz parte do grupo de algoritmos sequênciais. Foi escolhido porque para o caso de caracteres não pode haver perda de informação ou esqueletos com falhas. O algoritmo consiste em dois passos. Nos passos ímpares devemos utilizar o passo 1 e nos passos pares o passo 2.

Passo 1:

a- $2 \le NN(p_1) \le 6$

b- $CRN(p_1) = 1$

c- $p_2 . p_4 . p_6 = 0$

d- $p_4 . p_6 . p_8 = 0$

No passo 2 as equações (a) e (b) são mantidas enquanto que as (c) e (d) devem ser substituídas por:

e- $p_2 . p_4 . p_8 = 0$

f- $p_2.p_6.p_8 = 0$

Os *pixels* são definidos conforme a tabela 1.1:

Tabela 1.1- Vizinhança do pixel

p9	p2	p3
p8	p1	p4
p7	p6	p5

Onde: NN(p) => Número de pontos vizinhos diferentes de 0 do ponto p.

$CRN(p_1)$ => Número de transições de 0 para 1 na seqüência p2,p3,…,p9,p2.

O passo 1 é aplicado a todo o ponto da borda do objeto considerado. Se uma ou mais condições são violadas, o valor do ponto não é alterado. Se todas as condições são satisfeitas o ponto é marcado para exclusão. Contudo, os pontos não são excluídos até todos os da borda terem sido processados. Isto previne mudanças na estrutura dos dados durante o processamento. Após o passo 1 ter sido aplicado em todos os pontos, aqueles que forem marcados para ser excluídos são excluídos (valores alterados para 0). Então o passo 2 é aplicado a imagem da mesma forma que o passo 1.

Resumidamente, uma iteração do algoritmo de afinamento consiste de:

- Aplicar o passo 1
- Excluir os pontos marcados
- Aplicar o passo 2
- Excluir os pontos marcados

Este procedimento é aplicado interativamente até que nenhum ponto seja excluído. Quando o algoritmo terminar, o esqueleto da região estará definido.

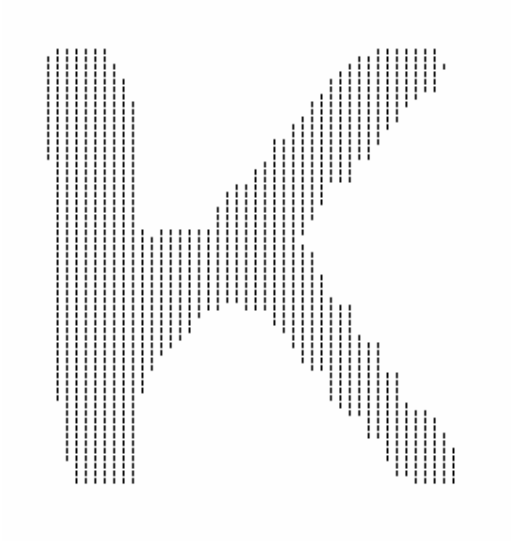

Figura 1.18 - Objeto antes da esqueletização

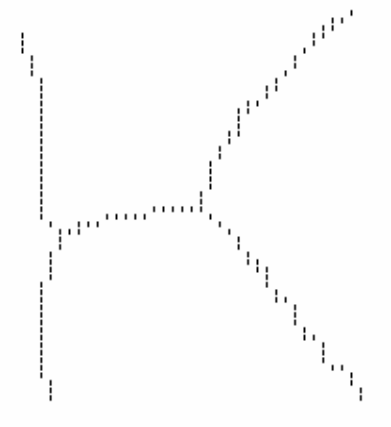

Figura 1.19 - Objeto depois da esqueletização

As figuras 1.17 e 1.18 mostram o objeto binarizado antes e depois do processamento do algoritmo de esqueletização.

Ao final da fase de esqueletização possuímos várias imagens binarizadas e esqueletizadas sendo cada uma delas um caracter. Os objetos que não eram caracteres e foram detectados foram descartados através dos filtros. Torna-se possível iniciar a extração de características que serão utilizadas nas fases posteriores para o reconhecimento dos caracteres.