UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE AUTOMAÇÃO E SISTEMAS

José Alexander Dueñas Salazar

RECONHECIMENTO AUTOMÁTICO DE PLACAS VEICULARES BRASILEIRAS EM AMBIENTES NÃO CONTROLADOS

Florianópolis
2017

José Alexander Dueñas Salazar

RECONHECIMENTO AUTOMÁTICO DE PLACAS VEICULARES BRASILEIRAS EM AMBIENTES NÃO CONTROLADOS

Dissertação submetida ao Programa de Pós-graduação em Engenharia de Automação e Sistemas para a obtenção do Grau de Mestre em Engenharia de Automação e Sistemas.
Universidade Federal de Santa Catarina
Orientador: Prof. Dr. Marcelo Ricardo Stemmer.

Florianópolis
2017

Ficha de identificação da obra elaborada pelo autor através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Dueñas , Jose Alexander
Reconhecimento automático de placas veiculares brasileiras em ambientes não controlados / Jose Alexander Dueñas ; orientador, Marcelo Ricardo Stemmer, 2017.
131 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós Graduação em Engenharia de Automação e Sistemas, Florianópolis, 2017.

Inclui referências.

1. Engenharia de Automação e Sistemas. 2. Engenharia de Automação e Sistemas. 3. Reconhecimento de objetos. 4. Reconhecimento automático de placas. I. Stemmer, Marcelo Ricardo . II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Automação e Sistemas. III. Título.

José Alexander Dueñas Salazar

RECONHECIMENTO AUTOMÁTICO DE PLACAS VEICULARES BRASILEIRAS EM AMBIENTES NÃO CONTROLADOS

Esta Dissertação foi julgada e aprovada para obtenção do Título de "Mestre em Engenharia de Automação e Sistemas" e aprovada em sua forma final pelo Programa de Pós-graduação em Engenharia de Automação e Sistemas.

Florianópolis, 28 de Julho de 2017.

_____
Prof. Dr. Daniel Ferreira Coutinho
Coordenador do Curso

_____
Prof. Dr.-Ing. Marcelo Ricardo Stemmer
Orientador

Banca Examinadora:

_____
Prof. Dr. Mário Lucio Roloff
Instituto Federal de Santa Catarina

_____
Prof. Dr. Maurício Edgar Stivanello
Instituto Federal de Santa Catarina

_____
Prof. Dr. Jorge Henrique Busatto Casagrande
Instituto Federal de Santa Catarina

Este trabalho é dedicado à minha família, sobretudo à minha mãe Carmen.

AGRADECIMENTOS

A Deus pela vida e a oportunidade de fomentar os meus sonhos.

À minha mãe Carmen por tudo o seu amor, conselhos e compreensão nos momentos difíceis. Aos meus irmãos Diana e Felipe por serem parte deste projeto de vida. Ao meu Pai Jose, que desde o céu, sempre está me protegendo.

A Dany Matos, que esteve do meu lado nos momentos difíceis, companheira de sempre. O seu apoio, fortaleça e amor incondicional foram essenciais para alcançar a meta. A Doroty pelo seu apoio, incentivo, carinho e visão de vida. A Jose, pelas risadas e exemplo de amizade. Foram minha família em todo momento.

A todos os meus familiares e amigos, sobretudo o Jaime pela sua amizade incondicional.

Aos mestrandos e doutorandos do grupo S2i, que de uma ou outra maneira contribuíram para o desenvolvimento desse projeto. Aos professores do PGEAS pelos conhecimentos e experiências.

Ao meu orientador, professor Dr. Marcelo Ricardo Stemmer pela confiança, apoio e compromisso no desenvolvimento do meu trabalho. À CAPES pelo apoio financeiro.

Seja a mudança que você quer ver no mundo.
(Ghandi)

RESUMO

Um dos tópicos mais importantes dentro dos Sistemas Inteligentes de Transporte (ITS, *Intelligent Transport Systems*) é o Reconhecimento Automático de placas veiculares (ALPR, *Automatic License Plate Recognition*), sendo parte fundamental em sistemas como: pedágios eletrônicos, análise de tráfego, contagem e rastreamento de veículos e monitoramento de infrações (fiscalização).
O ALPR é uma combinação de diferentes módulos onde estão envolvidas técnicas de detecção de objetos, processamento de imagens e reconhecimento de padrões (*Pattern recognition*) e tem como principal objetivo extrair as placas de uma imagem de entrada e reconhecer os caracteres que nela se encontram. Nesse contexto, são três as fases que compõem o ALPR: localização e detecção da placa, segmentação dos caracteres e reconhecimento dos caracteres. Em cada uma dessas fases apresentam-se problemas a serem considerados, como variações nos tamanhos das placas, variação do contraste, ângulos variantes, sujeiras, diferentes tipos de letras e semelhança dos caracteres. Embora na literatura encontram-se métodos para o ALPR, muitos deles são desenvolvidos sobre bancos de imagens capturadas em ambientes controlados onde as condições de luz, contraste, ângulo e distância de captura sempre são as mesmas. A resposta desses métodos sobre imagens capturadas em ambientes urbanos, onde são apresentadas condições climáticas cambiantes e luminosidade variante (ambientes não controlados), não é boa. Assim, neste trabalho é apresentado um método para o reconhecimento de placas veiculares Brasileiras em ambientes não controlados, neste caso rodovias da cidade de São Paulo, tendo como base um banco de imagens fornecido pela empresa Brasileira Brascontrol.

Palavras-chave: ALPR, Reconhecimento Automático de placas Veiculares, Detecção de Objetos, Reconhecimento de padrões, Visão computacional.

ABSTRACT

One of the most important topics within intelligent transport systems is the automatic recognition of vehicle license plates, because it is a fundamental part in systems such as: electronic tolls, analysis and control of the traffic, vehicle counting and monitoring of traffic violations.
The ALPR is a combination of different modules that involve techniques of object detection, image processing and pattern recognition. Its main objective is to extract the vehicular plates in an entrance image, and recognize the characters that compose them. In each one of these phases Problems such as variations in plate sizes, contrast variations, varying angles, dirt, and different kinds of letters and similarity of characters are found. Although ALPR methods can be found in the literature, many of them are developed on images captured in controlled environments, where light conditions, contrast, angle and capture distance are always the same. The response of those methods on images captured in urban environments where changing climatic conditions and varying luminosity (uncontrolled environments) are present is not good. Thus, in this work a method is proposed for the recognition of Brazilian vehicular plates in uncontrolled environments, in this case, avenues of the city of São Paulo, based on a database of images provided by the Brazilian company BrasControl.

Keywords: ALPR, Automatic License Plate Recognition, Object Detection, Pattern Recognition, Computer Vision.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

MMLPF	*Mathematic morphologic low-pass filter*
HOG	*Histogram of Oriented Gradients*
SVM	*Support Vector Machines*
ITS	*Intelligent transport Systems*
IA	Inteligência Artificial
OCR	*Optical character recognition*
RBF	*Radial Basis Functions*
MSER	*maximally stable region*
CCA	*Connected component Analysis*
ELM	*Extreme learning machine*
THT	*Top Hat Transform*
BHT	*Bottom Hat Transform*
PCA	*Principal Component Analysis*
LDA	*Linear Discriminant Analysis*
ALPR	*Automatic License Plate Recognition*

SUMÁRIO

1 INTRODUÇÃO

Com a constante evolução das diferentes áreas da ciência, muitas disciplinas relacionadas à tecnologia têm sido favorecidas, e entre elas encontram-se a visão computacional e a inteligência artificial (IA).

Dentro das técnicas usadas em visão computacional, o tratamento de imagens e o reconhecimento de padrões (*pattern recognition*) são áreas com uma pesquisa bastante relevante, devido as suas aplicações em diferentes campos: Robótica, medicina, trafego urbano, qualidade de produtos, vídeo games, etc. (OLAGUE, 2007). Nesse contexto, a importância da visão computacional dentro dos sistemas inteligentes de transporte (ITS, *Intelligent Transport systems*) tem passado por forte incremento recentemente. Esses sistemas desempenham um papel significativo para dar solução aos diferentes problemas de segurança, controle e gestão de trafego que se apresentam diariamente em ambientes urbanos (BUCH; VELASTIN; ORWELL, 2011).

Por outro lado, nos últimos anos a frota de veículos no Brasil tem crescido consideravelmente. Segundo o relatório feito pelo observatório das Metrópoles (DE QUEIROZ, 2015), a taxa de motorização passou de 14,4 automóveis para cada 100 habitantes em 2001 (a frota estava em torno de 24,5 milhões) para 28,1 automóveis para cada 100 habitantes em 2014 (representando um total de 56,9 milhões de automóveis), e a tendência é que os números continuem crescendo. Dessa perspectiva, os desafios para gerenciar o trafego nas grandes e pequenas cidades, além das rodovias, são cada vez maiores, tornando evidente a necessidade de implementar ITSs a fim de dar solução às diferentes problemáticas geradas por esse aumento.

Dentro dos ITSs encontra-se o reconhecimento automático de placas veiculares (ALPR, *Automatic License plate Recognition*), que é parte fundamental em diversas aplicações, tais como pedágios automáticos, análise de trafego, contagem de veículos, controle de acesso a ambientes restritos, rastreamento de veículos e monitoramento de infrações (fiscalização) (ZAHEDI; SALEHI, 2011) (BUCH; VELASTIN; ORWELL, 2011).

O ALPR é uma combinação de diferentes módulos onde estão envolvidas técnicas de detecção de objetos, processamento de imagens e reconhecimento de padrões (*Pattern recognition*) e tem como principal objetivo extrair as placas de uma imagem de entrada e reconhecer os caracteres que nela se encontram (AHMAD et al., 2015).

Além da aquisição da imagem, que para ambientes urbanos (especificamente estradas) é feita geralmente através de câmeras monocromáticas com projetores infravermelhos (BUCH; VELASTIN; ORWELL, 2011), o processo de reconhecimento de placas veiculares (LPR, *License Plate Recognition*) está dividido em três fases principais (ANAGNOSTOPOULOS et al., 2008) (AHMAD et al., 2015):

- Localização e detecção da placa: O objetivo aqui é localizar e segmentar na imagem de entrada regiões que podem representar uma placa veicular.
- Segmentação dos caracteres: Nesta fase os símbolos ou caracteres são separados uns dos outros e são deixados em blocos com um contorno definido para passar à etapa seguinte (JIAO; YE; HUANG, 2009).
- Reconhecimento dos caracteres: Esta fase é também conhecida como reconhecimento ótico de caracteres *(OCR, Optical character recognition)*, onde cada um dos caracteres segmentados é identificado (AHMAD et al., 2015).

Em cada uma dessas fases apresentam-se problemas a serem considerados como, por exemplo, variações nos tamanhos das placas, baixo contraste, iluminação e ângulos variantes, sujeiras, diferentes fontes de letras, pouca estandardização e semelhança dos caracteres. Em vista disso, são muitos os métodos propostos e testados em diferentes bases de dados a fim de solucionar os problemas citados (DU et al., 2013). Embora muitos desses métodos funcionem bem sobre ambientes controlados onde as condições de luz, contraste, ângulo e distancia de captura sempre são as mesmas, em ambientes urbanos onde são apresentadas condições climáticas cambiantes e luminosidade variante (ambiente não controlado), a resposta não é a mesma. É assim que, não existe uma solução geral para solucionar o problema de LPR, devido a que esta solução é altamente dependente das imagens de entrada (AHMAD et al., 2015).

De acordo com este contexto, neste trabalho foram investigadas diferentes propostas que solucionam o problema do reconhecimento de placas veiculares tanto em ambientes controlados, como em ambientes não controlados. Assim a pesquisa realizada visou o desenvolvimento de estratégias para superar o problema do ALPR em ambientes não controlados, neste caso, rodovias da cidade de São Paulo.

1.1 MOTIVAÇÃO

Como exposto anteriormente, a frota Brasileira de veículos passou de 24 milhões para quase 57 milhões no 2014. Como consequência, é possível identificar um forte crescimento das tecnologias dos sistemas inteligentes de transporte (MARIA et al., 2014) para atender necessidades como a segurança no transito e melhora da mobilidade urbana. Dessa forma, são muitas as empresas e grupos de pesquisa focados no desenvolvimento de sistemas que ajudam a suprir essas necessidades. Dentro desses sistemas, as tecnologias que envolvem a identificação de placas veiculares têm sido, com sucesso, cada vez mais utilizadas.

Nesse contexto, a identificação automática dos veículos que é realizada através do reconhecimento automático das placas, é um recurso importante em serviços como detecção de veículos roubados, controle de trafego em pedágios e estradas, gerenciamento e controle de fluxo de veículos e fiscalização e cumprimento de regras de transito (BERNARDI, 2015).

Dentro do grupo de empresas especializadas que atua na área de desenvolvimento de ITS, encontra-se a Brascontrol[1], empresa Brasileira fundada em 1981 e que opera em diversas cidades Brasileiras. Graças à parceria entre essa empresa e o grupo de pesquisa S2i -Sistemas industriais inteligentes-, pertencente ao departamento de automação e sistemas da Universidade Federal de Santa Catarina, foi possível que a empresa fornecesse um banco de imagens veiculares capturadas através de equipamentos de fiscalização eletrônica pertencentes à companhia.

Uma vantagem importante desse tipo de imagens, como foi mencionado anteriormente, é que são capturadas em ambientes onde as condições climáticas, de luminosidade e contraste não são controladas. Assim, através do desenvolvimento do método proposto, poderão ser atingidas as diferentes problemáticas que apresentam esse tipo de imagens, tornando a solução proposta, viável para futuras implementações em aplicações relacionadas com os ITS.

.

[1] http://www.brascontrol.com.br/

1.2 OBJETIVOS DO TRABALHO

1.2.1 Objetivo geral

O objetivo deste trabalho é desenvolver um método baseado em técnicas de visão computacional, para a detecção, segmentação e reconhecimento de placas veiculares Brasileiras em ambientes não controlados, a partir de um banco de imagens digitais fornecidas pela empresa Brascontrol.

1.2.2 Objetivos específicos

Para dar cumprimento ao objetivo geral deste trabalho, se tem os seguintes objetivos específicos:

- Propor um método de reconhecimento de placas veiculares baseado nas três fases principais de um sistema ALPR: detecção, segmentação e reconhecimento.
- Implementar um software para o reconhecimento de pacas Brasileiras em ambientes não controlados baseado no método proposto.
- Realizar testes com o método proposto para determinar a sua acurácia, além de comparar com algumas técnicas presentes na literatura.

1.3 ORGANIZAÇÃO DO TRABALHO

Este trabalho está organizado em seis capítulos. No capítulo 1 é apresentada a introdução do trabalho, onde são indicados os objetivos e as motivações que levaram à concepção do método ALPR. O capítulo 2 apresenta o referencial teórico, onde é exposta a fundamentação teórica necessária para o desenvolvimento do trabalho. No capítulo 3 são apresentados alguns trabalhos que propõem métodos para o reconhecimento automático de placas veiculares tanto Brasileiras como de outros países. A descrição do método proposto neste trabalho é realizada no capítulo 4. Aqui são apresentadas as técnicas utilizadas para o seu desenvolvimento e implementação. No capítulo 5 o método proposto é testado no banco de imagens fornecida pela Brascontrol, além disso são apresentados e analisados os resultados obtidos através da

realização do teste. Por fim, o capítulo 6 apresenta as conclusões do trabalho, assim como sugestões para pesquisas futuras visando melhorar expandir o método ALPR proposto.

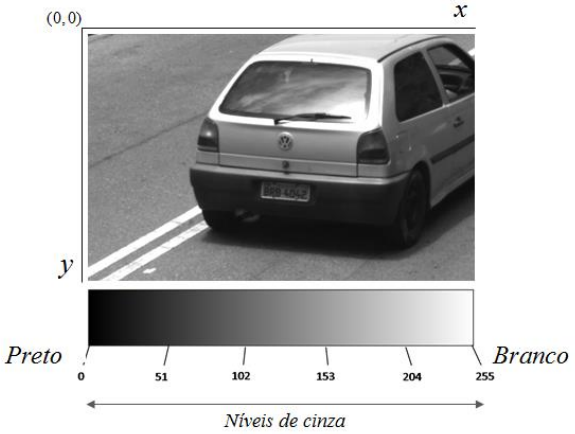
2 REFERENCIAL TEORICO

Neste capítulo serão abordados os principais conceitos teóricos que constituem a base para o desenvolvimento do método de reconhecimento de placas proposto neste trabalho.

2.1 PROCESAMENTO DIGITAL DE IMAGENS

Uma imagem pode ser definida como uma função bidimensional $f(x, y)$, onde x e y são coordenadas espaciais. A amplitude de f em qualquer par de coordenadas (x, y), é chamada de intensidade ou nível de cinza da imagem (para uma imagem monocromática), sendo os valores x, y e f finitos e discretos. Por tanto, a imagem pode ser descrita como uma matriz $M \times N$ onde M representa o eixo vertical y, e N o eixo horizontal x como ilustra-se na Figura 2. 1. Assim, cada elemento da matriz é chamado de *Picture element* ou Pixel (GONZALEZ; WOODS, 2006).

Figura 2. 1 - Exemplo de uma imagem digital monocromática.



Fonte: Elaborado pelo autor.

Para cada pixel da imagem é associado um número inteiro L que representa o nível de cinza nesse ponto (no caso de imagens monocromáticas). Dessa maneira, por convenção, os níveis de cinza da imagem variam desde $L = 0$ que representa a ausência de iluminação

(preto) até $L = 255$, representando o valor de máximo brilho (branco). Já a imagem binaria, que é um caso particular de uma imagem monocromática (GONZALEZ; WOODS, 2006), é utilizada normalmente para ressaltar algum tipo de característica ou como máscara em algum processo de segmentação, possui somente dois valores de intensidade para cada pixel, $L = 0 \vee L = 1$(PEDRINI; SCHWARTZ, 2007).

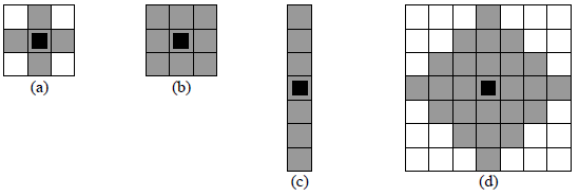
A partir dessas definições podem ser realizadas diferentes operações sobre os píxeis a fim de analisar, melhorar, transformar ou segmentar a imagem.

2.1.1 Morfologia Matemática

A morfologia Matemática foi introduzida em 1964 pelos pesquisadores George Matheron e Jean Serra em Fontaineblau, França. Ela descreve ou analisa a estrutura geométrica das imagens a partir de um conjunto perfeitamente definido e conhecido denominado elemento estruturante (máscara), este interage com os objetos contidos na imagem, modificando a sua aparência, a sua forma, e o seu tamanho (FACON, 2011).

O elemento estruturante pode assumir várias formas, valores (0 a 255) e a sua origem pode ser definida em qualquer ponto. A Figura 2. 2 apresenta alguns elementos estruturantes usados frequentemente. Assim, o elemento estruturante percorre a imagem realizando diversas operações morfológicas em cada pixel que compoe a imagem e dessa maneira podem ser extraídas informações relevantes sobre o tamanho e formas de objetos na imagem (GONZALEZ; WOODS, 2006).

Figura 2. 2 – Exemplo de elementos estruturantes. (a) Cruz. (b) Caixa ou retangular. (c) Linha Vertical



Fonte: (GONZALEZ; WOODS, 2006)

Por outra parte, a morfologia matemática está fundamentada na teoria de conjuntos. Assim, em imagens binarias, o conjunto de píxeis presentes é definido no espaço bidimensional dos números inteiros $\mathbb{Z}^2$,

em que cada elemento do conjunto é um vetor bidimensional com coordenadas (x, y) dos pontos dos objetos com respeito a uma origem (PEDRINI; SCHWARTZ, 2007).

Essencialmente todas as operações morfológicas existentes e desenvolvidas baseiam-se em duas operações primitivas: Dilatação e erosão. Antes de definir cada uma delas é importante levar em conta dois conceptos fundamentais que ajudam a seu entendimento. A reflexão e a translação (GONZALEZ; WOODS, 2006).

Sejam A e B conjuntos de $\mathbb{Z}^2$, com componentes $a = (a_1, a_2)$ e $b = (b_1, b_2)$, translação de A por $x = (x_1, x_2)$, denotada como $(A)_x$, é definida como:

$$(A)_x = \{c \mid c = a + x, para \ \ a \in A\} \tag{2. 1}$$

A reflexão de B, definida por B é descrita como:

$$B = \{x \mid x = -b, para \ \ b \in B\} \tag{2. 2}$$

O complemento do conjunto A é representado como:

$$A^c = \{x \mid x \notin A\} \tag{2. 3}$$

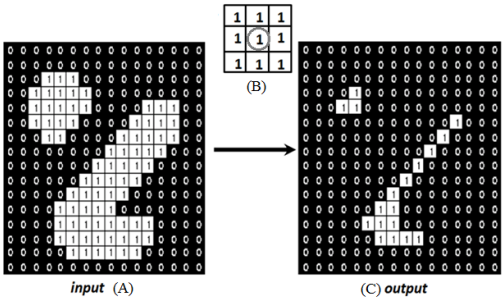
2.1.1.1 Dilatação e Erosão

A operação de dilatação tem como objetivo aumentar a área geométrica dos objetos na imagem, esse aumento depende do formato do elemento estruturante utilizado. Portanto, a dilatação de um conjunto A pelo elemento estruturante B, está dada pela seguinte equação:

$$A \oplus B = \left\{x \mid (B)_x \cap A \neq \varnothing\right\} = \left\{x \mid \left[(B)_x \cap A\right] \subseteq A\right\} \tag{2. 4}$$

Aqui, $A \oplus B$ é o conjunto de todos os deslocamentos, de forma que B (B refletido em torno da sua origem) e A se sobreponham pelo menos por um elemento (GONZALEZ; WOODS, 2006). A Figura 2. 3 ilustra um exemplo do processo.

Figura 2. 3 - Exemplo de dilatação. (a) Imagem de entrada (b) Elemento estruturante retangular 3x3. (c) Resultado da dilatação.



Fonte: Editado de http://cse19-iiith.vlabs.ac.in/theory.php?exp=morph

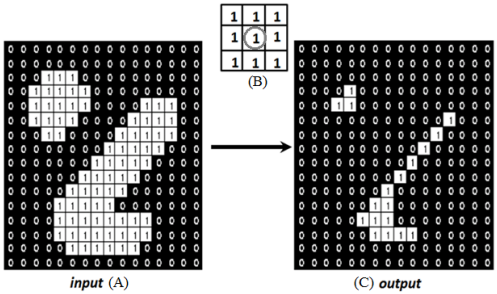
Em outras palavras, a dilatação da imagem A pelo elemento estruturante B consiste em mover a origem de B (no caso da Figura 2. 3 o elemento estruturante é retangular com origem no meio) sobre cada pixel dos objetos de A, atribuindo o valor de 1(branco) a cada posição que é sobreposta pelo elemento estruturante B.

A erosão é a operação contraria à dilatação, ou seja, visa diminuir a área geométrica dos objetos presentes na imagem, eliminando os detalhes que são menores que o elemento estruturante. Essa diminuição depende do elemento estruturante utilizado. Assim, a erosão de um conjunto A pelo elemento estruturante B, está dada pela seguinte equação.

$$A\theta B = \left\{ x \mid (B)_x \subseteq A \right\} = \left\{ x \mid (B)_x \cap A^c \neq \varnothing \right\} \qquad (2.\ 5)$$

Onde A e B são conjuntos de $\mathbb{Z}^2$ e $A\theta B$ é o conjunto de todos os pontos x de forma que B, transladado por x, está contido em A (GONZALEZ; WOODS, 2006). Na Figura 2. 4 apresenta-se um exemplo do processo de erosão onde o elemento estruturante B é retangular com a origem no meio. Nesse contexto, erodir a imagem entrante respeito ao elemento estruturante B consiste em mover a origem de B (ponto central neste caso) sobre cada pixel dos objetos de A, tal que, caso B esteja totalmente contido em A, o pixel da imagem sobre a origem de B pertencerá a erosão entre A e B.

Figura 2. 4 - Exemplo de Erosão. (a) Imagem de entrada. (b) Elemento estruturante. (c) Resultado da erosão.



Fonte: Editado de http://cse19-iiith.vlabs.ac.in/theory.php?exp=morph

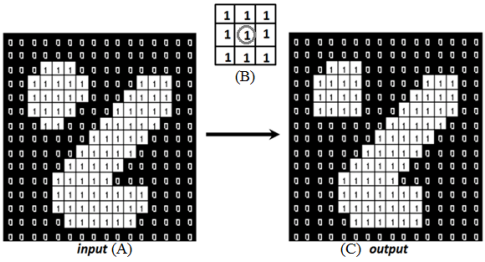
2.1.1.2 Abertura e fechamento

A combinação da erosão com uma dilatação é conhecida como abertura. É geralmente usada para suavizar os contornos dos objetos, eliminar contornos estreitos entre objetos e remover saliências finais (GONZALEZ; WOODS, 2006). Dessa maneira, a abertura de uma imagem A pelo elemento estruturante B, ambos pertencentes a $\square^2$ está dada por:

$$A \circ B = (A \theta B) \oplus B \qquad (2.6)$$

A Figura 2. 5 demonstra um exemplo da operação de abertura. Aqui, o elemento estruturante é também retangular com tamanho 3x3 e origem no meio.

Figura 2. 5 - Exemplo de Abertura. (a) Imagem de entrada. (b) Elemento estruturante. (c) Resultado da abertura.



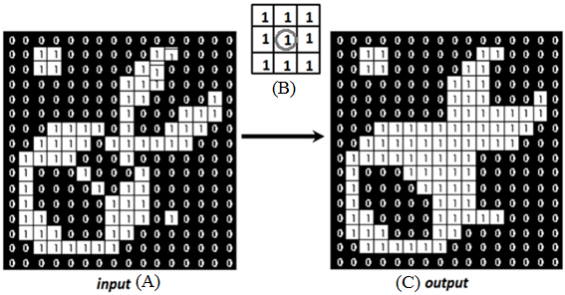
Fonte: Editado de http://cse19-iiith.vlabs.ac.in/theory.php?exp=morph

Já o fechamento é definido como uma operação onde uma abertura, é seguida de uma erosão em um objeto (equação (2.7)). Utilizado geralmente para fundir separações estreitas entre objetos e remover buracos mais pequenos que o elemento estruturante usado (GONZALEZ; WOODS, 2006).

$$A \bullet B = (A \oplus B) \theta B \qquad (2.7)$$

A Figura 2. 6 ilustra um exemplo da operação de fechamento com um elemento estruturante de tamanho 3x3.

Figura 2. 6– Exemplo de fechamento. (a) Imagem de entrada. (b) Elemento estruturante. (c) Resultado do fechamento.



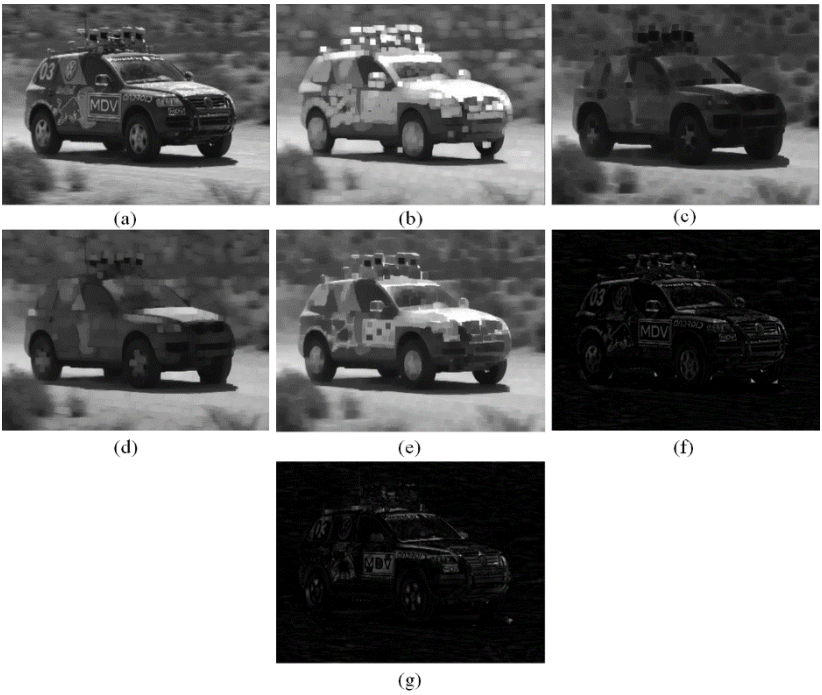
Fonte: Editado de http://cse19-iiith.vlabs.ac.in/theory.php?exp=morph

2.1.1.3 Extensão dos operadores morfológicos para imagens em cinza.

As operações vistas até agora são extensíveis para imagens em escala de cinza. Onde os píxeis tanto da imagem quanto do elemento estruturante tomam valores entre $0 e L-1$, ou seja, 0 a 255 níveis de cinza. Em níveis de cinza as imagens digitais podem ser vistas como uma função $f(x, y)$ e o elemento estruturante como $b(x, y)$.

A seguir citam-se as operações mais importantes. Já na Figura 2. 7 é ilustrado o resultado dessas operações aplicadas a uma imagem em tons de cinza.

Figura 2. 7 – Operações morfológicas em escala de cinza com um elemento estruturante retangular de tamanho 3x3. (a) Imagem original. (b) Dilatação. (c) Erosão. (d) Abertura. (e) Fechamento. (f) Top-Hat. (g) Bottom-Hat.



Fonte: Editado de (KAEHLER; BRADSK, 2016)

2.1.1.3.1 Dilatação e Erosão.

A dilatação de f pelo elemento estruturante b é obtida pelo máximo da imagem na região coincidente com b, quando b está em (x, y). A equação (2.8) representa esta operação.

$$\left[A \oplus B\right](x, y) = \max_{(m,n)\in b}\left[f(x-m, y-n)\right] \qquad (2.\ 8)$$

Como resultado da dilatação obtém-se uma imagem onde as partes mais claras são realçadas, reduzindo e minimizando padrões mais escuros. Frequentemente é usada para tentar encontrar componentes

conectados (regiões com píxeis onde os níveis de cinza ou intensidade são similares) (KAEHLER; BRADSK, 2016). A Figura 2. 7(b) demonstra o resultado deste processo.

Por outra parte, a operação de Erosão (equação 2.9)) ao igual que nas operações binarias, é o processo oposto à dilatação. Aqui, a operação faz com que a imagem seja escurecida, os padrões mais escuros são aumentados e os vales próximos são conectados. É utilizada geralmente para eliminar "pontos" espalhados pela imagem. Na Figura 2. 7 (c) apresenta-se um exemplo.

$$\left[A\theta B\right](x,y) = \min_{(m,n)\in b}\left[f(x+m,y+n)\right] \qquad (2.\ 9)$$

2.1.1.3.2 Abertura e Fechamento

Analogamente às operações em imagens binarias. A abertura é vista como uma erosão, seguida da dilatação de um objeto e o seu efeito na imagem original é de diminuir pequenos detalhes, além de remover picos de tamanho menor que o elemento estruturante (Figura 2. 7 (d)). Já o fechamento é definido como uma dilatação, seguida da erosão do objeto, tendo como resultado a união de picos próximos, conservação de picos afastados, deixa a imagem resultante mais regular que a imagem original, além de diminuir detalhes da imagem original (Figura 2. 7 (e)) (FACON, 2011) (KAEHLER; BRADSK, 2016). As equações (2.10) e (2.11) representam a abertura e o fechamento respetivamente.

$$f \circ b = (f\ \theta b) \oplus b \qquad (2.\ 10)$$

$$f \bullet b = (f \oplus b)\ \theta\, b \qquad (2.\ 11)$$

2.1.1.3.3 Transformadas Top-Hat e Bottom-Hat

A transformadas top-Hat e Bottom Hat, visam o descobrimento das estruturas da imagem eliminadas pelo processo de abertura e fechamento da imagem. Assim a transformada Top-Hat definida como a diferença entre a imagem original e o resultado de sua abertura (equação (2.12)) tem como função ressaltar os vales (regiões escuras) da imagem, ou seja,

realçar os objetos claros eliminados pelo processo de apertura (GONZALEZ; WOODS, 2006). Na Figura 2. 7(f) ilustra-se este processo.

$$THT(f) = f - (f \circ b) \tag{2. 12}$$

A transformada Bottom-Hat, conhecida também como Black-Hat é definida como a diferença entre o resultado do fechamento e a imagem original (equação (2.13)), e seu resultado é uma imagem onde são realçados os objetos escuros eliminados pelo processo de fechamento conforme apresentado na Figura 2. 7 (g).

$$BHT(f) = (f \bullet b) - f \tag{2. 13}$$

As operações de Top-Hat e Bottom-Hat são complementares entre si. Assim, sua combinação é muito útil para realçar imagens, realçar objetos quando a iluminação não é homogênea e realçar o contraste em geral (SOILLE, 2003).

2.1.2 Binarização ou Limiarização.

A Binarização é umas das tarefas mais comuns na análise de imagens digitais, além de que é um dos métodos mais simples de segmentação de imagens. Este consiste em aplicar uma função de transformação de maneira que a imagem final possua apenas dois níveis de intensidade, em outras palavras, é uma operação não linear que converte uma imagem em escala de cinza a uma imagem binaria onde os dois níveis são atribuídos a píxeis que estão abaixo ou acima de um valor limite especificado, chamado de limiar (GONZALEZ; WOODS, 2006). Desta forma, o tipo de binarização mais comum é chamado de binarização global. Aqui, se o pixel da imagem em escala de cinza possui um valor maior que o limiar, atribui-se a esse pixel o valor de 255 (branco) na imagem binarizada, caso contrário o pixel adquire o valor de 0 (preto). Por fim, os píxeis com valor 0 serão atribuídos ao fundo da imagem *(background*), enquanto que os objetos de interesse segmentados estarão no *foreground* (frente) da imagem. (JAIN; LALA, 2013).

2.1.2.1 Método do Limiar de Otsu.

O método de Otsu (OTSU, 1979) é usado para selecionar um limiar de forma automática em imagens em escala de cinza usando a informação

do histograma. Ele seleciona um limiar pela maximização da variância entre as classes. Através desse limiar são divididos os píxeis da imagem em *foreground e background* (LIU; YU, 2009). O método é considerado ótimo devido aos cálculos computacionais que são feitos no histograma 1-D da imagem (GONZALEZ; WOODS, 2006).

Seja uma imagem digital em escala de cinza com tamanho $M \times N$ píxeis, e L os níveis de cinza que variam entre 0 e $L-1$ (0 a 255). O número de píxeis no nível i é denotado por $N = n_1 + n_2 + ... + n_L$. A probabilidade do nível de cinza i é obtido pela seguinte expressão.

$$pi = \frac{n_i}{N},\ pi \geq 0,\ \sum_{i=0}^{L-1} pi = 1 \tag{2. 14}$$

Ao aplicar um limiar t em uma imagem, os píxeis são divididos em duas classes: C_1 com níveis de cinza $[0,1,...,t]$ e C_2 com níveis de cinza $[t+1,...,L-1]$. A distribuição probabilística do nível de cinza para as duas classes está dada por:

$$w_1 = \Pr(C_1)\sum_{i=0}^{t} pi \tag{2. 15}$$

$$w_2 = \Pr(C_2)\sum_{i=t+1}^{L-1} pi \tag{2. 16}$$

As medias das classes C_1 e C_2 são:

$$u_1 = \sum_{i=0}^{t} \frac{i\,pi}{w_1} \tag{2. 17}$$

$$u_2 = \sum_{i=t+1}^{L-1} \frac{i\,pi}{w_2} \tag{2. 18}$$

Já a média total dos níveis de cinza u_T é definida como

$$u_T = w_1 u_1 + w_2 u_2 \tag{2. 19}$$

As variâncias das classes são:

$$\sigma_1^2 = \sum_{i=0}^{t} (i - u_1)^2 \frac{pi}{w_1} \tag{2. 20}$$

$$\sigma_2^2 = \sum_{i=t+1}^{L-1} (i - u_2)^2 \frac{pi}{w_2} \tag{2. 21}$$

A variância dentro da classe é dada por:

$$\sigma_W^2 = \sum_{k=1}^{M} w_k \sigma_k^2 \tag{2. 22}$$

A variância entre as classes é definida como

$$\sigma_B^2 = w_1 (u_1 - u_T)^2 + w_2 (u_2 - u_T)^2 \tag{2. 23}$$

Por fim a variância dos níveis de cinza é dada por:

$$\sigma_T^2 = \sigma_W^2 + \sigma_B^2 \tag{2. 24}$$

Assim o método de Otsu (OTSU, 1979) escolhe o limiar t através da maximização da variância entre as classes, isto é equivalente à minimização da variância dentro das classes, desde que a variância total seja constante para as diferentes partições (LIU; YU, 2009).

$$t = \arg\left\{ \max_{0<t<L-1} \left\{ \sigma_B^2(t) \right\} \right\} = \arg\left\{ \min_{0<t<L-1} \left\{ \sigma_W^2(t) \right\} \right\} \tag{2. 25}$$

2.1.2.2 Limiar adaptativo Gaussiano

Outro método de binarização utilizado neste trabalho foi o proposto por (BRADLEY; ROTH, 2007) onde é usado um filtro de limiar adaptativo Gaussiano. O método baseia-se na ideia de dividir a imagem em várias regiões pequenas, com um tamanho predefinido $(BlockSize \times BlockSize)$, para assim, calcular o limiar em cada uma dessas regiões. Dessa maneira, são obtidos limiares diferentes em diferentes partes da imagem, melhorando o processo de binarização em imagens que tem variações de iluminação (BRADLEY; ROTH, 2007).

O valor de Limiar é definido como o valor da média (*mean*) das intensidades dos píxeis dentro da região subdividida da imagem. A equação (2.26) apresenta o processo de limiarização.

$$image(x,y) = \begin{cases} maxValue & if\ image(x,y) > T(x,y) \\ 0 & otherwise \end{cases} \quad (2.\ 26)$$

Desta forma, para calcular o valor do limiar $T(x,y)$ do pixel (x,y) na imagem, primeiro é definida uma região $(BlockSize \times BlockSize)$ ao redor do pixel examinado. Depois o algoritmo calcula a média ponderada Gaussiana dos valores dos píxeis que estão na região definida. Neste caso, os valores que ficarem próximos ao centro da região terão um maior peso. Assim, o valor da média ponderada é definido como $m(x,y)$.

Por fim para calcular o limiar $T(x,y)$, é subtraído um parâmetro constante C do valor meio ponderado $m(x,y)$ calculado para cada pixel na régio $(BlockSize \times BlockSize)$. Assim, o valor do limiar $T(x,y)$ na localização do pixel (x,y) é dado pela seguinte equação.

$$T(x,y) = m(x,y) - C \quad (2.\ 27)$$

O parâmetro *MaxValue* da equação (2.26) é o valor para o qual o pixel (x,y) da imagem é substituído caso ele seja maior que o valor de limiar da região. Para este caso o máximo valor é de 255.

Por outra parte, os elementos da matriz Gaussiana (de dimensão $BlockSize \times 1$) são obtidos através da seguinte equação:

$$G_i = \alpha e^{-\frac{\left(i-\frac{blockSize-1}{2}\right)^2}{\frac{1}{4\sigma^2}}} \quad (2.\ 28)$$

No trabalho proposto por (BRADLEY; ROTH, 2007) os componentes da função Gaussiana são definidos como: $i = 0,...,blockSize-1$, α é um fator escalar tal que $\sum_i G_i = 1$ e $\sigma = 0.3\left[\left(BlockSize-1\right)0.5-1\right]+0.8$.

2.1.3 Análise de Componentes Conectados *(CCA, connected Component Analysis)*

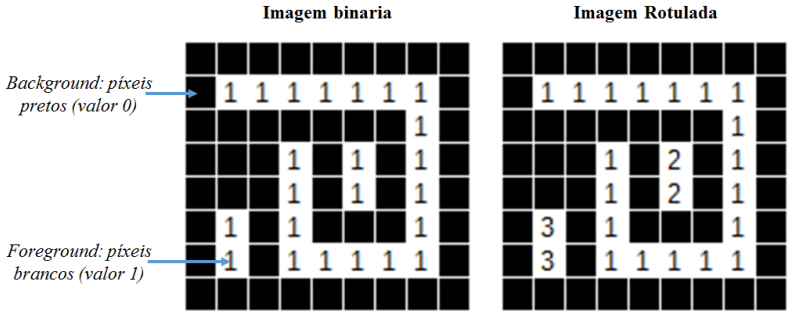
Um tópico fundamental e bastante utilizado no processamento digital de imagens e vídeo é a análise de componentes conectados (CCA, *Connected componente analysis*) (MUKHERJEE, 2014).

Após finalizar o processo de segmentação de uma imagem, feito geralmente através de técnicas de binarização (*Thresholding*), o resultado é uma imagem binaria que possui somente dois valores possíveis para cada pixel: 1(píxeis brancos), que faz referência às regiões que são de interesse e que estão no primeiro plano da imagem (*foreground*) e 0 (píxeis pretos), pertencentes a regiões sem interesse *(background*). Assim, as regiões de interesse formam componentes ou *blobs*, que podem ser analisados e isolados eficientemente por meio de técnicas como *CCA* (KAEHLER; BRADSK, 2016). Esta técnica permite escanear a imagem binarizada e assignar a cada um dos seus componentes um único rotulo através da conectividade dos seus pixeis (conectividade-4 ou conectividade-8). Dessa maneira é gerada uma imagem onde cada pixel com valor 1 (branco) é rotulado com o valor da etiqueta do componente ao qual foi assignado (ANAGNOSTOPOULOS et al., 2008). A Figura 2. 8 ilustra um exemplo do exposto anteriormente. Observe como os píxeis com valor 1 da imagem binarizada são etiquetados segundo a sua conectividade, fazendo com que o algoritmo possa identificar e rotular os três componentes presentes com valores de 1, 2 e 3.

Devido à identificação e rotulagem dos objetos presentes na imagem binarizada, são obtidas informações relevantes como: área, altura, largura e posição. Graças a essas informações podem ser utilizados algoritmos de filtragem para eliminar componentes indesejados. O

anterior faz com que a técnica de CCA possa ser integrada em algoritmos de detecção de placas veiculares (ANAGNOSTOPOULOS et al., 2008).

Figura 2. 8 – Rotulação de componentes conectados. (a) Imagem binarizada e (b) Rotulação dos componentes da imagem



Fonte: Editado de https://www.codeproject.com/Articles/825200/An-Implementation-Of-The-Connected-Component-Label

2.1.4 Transformada de Hough

A Transformada de Hough é um dos métodos mais utilizados para encontrar linhas, círculos ou outras formas geométricas simples em uma imagem. Foi proposta em 1962 por Paul Hough e deu como nome "Transformada de Hough Generalizada". Foi introduzida na visão computacional através do trabalho proposto por (DUDA; HART, 1972) (KAEHLER; BRADSK, 2016). A transformada não é aplicável sobre a imagem original, sendo necessário um pré-processamento (detecção de bordes e uma binarização) (GONZALEZ; WOODS, 2006).

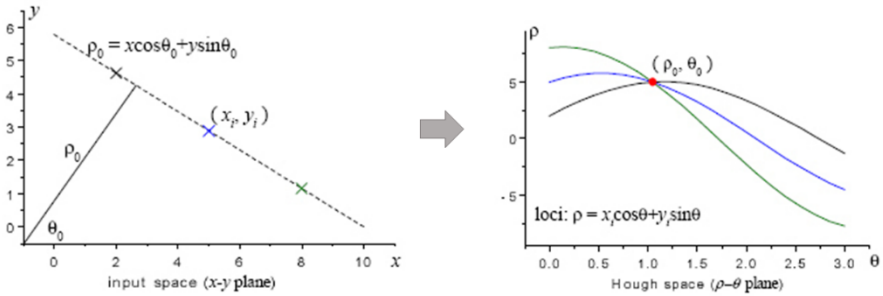
Seja (x, y) a localização em coordenadas cartesianas de um pixel na imagem binarizada, a transformada de Hough converte os pares de coordenadas cartesianas (x, y) em curvas nas coordenadas polares (ρ, θ) ao plano paramétrico apresentado na Figura 2. 9. Aqui, as retas são definidas através da equação (2.29):

$$\rho = x \cdot \cos(\theta) + y \cdot sen(\theta) \qquad (2.\ 29)$$

Utiliza também um vetor auxiliar acumulador que computa a quantidade de vezes que cada combinação de (ρ, θ) aparece na imagem.

Os pontos colineares na imagem binarizada geram picos no plano de Hough e permitem identificar o ângulo (θ) das linhas que formarem esses pontos, ou seja neste casso, o ângulo de inclinação da placa

Figura 2. 9 – Transformação dos pontos colineares de uma imagem binaria (a) ao espaço paramétrico (b).



(a) (b)

Fonte: (LIN; OTOBE, 2001)

2.2 RECONHECIMENTO

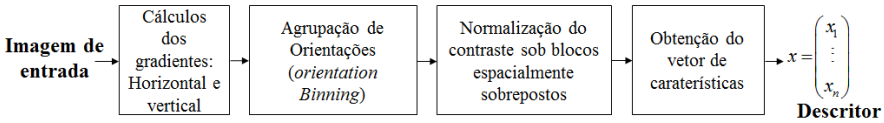
2.2.1 HOG (Histogram Of Oriented Gradients)

Histogram of Oriented Gradients, ou descritor HOG, é um descritor de características proposto inicialmente por (DALAL; TRIGGS, 2005) para a detecção de pedestres, e que devido aos seus bons resultados foi usado posteriormente para atingir diversos problemas de detecção e reconhecimento, tais como reconhecimento de assinaturas (TASKIRAN; CAM, 2017), detecção de sinais de transito (SUGIHARTO; HARJOKO, 2016), detecção de carros (SU et al., 2016), e reconhecimento de placas veiculares (GOU et al., 2016) (WU et al., 2013) (ZHENG et al., 2012).

O método baseia-se na ideia de contar -em histogramas- as ocorrências de orientações do gradiente (direções das bordas) em determinadas partes da imagem. Ao final, os histogramas calculados são chamados de histogramas de gradientes orientados, e formam um vetor único de características HOG que representa à imagem analisada (DALAL; TRIGGS, 2005).

Para calcular esse vetor de características (descritor), os autores propõem uma série de operações descritas na Figura 2. 10 que serão explicadas a seguir.

Figura 2. 10 - Processo de extração de características HOG.



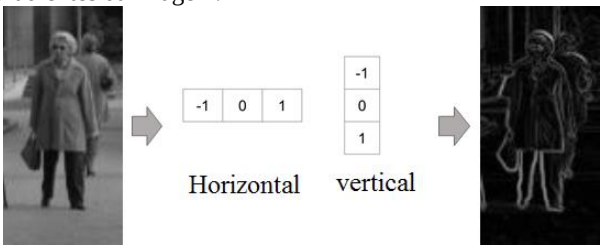
Fonte: Elaborado pelo autor.

2.2.1.1 Calculo do gradiente

A primeira etapa consiste no cálculo do gradiente horizontal (g_x) e vertical (g_y), ou seja, as bordas na imagem de entrada, já que o objetivo é a obtenção de histogramas dos gradientes.

(DALAL; TRIGGS, 2005) notaram através dos testes realizados que com os esquemas mais simples para o cálculo do gradiente (máscaras unidimensionais), eram obtidos melhores resultados. Nesse sentido, para obter o gradiente, basta com filtrar a imagem com as máscaras ilustradas na Figura 2. 11.

Figura 2. 11 - Calculo dos gradientes numa imagem. (a) Imagem original em tons de cinza. (b) Gradientes da imagem.



Fonte: Editado de (VALVENY, 2015)

Depois, os valores resultantes do cálculo dos gradientes (horizontal e vertical) são usados para calcular a magnitude e orientação do gradiente em cada um dos píxeis da imagem, através das seguintes equações:

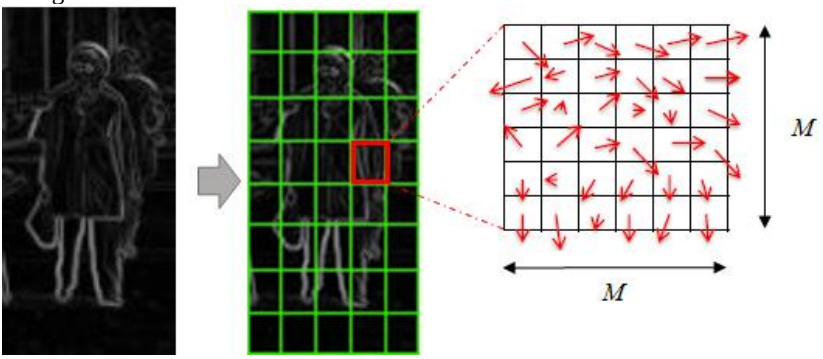
$$g = \sqrt{g_x^2 + g_x^2} \tag{2. 30}$$

$$\theta = \arctan\left(\frac{g_y}{g_x}\right) \tag{2. 31}$$

Onde g representa a magnitude total dos gradientes g_x (horizontal) e g_y (vertical), e θ a direção do ângulo do gradiente respeito ao eixo x. Desde esse ponto de vista, observe que na imagem da Figura 2. 11 (a) que tem um tamanho $A \times L$, o gradiente é calculado para cada um dos píxeis da imagem, tendo como resultado dois vetores de tamanho $A \times L$ onde o primeiro é usado para armazenar o gradiente e o segundo para armazenar o ângulo relacionado a este gradiente (Figura 2. 12 (a)).

Posteriormente a imagem é dividida em células com um tamanho fixo $M \times M$ [2] (Figura 2. 12 (b)) e para cada uma das células calcula-se um histograma das orientações dos gradientes (Figura 2. 12 (c)). Na Figura 2. 12 ilustra-se este processo.

Figura 2. 12 - Cálculo do histograma de orientações do gradiente. (a) Gradiente da imagem. (b) Divisão da imagem em células com tamanho fixo e (c) cálculo do histograma.



(a) (b) (c)

Fonte: Elaborado pelo Autor

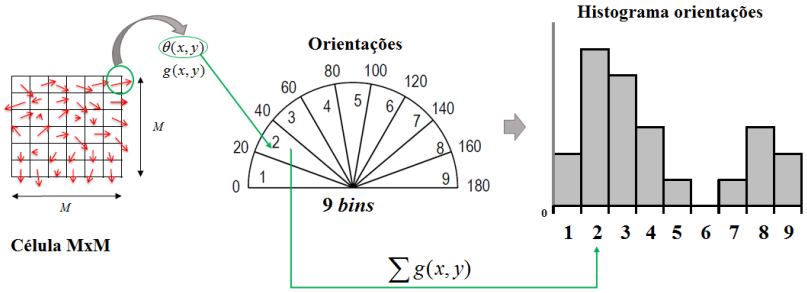
[2] Geralmente usam-se células com tamanho 6x6 ou 8x8 (DALAL; TRIGGS, 2005).

2.2.1.2 Cálculo do histograma das orientações.

A célula ilustrada na Figura 2. 12 (c) faz referência aos gradientes nessa região da imagem. Assim, o sentido das flechas representa a direção do gradiente, enquanto que sua longitude denota a magnitude do gradiente (MALLICK, 2016). Dessa maneira, o histograma resultante irá a representar as distribuições das orientações (direções) do gradiente nessa célula.

Para poder criar o histograma, primeiro são divididas as orientações num número de intervalos fixos, chamados de *bins*. Esses intervalos podem tomar valores de $0^\circ - 180^\circ$ (gradientes sem signo) ou de $0^\circ - 360^\circ$ (gradientes com signo). Geralmente são usados os valores de gradientes sem signo já que um gradiente negativo é considerado o mesmo que seu oposto positivo e ambos os dois irão ficar no mesmo intervalo (DALAL; TRIGGS, 2005). Nesse contexto o número de *bins* que apresentou melhores resultados para os autores foi de 9. Consequentemente o histograma de orientações terá 9 divisões correspondente aos ângulos $0, 20, 40, 60, 80, \ldots 180$. Na Figura 2. 13 pode-se observar o planteado anteriormente.

Figura 2. 13 – Cálculo do histograma de orientações



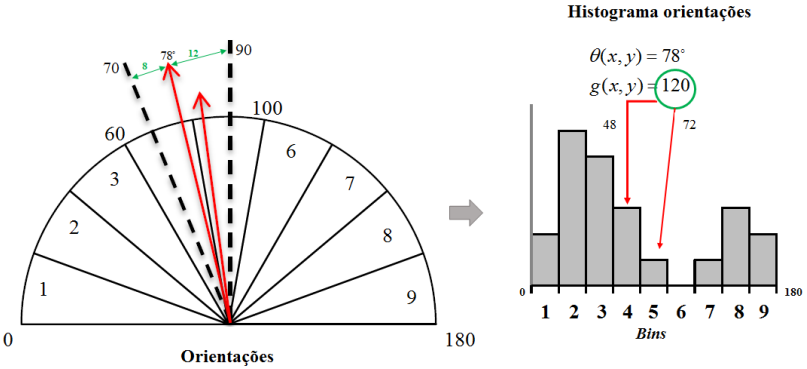
Fonte: Elaborado pelo autor.

Conforme ilustrado na Figura 2. 13, cada gradiente da célula terá uma direção $\theta(x, y)$ e uma magnitude $g(x, y)$. Assim, o gradiente será assignado a um respetivo intervalo em função da sua orientação. Dessa forma, cada intervalo acumula as orientações de todos os gradientes que estejam dentro dos limites definidos. Por fim, a somatória de todas as

magnitudes pertencentes aos gradientes acumulados nesse intervalo é enviada ao histograma de orientações.

A fim de minimizar o *aliasing* que é provocado quando gradientes com orientações similares são atribuídos a intervalos diferentes, o método propõe que os votos (valores atribuídos para cada *bin*) sejam interpolados de forma bi-linear entre os centros vizinhos em ambas orientações e posições. Na Figura 2. 14 demonstra-se um exemplo.

Figura 2. 14 – Interpolação na orientação.



Fonte: Elaborado pelo Autor

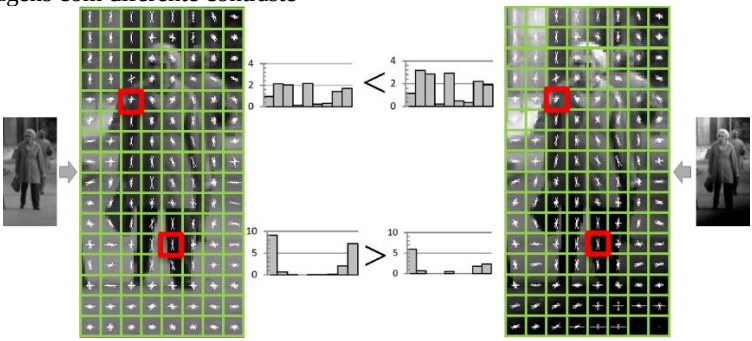
No exemplo da Figura 2. 14 se tem um gradiente com um ângulo de 78° e uma magnitude de 120. Observe que esse ângulo não pode ser representado por nenhum dos centros dos 9 *bins* fazendo com que seja necessário a interpolação da sua magnitude. Nesse contexto, os centros mais próximos são 90 graus para o lado direito e 70 graus do lado esquerdo, note que as distancias para cada um dos centros é 12 graus e 8 graus respetivamente. Desta forma, a magnitude desse gradiente será distribuída da seguinte maneira:

Assigna-se $\frac{12}{20}$ do valor da magnitude total do gradiente, ou seja 72, no *bin* onde encontra-se o centro de 90 graus. Do outro lado é assignado um $\frac{8}{20}$ do valor total do gradiente, ou seja 48, no *bin* onde encontra-se o centro de 70 graus.

2.2.1.3 Normalização e cálculo do descritor.

Como resultado da etapa anterior é calculado um histograma de orientações por cada uma das células definidas na imagem. No entanto, em muitas ocasiões apresentam-se problemas de variações de luz e aumento ou diminuição do contraste na imagem. Esses problemas fazem com que a intensidade do gradiente mude, levando a que o descritor final não seja invariante ante condições de iluminação. Na Figura 2. 15 se demonstra um exemplo. Observe que o cálculo do histograma de orientações da célula superior da segunda imagem é maior que o histograma da primeira imagem na mesma célula, isto é devido a que nesse ponto o contraste é maior na segunda imagem respeito da primeira. Em contrapartida, o histograma da célula inferior da primeira imagem é maior que o da segunda imagem devido à mesma condição do contraste.

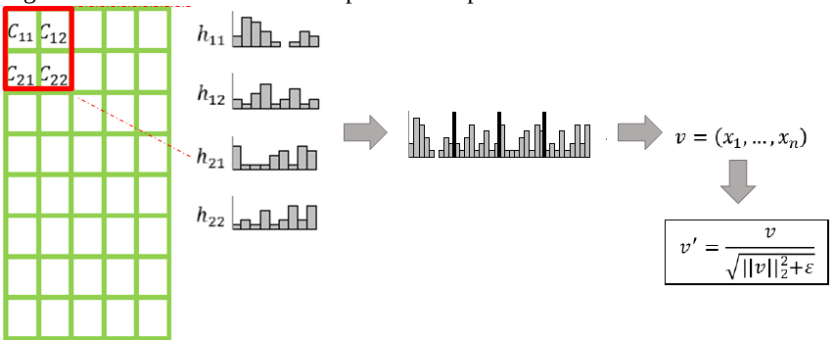
Figura 2. 15 – Necessidade de normalizar. Diferenças dos histogramas de imagens com diferente contraste



Fonte: Editado de (VALVENY, 2015)

Para solucionar esses problemas os autores introduzem o conceito de bloco (*block*), que consiste na agrupação das células em blocos que contem $C \times C$ células. Na Figura 2. 16 pode ser observado um exemplo de bloco na imagem.

Figura 2. 16 – Calculo do vetor representante por cada bloco.



Fonte: Editado de (VALVENY, 2015)

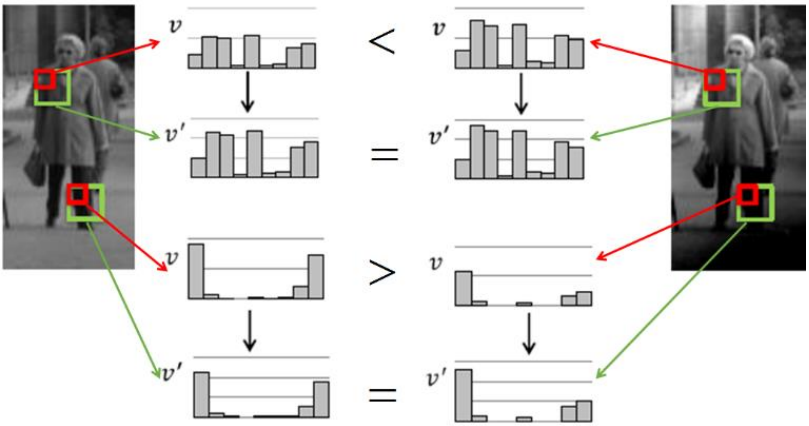
Observe na Figura 2. 16 que para cada bloco é formado um único vetor onde estão concatenados os histogramas das células que contem cada bloco. Esse vetor resultante é normalizado através da Norma L2 descrita na equação (2.32).

$$Norma\ L2 = \frac{v}{\sqrt{\|v\|_2^2 + \varepsilon^2}} \qquad (2.\ 32)$$

Onde v é o descritor de caraterísticas no normalizado. $\|v\|_k$ é a k-norma para $k = 1, 2$ e ε é uma constante pequena.

Na Figura 2. 17 os histogramas originais para cada célula antes do processo de normalização são denotados por v, conforme foi apresentado na Figura 2. 15. Já os histogramas denotados por v' é o resultado de juntar as células nos blocos (em cor verde) e aplicar a normalização. Note como as diferenças entre os histogramas de ambas as duas imagens são reduzidas consideravelmente, fazendo com que os seus valores globais sejam iguais. É desta forma que a normalização contribui a reduzir as diferenças na representação final em imagens similares, permitindo assim, que o descritor final obtido seja mais robusto (VALVENY, 2015).

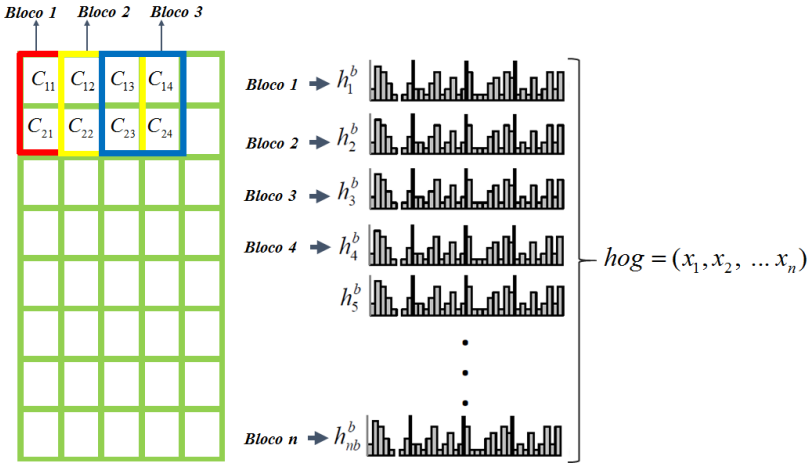
Figura 2. 17 – Resultado do processo de normalização.



Fonte: Editado de (VALVENY, 2015)

Após a normalização, é feito um deslocamento (*stride*) dos blocos conforme ilustra a Figura 2. 18, esse deslocamento serve para que o descritor final seja mais robusto ante deformações e variações da forma do objeto (DALAL; TRIGGS, 2005).

Figura 2. 18– Normalização e cálculo do descritor HOG.



Fonte: Elaborado pelo autor.

Por fim, para calcular o descritor final HOG são concatenadas todas as representações normalizadas dos bloques deslocados. O vetor final é composto por n dimensões. Esse valor de n pode ser calculado usando a equação (2.33).

$$n = Nro\_B \times Nro\_CB \times Nro\_IH \tag{2.33}$$

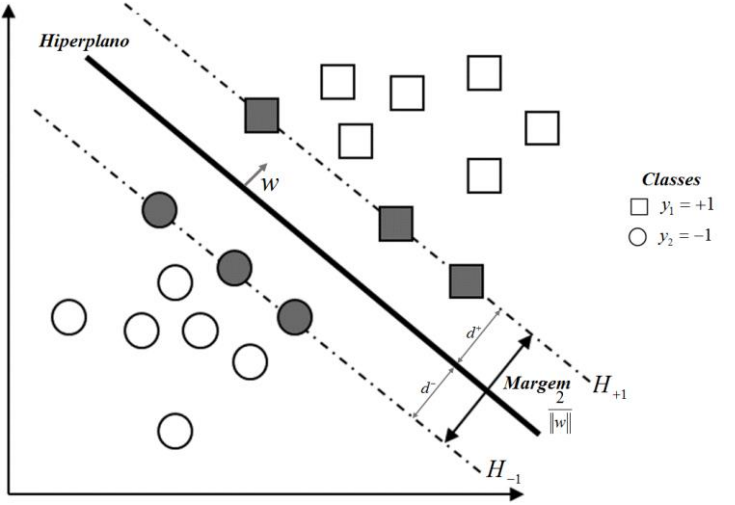
B é o número de blocos que podem ser inseridos na imagem, CB é o número de Células que compõem cada bloque e IH é o número de intervalos do histograma. Finalmente o vetor HOG servirá para descrever as características de um objeto em uma imagem. Posteriormente esse vetor poderá ser treinado através de alguma técnica de classificação para realizar o processo de reconhecimento. Nesse sentido cabe dizer que (DALAL; TRIGGS, 2005) usaram maquinas de vetores de Suporte linear *(SVM, Support vector Machine)* como técnica de classificação.

2.2.2 SVM (Support Vector Machines)

SVM é uma técnica de aprendizado de máquina baseada em aprendizado estatístico supervisionado, usada para resolver problemas relacionados à classificação e regressão (VAPNIK, 1998) (TOBERGTE; CURTIS, 2013). A ideia principal da técnica é que dois conjuntos de dados podem ser separados por um hiperplano ótimo de separação *(OHDR, optimal hyperplane decision rule*) que maximiza a distância entre ele e as amostras mais próximas (chamadas de vetores de suporte) de cada um dos conjuntos analisados, essa distância denomina-se margem *(margin*). Na Figura 2. 19 é ilustrado o planteado anteriormente. Observe que os vetores de suporte, quadrados e círculos de cor cinza, definem a margem de separação máxima entre as duas classes (y_i) rotuladas como $y_1 = +1$ e $y_2 = -1$.

Essa representação pode ser usada no caso em que as classes sejam linearmente separáveis, e é conhecida como uma abordagem com margens rígidas. A seguir serão expostos alguns conceitos teóricos básicos do método para o seu desenvolvimento matemático. Para um estudo mais aprofundado e detalhado podem-se consultar os trabalhos de (VAPNIK, 1998), (BURGES, 1998) e (SMOLA; SCHÖLKOPF, 2004).

Figura 2. 19 - Exemplo de um problema binário linearmente separável em um espaço bidimensional. A margem é delimitada pelos dois hiperplanos H_{+1} e H_{-1}, e o hiperplano separador é definido como $f(x) = w^t x_i + b$. Já os vetores de suporte são representados pelos quadros e círculos de cor cinza.



Fonte: Elaborado pelo autor.

Seja T um conjunto de treinamento composto por n dados do tipo (x_n, y_n), onde $x_n \in X$ e $y_n \in Y$. O vetor x_n pertence ao espaço de características d-dimensional e $Y = \{-1, 1\}$ é o seu respetivo rotulo. No caso em que os dados das classes +1 (positivos) e -1 (negativos) sejam separáveis por um plano, T é considerado linearmente separável. Assim o classificador SVM é considerado como linear. Essa condição de linearidade implica que existe um hiperplano solução que pode ser expressado pela equação (2.34):

$$f(x) = w^t x_i + b \tag{2. 34}$$

Onde w é um vetor ortogonal ao hiperplano solução e b um coeficiente de interseção conhecido como *bias.*

Esse hiperplano solução é o hiperplano meio entre outros dois hiperplanos H_{+1} e H_{-1} que contém os vetores de suporte (quadros e círculos de cor cinza) e são definidos como:

$$\begin{aligned} H_{+1} &\rightarrow w^t x_i + b = +1 \\ H_{-1} &\rightarrow w^t x_i + b = -1 \end{aligned} \tag{2. 35}$$

Onde +1 representa os exemplos positivos e -1 os exemplos negativos. Sequencialmente para encontrar o hiperplano separador que possui a maior margem, as amostras devem satisfazer as seguintes condições:

$$\begin{cases} w^t x_i + b \geq +1, \ \ se\ y_n = +1 \\ w^t x_i + b \leq +1, \ \ se\ y_n = -1 \end{cases} \tag{2. 36}$$

Ambas condições podem-se juntar através da seguinte equação:

$$y_i (w^t x_i + b) \geq 1 \tag{2. 37}$$

Onde y_i corresponde aos rótulos das amostras, +1 para amostras positivas e -1 para amostras negativas.

Com essa condição de classificação assegura-se que as amostras classificadas corretamente estejam localizadas além da margem calculada.

Por outra parte a margem é calculada a partir das distancias d^+ e d^- (veja a Figura 2. 19) dos hiperplanos H. Desta forma:

$$d^+ = d^- = \frac{|wx + b|}{\|w\|} = \frac{1}{\|w\|}, \text{ assim}$$

$$m\arg em = d = d^+ + d^- = 2\frac{1}{\|w\|} \tag{2. 38}$$

Onde $\|w\|$ é a norma do vetor w. Dessa maneira a margem vai depender somente dos parâmetros do hiperplano w.

Agora para encontrar o hiperplano que maximiza a margem, ou seja, maximizar a distância d, o problema pode ser reescrito como um problema de otimização quadrática, onde basta com minimizar a seguinte função:

$$\underset{w,b}{Minimizar}\ \phi(w) = \frac{1}{2}\|w\|^2 \qquad \text{Onde } \|w\|^2 = w^T w \qquad (2.\ 39)$$

Com a restrição

$$y_i(w^t x_i + b) \geq 1,\ \forall i = 1,...,n \qquad (2.\ 40)$$

Para solucionar o problema pode-se fazer uso da formulação Lagrangiana definida como:

$$L(x,\alpha) = f(x) + \sum_i \alpha_i g_i(x) \quad \forall \alpha_i \geq 0 \qquad (2.\ 41)$$

Aqui:

α_i são os multiplicadores de Lagrange com $i = 1,..., N$

$f(x)$ é a função a otimizar, ou seja, $f(x) \rightarrow \phi(w)$,

e $g(x)$ são as restrições, ou seja, $g(x) \rightarrow y_i(w^t x_i + b) - 1 \geq 0$

Dessa forma chega-se à seguinte equação conhecida como forma **primal**:

$$L_p = L(w,b,\alpha_i) = \frac{1}{2} w^T w - \sum_{i=1}^{N} \alpha_i \left[y_i(w^T x_i + b) - 1 \right] \qquad (2.\ 42)$$

Onde:

α_i são os multiplicadores de Lagrange.

A minimização de L_p implica, realizar as derivadas parciais respeito a w, b e igualar a 0, assim:

$$\frac{\partial L_p}{\partial w} = w - \sum_{i=1}^{N} \alpha_i y_i x_i = 0 \quad \Rightarrow w = \sum_{i=1}^{N} \alpha_i y_i x_i \qquad (2.\ 43)$$

$$\frac{\partial L_p}{\partial b} = w - \sum_{i=1}^{N} \alpha_i y_i = 0 \qquad \Rightarrow \sum_{i=1}^{N} \alpha_i y_i = 0 \;\; \alpha_i \geq 0 \qquad (2.\ 44)$$

Agora, substituindo as duas expressões na equação (2.42) que representa a forma primal, se obtém uma nova expressão do Lagrangiano.

$$L(w,b,\alpha_i) = \frac{1}{2}\left(\sum_{i=1}^{N} \alpha_i y_i x_i\right)^T \left(\sum_{j=1}^{N} \alpha_i y_i x_i\right) - \sum_{i=1}^{N} \alpha_i y_i \left(\sum_{j=1}^{N} \alpha_i y_i x_i\right)^T x_i + \sum_{i=1}^{N} \alpha_i$$

$$L_d = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j + \sum_{i=1}^{N} \alpha_i \qquad (2.\ 45)$$

Através da equação (2.45) chega-se a um novo problema de otimização. Esta nova formulação é conhecida como forma **dual** e consiste na maximização de uma nova função $\theta(\alpha)$.

$$\underset{\alpha}{Maximizar}\ \theta(\alpha) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j + \sum_{i=1}^{N} \alpha_i \qquad (2.\ 46)$$

Com as restrições: $\begin{cases} \sum_{i=1}^{N} \alpha_i y_i = 0 \\ \alpha_i \geq 0, \quad \forall i = 1,...,N \end{cases}$

Com o anterior, a solução ótima α^* do problema dual deve satisfazer as condições de Karush-Kuhn-Tucker, que incluem as restrições apresentadas na equação (2.46) , a equação (2.42) e a expressão definida a continuação (BURGES, 1998):

$$\alpha_n \left[y_n \left(w^T x_n + b \right) - 1 \right] = 0 \quad \forall n. \tag{2. 47}$$

Com isso $\alpha_n > 0$ somente para aqueles pontos que estão sobre os hiperplanos H^+ e H^-, ou seja os vetores de suporte. Já para os outros pontos, as condições de Karush-Kuhn-Tucker são respeitadas somente para $\alpha_n = 0$.

Com a solução do problema dual α^* e atendidas as condições, o hiperplano ótimo definido por w^* e b^* é obtido através das equações (2.43) e (2.47). Assim o classificador SVM é definido como:

$$g(x) = sign\left(\sum_{i \in SV} y_i \alpha_i^* x_i^T x + b^* \right) \tag{2. 48}$$

$$b^* = \frac{1}{n_{SV}} \sum_{i \in SV} \left(\frac{1}{y_i} - \sum_{j \in SV} \alpha_j^* y_j x_j^T x_j \right) \tag{2. 49}$$

Onde SV é o conjunto formado pelos vetores de suporte e n_{SV} é a quantidade de elementos do conjunto. Por outra parte b^* é obtido a partir da equação (2.47), computando a média sobre todos os vetores de suporte.

O classificador SVM descrito até agora representa a separação de dados num espaço bidimensional, onde os dados são linearmente separáveis, porem na vida real é muito difícil encontrar dados que sejam linearmente separáveis, devido principalmente a ruídos e *outliers.* Para poder lidar com essa condição de não linearidade uma estratégia utilizada é suavizar a condição da margem e introduzir tolerância aos erros de classificação. Isto é feito através da introdução de umas variáveis de folga (*slack variables*) denotadas como $\xi_i \geq 0$. Estas variáveis são incorporadas na condição expressada pela equação (2.40) da formulação do problema SVM Primal. Dando como resultado,

$$y_i (w^t x_i + b) \geq 1 - \xi_i, \quad onde\ \xi_i \geq 0, \quad \forall_i = 1, ..., N \tag{2. 50}$$

Essa nova condição, que é chamada de SVM com margens suaves *(soft Margins)* suaviza as margens fazendo com que seja permitida a

inclusão de alguns dados entre os Hiperplanos H^+ e H^- , conforme ilustra a Figura 2. 20. Dessa maneira a reformulação da forma primal descrita na equação (2.39) fica da seguinte forma:

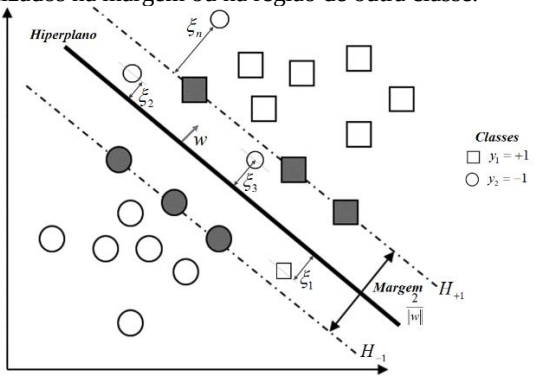
$$\underset{w,b,\xi}{Minimizar}\ \phi(w) = \frac{1}{2} w^T w + C\left(\sum_{i=1}^{N} \xi_i\right) \tag{2. 51}$$

Com a restrição:

$$y_i(w^t x_i + b) - 1 + \xi_n \geq 0 \tag{2. 52}$$

O parâmetro C pode ser interpretado como um parâmetro de regularização que permite controlar o equilíbrio entre maximizar a margem e minimizar o erro nas amostras de treinamento. Assim, quando $C \to \infty$ o parâmetro anula a condição de margem suave e o resultado é a formulação de margens duras (RYCHETSKY, 2001), fazendo com que o problema se resolva da forma standard descrita anteriormente.

Figura 2. 20 – SVM com margem suave. As linhas tracejadas representam os hiperplanos que delimitam a margem. Observe como alguns vetores de suporte podem estar localizados na margem ou na região de outra classe.



Fonte: Elaborado pelo Autor.

Este novo problema de otimização, é solucionado de forma similar ao problema de margens rígidas. Assim, sendo μ_n multiplicadores de

Lagrange, a forma Primal do SVM com margens suaves é expressada como:

$$L_p = \frac{1}{2} w^T w + C\sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i \left[y_i (w^T x_i + b) - 1 + \xi_i \right] - \sum_{i=1}^{N} \mu_n \xi_i \quad (2.53)$$

Com a seguinte restrição:

$$\alpha_i, \mu_n \geq 0 \quad (2.54)$$

Agora realizando as derivadas parciais respeito a w, b, ξ e igualar a 0, obtém-se as seguintes expressões:

$$w = \sum_{i=1}^{N} \alpha_i y_i x_i \quad (2.55)$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \quad (2.56)$$

$$C - \alpha_n - \mu_n = 0 \quad (2.57)$$

Agora, substituindo as anteriores expressões na equação (2.53), obtém-se a forma Dual do SVM:

$$L_d = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j + \sum_{i=1}^{N} \alpha_i \quad (2.58)$$

Com as restrições:
$$\begin{cases} \sum_{i=1}^{N} \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq C, \quad \forall i = 1, ..., N \end{cases} \quad (2.59)$$

Observe que incluir as variáveis de folga na forma dual, implica a inclusão de uma nova inequação descrita na equação (2.59), esta tem como objetivo modular a contribuição dos vetores de suporte com $\alpha_i \neq 0$

Depois, as condições de Karush-Kuhn-Tucker são dadas pelas equações (2.52), (2.54), (2.55), (2.56), (2.57) e as seguintes expressões (BURGES, 1998):

$$\alpha_n \left[y_n \left(w^T x_n + b \right) - 1 + \xi_n \right] = 0 \quad \forall n. \tag{2. 60}$$

$$\mu_n \xi_n = 0 \tag{2. 61}$$

Resolvendo o problema dual α^* pode-se obter o w^* e b^* das equações (2.55) e (2.60) respetivamente. Dessa maneira, ao igual que no caso das margens rígidas, a solução do classificador final é dada por:

$$g(x) = sign\left(\sum_{i \in SV} y_i \alpha_i^* x_i^T x + b^* \right) \tag{2. 62}$$

$$b^* = \frac{1}{n_{SV}} \sum_{i \in SV} \left(\frac{1}{y_i} - \sum_{j \in SV} \alpha_j^* y_j x_j^T x_j \right) \tag{2. 63}$$

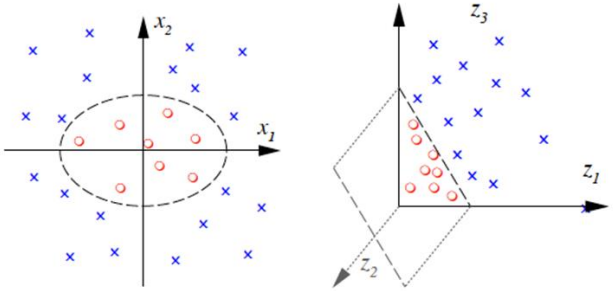
Onde SV é o conjunto formado pelos vetores de suporte que satisfazem a inequação $0 \le \alpha_i \le C$ e n_{SV} é a quantidade de amostras no conjunto. Observe como na solução final da forma dual do classificador não influem de forma explicita nem as variáveis de folga, nem o parâmetro C.

Em muitas ocasiões não é possível dividir o conjunto de dados de treinamento por um hiperplano, já que os dados apresentam uma forma não lineal, conforme se ilustra na Figura 2. 21.

Para esses casos pode ser utilizada a versão não Lineal do SVM que é conhecida como o *truque do Kernel* (AIZERMAN; BRAVERMAN; ROZONOER, 1964). Baseia-se em um mapeamento não linear do conjunto de dados de entrada em um espaço de Hillbert *H* de maior dimensão, onde os dados já poderão ser separados de uma forma lineal através de um hiperplano. Assim, o mapeamento é dado pela seguinte função:

$$\Phi : \mathbb{R}^d \rightarrow H \tag{2. 64}$$

Figura 2. 21 – Forma não linear dos dados. (a) Conjunto de dados em duas dimensões ($\mathbb{R}^2$) separados por uma fronteira não linear. (b) Fronteira linear no espaço de caraterísticas ($\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$).



Fonte: (GEHLER; SCHÖLKOPF, 2009)

Devido a que na formulação Dual do problema (equação (2.58)) somente se tem produtos escalares entre os vetores ($x_i^T x_j$), o *truque do kernel* permite que não seja necessário definir explicitamente a função de mapeio Φ, mas sim pode-se definir uma função *kernel* $K : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ que represente o produto escalar no espaço transformado. Dessa forma o produto escalar estará definido como:

$$K(x_i, x_j) = \Phi^T(x_i)\Phi(x_j) \tag{2. 65}$$

Consequentemente esse *Kernel* escalar poderá ser integrado na solução dual do problema, dando como resultado a seguinte expressão:

$$L_d = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \Phi^T(x_i)\Phi(x_j) + \sum_{i=1}^{N}\alpha_i \tag{2. 66}$$

Com as restrições:

$$\text{Com as restrições: } \begin{cases} \sum_{i=1}^{N} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C, \quad \forall i = 1,...,N \end{cases} \tag{2. 67}$$

Consequentemente a regra de classificação é dada por:

$$g(x) = sign\left(\sum_{i \in SV} y_i \alpha_i^* \Phi^T(x_i)\Phi(x_j) + b^* \right) \tag{2. 68}$$

$$b^* = \frac{1}{n_{SV}} \sum_{i \in SV} \left(\frac{1}{y_i} - \sum_{j \in SV} \alpha_j^* y_j \Phi^T(x_i)\Phi(x_j) \right) \tag{2. 69}$$

Onde SV é o conjunto formado pelos vetores de suporte e n_{SV} é a quantidade de elementos do deste conjunto.

As condições necessárias e suficientes para que uma função simétrica e continua $K(x_i, x_j)$ represente um produto escalar num espaço de HIllbert (*RKHS, reproducing Kernel Hilbert space*) estão dadas pelo teorema de Mercer (MERCER, 1909). Este teorema estabelece que existe uma transformação $\Phi : \mathbb{R}^d \to H$ e uma função de *Kernel* que representa um produto escalar no espaço de caraterísticas associado $K(x_i, x_j) = \Phi^T(x_i)\Phi(x_j)$ se e somente se for cumprido que:

$$\int_x K(x,z)g(x)g(z)dxdz \ge 0 \tag{2. 70}$$

Onde $g(x)$ é uma função qualquer tal que:

$$\int_x g^2(x)dx < \infty \tag{2. 71}$$

A seleção de uma função de *Kernel* adequada depende muito das caraterísticas do problema a ser abordado. Assim, através dos anos foram propostos vários tipos de *kernels* para solucionar diversos tipos de problemas no campo de aprendizado de máquina. Em (SHAWE-TAYLOR; CRISTIANINI, 2004) encontra-se um compendio completo das funções de *Kernel* propostas nos últimos anos. Onde algumas das mais usadas são:

Tabela 2. 1- Tipo de funções Kernel

Tipo de não linearidade	**Função *Kernel***	**Parâmetros**
Polinomial	$K(x_{i,}x_j) = (x_i x_j + 1)^d$	d
Gaussiano	$K(x_{i,}x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$	$\sigma > 0$
Sigmoidal	$K(x_{i,}x_j) = \tanh(kx_i x_j - \delta)$	$K\ e\ \delta$

Fonte: Elaborado pelo Autor

A Função Gaussiana é conhecida também como RBF *(Radial Basis Functions*), onde σ define o comportamento da exponencial. Observe como as funções de *kernel* introduzem alguns parâmetros na sua formulação que devem ser ajustados no processo de treinamento.

O SVM na sua forma original é um classificador binário. Porém, na prática apresentam-se problemas onde tem que determinar-se a classe correta entre k classes, com $k > 2$. Para resolver este inconveniente, existem dois métodos que permitem a extensão do SVM para problemas multiclasse.

O primeiro, chamado de *one-vs-all* proposto por (VAPNIK, 1998). Utiliza *k* classificadores *SVMs* binários de forma independente, um para cada classe. Assim, treina-se o i-ésimo classificador enquanto são rotuladas todas as amostras da i-ésima classe como sendo positivas, e as restantes como negativas (VAPNIK, 1998).

O segundo método, proposto por (KNERR; PERSONNAZ; DREYFUS, 1990) consiste em construir $\frac{k(k+1)}{2}$ classificadores binários, realizando todas as combinações possíveis das k classes, para depois por meio de um mecanismo de votação, assignar à amostra a etiqueta correspondente à classe que haja resultado vencedora em um

maior número de classificadores binários (KNERR; PERSONNAZ; DREYFUS, 1990). Esse método é chamado de *one-vs-one*.

3 TRABALHOS RELACIONADOS

O desenvolvimento e melhora constante das técnicas de visão computacional e inteligência artificial tem permitido que sejam apresentadas muitas propostas de ALPR. Porém, é importante esclarecer que muitas das abordagens encontradas na literatura são aplicadas sobre imagens onde as condições ambientais da cena são favoráveis para o reconhecimento da placa veicular. Com esta premissa, e visando atingir os objetivos planteados neste trabalho, primeiro foram estudados os trabalhos propostos por (KULKARNI et al., 2012), (DU et al., 2013), (AHMAD et al., 2015) e (ALI et al., 2016), para ter uma visão global das propostas mais relevantes presentes na literatura respeito ao ALPR. Dessa forma, logrou-se obter um compendio de trabalhos científicos que serviram de base teórica para o desenvolvimento do método ALPR proposto neste trabalho.

Com base nas três etapas fundamentais que compõem um método ALPR (detecção da placa, segmentação de caracteres e reconhecimento de caracteres) apresentadas na introdução do trabalho, neste capítulo serão expostas e discutidas as principais técnicas que visam a resolução dos problemas presentes em cada uma de essas etapas.

3.1 DETECÇÃO DA PLACA

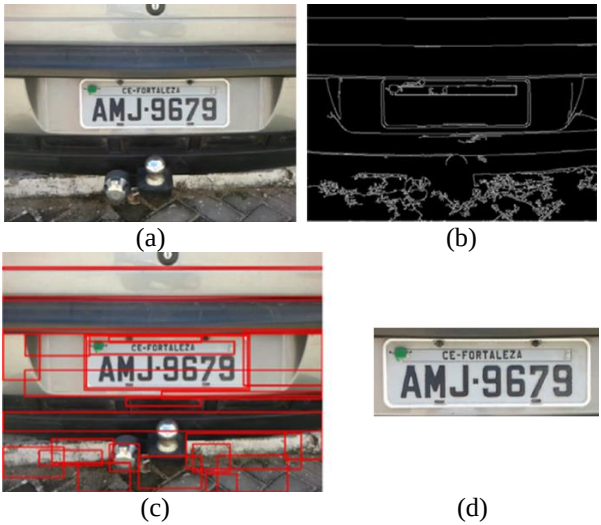
A fase de detecção da placa é uma das mais importantes dentro do sistema ALPR já que influi diretamente na acurácia do sistema (DU et al., 2013). Neste contexto, são muitos os algoritmos propostos para segmentar a placa de uma maneira eficiente.

Com o argumento de que as placas veiculares possuem propriedades geométricas predeterminadas como: uma forma retangular, área e *aspect ratio (*proporção*),* alguns métodos baseiam-se nessas caraterísticas a fim de encontrar todos os possíveis retângulos dentro da imagem de entrada para após serem extraídos através de verificações geométricas.

Trabalhos como os propostos por (NETO et al., 2015), onde usam placas Brasileiras ou (PRABHAKAR; ANUPAMA; RESMI, 2014), utilizam a transformada de Hough (DUDA; HART, 1972) como base para encontrar todos os retângulos presentes na imagem e assim, poder segmentar a região de interesse que contém a placa veicular. Para isso, primeiro é convertida a imagem de cor a escala de cinza, seguidamente é usado o operador Canny (CANNY, 1986) a fim de detectar todas as bordas presentes na imagem. Depois por meio da transformada de Hough

(DUDA; HART, 1972) são identificadas as linhas horizontais e verticais que representam as bordas detectadas, desta maneira formam-se retângulos que ao final serão filtrados tendo em conta aspectos geométricos da placa como: o rádio, o *aspect ratio* e a área, para finalmente segmentar a placa. Na Figura 3. 1 ilustra-se um exemplo do método aplicado.

Figura 3. 1 - Detecção da placa veicular: (a) imagem de entrada, (b) detecção de bordas por Canny, (c) Detecção de retângulos através de Hough e (d) Placa segmentada.



Fonte: (NETO et al., 2015).

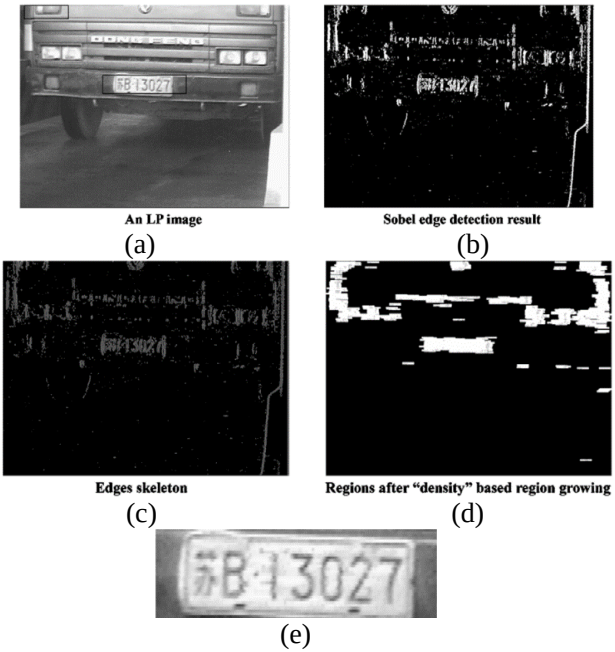
Para testar o método os autores usaram 700 vídeos de onde foram extraídas 12000 imagens, ao final reportam uma acurácia do 97, 02% na etapa de detecção.

Este método não funciona eficientemente quando apresentadas placas com bordas indefinidas, tamanho variável ou baixo contraste, onde as bordas não são facilmente detectáveis. Além disso, a acurácia de aplicar o método também depende de uma captura da imagem a uma distância fixa entre a câmara e o veículo.

Os sistemas propostos por (JIAO; YE; HUANG, 2009) e (ELBAMBY; HEMAYED, 2016) usam o operador de gradiente de Sobel como ferramenta principal para segmentar o retângulo que forma a placa veicular. Aqui, primeiro é suavizada a imagem com um filtro Gaussiano

para reduzir a resposta aos ruídos, depois é usado o operador Sobel (SOBEL, 1990) a fim de salientar as bordas verticais na imagem, em seguida é aplicado um algoritmo de esqueletização para filtrar as bordas indesejadas. Como resultado dessas operações obtém-se uma serie de bordas densas e repetitivas -que representam os alfanuméricos- na região onde encontra-se a placa. Finalmente é usado um método baseado no algoritmo de região de crescimento (YE; GAO; ZENG, 2003) para conectar os píxeis e segmentar a placa. A Figura 3. 2 ilustra o procedimento descrito.

Figura 3. 2 - Localização da placa: (a) imagem de entrada (b) detecção de bordas Sobel (c) esqueletização (d) conexão de píxeis (e) segmentação da placa.



Fonte: Adaptado de (JIAO; YE; HUANG, 2009).

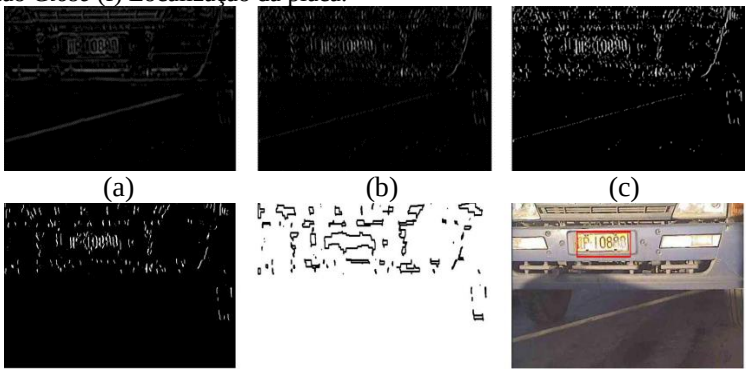
Um limitante deste método é o mau desempenho sob imagens onde a placa apresenta sombra ou condições de baixa resolução, um exemplo disto se apresenta na Figura 3. 3. Aqui, o operador Sobel não encontra as bordas verticais de uma maneira eficiente, principalmente devido ao fato de que o fundo da placa é confundido facilmente com os caracteres.

Figura 3. 3 – Erro na detecção de bordas por Sobel.

Fonte: Elaborado pelo Autor.

No sistema proposto por (GOU et al., 2016) primeiro a imagem de entrada é convertida a escala de cinza, depois utiliza-se a transformada matemático-morfológica *Top Hat* para pre-processar a imagem. Esta transformada funciona como filtro para minimizar ruídos em imagens com fundos complexos e realça detalhes especialmente nas regiões onde encontra-se os textos (GOU et al., 2016). Para os autores a transformada se desempenha eficientemente usando como parâmetro principal um kernel com uma estrutura retangular de tamanho 3x3 píxeis. Posteriormente é usado o operador Sobel da mesma forma que o método proposto por (JIAO; YE; HUANG, 2009) para salientar as bordas verticais na imagem, seguido de uma binarização da imagem com o método de (OTSU, 1979), neste ponto é utilizado o algoritmo proposto por (ZHENG; ZHAO; WANG, 2005) que tem como objetivo remover contornos curvados, com áreas muito grandes, ou aqueles pequenos que encontram-se espalhados pela imagem. O algoritmo é repetido três vezes, para escanear a totalidade da imagem. Finalmente é usada a operação morfológica de fechamento (*close*) para conectar os píxeis que representam o texto presente na placa. Na Figura 3. 4 (e) apresenta-se o resultado desta operação. Neste ponto são obtidos os retângulos delimitadores dos contornos presentes, após utiliza-se um algoritmo baseado na proporção (*aspect ratio*) e a área da placa para filtrar todos os retângulos e assim, segmentar a região de interesse final (Figura 3. 4 (f)).

Figura 3. 4 - Método localização (GOU et al., 2016). (a) Transformada *Top Hat* (b) Sobel (c) Binarização OTSU (d) Removendo curvas e áreas maiores (e) Operação *Close* (f) Localização da placa.



(a) (b) (c)

(d) (e) (f)

Fonte: (GOU et al., 2016).

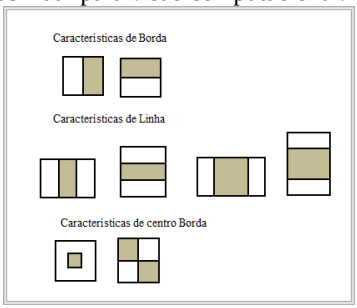
Os autores reportaram uma taxa de acerto do 95.9% sobre uma base de 4242 imagens com tamanhos entre 1936×2592 píxeis e 720×280 píxeis, que foram capturadas em diferentes cenas de monitoramento de trafego urbano da China, ou seja ambientes não controlados.

Os sistemas propostos por (ZHENG et al., 2012) e (CASTELAN, 2009) visam à utilização de classificadores Haar, introduzidos originalmente por (VIOLA; JONES, 2004) para a detecção de faces em imagens, mas também identificam qualquer classe de objeto em condições de fundos complexos independentemente de variações de luz, posição, inclinação e tamanho (ZHENG et al., 2012). Em (ZHENG et al., 2012) primeiro é pre processada a imagem com um filtro da mediana com uma janela deslizante de tamanho 5×5 píxeis a fim de remover os ruídos presentes na imagem. Já o treinamento do classificador proposto é feito com 6500 imagens de placas veiculares (amostras positivas) e 12000 imagens de não-placas (amostras negativas) , enquanto que (CASTELAN, 2009) não reporta o número de amostras utilizadas para o treinamento.

Com base no conjunto para o treinamento, as características Haar são extraídas para cada uma das imagens a treinar, estas características são um conjunto de regiões retangulares especificas que servem como máscara para aplicar um limiar às somas e diferenças de regiões retangulares da imagem (Figura 3. 5). Tais características como

demostrado no trabalho de (PAPAGEORGIOU; POGGIO, 2000) são suficientes para à extração de informações relevantes das formas de um objeto.

Figura 3. 5 – Características Haar para visão computacional.



Fonte: Adaptado de (ZHENG et al., 2012).

Cada característica Haar é considerada um classificador fraco, mas quando somadas em cascata, torna-se num classificador forte (VIOLA; JONES, 2004), com isso, e tendo em conta que numa imagem encontram-se um grande número (milhões) de características Haar, os autores propõem usar um classificador *Gentle AdaBoost* (SCHAPIRE et al., 1998) para selecionar os classificadores fracos mais eficientes e depois treinar o classificador Haar em cascata.

Um classificador em cascata conta com uma serie de estágios (*layers*), que para este caso são os classificadores criados pelo algoritmo *Gentle Adaboost*, cujo principal objetivo é otimizar o reconhecimento do objeto procurado (a placa veicular). A ideia principal é fazer com que seus estágios iniciais descartarem um grande número de regiões que não contém o objeto desejado, assim os estágios posteriores serão cada vez mais precisos no reconhecimento, ao final a área da imagem que passe pelo último estágio da cascata será considerada como uma área que contém o objeto desejado (PAPAGEORGIOU; POGGIO, 2000). Para o sistema proposto por (ZHENG et al., 2012), foram escolhidos 18 estágios com uma taxa de detecção de 99.9% por cada um deles. Os autores reportaram que o treinamento do classificador demorou uma semana.

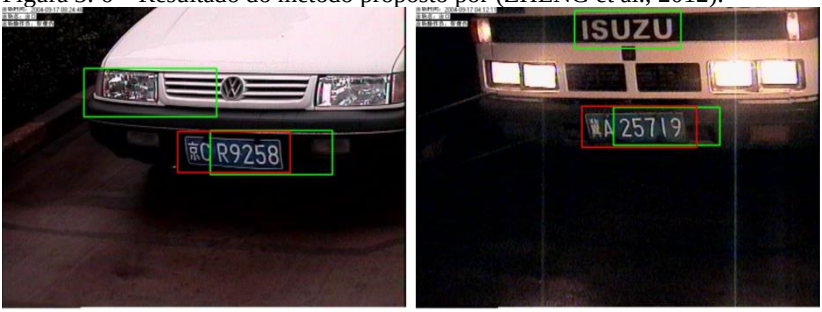
Uma das principais desvantagens dos classificadores baseados nas características Haar é o grande número de falsos positivos que são gerados no momento da detecção do objeto desejado (ZHENG et al., 2012). Nesse contexto, (CASTELAN, 2009) reporta que nas imagens

usadas para o teste, em 50% delas foram detectados falsos positivos, diminuindo assim a acurácia do sistema.

Para resolver o problema, e visando uma maior taxa de acerto final, na proposta de (ZHENG et al., 2012) propõem usar um classificador baseado no descritor HOG, cujo objetivo é estabelecer quais das imagens geradas pelo classificador em cascata contêm placas veiculares.

Para o treinamento deste classificador foram usadas 6000 imagens de placas, que são consideradas amostras positivas, e 9000 imagens de amostras negativas (não placas). Estas imagens foram normalizadas a um tamanho de 152x56 píxeis, após foram extraídas as características HOG de cada uma delas, através dos seguintes parâmetros: *cell size* de 8x8, *block size* de 16x16, e um *step size* de 8. Ao final por cada imagem é obtido um vetor de 3888 características HOG. Usando esses vetores o classificador HOG é treinado novamente por um método *Gentle Adaboost*. Ao final os autores amostram que o método proposto consegue uma taxa de acerto de 94 % sobre um banco de dados de 3000 imagens capturadas em estacionamentos de cidades Chinesas. A Figura 3. 6 ilustra dois exemplos do resultado de aplicar o método baseado nos descritores Haar e HOG. Observe como geram-se falsos positivos nas duas imagens e como o classificador HOG filtra os resultados para identificar a placa enquadrada em vermelho.

Figura 3. 6 – Resultado do método proposto por (ZHENG et al., 2012).



Fonte: (ZHENG et al., 2012).

Como dito anteriormente uma das desvantagens da metodologia proposta pelos autores citados anteriormente, além da complexidade computacional, é a grande quantidade de falsos positivos gerados, pelo que é necessário usar outras ferramentas de classificação para minimizá-los. Outra desvantagem é que para ter um bom resultado no classificador Haar é preciso ter bases de dados com uma quantidade considerável de

exemplos de cada uma das diferentes classes de placas veiculares a reconhecerem.

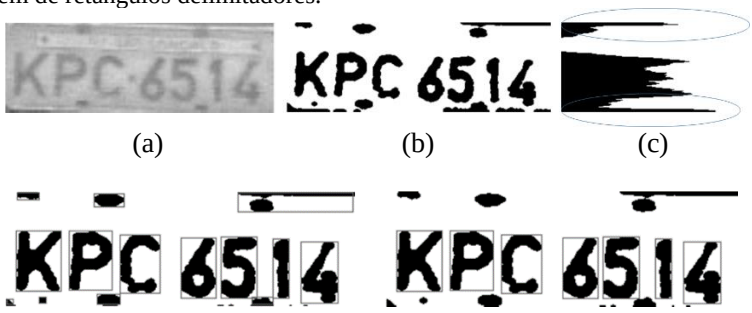
3.2 SEGMENTAÇAO DE CARACTERES

Esta etapa do processo é muito importante dentro de um do ALPR já que, a acurácia da etapa seguinte, a de reconhecimento, depende diretamente de uma boa segmentação dos caracteres.

No trabalho proposto por (CORNETO et al., 2017) -que é baseado em placas Brasileiras-, primeiro são realçadas as bordas da imagem obtida na etapa de detecção por meio de um filtro Gaussiano, depois a imagem resultante é binarizada através do método de OTSU (OTSU, 1979). Na sequência utiliza-se a operação morfológica de fechamento *(close)* para minimizar os ruídos da imagem.

Após o fechamento, os autores usam projeções horizontais para remover as bordas da placa, a Figura 3. 7 (c) ilustra esse processo. Depois detectam-se os contornos finais por meio do método de (SUZUKI; BE, 1985), que é baseado no algoritmo de *border-Following*, onde é criada uma estrutura hierárquica entre os contornos encontrados, definindo os mais externos de uma imagem (CORNETO et al., 2017). Por último são detectados os retângulos delimitadores para cada contorno final. Neste ponto, os retângulos que não possuírem uma altura ou largura suficientes para ser um alfanumérico são rejeitados. Na Figura 3. 7 ilustra-se o método proposto.

Figura 3. 7 – Segmentação dos caracteres. (a) imagem de entrada, (b) OTSU, (c) Projeção horizontal e eliminação de bordas, (d) detecção de contornos e (e) Filtragem de retângulos delimitadores.



Fonte: (CORNETO et al., 2017)

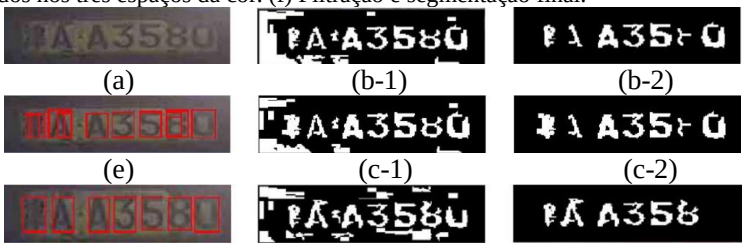
Para este método os autores relatam que obtiveram uma taxa de acerto do 88,71 %, contudo não expõem o número de imagens que utilizaram para realizar o teste, mas sim descrevem que a sua base de dados para treinamento e teste consta de 1410 imagens.

O sistema proposto por (GOU et al., 2016) e Li et al. (2012), usam o detector MSER (*maximally stable region*) (MATAS et al., 2002) para extrair os caracteres diretamente da placa segmentada.

Dado que o MSER pode identificar regiões locais quase invariantes a variações de iluminação e rotação, e devido a que em geral, existe um contraste marcante entre o fundo da placa e os caracteres impressos nela, torna-se possível usar o detector. Porém, nem sempre as placas apresentam condições ideais de contraste fazendo com que o detector não seja eficaz na detecção dos caracteres. Para resolver este problema os autores propõem trabalhar separadamente nos três canais da cor (R-G-B) da imagem.

As regiões obtidas são filtradas tendo em conta a relação geométrica que possuem os caracteres nos tipos de placa usadas pelos autores (placas chinesas). Ao final são segmentados todos alfanuméricos para serem enviados à etapa do reconhecimento. A Figura 3. 8 mostra o resultado obtido ao aplicar MSER.

Figura 3. 8 – Detecção de ERs (*Extremal Regions*) para cada canal da imagem e segmentação de caracteres. (a) imagem original (RGB). (b-1) Binarização do canal B (*azul*). (b-2) Filtração em B. (c-1) Binarização do canal G (*verde*). (c-2) Filtração em G. (d-1) Binarização do canal R (*vermelho*). (d-2) Filtração em R. (e) Retângulos detectados nos três espaços da cor. (f) Filtração e segmentação final.



(a) (b-1) (b-2)

(e) (c-1) (c-2)

(f) (d-1) (d-2)

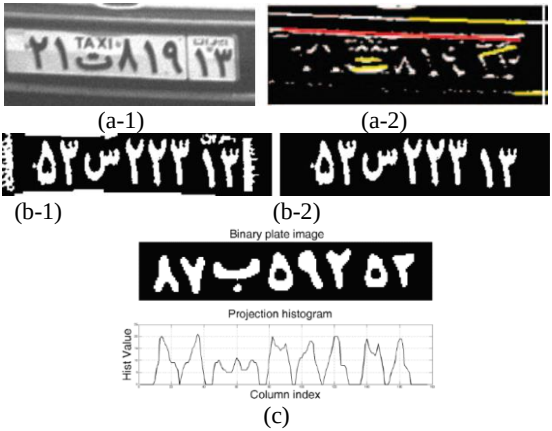
Fonte: (GOU et al., 2016).

O método proposto foi testado sobre 4030 imagens produto da etapa de detecção da placa, atingindo uma taxa de acerto de 98.2%[3].

O estudo de (NEJATI; MAJIDI; JALALAT, 2015) visa a segmentação dos caracteres propondo um método baseado no análises de componentes conectados *(CCA)* e as projeções horizontais e verticais da imagem. Para isso primeiramente é feita uma correção do ângulo da imagem através da transformada de Hough (DUDA; HART, 1972), em seguida a imagem resultante é binarizada com um método local adaptativo Gaussiano (BRADLEY; ROTH, 2007) que, segundo os autores, entregou melhores resultados em termos da acurácia na segmentação.

Produto da binarização são geradas algumas regiões consideradas como ruído na imagem. Para remove-las os autores utilizam o *CCA* levando em consideração características dos componentes como: Área, largura, altura e *aspect ratio.* Finalmente os componentes resultantes correspondentes aos caracteres da placa são segmentados utilizando a projeção vertical da imagem. O resultado deste método é apresentado na Figura 3. 9.

Figura 3. 9 – Processo segmentação de caracteres proposto por (NEJATI; MAJIDI; JALALAT, 2015). (a-1) imagem original. (a-2) Resultado Transformada Hough. (b-1) Imagem binarizada (b2) CCA. (c) Projeção vertical da imagem.



(a-1) (a-2)

(b-1) (b-2)

(c)

Fonte: (NEJATI; MAJIDI; JALALAT, 2015)

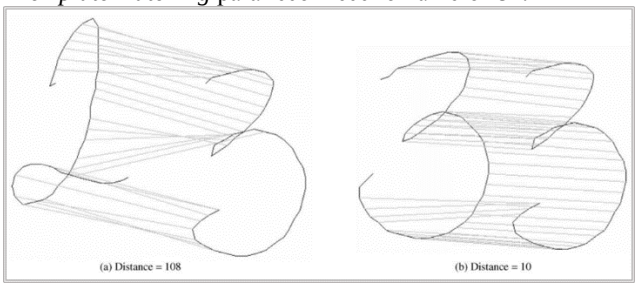
[3] Esse valor foi obtido considerando a etapa de segmentação e a etapa de reconhecimento. Nesse contexto, cabe dizer que na literatura são poucos os trabalhos que reportam à acurácia só na etapa de segmentação dos caracteres.

A taxa de acerto reportada pelos autores é de 92.45% sobre uma base de dados de 2032 imagens iranianas com um tamanho de 1280×960 píxeis e capturadas em diferentes horas do dia em ambientes não controlados.

3.3 RECONHECIMENTO DOS CARACTERES

Uma das técnicas mais simples para o reconhecimento de caracteres é a de *Template matching*. Esta consiste em comparar os caracteres extraídos da placa com um *template* (ou modelo ideal de um caractere), desta maneira, o modelo que tiver maior correspondência com o *template* será escolhido como o caractere correspondente. Na Figura 3. 10 pode-se observar um exemplo. Aqui, o *template* é o número "3" e as entradas são dois caracteres (2 e 3). O caractere é reconhecido como "3" já que a distância entre os pontos de correspondência entre os dois caracteres "3" é menor, quanto a distância entres os pontos de correspondência entre "2" e "3".

Figura 3. 10– *Template matching* para reconhecer o número "3".



Fonte: (CONNELL; JAIN, 2001).

Na proposta de (SARFRAZ; AHMED; GHAZI, 2003) os *templates* são normalizados a um tamanho de 40x40 píxeis e armazenados numa base de dados. Após serem normalizados ao mesmo tamanho dos *templates* (40x40 pixeis), os caracteres extraídos são comparados com os modelos da base de dados através do método de *Hamming distance*. Neste método são encontradas as não correspondências entre o caractere de entrada e todos os *templates* da base de dados, assim, o caractere que tiver o menor valor de não correspondência é tomado como o caractere reconhecido (SARFRAZ; AHMED; GHAZI, 2003). Ao final são

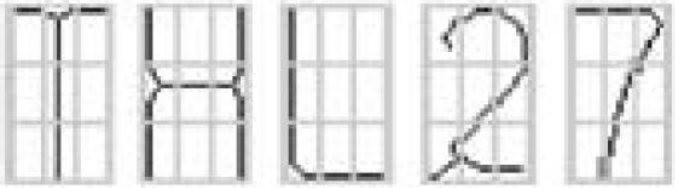
reconhecidas um total de 581 placas sobre um total de 610, tendo como taxa de acerto 95.24%.

É importante aclarar que a efetividade do método depende de que as placas não contenham ruídos indesejados, caracteres quebrados, com tamanhos diferentes ou algum ângulo de inclinação (DU et al., 2013).

Nos últimos anos uma das alternativas para o reconhecimento de caracteres mais usadas por vários autores é a de extração de características acompanhada do uso de classificadores (DU et al., 2013). O método fundamenta-se na extração de algumas características para ajudar a identificar o caractere que está sendo analisado. Essas características são armazenadas num vetor (chamado vetor de características), a fim de serem comparadas com vetores característicos previamente armazenados para medir a sua similaridade.

Em (ANGELINE et al., 2012) primeiro é feita a esqueletização de cada caractere de entrada através do algoritmo de *Thinning* (ZHANG; SUEN, 1984), depois a imagem que contém o caractere é subdividida em 9 partes iguais, conforme amostra a Figura 3. 11. Cada uma destas partes pode conter ou não uma linha que é categorizada entre 4 tipos de linhas: horizontal, vertical, diagonal direita ou diagonal esquerda. Essa informação é usada para gerar o vetor de características que terá um tamanho de 56. Este vetor é a entrada para uma rede neural multi-camada *feed-forward back-propragation* com 56 entradas e 34 neurônios em sua camada de saída para identificar as letras. Além disso, é usada uma camada escondida com 10 neurônios. A taxa de reconhecimento obtida pelos autores é de 97.14%.

Figura 3. 11 – *Thinning* de caracteres e subdivisão das imagens que contém os caracteres.



Fonte: (ANGELINE et al., 2012)

No trabalho proposto por (GOU et al., 2014), utiliza-se o descritor de características HOG para o reconhecimento dos caracteres. Aqui, os autores coletaram manualmente 1358 caracteres contidos na sua própria

basede dados para serem treinados através de uma máquina de aprendizado extremo (ELM, *extreme learning machine*) (HUANG; ZHU; SIEW, 2006). Na Figura 3. 12 ilustra-se alguns dos exemplos coletados para o treinamento.

Figura 3. 12– Exemplos de caracteres para treinamento HOG.

Fonte: (GOU et al., 2014)

Para extrair as características HOG de cada um dos caracteres, os autores configuram os parâmetros de *cell size*, *block size* e *step size* como 10x10, 20x20 e 5x5 respetivamente. Com isso, obtém-se um vetor de 180 dimensões representativo para cada caractere. Depois, cada um dos vetores será uma entrada para treinar o classificador ELM, onde são usados 1100 nós na camada escondida e 65 neurônios na camada de saída. Os experimentos realizados pelos autores foram feitos sobre 2709 caracteres contidos em um total de 387 placas veiculares. Aqui foram reconhecidos um total de 2652 caracteres de forma correta, o que representa uma acurácia de 97,90%.

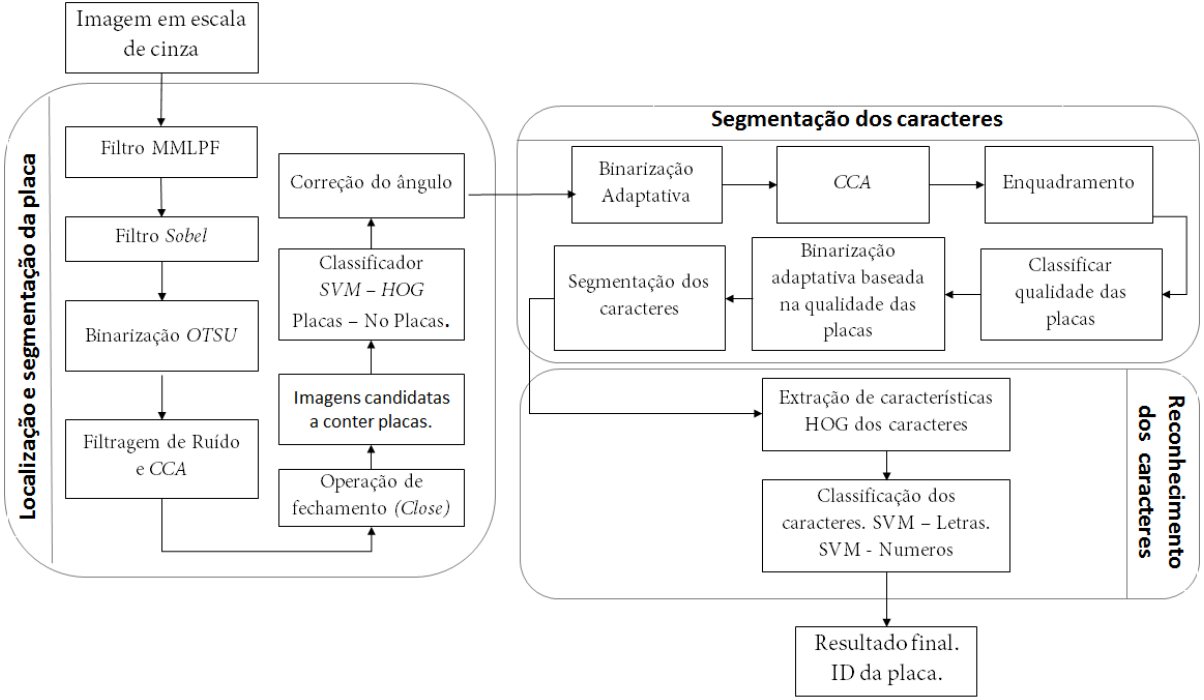
4 PROPOSTA DE UM METODO ALPR BASEADO EM HOG E SVM.

Conforme foi mencionado na introdução do trabalho, o processo de reconhecimento de placas veiculares é composto por três etapas fundamentais: a primeira, localização e segmentação da placa, cujo objetivo é a localização e segmentação da placa na imagem de entrada. A segunda, chamada de segmentação dos caracteres, onde cada caractere é separado, segmentado e enviado à última etapa denominada reconhecimento dos caracteres, baseada em técnicas de OCR (*optical character recognition*). Nesse contexto, este capítulo visa o desenvolvimento de um método ALPR, objetivo geral deste estudo, atendendo as problemáticas relacionadas às imagens que são adquiridas em ambientes não controlados.

A Figura 4. 1 ilustra o diagrama com os processos e técnicas utilizadas para atingir os problemas apresentados em cada uma das três etapas. Assim, na primeira etapa é feito um pré-processamento da imagem de entrada, seguido de uma série de operações morfológicas e a utilização de um sistema de classificação SVM-HOG. O resultado dessa etapa, é a entrada para o modulo de segmentação dos caracteres, onde é realizado um enquadramento da imagem de entrada, baseando-se na transformada de Hough (DUDA; HART, 1972), a fim de isolar a ROI que contem só os caracteres da placa veicular. Posteriormente, a placa resultante é classificada de acordo aos valores de níveis de cinza da imagem analisada. Esses valores são usados para binarizar a imagem através do método de binarização adaptativa Gaussiana (BRADLEY; ROTH, 2007). Por fim, os componentes resultantes são filtrados através de um CCA para poder realizar a segmentação final dos caracteres.

Na última etapa de reconhecimento dos caracteres, são extraídas as características HOG (*histogram oriented gradients*) de cada um dos caracteres segmentados, dessa maneira é formado um vetor que representa o caractere de forma única. Logo o sistema verifica a classe a que pertence cada caractere através de dois classificadores SVM – HOG multiclasse treinados previamente.

Figura 4. 1 – Diagrama do método ALPR proposto.



Fonte: Elaborado pelo Autor.

Nas seguintes seções serão abordados cada um dos módulos apresentados na Figura 4. 1. A fim de ter uma visão detalhada do método proposto.

4.1 ANALISE BASE DE DADOS

Antes de serem explicados os processos efetuados para o desenvolvimento do método ALPR proposto neste trabalho, é importante fazer uma análise das imagens contidas na base de dados utilizada, para assim ter uma visão global dos problemas que terá que resolver o algoritmo a desenvolver.

O treinamento, testes e verificações das técnicas propostas foram feitas sobre uma base de dados que consta de 1970 imagens. Estas foram capturadas com uma câmara monocromática com iluminação instantânea infravermelha a fim de não causar ofuscamento ao motorista (luz infravermelha não é visível ao olho humano). Este tipo de iluminação faz com que todas as imagens geradas sejam em escala de cinza (BERNARDI, 2015). Devido a que as condições do ambiente são variantes em cenários não controlados[4] abertos (neste caso uma rodovia), as imagens capturadas apresentam algumas características importantes que devem ser levadas em conta. Alguns exemplos são:

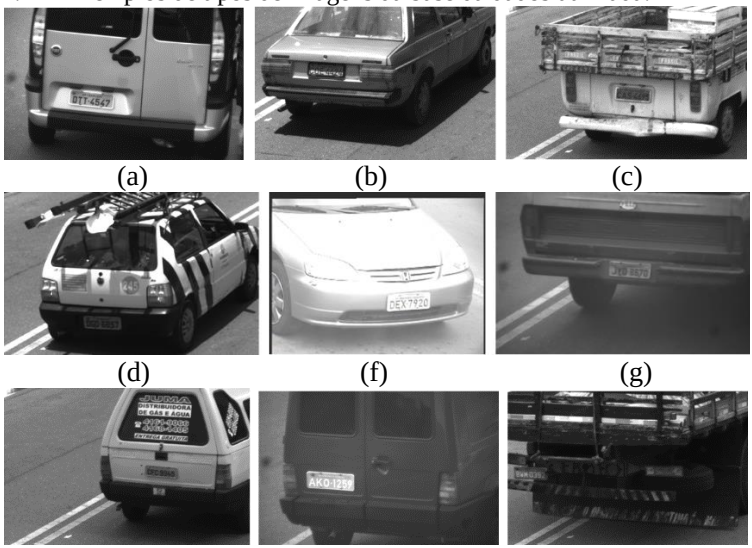
- Imagens em escala de cinza.
- Cenas com tamanho variante de 752x480 e 640x480 pixeis.
- Imagens com vista frontal e traseira dos veículos.
- Distribuições de luz tanto homogênea quanto heterogênea nas imagens.
- Variação da distância da câmera até o veículo.
- Ângulos variantes das placas presentes nas imagens.
- Quantidade excessiva de texto ou desenhos que dificultam o reconhecimento.
- Tipos de placas diferentes: Caracteres escuros sob fundos claros, caracteres claros sob fundos escuros.

A Figura 4. 2 ilustra alguns tipos de imagens da base de dados utilizada para o desenvolvimento do sistema. Note que os problemas

[4] Cenário não controlado para o contexto do presente trabalho refere-se a cenários onde as condições do ambiente não são controladas, por exemplo: a iluminação, o fundo, o contraste, o posicionamento da câmera, etc.

quando apresentadas imagens em cenários naturais aumentam notavelmente.

Figura 4. 2 – Exemplos de tipos de imagens da base da dados utilizada.



Fonte: Elaborado pelo autor.

Feito a análise da base de dados utilizada para o desenvolvimento do trabalho, nas seguintes sessões serão apresentadas as técnicas utilizadas e desenvolvidas para resolver o problema de detecção e reconhecimento de placas veiculares, objetivo geral deste estudo

4.2 DETECÇÃO E SEGMENTAÇAO DA PLACA VEICULAR

Conforme foi mencionado na seção 3.1, nesta primeira etapa, a entrada é uma imagem contendo um veículo, e a saída esperada é uma porção de imagem contendo a placa que identifica o veículo. Na sequência são apresentados os processos utilizados para detectar e segmentar a placa.

4.2.1 Pré-processamento da imagem

Dentro de um sistema de visão artificial, a etapa de pré-processamento da imagem tem como principal objetivo melhorar a

qualidade da imagem. De acordo aos objetivos propostos no sistema, utilizam-se técnicas para diminuir ou aumentar algumas características das imagens (SILVA et al., 2015).

Segundo a análise feita nas imagens presentes na base de dados, concluiu-se que o pré-processamento da imagem devia atingir problemas relacionados aos ruídos presentes nas imagens, bem como suavizar a textura dos fundos, normalizar a luminosidade, além de aumentar o contraste entre o fundo e as áreas de texto.

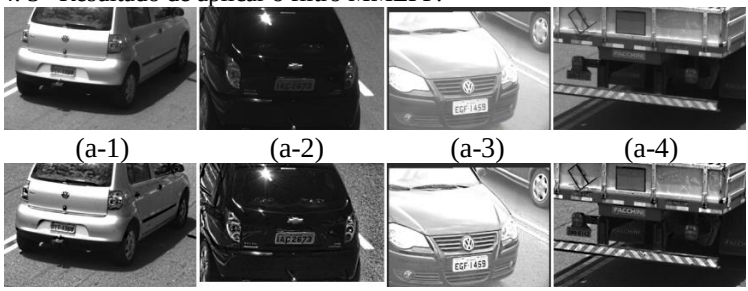
Para solucionar os problemas descritos anteriormente, neste trabalho optou-se por usar o filtro proposto por (WANG et al., 2012). Este filtro fundamenta-se nas operações morfológicas de *bottom-hat* e *top-hat* descritas na seção 2.1.1.3.3. Os autores definem o filtro como um filtro Matemático Morfológico passa baixa (MMLPF, *mathematic morphologic low-pass filter*), e tem como funções principais: incrementar o contraste na imagem sem importar a variação de luz presente, suavizar a imagem e ser robusto a sombras que dificultem o reconhecimento da placa. O filtro é definido na equação (4.1).

$$MMLPF(f) = f + THT(f) - BHT(f) \tag{4. 1}$$

Aqui, f representa a imagem de entrada em escala de cinza. THT simboliza a transformada morfológica *top-hat* e BHT a transformada morfológica *bottom-hat*. A Figura 4. 3 ilustra alguns dos resultados ao aplicar o filtro sobre imagens com diferentes tipos de problemas. Observe como na Figura 4. 3 (b-1) o filtro atua sob a sombra projetada na região de interesse (a placa veicular), diminuindo notavelmente o efeito desta sobre a placa, e fazendo com que os caracteres fiquem mais notórios.

Nas Figura 4. 3 (a-2) e (a-4) apresentam-se imagens onde o contraste entre os caracteres e o fundo da placa é muito baixo, impossibilitando assim a sua posterior localização. Para solucionar este problema utiliza-se o filtro com um elemento estruturante retangular de tamanho 5x5 a fim de serem realçados os caracteres sobre o fundo escuro. Por último na Figura 4. 3 (a-3) ilustra-se uma imagem obtida em condições de iluminação não uniforme, pode-se observar como o filtro através do mesmo elemento estruturante retangular realça os caracteres e faz que o fundo da placa permaneça na mesma tonalidade.

Figura 4. 3– Resultado de aplicar o filtro MMLPF.



(a-1) (a-2) (a-3) (a-4)

(b-1) (b-2) (b-3) (b-4)

Fonte: Elaborado pelo Autor

4.2.2 Localização de possíveis placas veiculares

Esta fase do processo tem como objetivo principal identificar potenciais regiões da imagem pre-processada a serem consideradas placas veiculares. Para isso, propõe-se a utilização de uma série de técnicas que são detalhadas a seguir:

4.2.2.1 Binarização da imagem

Uma das vantagens de aplicar o filtro MMLPF é o aumento significativo do contraste entre o fundo da placa e os caracteres presentes nela. Logo, esse aumento permite que possa ser aproveitada a característica de que as placas estão compostas majoritariamente por bordas verticais. Desta maneira, a detecção destas bordas permite diferenciar as áreas dentro da imagem onde pode estar localizada a placa veicular.

Para encontrar as bordas em uma imagem existem algoritmos bastante estudados e utilizados na literatura como o proposto por (ROBERTS, 1965), Sobel (SOBEL, 1990), Prewitt (PREWITT, 1970), ou Canny (CANNY, 1986). Neste trabalho propõe-se usar o detector de bordas de Sobel (SOBEL, 1990) já que apresenta boa supressão do ruído e pode ser aplicado tanto na direção horizontal quanto na direção vertical (BARBOSA; SILVA, 2009).

Antes de aplicar o detector, a imagem de entrada é suavizada com um filtro Gaussiano de tamanho 5 x 5, com o objetivo de minimizar possíveis ruídos, além de ajudar a gerar bordas verticais mais fortes no resultado final. Na sequência, aplica-se o operador vertical de Sobel com

um kernel de tamanho 3 x 3 ilustrado na Figura 4. 4, o tamanho do Kernel foi determinado experimentalmente.

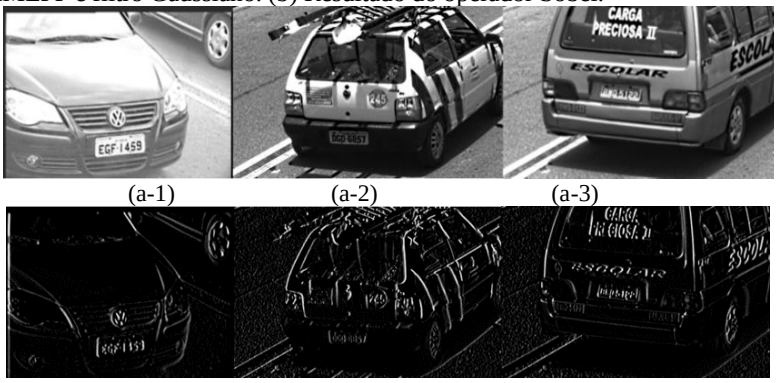
Figura 4. 4 - Kernel para encontrar bordas verticais.

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

Fonte: Elaborado pelo autor.

A Figura 4. 5 ilustra o resultado de aplicar o filtro Sobel (SOBEL, 1990). Observe a alta densidade de bordas verticais na região onde localizada a placa veicular.

Figura 4. 5 – Resultado de aplicar o operador Sobel. (a) Imagens com filtro MMLPF e filtro Gaussiano. (b) Resultado do operador Sobel.



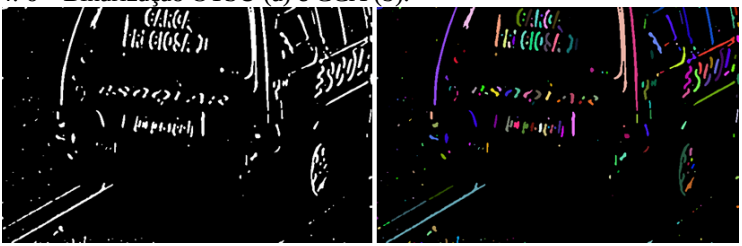
(a-1) (a-2) (a-3)

(b-1) (b-2) (b-3)

Fonte: Elaborado pelo autor.

Após aplicar o filtro, é utilizado um filtro da média com um tamanho de 5 píxeis sobre a imagem do veículo, com o objetivo de remover pequenos ruídos gerados (*salt-and-pepper noise*).
Na sequência, a imagem resultante é binarizada com o objetivo de separar as regiões de interesse (as bordas) do fundo. Para isso, utiliza-se, similar ao trabalho de (GOU et al., 2014), o método de Otsu (OTSU, 1979), onde é detectado automaticamente o limiar que deve ser empregado. Contudo, a imagem obtida apresenta bordas fora da região de interesse, além de pontos espalhados que atrapalham a identificação da placa veicular. A Figura 4. 6 (a) apresenta o resultado da binarização da imagem Figura 4. 5 (b-3).

Figura 4. 6 – Binarização OTSU (a) e CCA (b).



(a) (b)

Fonte: Elaborado pelo autor.

Algumas das bordas obtidas na binarização da imagem são diagonais. Outras, apresentam alturas ou larguras maiores se comparadas com as bordas que representam os caracteres da placa. Para que possam ser eliminadas essas bordas propõe-se utilizar a técnica de análises de componentes conectados (*CCA, Connected Component Analysis*) (Figura 4. 6 (b)), conforme apresentado nos trabalhos de (DALIDA et al., 2016) , (NEJATI; MAJIDI; JALALAT, 2015) e (WU et al., 2006).

4.2.2.2 Análise dos componentes Conectados (*CCA*)

Partindo de que os píxeis com valor 0 (preto) dentro da imagem são pertencentes a regiões sem interesse e os píxeis com valor 1 (branco) equivalem a regiões de interesse. A análise de componentes, como mencionado na seção 2.1.3, permite etiquetar todas as bordas (componentes) presentes, gerando uma matriz com informação relevante sobre a altura, largura, área e posicionamento de cada um dos componentes dentro da imagem. A Figura 4. 6 (b) ilustra o resultado de aplicar CCA.

Com a informação obtida através do CCA, definem-se premissas para poder filtrar as bordas não desejadas, por tanto, as bordas (componentes) serão eliminadas se:

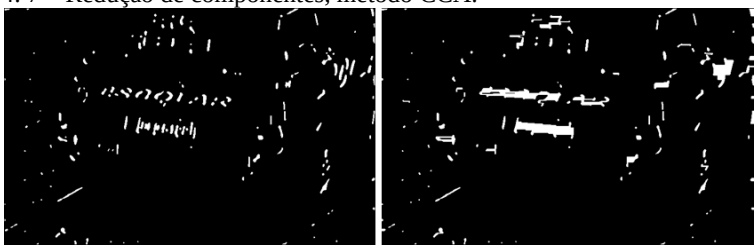
- Sua altura está por fora do rango da altura máxima ou mínima predeterminada no sistema. Neste trabalho a altura máxima é de 41 píxeis e a mínima é de 6 píxeis.
- Sua largura está por fora do rango de largura máxima ou mínima predeterminada no sistema.

- A área do componente está por fora do rango de área máxima ou mínima predeterminada no sistema. Para este trabalho a área máxima é de 3800 e a mínima de 1050 píxeis.

A Figura 4. 7 (a) demonstra que o resultado é uma imagem binarizada com uma redução considerável de componentes. Observe a alta densidade de bordas e a pouca distância horizontal entre elas na área central da imagem, isso pressupõe que trata-se dos caracteres da placa veicular.

Neste ponto é utilizado o operador morfológico de fechamento com um elemento estruturante de linha horizontal de tamanho igual a 17 píxeis, que para o caso deste trabalho é a maior distância possível que pode haver entre os caracteres da placa veicular, desta forma conectam-se os componentes como ilustrado na Figura 4. 7 (b).

Figura 4. 7 – Redução de componentes, método CCA.



(a) (b)

Fonte: Elaborado pelo Autor.

4.2.2.3 Filtragem e detecção de regiões candidatas.

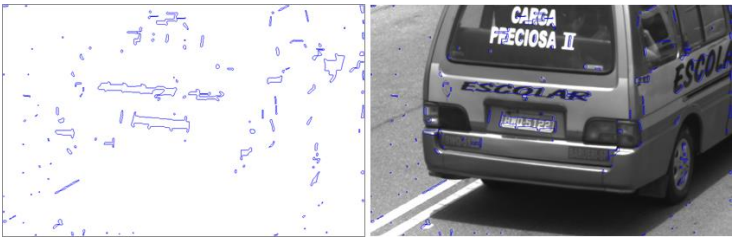
Após o processo de fechamento, a imagem gerada contém regiões onde pode estar localizada a placa, porém, muitas dessas regiões não contêm o objeto desejado. Para poder fazer uma filtragem dessas regiões propõe-se primeiro encontrar os contornos externos que delimitam cada uma delas, como demonstra a Figura 4. 8. Depois, um *bounding box* (retângulo delimitador) é definido para cada uma das regiões candidatas, para assim, fazer algumas validações geométricas a fim de obter a ROI.

As validações estão baseadas nos valores máximos e mínimos de altura, largura e área que podem ter os *boundings boxes*, além do *aspect ratio (AR)*, que é calculado a traves da Equação (4.2). Para o caso das placas brasileiras os valores de altura e largura são de 13 cm e 40 cm respetivamente ((DENATRAN, 2007)). Esse cálculo impossibilita que

objetos com formas estranhas (muito largas ou muito compridas), sejam consideras placas veiculares (ROY; HOSSEN; NAG, 2016).

$$AR = \frac{L\,\mathrm{arg}\,ura}{Altura} \tag{4. 2}$$

Figura 4. 8 – Detecção de contornos externos.



Fonte: Elaborado pelo Autor.

Na Figura 4. 9 ilustra-se o resultado de aplicar as operações citadas anteriormente. Note como a filtragem dos contornos define duas regiões de interesse finais que são consideradas como potenciais placas veiculares (Figura 4. 9 (a)). A região que não contém a placa é catalogada como falso positivo. Após, com a validação das regiões através da condição de *aspect ratio* observa-se como é definida uma última ROI (retângulo verde na Figura 4. 9 (b)) onde está contida a placa veicular, eliminando-se assim o falso positivo.

Figura 4. 9 – Detecção ROI.

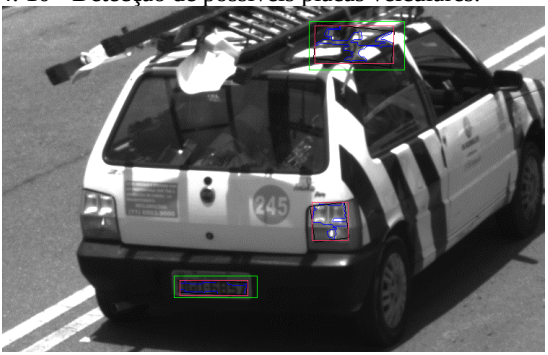
(a) (b)

Fonte: Elaborado pelo autor.

4.2.2.4 Classificação das regiões candidatas

Na seção anterior mostrou-se como através dos algoritmos de filtragem obtém-se uma ROI que contem a placa veicular. Embora a filtragem funcione corretamente, em alguns casos é detectada mais de uma ROI que satisfaz as condições necessárias para ser catalogada como placa veicular, isso faz com que sejam gerados falsos positivos conforme amostrado na Figura 4. 10. Observe que são detectadas duas regiões de interesse enquadradas em retângulos que o sistema reconhecerá como placas veiculares. Para solucionar este problema propõe-se usar, similar ao trabalho proposto por (ZHENG et al., 2012), um classificador binário SVM, treinado com descritores de características HOG (DALAL; TRIGGS, 2005) que determina se a região detectada é ou não é uma placa veicular.

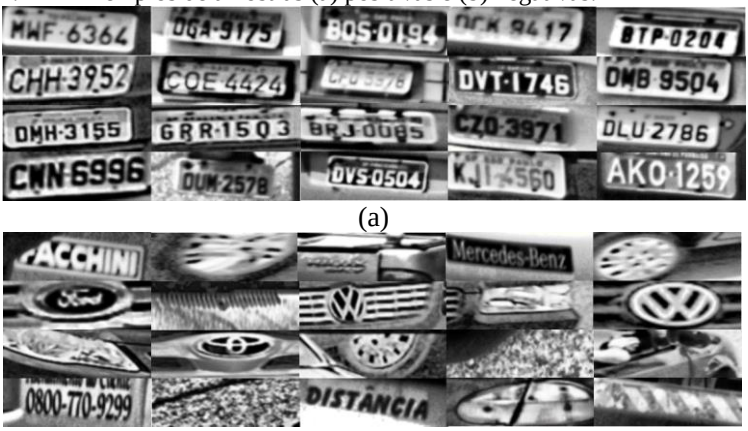
Figura 4. 10 - Detecção de possíveis placas veiculares.



Fonte: Elaborado pelo Autor.

O treinamento do classificador inicia-se com a construção de um conjunto de imagens que contem placas veiculares e que são redimensionadas a um tamanho fixo, estas imagens são denominadas como exemplos positivos. Para o casso deste trabalho utilizou-se 900 amostras positivas (placas veiculares), normalizadas a um tamanho de 144 x 48 píxeis. De igual maneira foi construído outro conjunto de 700 exemplos negativos, ou seja, imagens que não contem placas veiculares (não placas), redimensionados ao mesmo valor de 144 x 48 píxeis. A Figura 4. 11 apresenta alguns exemplos positivos e negativos usados para o treinamento do classificador.

Figura 4. 11 – Exemplos de amostras (a) positivas e (b) negativas.



(a)

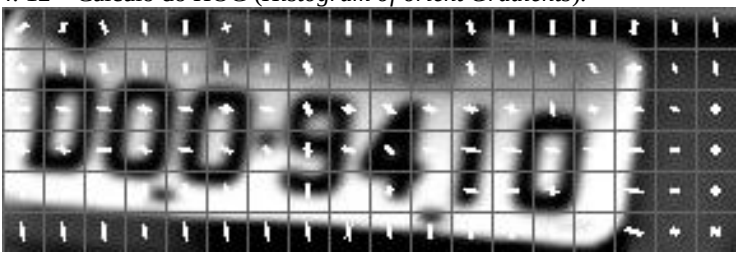
(b)

Fonte: Elaborado pelo autor.

Para cada imagem contida tanto no conjunto de exemplos positivos quanto no conjunto de exemplos negativos é extraído o vetor de características (descritor) HOG conforme apresentado na seção 2.2.1.

Este descritor conta a ocorrência de orientações de gradientes em porções localizadas das imagens. Para isso a imagem é dividida em células (*cells*) não sobrepostas e para cada uma dessas células é calculado o histograma de gradientes orientado. Neste trabalho, utilizou-se células de tamanho 8 x 8 píxeis. Já logo para que o descritor seja independente às variações de luz os autores propõem normalizar os histogramas usando a normalização de contraste local (DALAL; TRIGGS, 2005). Para esse fim as células são agrupadas em blocos e cada bloco é normalizado de forma separada. Neste trabalho, cada bloco tem um tamanho de 16 x 16 píxeis, assim cada bloco estará composto por 4 células. Já o passo de deslocamento da janela deslizante entre blocos (*stride*) é definido como 8 píxeis (valor de uma célula). Por último o número de intervalos para o histograma é de 9 *bins*. Dessa maneira, para cada imagem de entrada com tamanho de 144 x 48 píxeis será calculado um vetor de 3060 características HOG. Na Figura 4. 12 pode se observar o resultado de aplicar o descritor HOG numa placa veicular da base de dados usada para o treinamento. Observe como muda a orientação dos gradientes nas áreas onde apresentadas os caracteres da placa.

Figura 4. 12 – Cálculo do HOG (*Histogram of orient Gradients*).



Fonte: Elaborado pelo autor.

O classificador SVM foi treinado tendo como base os vetores de características HOG extraídos dos exemplos positivos e negativos do conjunto de imagens usadas para o treinamento. Como o objetivo é classificar as imagens como placas ou não placas, foi usado um classificador binário com um *kernel* linear e margem flexível (*soft margin*) com o parâmetro $C = 0.01$, assim como proposto por (DALAL; TRIGGS, 2005). O resultado é um classificador com duas classes: placas, representado por "1" e não placas, representado por "0".

Após o treinamento do classificador SVM – HOG, as regiões de interesse provenientes da etapa de detecção, e que são catalogadas como potenciais placas veiculares, podem ser classificadas em tempo de execução (etapa de teste).

Cada uma destas ROI é redimensionada para um tamanho de 144 x 48 píxeis. Em seguida é extraído o vetor de características HOG usando os mesmos parâmetros que foram utilizados na etapa de treinamento. O vetor resultante é avaliado pelo classificador para gerar uma resposta final de classificação.

4.2.2.5 Correção do ângulo

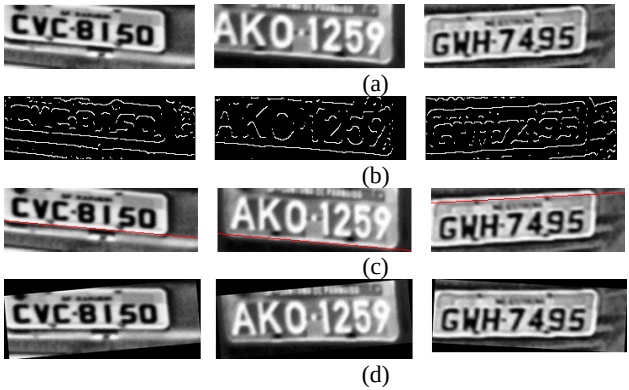
As regiões que são classificadas como placas veiculares, como ilustra-se na Figura 4. 13 (a), apresentam diferentes rotações e perspectivas devido principalmente ao ângulo e posição da câmara, o que faz com que seja prejudicada a etapa posterior de segmentação da placa (DALIDA et al., 2016). Para corrigir esses inconvenientes propõe-se implementar um sistema baseado no método apresentado por (ARULMOZHI et al., 2012) onde é usada a transformada de Hough

(DUDA; HART, 1972) e um algoritmo de esqueletização como principais ferramentas para corrigir o ângulo da placa.

Para estimar o ângulo de rotação efetua-se primeiro uma binarização da imagem usando um limiar adaptativo, depois é esqueletizada a imagem binarizada, salientando as bordas horizontais predominantes, reduzindo assim, a quantidade de pontos a serem transformados para o plano de Hough, o que faz melhorar a eficiência computacional do algoritmo (MASCARO, 2010). A Figura 4. 13(b) ilustra alguns dos resultados ao aplicar este processo.

Seguidamente é aplicada a transformada de Hough (DUDA; HART, 1972) conforme apresentado na 2.1.4, a fim de extrair a linha desejada. A Figura 4. 13 (c) ilustra o resultado desta única linha que representa o ângulo de inclinação no qual encontra-se a placa veicular. Por último a imagem de entrada é rotada tendo em conta esse ângulo de inclinação (Figura 4. 13 (d)).

Figura 4. 13 – Correção do ângulo de rotação. (a) Placas com ângulo de rotação. (b) Binarização e detecção de bordas. (c) Detecção de Hough. (d) Rotação da placa.



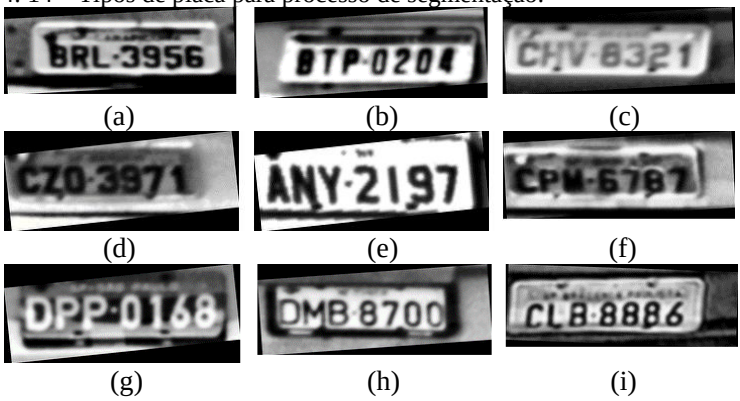
Fonte: Elaborado pelo autor.

4.3 SEGMENTAÇÃO DOS CARACTERES

O objetivo principal desta fase é segmentar os caracteres presentes na placa veicular para depois serem reconhecidos. Antes de executar esta fase é importante observar que as placas obtidas após o processo de correção do ângulo apresentam algumas características descritas a continuação e que podem ser observadas na Figura 4. 14:

- Mais informação do que a placa e os caracteres.
- Iluminação e contraste irregulares devido a condições ambientais (Figura 4. 14 (c), Figura 4. 14 (d), Figura 4. 14 (f), Figura 4. 14 (g)).
- Ausência de bordas definidas, devido ao desgaste das placas (Figura 4. 14 (f), Figura 4. 14 (i)).
- Diferentes tipos de letras, devido ao padrão Brasileiro (Figura 4. 14 (b), Figura 4. 14 (g), Figura 4. 14 (h)).
- Diferente espaçamento entre as letras (Figura 4. 14 (b), Figura 4. 14 (d), Figura 4. 14 (h)).

Figura 4. 14 – Tipos de placa para processo de segmentação.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Fonte: Elaborado pelo Autor.

Essas características influem diretamente no processo de binarização da imagem (etapa importante no processo de segmentação), o que faz com que a eficiência da segmentação diminua notavelmente. É por isso que neste trabalho propõe-se realizar primeiro um enquadramento da placa para assim extrair somente a região dos alfanuméricos presentes na placa, visando uma efetiva binarização.

4.3.1Binarização inicial

Devido a que não há um controle de iluminação sobre a imagem, além de que o fundo não possui uma intensidade luminosa constante conforme apresentado na Figura 4. 14, não é possível usar um valor de limiar constante (ou global) em toda a imagem no processo de binarização, já que um valor que fornecesse bom resultado em uma

determinada região poderá não ser adequado em outra. Daí que sejam gerados problemas como uma separação errada nos componentes na imagem resultante e perdas indesejadas da informação dos caracteres. Para minorar esses problemas na literatura apresentam-se técnicas de binarização adaptativa fundamentadas na utilização de um limiar variável que possa se adaptar às diferentes condições de iluminação na imagem (DU et al., 2013), tais como as propostas por Niblack (NIBLACK, 1986) ou Sauvola (SAUVOLA; PIETIKÄINEN, 2000). Neste trabalho foram testados vários métodos, e encontrou-se empiricamente que com o método de binarização adaptativa Gaussiana (BRADLEY; ROTH, 2007), (NEJATI; MAJIDI; JALALAT, 2015), (KAEHLER; BRADSK, 2016), obtiveram-se os melhores resultados em termos de perdas de informação dos caracteres e iluminação da imagem. Resultados desta binarização são ilustrados na Figura 4. 15, contudo as imagens apresentam regiões indesejadas a serem eliminadas. Note-se por exemplo que, na Figura 4. 15 (a) e Figura 4. 15 (b) apresentam-se uniões de píxeis entre os caracteres e regiões que não são de interesse, já na Figura 4. 15 (c) ilustra-se outro caso onde os píxeis da parte inferior dos caracteres estão unidos, por último na Figura 4. 15 (d) demonstra-se uma quebra dos componentes dos caracteres. Por isso é muito importante a etapa de enquadramento.

Figura 4. 15 – Binarização Inicial.

(a) (b) (c) (d)

Fonte: Elaborado pelo autor.

4.3.2 Análise de componentes conectados

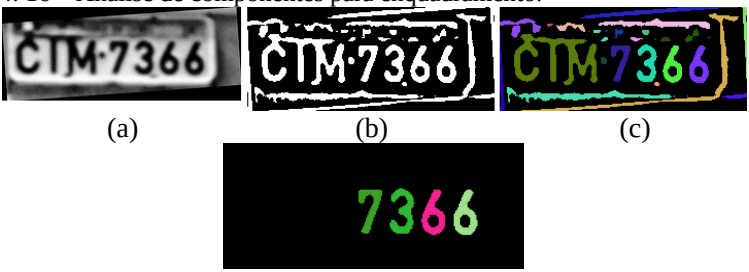
O objetivo principal desta etapa é eliminar todas as regiões que não são caracteres dentro da imagem analisada. Para isto é utilizado o método de *CCA* descrito anteriormente na seção 2.1.3 e usado na etapa de localização e detecção da placa (seção 4.2.2.2). A seguir apresentam-se as regras definidas para eliminar os componentes indesejados, tendo como referência o tamanho das imagens (240 x 100 píxeis, para o casso deste trabalho):

- Contornos com mais do 20% da largura total da imagem são eliminados.

- Contornos com mais do 50% da altura da imagem são eliminados.
- Os contornos posicionados numa distância menor a 10 píxeis tanto nas bordas laterais quanto nas bordas superior e inferior são eliminados devido a que os caracteres estão posicionados numa distância relativamente central da imagem.
- Os contornos com uma área menor a 8 píxeis são eliminados.

Na Figura 4. 16 apresentam-se alguns dos resultados obtidos nesta etapa. Note por exemplo que os três primeiros caracteres correspondentes às letras "CIM" da Figura 4. 16 (b) foram eliminadas devido a sua juntura com a parte superior da placa, daí que sejam etiquetados como um componente só (verde escuro) conforme amostra a Figura 4. 16 (c). Ao fim do processo é obtida uma imagem com 4 caracteres como ilustrado na Figura 4. 16 (d). Neste ponto, é importante esclarecer que para a seguinte etapa (segmentação da área dos caracteres), basta com pelo menos ter detectado uma letra para poder obter a região desejada.

Figura 4. 16 – Análise de componentes para enquadramento.



(a) (b) (c)

(d)

Fonte: Elaborado pelo autor.

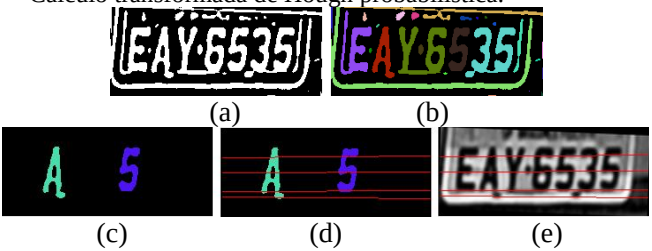
4.3.3 Segmentar região correspondente aos caracteres.

Conforme apresentado na Figura 4. 16 a análise de componentes gerou uma imagem onde a localização dos caracteres está definida. A partir dessas posições, aplica-se a transformada de Hough probabilística progressiva proposta por (MATAS; GALAMBOS; KITTLER, 1998) para encontrar as linhas horizontais presentes na imagem. Esta transformada é uma otimização da transformada de Hough clássica (DUDA; HART, 1972), caracterizada por explorar a diferença entre o número de votos necessários para suportar uma correta decisão na

detecção de linhas com um número diferente de pontos. O número de pontos da linha a exercer o voto em função do comprimento da linha, por conseguinte, para uma linha longa a proporção de votos é pequena, diferente a uma linha pequena onde a proporção de votos é maior. É assim que a técnica permite determinar a longitude dos segmentos a serem aceitos como linhas. Tornando o processo mais rápido devido ao seu mínimo custo computacional, permitindo que seja ideal para aplicações em tempo real (MATAS; GALAMBOS; KITTLER, 2000).

Com o anterior, podem-se definir dois parâmetros importantes dentro do algoritmo: o primeiro é a mínima longitude permitida para que um segmento seja considerado como uma linha e o segundo é o máximo valor (determinado em píxeis) do espaço entre dois segmentos a serem tratados como uma linha (LAGANIÈRE, 2011). Para este trabalho determinou-se esses valores como 5 e 4 píxeis respetivamente. Daí que seja suficiente somente um caractere para segmentar a região desejada. A Figura 4. 17 demonstra um exemplo onde o resultado do *CCA* faz com que seja gerada uma região com dois caracteres (Figura 4. 17 (c)), e como a transformada probabilística de Hough (MATAS; GALAMBOS; KITTLER, 1998) detecta as linhas horizontais com os parâmetros descritos anteriormente (Figura 4. 17 (d) (e)).

Figura 4. 17 – Calculo transformada de Hough probabilística.



Fonte: Elaborado pelo autor.

Utilizando o resultado obtido pela transformada, utilizam-se algumas propriedades geométricas da imagem para segmentar a ROI.

Primeiro são definidas duas linhas "guias" ($l1$ e $l2$) separadas 15 píxeis tanto da borda superior, quanto da borda inferior da imagem como ilustrado na Figura 4. 18. Esse valor foi definido visando que os caracteres -cuja localização na imagem é variável-, sempre fiquem localizados entre as duas linhas. Assim, O objetivo de $l1$ e $l2$ é percorrer a imagem desde os seus extremos até o interior, procurando os valores de p_1 e p_2 respetivamente, os quais representam os vértices do retângulo que servirá

para segmentar a ROI pertencente aos caracteres da placa. Dessa forma, por exemplo, a linha $l1$ que começa no ponto y_{o1} fará uma busca vertical (eixo y) até encontrar o ponto p_1, que representa o vértice superior do retângulo. Neste ponto é importante considerar a ubiquação das linhas detectadas pela transformada probabilística. Observe por exemplo, que a linha que começa no ponto p_1 entrecruza a parte superior de todos os caracteres da placa, deste modo, se fosse feita uma segmentação desde esse ponto, se perderia informação importante dos caracteres. Por essa razão, e considerando que essa condição se apresenta na maioria das placas, experimentalmente foi determinado que o vértice 1 denotado por p_{v1} fosse calculado como: $p_{v1} = p1 - 7px$, conforme ilustra a Figura 4. 18, dessa maneira assegura-se que a ROI seja segmentada sem perder informação dos caracteres.

Da mesma forma, a linha $l2$ que começa no ponto y_{o2} fará a busca até chegar ao ponto p_2, que representa o vértice oposto de p_{v1} e é denotado como p_{v2}. Já neste casso $p_2 = p_{v2}$.

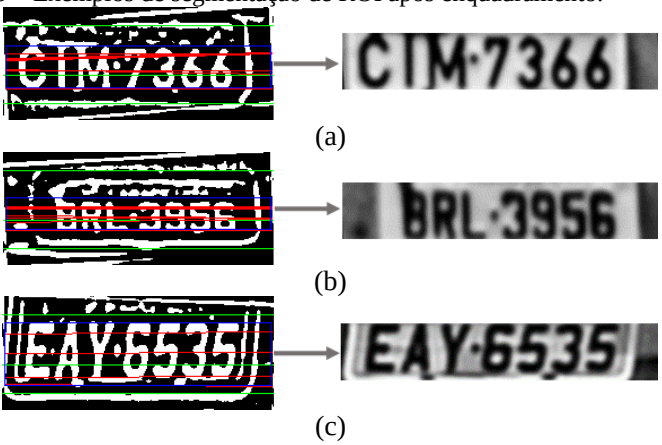
Figura 4. 18 – Detecção da ROI para segmentar caracteres

Fonte: Elaborado pelo Autor.

Por fim, com base nos dois vértices encontrados, é formado e segmentado um retângulo que delimita a ROI pertencente aos caracteres da placa, este pode ser observado na Figura 4. 18 em cor azul. Já a Figura 4.19 demostra alguns exemplos do resultado obtido.

Cada um dos retângulos obtidos é então redimensionado a um tamanho de 280×50 píxeis para passar à seguinte etapa do processo.

Figura 4. 19 – Exemplos de segmentação de ROI após enquadramento.



Fonte: Elaborado pelo Autor.

4.3.4 Classificação qualidade das placas.

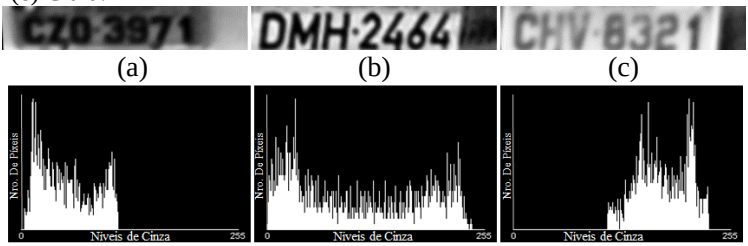
Após o enquadramento das placas, as imagens retangulares obtidas caracterizam-se por apresentar mudanças de luminosidade e contraste devido principalmente ao fato da aquisição em ambientes não controlados. Embora o filtro MMLPF ajude a minimizar essas mudanças, nem todas as imagens podem apresentar valores fixos de contraste e luminosidade, impossibilitando assim uma correta binarização. Para resolver esse problema, neste trabalho propõe-se uma técnica de binarização adaptativa baseada na variação da iluminação das placas.

Fundamentando-se na base de dados fornecida para o desenvolvimento do projeto encontrou-se que a qualidade das placas está dividida em três classes: “escura”, “normal” e “clara”. As placas escuras caracterizam-se por ter condições de baixa luminosidade e contraste, enquanto as “claras” apresentam valores altos de brilho.

A Figura 4. 20 ilustra os três tipos de placas presentes na base de dados e seus respectivos histogramas, aqui o eixo horizontal representa os níveis de cinza e o eixo vertical a quantidade de píxeis de cada nível de cinza na imagem. Observe como a distribuição dos píxeis no histograma muda conforme o tipo de placa. Assim, por exemplo nas placas normais (Figura 4. 20 (b)) a distribuição dos píxeis é uniforme, nas placas escuras (Figura 4. 20 (a)) os píxeis encontram-se na região de

valores de cinza pequenos, contrário das placas claras (Figura 4. 20 (c)) onde os píxeis distribuem-se sobre valores de cinza altos.

Figura 4. 20 – Classificação tipos de placas e os seus histogramas. (a) Escura (b) Normal (c) Clara.



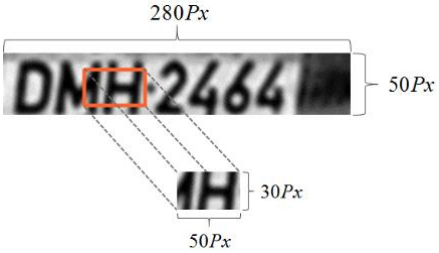
(a) (b) (c)

(a-1) (a-2) (a-3)

Fonte: Elaborado pelo autor.

A partir das distribuições definiu-se uma simples técnica para classificar o tipo de placa analisada. Para isso, primeiramente é definida uma região de interesse com um tamanho de 10% do tamanho total da imagem, conforme se ilustra na Figura 4. 21.

Figura 4. 21– ROI para determinar o tipo de placa



Fonte: Elaborado pelo Autor.

Experimentalmente determinou-se que uma janela desse tamanho possui suficiente informação para classificar a placa. Logo é calculada a média dos valores de níveis de cinza dos píxeis concentrados na ROI. Assim, o tipo de placa é determinado a partir da seguinte formulação:

$$Tp = \begin{cases} Escura & se \quad 0 < n_c \leq 69 \\ Normal & se \quad 69 < n_c \leq 125 \\ Clara & se \quad 125 < n_c \leq 255 \end{cases} \tag{4. 3}$$

Onde Tp representa o tipo de placa e n_c o valor da média dos níveis de cinza. Esses valores foram determinados empiricamente com base em observações experimentais. Assim por exemplo, a placa ilustrada na Figura 4. 20 (a) tem um valor de $n_c = 65,188$ o que faz com que seja classificada como uma placa escura. Já para a placa da Figura 4. 20 (b) o valor de $n_c = 106,44$ que corresponde a Tp igual a normal, enquanto que a Figura 4. 20 (c) apresenta uma média de $n_c = 178,398$, sendo assim classificada como clara.

4.3.5 Binarização final

Após determinar o tipo de placa, a imagem classificada precisa ser binarizada a fim de segmentar os caracteres. Para atingir este objetivo, utilizou-se o método de binarização adaptativa gaussiana (BRADLEY; ROTH, 2007) descrito na seção 2.1.2.2, tendo como principal referência os parâmetros de n_c obtidos na etapa anterior. Deste modo os valores da constante *"C"* -definida como um coeficiente de compensação da iluminação, que varia de acordo às propriedades da imagem de entrada em escala de cinza (WANG et al., 2012) -, e o tamanho da janela Gaussiana mudam de acordo aos valores de níveis de cinza representados por n_c. A Tabela 4. 1 apresenta os valores definidos para C e a janela Gaussiana em relação aos valores de n_c.

Tabela 4. 1 – Valores de C e *Block Size* em relação a n_c.

Tipo placa (Tp)	Valor de (n_c)	Valor de C	Janela Gaussiana (*Block Size*)
Imagem Escura (Ie)	$0 < n_c \leq 69$	3	(9×9)
Imagem Normal (In)	$69 < n_c \leq 125$	9	(17×17)
Imagem Clara (Ic)	$125 < n_c \leq 255$	11	(24×24)

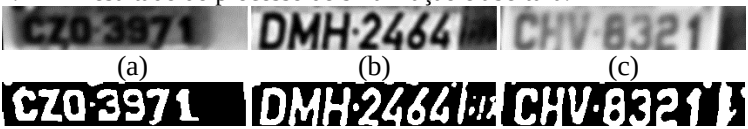
Fonte: Elaborado pelo Autor.

Observe como para valores pequenos de n_c, os valores de C também são menores e a janela Gaussiana é pequena.

Uma vez binarizada a imagem, é utilizada a operação de abertura com um elemento estruturante retangular com um tamanho de 3×3 píxeis a fim de suavizar os contornos resultantes, quebrar pequenas uniões e minimizar pontos agudos (HARALICK; STERNBERG; ZHUANG,

1987). A Figura 4. 22 demonstra o resultado de aplicar o processo mencionado anteriormente nas placas da Figura 4. 20. Note como os caracteres preservam a sua conectividade *(stroke connectivity).*

Figura 4. 22 – Resultado do processo de binarização e abertura.



(a) (b) (c)

(a-1) (b-1) (c-1)

Fonte: Elaborado pelo Autor.

Terminada a etapa de enquadramento e binarização, pode se observar que as imagens binarias resultantes contém, além dos caracteres, informação indesejável como bordas, manchas, sujeiras e pequenos componentes que, em seu conjunto conformam ruído a ser eliminado. Desta forma, e visando a correta localização e extração dos 7 caracteres que compõem a placa veicular, propõe-se usar uma serie de técnicas que são descritas a continuação.

4.3.6 Eliminando primeiros componentes

A primeira operação é encontrar os componentes conectados da imagem binarizada. Logo, para cada componente detectado determina-se a sua altura *(height)*, largura *(Width)* e área, definidas aqui como H_c, W_c e A_c respetivamente. Com essa informação são elaboradas uma série de regras com a finalidade de remover os componentes com tamanhos muito grandes ou muito pequenos com base ao tamanho da imagem e considerando a condição de que podem-se apresentar caracteres unidos como se ilustra na Figura 4.23.

Figura 4. 23 – Características de componentes nas imagens.

Fonte: Elaborado pelo autor.

Sejam I_H, I_W e I_A a altura, largura e área da imagem respetivamente. O componente será eliminado se:

$$W_c > (0.27 \times I_W) \vee W_c < (0.03 \times I_W) \quad (4.\ 4)$$

$$H_c > (0.2 \times I_H) \quad (4.\ 5)$$

$$A_c < 10\ pixeis \quad (4.\ 6)$$

4.3.7 Separando componentes

Embora a etapa de enquadramento funcione de forma adequada para remover muitas regiões indesejáveis. Em ocasiões geram-se imagens onde os caráteres apresentam uniões com outros componentes considerados como ruído e que geralmente representam alguma falha ou desgaste da placa. Na Figura 4. 24 pode-se observar três exemplos de placas contendo caracteres unidos com outros componentes. Note-se como este problema é refletido na binarização da imagem, o que impossibilitaria posteriormente uma apropriada segmentação da letra.

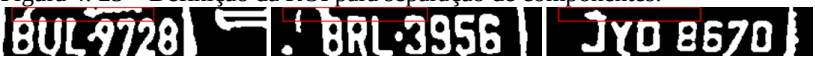
Figura 4. 24 – União de componentes no processo de binarização.

Fonte: Elaborado pelo Autor.

Para solucionar o problema, primeiramente foi delimitada uma região de interesse na imagem baseando-se no seguinte critério: observou-se que quando apresentadas uniões de caracteres com outros

componentes na parte superior da imagem, essas uniões estavam localizadas entre o primeiro e terceiro caractere da placa, como ilustrado anteriormente na Figura 4. 24. Bastaria então com definir uma região que assegurasse conter sempre os quatro primeiros caracteres da placa, com isso diminui-se o custo computacional desta etapa. Neste contexto, a ROI definida para este trabalho -ilustrada na Figura 4. 25-, têm uma altura de 15 píxeis (20% da altura da imagem) e uma largura de 140 píxeis (50% da largura total da imagem).

Figura 4. 25 – Definição da ROI para separação de componentes.



(a) (b) (c)

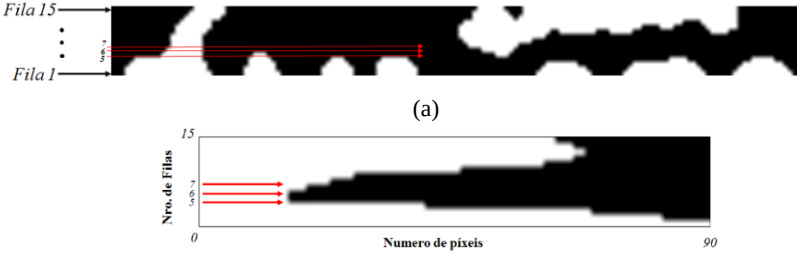
Fonte: Elaborado pelo Autor.

O seguinte passo do processo é obter o número de píxeis presentes por cada fila da ROI delimitada. Para isso, a técnica utilizada foi a projeção horizontal da imagem. Aqui, soma-se a quantidade de píxeis brancos (com valor 1) para cada uma das filas da imagem, gerando assim um vetor de projeção. A técnica é definida então pela seguinte equação:

$$V_p(x) = \sum_{i=0}^{lr} I(x,i) \qquad (4.\ 7)$$

Onde Vp é a projeção horizontal, I a imagem binarizada, lr a largura da imagem, enquanto x varia desde 0 até a altura total de I. A Figura 4. 26 ilustra um zoom da ROI definida na Figura 4. 25 (a) e sua projeção horizontal.

Figura 4. 26 – Região de interesse da Figura 4. 25 (a) para separação de componentes. (a) zoom da ROI e (b) Projeção vertical da ROI.



Fonte: Elaborado pelo Autor.

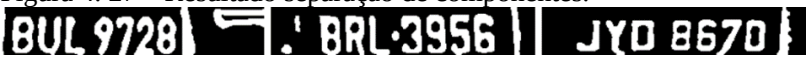
Como resultado da projeção horizontal obtém-se os valores da quantidade de píxeis por cada uma das filas da ROI, assim por exemplo, na fila 1 da Figura 4. 26 (a) foram calculados 90 píxeis, enquanto na fila 15 obtiveram-se 65 píxeis, esses valores são refletidos na projeção horizontal da Figura 4. 26 (b). Observe como para as filas 5, 6, 7 a quantidade de píxeis é menor (15, 15 e 18 respetivamente) comparada com outras filas. Esses píxeis representam a união do componente de ruído com o caractere da primeira placa, portanto são píxeis a serem eliminados. Nesse contexto, define-se uma simples regra a fim de eliminar a união:

Seja $Q_{px}(i)$ a quantidade de píxeis da i-ésima fila da ROI definida. Essa fila será eliminada se:

$$Q_{px}(i) < l_p \tag{4. 8}$$

Onde l_p representa o número limite de píxeis permitido para que a fila permaneça na ROI. Neste trabalho definiu-se o valor de l_p como 40 píxeis. Assim, todas as filas da ROI cujo valor de $Q_{px}(i)$ seja menor a 40 píxeis serão eliminadas. O resultado de aplicar essa regra é apresentado na Figura 4. 27.

Figura 4. 27 – Resultado separação de componentes.



(a) (b) (c)

Fonte: Elaborado pelo Autor.

A partir da imagem gerada pela aplicação da regra mencionada anteriormente é feita uma análise dos componentes resultantes a fim de serem segmentados os 7 caracteres da placa. Para tal fim, primeiro são filtrados os componentes a traves das equações (4.4) (4.5) e (4.6), definidas na Seção 4.3.6. Além disso, aplica-se uma última regra fundamentada na posição dos componentes dentro da imagem. Assim, todos os componentes que possuam uma distância menor ou igual a 3 px respeito à borda superior da imagem serão eliminados. A

Figura 4. 28 ilustra o resultado do procedimento. Observe como os componentes localizados na parte superior da placa são removidos através da condição exposta anteriormente.

Figura 4. 28 - Análise de componentes para remoção de ruídos.

Fonte: Elaborado pelo autor.

Finalmente, a partir da imagem obtida pelas operações anteriores, calculam-se os contornos externos dos componentes conforme ilustra a Figura 4. 29 (a). Dessa maneira é possível definir um retângulo delimitador (*bounding box*) para cada um dos contornos com o objetivo de segmentar os caracteres da placa. Na Figura 4. 29 (b) é ilustrado um exemplo desta operação, aqui, os retângulos brancos representam os *bounding boxes*. Por último a Figura 4. 29 (c) demonstra o resultado final na imagem original.

Figura 4. 29 – Detecção de contornos finais e *bounding boxes.*

(a) (b) (c)

Fonte: Elaborado pelo Autor.

Os retângulos resultantes contêm atributos importantes que podem ser utilizados nas seguintes etapas do processo, visando atingir uma boa segmentação dos caracteres. Esses atributos são listados a continuação:

- Área denotada por A_R.
- Largura definida como W_R.
- Altura denotada por H_R.
- Centro do retângulo definido como (x, y).

4.3.8 Separando uniões de caracteres.

Em alguns casos, como o ilustrado na Figura 4. 30 (a), após a binarização da imagem apresentam-se uniões de caracteres, devido principalmente a ruídos produzidos por pequenas sombras ou sujeiras nas placas. Para separar esses caracteres é usada a média da largura dos *bounding box* presentes na imagem. Deste modo estipulou-se que os retângulos devem ser particionados em duas partes quando sua largura satisfaça a seguinte condição:

$$W_R < 1,4 * \overline{X}_{largura} \quad (4.\ 9)$$

Onde W_R é a largura do *bounding box* avaliado e $\overline{X}_{largura}$ a média das larguras dos *bounding Boxes*. A Figura 4. 30 (c) ilustra o resultado da separação.

Figura 4. 30 – Separação de caracteres. (a) imagem binarizada (b) Contornos detectados. (c) separação de caracteres unidos.

(a) (b) (c)

Fonte: Elaborado pelo Autor.

4.3.9 Remoção dos Falsos caracteres.

Neste ponto do processo, muitos dos componentes indesejados foram removidos e na maioria das imagens os caracteres foram localizados corretamente conforme ilustrado na Figura 4. 29 e Figura 4. 30, no entanto apresentam-se casos onde alguns componentes são confundidos facilmente como caracteres, já que cumprem com as condições mínimas de área, largura, altura e *aspect ratio* para serem classificados corretamente como alfanumérico na imagem, na Figura 4. 31 é apresentado um exemplo desta situação. Observe que os componentes confundidos como caracteres estão localizados no início e no final da placa.

Figura 4. 31 – Imagens que apresentam falsos caracteres.

(a) (b) (c) (d)

Fonte: Elaborado pelo Autor.

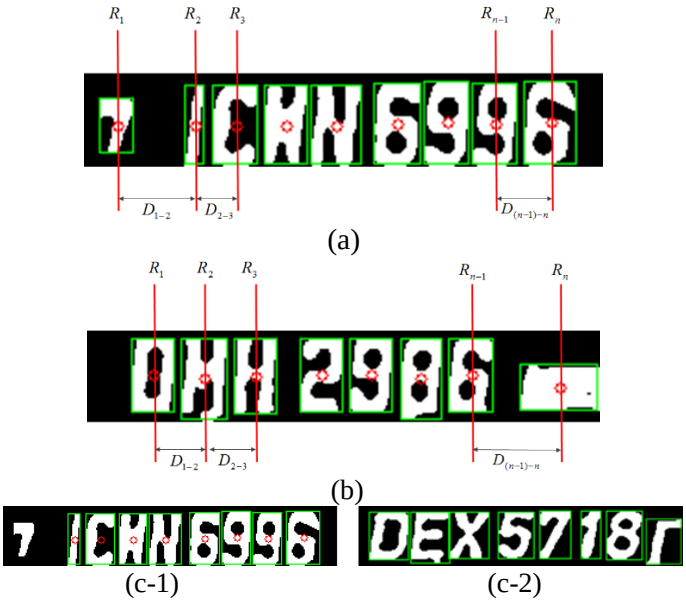
Para resolver o problema, primeiro é feita uma validação da quantidade de *bounding boxes* (denotado por N_{BB}) presentes na imagem. Assim, lembrando que o número de retângulos esperado por cada placa é de 7 (3 letras e 4 números para o padrão de placas Brasileiras), determinam-se uma série de regras a seguir expostas:

Quando apresentada a condição ideal em que $N_{BB} = 7$, conforme ilustrou-se na Figura 4. 29, o processo de remoção dos falsos caracteres é encerrado por parte do algoritmo, passando assim à etapa de reconhecimento dos caracteres.

Já no caso em que $N_{BB} > 7$, o primeiro passo a seguir é calcular a média das distancias horizontais entre os centros dos retângulos, definida aqui como $\bar{X}_{DR}$. Seguidamente calcula-se a distância horizontal entre o primeiro e o segundo retângulo, de igual forma entre o ultimo e o penúltimo retângulo. A Figura 4. 32 ilustra o anteriormente planteado. Note como a distância (D) entre o primeiro e segundo retângulo (R) da Figura 4. 32 (a) é relativamente maior quanto a distância entre o segundo e o terceiro retângulo. Portanto, se a distância entre R_1 e R_2, isto é $D(R_1, R_2) > 1.5 * \bar{X}_{DR}$, então o primeiro retângulo é removido da imagem. Da mesma maneira, conforme demonstra a Figura 4. 32 (b), aplica-se o procedimento para os últimos retângulos da imagem, ou seja, se $D(R_{n-1}, R_n) > 1.5 * \bar{X}_{DR}$, então o último retângulo da imagem é removido. Aqui n denota o último retângulo, enquanto $n-1$ representa o último.

Como último passo é feita uma nova verificação do número de *bounding Boxes* presentes na imagem. Se o seu número é maior a 7 então o algoritmo envia a imagem às seguintes etapas de verificação.

Figura 4. 32 – Removendo componentes se baseando na distância dos *bounding boxes*. (a) Remoção do primeiro componente. (b) Remoção do ultimo componente. (c) Resultado do procedimento.



Fonte: Elaborado pelo autor.

Em alguns casos, como o apresentado na Figura 4. 32 (c-1), a imagem resultante ainda contém componentes indesejados, devido principalmente a que a distância entre o primeiro *bounding box* (que representa uma borda da placa) e o segundo (que é um caractere) é muito parecida ao valor da média das distancias de todos os retângulos da imagem, isso faz com que não funcione a condição de remoção planteada anteriormente. Na Figura 4. 32 (c-2) ilustra-se outro caso com a mesma situação, mas com o componente indesejado localizado no último *boundig box* da imagem.

Para esses casos a remoção efetua-se usando os valores da metade das alturas (H_R) e as larguras (W_R) dos *bounding boxes*. Para isso, inicialmente verifica-se a quantidade de retângulos da imagem. Portanto, se $N_{BB} > 7$ calcula-se a meia das alturas -denotada como $\bar{X}_{HBB}$- e a

meia das larguras -definida como $\bar{X}_{WBB}$ -. Depois cada um dos retângulos é analisado através das seguintes equações:

$$H_R \leq 0.8 * \bar{X}_{HBB} \vee \ H_R \geq 1.2 * \bar{X}_{HBB} \quad (4.10)$$

$$W_R \leq 0.6 * \bar{X}_{WBB} \quad (4.11)$$

Consequentemente, os retângulos que atenderem qualquer das condições apresentadas nas equações (4.10) e (4.11) serão removidos da imagem. Esse processo é repetido até atingir o valor ideal de $N_{BB} = 7$. Ao final desta etapa obtém-se uma imagem em níveis de cinza contendo os caracteres presentes na placa veicular e um vetor com retângulos que determinam cada uma das posições dos caracteres. Ambos serão os parâmetros de entrada para a próxima etapa de reconhecimento dos caracteres. A Figura 4. 33 demonstra alguns dos resultados obtidos.

Figura 4. 33 – Resultados finais de segmentação.

(a) (b) (c) (d)

Fonte: Elaborado pelo Autor.

4.4 RECONHECIMENTO DOS CARACTERES

O reconhecimento dos caracteres foi realizado utilizando o método de aprendizado supervisionado *Support Vector Machine (SVM)* descrito na 2.2.2 com o objetivo de reconhecer os caracteres. Já para a extração de características e posterior treinamento do classificador utilizou-se o descritor de características HOG proposto por (DALAL; TRIGGS, 2005)

conforme exposto na 2.2.1 e usado na etapa de classificação das placas veiculares (Seção 4.2.2.4). Esse descritor demonstrou trabalhar de maneira efetiva para representar os caráteres da placa em trabalhos como os apresentados por (GOU et al., 2016), (SUBHADHIRA et al., 2014) e (WU et al., 2013). A seguir será descrito o processo do reconhecimento.

Para aumentar a acurácia do processo de reconhecimento decidiu-se dividir o classificador em dois: um deles destinado ao reconhecimento das letras e chamado de *SVM – Letras*, composto por 26 classes (cada classe representa uma letra do alfabeto inglês, que é usado nas placas Brasileiras), e outro designado ao reconhecimento dos números e denominado como *SVM – Números*, que contém 10 classes (cada uma correspondente a um número entre 0 e 9).

O processo de treinamento dos classificadores *SVM* inicia-se pela seleção de exemplos de imagens para cada um dos tipos de letras e números das placas veiculares contidas na base de dados usada para o treinamento. Para isto, neste trabalho foram usadas 6097 amostras entre números e letras que foram redimensionadas a um tamanho fixo de 32×48 píxeis. Essas amostras foram coletadas da base de dados usada para o reconhecimento das placas visto na seção 4.2.2.4, além disso foram geradas algumas outras amostras artificiais adicionando algum tipo de ruído ou mudando seu ângulo de rotação. A Figura 4. 34 ilustra algumas amostras que compõem o conjunto de treinamento.

Figura 4. 34 – Exemplos de imagens usadas para o treinamento dos classificadores SVM-HOG.

Fonte: Elaborado pelo Autor.

A partir do conjunto de treinamento, realiza-se o processo de extração de características para cada uma das amostras. Para tal fim, é utilizado o descritor de caraterísticas HOG, que como foi explicado anteriormente, é um descritor que conta a ocorrência de orientações de gradientes em porções localizadas das imagens, fazendo com que seja robusto ante as mudanças de iluminação, sombras ou baixo contraste (WU et al., 2013).

Neste trabalho foram usadas células (*cells*) com tamanho de 8×8 píxeis, blocos (*block Size*) de tamanho 8×8 píxeis e um deslocamento entre blocos (*Block Stride*) de 4×4 pixeis. Já o número de *bins* (orientações) é de 9. Essa configuração produz um vetor de 693 caraterísticas HOG por cada imagem de entrada com tamanho de 32×48 píxeis.

Após calcular os descritores, é realizado o treinamento dos dois classificadores *SVM* lineares.

Para o ajuste dos hiperplanos utiliza-se a o esquema de validação cruzada *k-fold,* sendo 10 o valor escolhido para o parâmetro *k,* e os descritores HOG são normalizados por meio da norma *L2.* Já o tipo de treinamento utilizado foi o *one-against-one (também chamado One-vs-one),* devido ao fato dos classificadores serem multiclasse. Esse método consiste na construção de um classificador binário por cada par de classes usadas para o treinamento. Deste modo, para um problema com k classes são treinados $k\,(k-1)/2$ *SVMs (*todos os possíveis pares de classes*)* para distinguir as amostras de uma classe das amostras de outra classe (DAENGDUANG; VATEEKUL, 2016) (MILGRAM; CHERIET; SABOURIN, 2006). Assim por exemplo, para o classificador *SVM – Numeros* são treinados 45 *SVMs* binários, enquanto para o problema do SVM – Letras são treinados 325 *SVMs* binários. Posteriormente esses classificadores são armazenados a fim de serem usados na etapa de teste.

Após o treinamento dos classificadores, o vetor com as posições dos caracteres obtido na etapa anterior de segmentação de caracteres é distribuído, em tempo de execução, da seguinte maneira: os três primeiros caracteres são enviados para o *SVM – Letras,* e os quatro últimos para o *SVM – Números.*

Cada caractere segmentado é redimensionado a um tamanho fixo de 32×48 píxeis, a fim de normalizar o tamanho das entradas para os classificadores SVMs treinados. Posteriormente para a etapa de teste (em

tempo de execução) é extraído o descritor de caraterísticas HOG para cada um dos caracteres, usando os mesmos parâmetros utilizados na etapa de treinamento. Desta maneira cada caractere terá um vetor HOG característico que será avaliado por cada um dos classificadores binários treinados.

Por último para estabelecer a classe à que pertence cada caractere. O método de *one vs one* geralmente usa uma estratégia de votos, onde cada um dos classificadores binários é considerado como um voto, assim cada vetor HOG (que representa o caractere) recebe um voto por cada classificador que o prediz. Dessa maneira o caractere é designado para estar na classe com o maior número de votos (BURGES, 1998) (CHANG; LIN, 2013).

5 RESULTADOS.

Neste capítulo serão apresentados os resultados obtidos ao aplicar o método proposto sobre as etapas de detecção, segmentação e reconhecimento de caracteres que compõem o ALPR. Esses resultados serão comparados com algumas das soluções propostas na literatura.

Todos os experimentos, testes e validação do método foram implementados em um computador com sistema operacional Windows 7, processador intel i3 2.2 Ghz e memória de 2 GB DDR2 RAM. Desde o ponto de vista software, o algoritmo foi desenvolvido em C++ sobre a plataforma Visual Studio 2011 e a biblioteca de processamento de imagens OpenCV na versão 3.0.

A base de dados utilizada nos experimentos e avaliação do método proposto foi fornecida pela empresa Brasileira Brascontrol[5]. Essa base de dados, conforme apresentado na seção 4.1 é composta por 1970 imagens que foram capturadas em rodovias da cidade de São Paulo, em diferentes horas do dia. Dessas 1970 imagens, 850 foram utilizadas para formar um conjunto de teste (diferente do conjunto de imagens para o treinamento) com o objetivo de avaliar o desempenho do sistema. As imagens contidas neste conjunto possuem as mesmas características relativas à iluminação e contraste que as imagens usadas para o treinamento dos classificadores.

Por outra parte, é importante mencionar que a maioria de estudos publicados não utilizam as mesmas bases de dados, devido principalmente a que cada país tem especificações concretas respeito à forma e conteúdo das placas veiculares (DU et al., 2013). Além disso, são poucas as bases de dados disponíveis publicamente, sendo a grande maioria próprias ou de empresas. No caso deste trabalho, a base de dados foi fornecida graças à parceria entre a empresa Brascontrol e o grupo de pesquisa S2i.

Com o anterior, e tendo em conta que na literatura são poucos os trabalhos publicados em relação ao reconhecimento de placas Brasileiras, os resultados foram comparados com os resultados das propostas desenvolvidas por (CORNETO et al., 2017) cuja base de dados pertence a imagens com placas Brasileiras capturadas em um estacionamento. (NETO et al., 2015), onde as imagens foram capturadas por uma câmera fotográfica convencional a uma distância fixa inferior a 4 metros; e (CASTELAN, 2009), onde foram usadas imagens da mesma empresa Brasileira Brascontrol em cenários não controlados; (ZHENG et al.,

[5] http://www.brascontrol.com.br/

2012) cuja base de dados é de placas veiculares chinesas capturadas também em um estacionamento; e a proposta de (GOU et al., 2016) onde é usada uma base de dados com imagens capturadas em rodovias Chinesas, com condições de iluminação e contraste complexas. (CHEN et al., 2009)

5.1 DETECÇÃO E SEGMENTAÇÃO DA PLACA

Como já foi mencionado anteriormente, o objetivo desta etapa é a segmentação da placa veicular desde a imagem de entrada. Nesse sentido, o detector desenvolvido percorre a imagem de entrada a procura de regiões que possam ser consideradas placas veiculares, delimitando-as em caixas *(bounding box).*

Para este trabalho as imagens de entrada têm dimensões entre 640x480 e 752x480 píxeis. As métricas utilizadas para medir o desempenho da etapa de detecção foram a sensitividade (*recall*), definida na equação (5.1) e, que corresponde à eficiência na classificação de todos os objetos detectados corretamente, ou seja, as placas. E a precisão (equação (5.2)) que mede a qualidade das respostas positivas da etapa de detecção.

$$\underset{(\mathrm{Re}\,call)}{Sensitividade} = \frac{Positivos\ \mathrm{Re}\,ais}{Positivos\ \mathrm{Re}\,ais + Falsos\ Negativos} \quad (5.1)$$

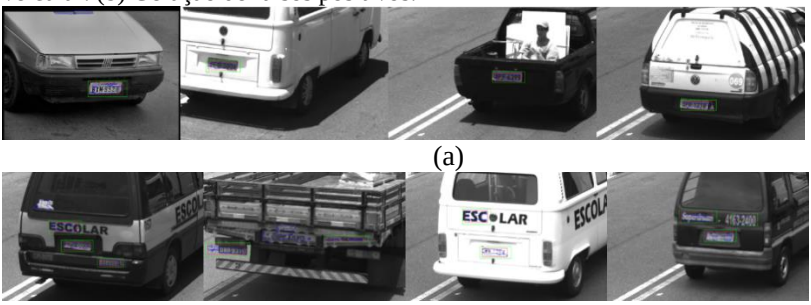
$$\mathrm{Pr}\,ecisão = \frac{Positivos\ \mathrm{Re}\,ais}{Positivos\ \mathrm{Re}\,ais + Falsos\ Positivos} \quad (5.2)$$

$$CCR = \frac{Positivos\ \mathrm{Re}\,ais + Negativos\ \mathrm{Re}\,ais}{Positivos\ \mathrm{Re}\,ais + Falsos\ Negativos + Falsos\ Positivos + Negativos\ \mathrm{Re}\,ais} \quad (5.3)$$

Assim, em uma primeira instância, onde foram consideradas somente as operações morfológicas e geométricas para a detecção das placas veiculares, o sistema detectou corretamente 830 placas veiculares nas 850 imagens do conjunto de teste, o que corresponde a uma sensitividade de 0,96. No entanto, em 375 imagens (que equivale a 43% do total das imagens), foram originados um total de 510 falsos positivos, fazendo com que a precisão total do sistema na etapa de detecção fosse de 0,619. Que é um valor baixo tendo em conta que as saídas desejadas são placas

veiculares isoladas. Na Figura 5. 1 são ilustrados alguns casos do exposto anteriormente.

Figura 5. 1 – Resultado do processo de detecção. (a) Isolamento sucedido da placa veicular. (b) Geração de falsos positivos.



(a)

(b)

Fonte: Elaborado pelo autor.

Para diminuir os falsos positivos e aumentar a precisão do detector desenvolvido foi usado um classificador binário SVM-HOG que determinará se as imagens delimitadas nos *bounding boxes* são placas ou não-placas conforme foi apresentado na seção 4.2.2.4.

Para o treinamento do classificador foi utilizado um *kernel* linear de maneira semelhante ao trabalho proposto por (DALAL; TRIGGS, 2005) com um valor de C=0,01.

Já para evitar a ocorrência de *overfitting* no classificador, ou seja, fazer com que seja ajustado extremamente bem aos dados de treinamento porém, por conta disso, diminua capacidade de generalização para poder classificar imagens de outra fonte de dados (KAEHLER; BRADSK, 2016), utilizou-se o método *k-folds* de validação cruzada (*k-folds Cross-Validation*) no treinamento, com um valor de K=10.

Na Tabela 5. 1 apresentam-se o número de dados utilizados para o treinamento e teste do classificador.

Tabela 5. 1 - Dados base para o treinamento e teste do classificador para placas e não-placas SVM-HOG.

Classe	Etapa de treinamento	Teste	Total
Placa	900	830	1730
Não placa	700	510	1210
Total	1600	1340	2940

Fonte: Elaborado pelo autor.

Após a etapa de treinamento, o classificador é testado em tempo de execução nas imagens geradas pelo detector baseado nas operações morfológicas e geométricas. Como resultado é gerada uma matriz de confusão (Tabela 5. 2) de onde podem ser extraídos resultados como a sensitividade e a precisão do classificador final. Observe que a soma de cada uma das linhas da matriz corresponde ao número total de dados usados para cada classe.

Tabela 5. 2 – Matriz de confusão classificação das placas.

		Resultado classificação	
		Placa	**Não placa**
Ground Truth	**Placa**	Positivos Reais	Falsos negativos
	Não placa	Falsos Positivos	Negativos Reais

		Resultado classificação	
		Placa	**Não placa**
Ground Truth	**Placa**	810	13
	Não placa	20	497

Fonte: Elaborado pelo autor.

Pode-se observar na matriz de confusão que o número de falsos positivos e falsos negativos é de 20 e 13 respetivamente. Sendo assim, a sensitividade do classificador foi de 0,984 e a nova precisão foi de 0,975. Neste ponto é importante observar como o valor alto da precisão

corresponde a uma minimização significativa dos falsos positivos gerados pelo detector. Por fim, o valor de acurácia total do classificador *(CCR, Correct classification rate)* calculada através da equação (5.3) foi de 0,98 que equivale a 98%, esse valor é considerado bom, se comparado com os valores presentes na literatura e considerando os problemas de contraste presentes na base de dados. A Figura 5. 2 se ilustra alguns dos resultados obtidos nesta etapa.

Figura 5. 2 – Resultados da etapa de detecção. (a) positivos reais. (b) Falsos Positivos. (c) Negativos reais. (d) Falso Negativo.

(a) (b) (c) (d)

Fonte: Elaborado pelo autor.

5.2 SEGMENTAÇÃO DOS CARACTERES.

Esta etapa foi avaliada em base às 810 imagens usadas na etapa de teste, além das 900 imagens utilizadas para o treinamento dos classificadores. Assim, o número total de imagens foi de 1710.

Inicialmente foi testada a base de dados com as imagens usadas para o treinamento dos classificadores (base de dados 1). O *ground Truth*, ou seja, o número de caracteres totais que em teoria deveriam ter as 900 imagens é de 6300: sendo 2700 letras e 3600 números, tendo em conta o padrão Brasileiro de 7 caracteres por placa veicular: 3 letras e 4 números (DENATRAN, 2007). Neste ponto, cabe mencionar que o objetivo deste primeiro teste não é observar o número de placas onde foram segmentados corretamente os 7 caracteres, mas sim o máximo de caracteres segmentados dentro da base de dados. Isto devido principalmente a que essas imagens são usadas exclusivamente para o treinamento dos classificadores. Dessa maneira o algoritmo segmentou corretamente um total de 6097 caracteres, equivalente a 96,7% do total de caracteres. A Tabela 5. 3 e Tabela 5. 4 demonstram o resultado do experimento. Em cada uma delas indica-se a quantidade de letras e números segmentados através da técnica proposta. Observe como a distribuição das letras dentro da base de dados não é uniforme, sendo as primeiras 5 as que mais se repetem, enquanto que as letras “I” e “Z” se encontram em uma menor quantidade.

Tabela 5. 3 - Resultado da etapa de segmentação das letras na base de dados 1

Letra	Total de caracteres segmentados
A	123
B	147
C	237
D	312
E	149
F	92
G	103
H	74
I	51
J	71
K	64
L	71
M	82
N	61
O	120
P	69
Q	67
R	110
S	65
T	114
U	66
V	67
W	68
X	68
Y	90
Z	59
Total	2620

Fonte: Elaborado pelo autor.

Por outra parte, a distribuição dos números segmentados (Tabela 5. 4) é mais uniforme, com exceção do primeiro digito “0” que é repetido mais vezes.

Tabela 5. 4 Resultado da etapa de segmentação dos números da base de dados 1.

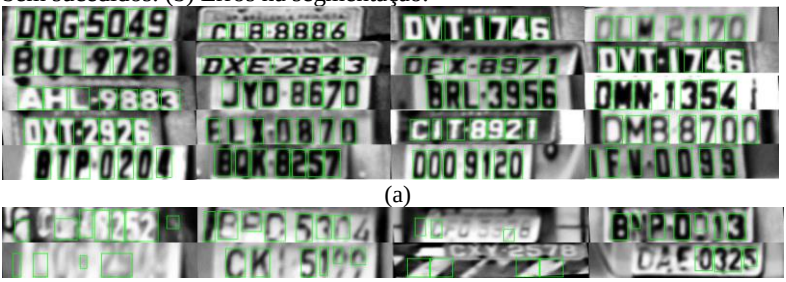
Numero	Total de caracteres segmentados
0	402
1	337
2	295
3	342
4	356
5	352
6	320
7	367
8	359
9	347
Total	3477

Fonte: Elaborado pelo Autor

A segunda base de dados, ou seja, a composta pelas 810 imagens para o teste geral do sistema, foi avaliada de uma forma diferente, considerando que o objetivo principal aqui é segmentar os 7 caracteres de forma eficiente para que possam ser reconhecidos pelos classificadores na seguinte etapa do processo. Desta forma, a placa será segmentada corretamente só se os 7 caracteres são segmentados acertadamente.

Com o anterior, das 810 imagens com placas, foram segmentadas corretamente 786 que corresponde a um 97,03% de acurácia por parte do método proposto. Os erros de segmentação são apresentados em aquelas imagens degradadas ou com caracteres muito ilegíveis. Na Figura 5. 3, ilustra-se alguns dos resultados desta etapa.

Figura 5. 3 - Resultado do processo de segmentação de caracteres. (a) Exemplos bem sucedidos. (b) Erros na segmentação.



(a)

(b)

Fonte: Elaborado pelo autor.

5.3 RESULTADOS DO RECONHECIMENTO.

Como foi mencionado na Seção 4.4, para classificar os caracteres foram treinados dois classificadores SVM multiclasse denominados SVM-Letras e SVM-Números. Para o seu treinamento foram utilizados os caracteres segmentados das imagens da base de dados 1 (treinamento).

Em razão de que alguns exemplos de letras apresentam uma quantidade pequena de amostras por causa da distribuição não uniforme na base de dados conforme apresentado na Tabela 5. 3, foram geradas artificialmente algumas amostras adicionando ruído ou variando a sua rotação. Assim, o SVM-Letras foi treinado com 26 classes onde cada uma das classes teve pelo menos 80 amostras. Já para treinar o classificador SVM-Números constituído por 10 classes foram usadas a quantidade de amostras originais adquiridas na segmentação, apresentadas na Tabela 5. 4.

Para o treinamento utilizou-se a validação cruzada *k-folds* com um valor de k=10. O tipo de Kernel usado foi um RBF (*Radial Basis Function*), de forma similar aos trabalhos propostos por (ZHAO et al., 2008), (HO; LIM; TAY, 2009) , (YANG, 2011) e (GOU et al., 2016). Já os valores do hiperplano C e do fator gamma (γ) foram de 10 e 0,54 respetivamente [6].

Finalmente os erros e acertos do processo de classificação tanto dos números quanto das letras são apresentados em duas matrizes de confusão ilustradas na Tabela 5. 5 e Tabela 5. 6. Através dessas matrizes é possível analisar o desempenho da etapa do reconhecimento dos caracteres.

[6] Esses valores foram encontrados através da função ml::SVM::trainAuto fornecida pela biblioteca OpenCV. Informação adicional desta função encontra-se em http://docs.opencv.org/3.0-beta/modules/ml/doc/support_vector_machines.html e (KAEHLER; BRADSK, 2016).

Tabela 5. 5 - Matriz de confusão para classificação dos números.

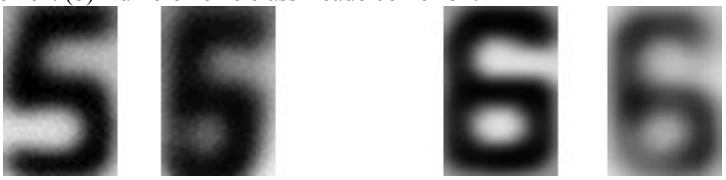
		Classificação Números										
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**Acurácia**
Ground Truth	**0**	365	1	0	0	0	0	0	0	1	0	99,40%
	1	0	295	0	0	0	1	0	2	0	1	98,60%
	2	0	0	276	0	0	1	0	2	0	0	98,90%
	3	1	0	0	308	0	1	0	0	2	1	98,40%
	4	0	2	1	0	309	0	3	0	0	0	98%
	5	1	0	0	1	0	302	8	0	0	0	96,80%
	6	0	0	0	0	0	6	279	0	2	0	97,20%
	7	0	1	1	0	0	0	0	330	0	2	98,00%
	8	0	0	0	0	0	0	2	0	320	1	99,10%
	9	0	0	0	0	0	2	0	0	2	311	98,70%

Fonte: Desenvolvido pelo autor.

Na matriz de confusão correspondente aos números é possível observar como são confundidos os números 5 e 6, motivo pelo qual, a acurácia nesses dois algarismos é a mais baixa se comparada com os outros.

A Figura 5. 4 (a) ilustra um exemplo onde um número "5" é reconhecendo como um número "6", isso é devido principalmente ao alto contraste na imagem classificada. Porém, pode ser considerado como um ruído aceitável dentro do algoritmo já que não se apresenta em muitas ocasiões. Já o caso contrário é apresentado na Figura 5. 4 (b), aqui o número "6" é classificado como "5" em razão do desgaste do caractere apresentado na imagem.

Figura 5. 4 - Exemplo de erro no reconhecimento. (a) Número "5" é classificado como "6". (b) Número "6" é classificado como "5".



Fonte: Elaborado pelo autor.

Com tudo, o reconhecimento dos números atingiu uma acurácia ou CCR de 0,983 que representa 98,3%, valor bom se comparado com os valores obtidos em diferentes trabalhos propostos.

Na Tabela 5. 6 é apresentada uma versão simplificada da matriz de confusão relativa às letras, a fim de ter uma melhor visualização. Aqui, são apresentadas as linhas que tiveram a menor acurácia e as colunas onde são apresentados os erros de classificação. Já no apêndice A é possível observar a matriz com os resultados de todas as 26 classes.

Como primeira observação, pode se notar que a distribuição das letras no *dataset* de teste não é uniforme. Sendo as primeiras letras do abecedário (A, B, C, D, E) as que mais aparecem.

Tabela 5. 6 - Matriz de confusão simplificada para classificação das letras.

		Classificação Letras														
		D	**E**	**F**	**G**	**M**	**N**	**O**	**P**	**Q**	**R**	**S**	**U**	**V**	**W**	**Acurácia**
Ground Truth	**D**	243	0	0	0	0	0	29	0	11	0	0	0	0	0	85,50%
	E	0	124	3	0	0	0	0	0	0	0	1	0	0	0	96,80%
	F	0	2	77	0	0	0	0	2	0	0	0	0	0	0	95,10%
	G	1	0	0	95	0	0	1	0	0	0	1	0	0	0	96,90%
	M	0	0	0	0	68	2	0	0	0	0	0	0	0	1	90,60%
	N	0	0	0	0	3	53	0	0	0	0	0	0	2	0	89,80%
	O	9	0	0	0	0	0	81	0	3	0	0	0	0	0	87,10%
	P	0	0	2	0	0	0	0	69	0	0	0	0	0	0	95,80%
	Q	3	0	0	0	0	0	5	0	48	0	0	0	0	0	85,70%
	R	0	0	0	0	0	0	0	0	0	96	0	0	0	0	96,90%
	U	0	0	0	0	0	0	1	0	0	0	0	46	1	0	95,80%
	V	0	0	0	0	0	0	0	0	0	0	0	1	49	2	92,40%
	W	0	0	0	0	2	3	0	0	0	0	0	0	0	46	90,20%

Fonte: Elaborado pelo autor.

Outra observação é a baixa acurácia no reconhecimento das letras “D”, “O” e “Q”. Isto é devido principalmente a não padronização das letras nas placas veiculares Brasileiras, o que faz com que sejam apresentados erros como os apresentados na Figura 5. 5, Porém, segundo a Brascontrol e a pesquisa feita por (CASTELAN, 2009), em sistemas de reconhecimento empresariais e comercias Brasileiros não são considerados estes caracteres para o cálculo de taxa de acerto total. Contudo, a acurácia do reconhecimento das letras, tendo em conta as baixas taxas de acerto desses três caracteres, foi de 95,1%.

Figura 5. 5 – Erro no reconhecimento.


Ground Truth: DDI 0984
Result: : OOI 0984


Ground Truth: DOO 9120
Result: : DDO 9120


Ground Truth: DQR 0388
Result: : DDR 0388

Fonte: Elaborado pelo autor.

Por fim, para avaliar esta etapa, foram considerados dois critér~~ios que são apresentados nas equações (5.4) (5.5).

$$AR1 = \frac{NCR}{NCS} \tag{5. 4}$$

$$AR2 = \frac{NPR}{NPV} \tag{5. 5}$$

Onde NCR é o número total de caracteres reconhecidos satisfatoriamente e NCS o número de total de caracteres segmentados na etapa anterior, que para este caso são 5502, correspondente às 786 imagens de teste. Já na equação (5.5), NPR descreve o número total de placas onde os 7 caracteres foram reconhecidos satisfatoriamente e NPR o número total de placas veiculares usadas para o teste (786). Nesse contexto os valores de $AR1$ e $AR2$ foram:

$$AR1 = \frac{5333}{5502} = 0,97 \tag{5. 6}$$

$$AR2 = \frac{754}{786} = 0,96 \tag{5. 7}$$

Neste ponto é importante relatar que o valor utilizado para avaliar a acurácia total do sistema foi $AR2$, lembrando que este foi obtido sem considerar os erros gerados pelas letras “D”, “O”, “Q”.

Assim, a acurácia total do sistema é dada pela seguinte equação:

$$Acuracia\ Total = (D \times S \times R)\% \tag{5. 8}$$

Onde D é a taxa de acerto na etapa de detecção e segmentação da placa veicular, S a taxa de acerto da segmentação dos caracteres e R o valor concernente à taxa de acerto para o reconhecimento dos caracteres. Assim, a acurácia total é:

$$Acuracia\ Total = (0,98 \times 0,97 \times 0,97)\% = 92,2\% \quad (5.\ 9)$$

Por fim, os experimentos realizados mostraram um tempo médio de processamento para cada imagem de teste de 446,75 ms para a etapa de detecção da placa e 341 ms na segmentação e reconhecimento dos caracteres.
Na Tabela 5. 7 é apresentado um comparativo entre a metodologia proposta e algumas outras técnicas encontradas na literatura.

Tabela 5. 7 - Comparação metodologia proposta e trabalhos da literatura.

Sistema	Acuracia Reportada				Tipo de placas	Nro de placas	Tempo de Procesamento (ms)	Condiçoes das imagens
	Detecçao	Segmentação	Reconhecimento	Acuracia Total				
(CASTELAN, 2009)	91,75%	80,10%	62%	45,50%	Brasileiras	170	1277	Captura em rodovias. Iluminaçao, contraste e angulos variantes. Desgaste de placas.Diferentes tamanhos e perspecitvas.
(CHEN et al., 2009)	97,10%	NR	96,4%	93,60%	Chinesas	332	594	Captura em rodovias. Iluminaçao e contraste variantes. Diferentes tamanhos.
(ZHENG et al., 2012)	94,0%	NR	NR	NR	Chinesas	3000	76	estacionamento, iluminaçao e contraste fixo, ângulo fixo.
(NETO et al., 2015)	97,0%	NR	96,70%	94,00%	Brasileiras	12000	400	Captura com Camera convencional, ângulo fixo, iluminaçao e contraste fixos, tamanho fixo.
(GOU et al., 2016)	95,90%	NR	94,10%	94,10%	Chinesas	4242	470	Captura em rodovias. Iluminaçao, contraste e angulos variantes. Desgaste de placas.Diferentes tamanhos e perspecitvas.
(CORNETO et al., 2017)	92,55%	NR	88,40%	82,0%	Brasileiras	1410	500	Captura em estacionamento, iluminaçao variante, ângulo fixo. Desgaste de placas
Proposto	98,0%	97,0%	97,0%	92,2%	Brasileiras	850	788	Captura em rodovias. Iluminaçao, contraste e angulos variantes. Desgaste de placas.Diferentes tamanhos e perspecitvas.

NR: Não reportado

Fonte: Elaborado pelo autor.

Na Tabela 5. 7 é importante considerar que cada uma das metodologias propostas foi avaliada em suas próprias bases de dados, com exceção do trabalho de (CASTELAN, 2009), onde as imagens utilizadas foram extraídas da mesma base de dados usada para desenvolver o método aqui proposto. No entanto, algumas condições das imagens são similares, daí que possa ser feita uma comparação mais intuitiva.

Observa-se que a maioria dos trabalhos encontrados na literatura não possibilitam resultados referentes a acurácia no processo de segmentação, fazendo o cálculo da acurácia total somente com os valores de detecção e reconhecimento dos caracteres.

Os trabalhos de (CHEN et al., 2009) e (GOU et al., 2016) apresentam tipos de imagens onde as condições de iluminação, contraste e perspectiva são similares às fornecidas pela Brascontrol. Nesse sentido, a acurácia total atingida pelo sistema proposto é parecida à encontrada na literatura. Além disso, segundo a (DENATRAN, 2007), o valor mínimo de acurácia para que um sistema de reconhecimento possa ser usado em aplicações comerciais é de 90%.

Em relação ao tempo de processamento, a metodologia proposta apresenta um maior valor se comparada com as propostas da Tabela. Isso é devido principalmente a que a técnica de CCA tem um maior custo computacional.

6 CONCLUSÃO

Neste trabalho, foi desenvolvida uma proposta para a detecção, segmentação e Reconhecimento de placas veiculares Brasileiras em cenários abertos onde são apresentadas condições de iluminação, contraste e ângulos variantes, além de diferentes perspectivas, tamanhos e *backgrounds* complexos.

Para desenvolver a proposta aqui citada, efetuou-se uma revisão bibliográfica das técnicas mais relevantes para reconhecer placas veiculares. Nesse contexto verificou-se que são poucos os trabalhos encontrados na literatura que fazem referência à detecção e o reconhecimento das placas brasileiras em ambientes reais. Isto é devido principalmente a que as bases de dados existentes são privativas e de uso empresarial. Por esta razão a proposta foi desenvolvida levando em conta técnicas usadas em trabalhos relacionados com imagens de trafego Chinês.

Através do esquema de pré-processamento da imagem de entrada proposto neste trabalho foi possível minimizar os efeitos de sombras, baixo contraste e iluminação não uniforme nas placas. Assim, por intermédio do filtro de Sobel, a técnica de CCA e as operações matemáticas aqui mostradas foi possível atingir uma taxa de acerto boa na etapa de detecção e segmentação da placa se comparada com os trabalhos presentes na literatura. Porém, o custo computacional que demanda o CCA faz com que o tempo de processamento aumente em relação ao tempo das propostas estudadas. Esse aumento é devido principalmente ao fato de usar o filtro MMLPF, já que o aumento da qualidade da imagem fará que sejam acentuados muitos detalhes considerados como ruídos, além de realçar os caracteres da placa.

Por outra parte o classificador binário SVM – HOG utilizado para reduzir os falsos positivos gerados na etapa detecção da placa mostrou melhores resultados aos obtidos por (ZHENG et al., 2012) onde foi usado o mesmo descritor de características e um classificador em cascata para a classificar as imagens.

Através das experimentações realizadas na etapa de segmentação dos caracteres comprovou-se a premissa de que a eficiência desta etapa depende em grande medida do método de binarização que seja utilizado (PAN; YAN; XIAO, 2008), (DU et al., 2013). Nesse contexto a técnica aqui proposta, baseada no limiar adaptativo Gaussiano (BRADLEY; ROTH, 2007) fez com que a etapa de segmentação dos caracteres atingira resultados similares aos trabalhos propostos na literatura. No entanto, a

técnica apresenta limitações na binarização de caracteres que evidenciam uma alta degradação, fazendo com que o resultado da detecção seja infrutífero nestes casos, como apresentado na Figura 5. 3 (b).

Quanto à etapa de reconhecimento dos caracteres, foi possível comprovar que a utilização dos classificadores SVM-HOG, para reconhecer tanto as letras como os números influenciou diretamente na acurácia global desta etapa, posto que foram evitados erros produto da confusão entre caracteres como “S” e o número “5”, a letra “I” e o número “1”, as letras “O”, “Q”, “D” e o número “0”.

Os testes verificaram que a acurácia do classificador SVM-HOG para números é maior que a acurácia apresentada pelo SVM-HOG para letras. Isto é devido principalmente ao menor número de classes que tem os números (10) em relação às letras (26), além da diferencia entre o número de exemplos de treinamento usados para cada um dos classificadores. Para o treinamento do classificador de números foram usados em média 347 exemplos por classe, enquanto que para o classificador de letras o número de exemplos foi de 100, lembrando que foram gerados alguns exemplos artificiais de letras. Por outra parte a tarefa do reconhecimento das letras é mais complexa no sentido de que algumas letras são facilmente confundíveis entre si, exemplo disso são os grupos (“M”, “N”, “W”) e (“O”, “Q”, “D”).

O resultado da acurácia final no reconhecimento foi calculado sem considerar os erros gerados pelo grupo de letras (“O”, “Q”, “D”). Isto devido à não padronização das placas Brasileiras (CASTELAN, 2009). Dessa forma, a acurácia total do sistema tendo em consideração as três etapas do sistema, foi de 92,2%.

Por outra parte, se fossem considerados os erros gerados pelo grupo de letras mencionado anteriormente a acurácia total do sistema seria de 90,4%, cumprindo assim com a condição mínima de que um método pode ser considerado comercial no Brasil se atingir um mínimo de acerto de 90% (DENATRAN, 2007) (NETO et al., 2015).

6.1 TRABALHOS FUTUROS.

Uma das limitações da proposta desenvolvida neste trabalho é o tempo de processamento total se comparado com alguns dos trabalhos encontrados na literatura. Para diminuir esse tempo podem ser exploradas soluções na etapa de detecção e segmentação da placa veicular que visem minimizar a quantidade de ruído gerado pelo filtro MMLPF, dessa maneira o etiquetado dos componentes por parte do CCA será menor.

Outra alternativa para diminuir o tempo de processamento, além de melhorar a acurácia na classificação dos caracteres, pode ser utilizar técnicas de redução de dimensionalidade dos vetores HOG resultantes, tais como LDA (*Linear Discriminant Analysis*) ou PCA *(Principal Componente Analysis).*

Para conseguir segmentar e reconhecer eficientemente os caracteres desgastados ou que apresentam pouca conectividade, podem ser exploradas propostas como as apresentadas por (ZEMOURI; CHIBANI; BRIK, 2014) (D; HOLI, 2015) que visam a restauração e binarização de caracteres em documentos históricos.

REFERÊNCIAS

AHMAD, I. S. et al. Automatic license plate recognition: A comparative study. **2015 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)**, p. 635–640, 2015.

AIZERMAN, M. A.; BRAVERMAN, E. A.; ROZONOER, L. Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning. **Automation and Remote Control**, v. 25, p. 821–837, 1964.

ALI, M. et al. A Review on License Plate Recognition System Algorithms. n. May, p. 84–89, 2016.

ANAGNOSTOPOULOS, C. N. E. et al. License plate recognition from still images and video sequences: A survey. **IEEE Transactions on Intelligent Transportation Systems**, v. 9, n. 3, p. 377–391, 2008.

ANGELINE, L. et al. License Plate Character Recognition via Signature Analysis and Features Extraction. **2012 Fourth International Conference on Computational Intelligence, Modelling and Simulation**, p. 1–6, 2012.

ARULMOZHI, K. et al. Skew detection and correction of Indian vehicle license plate using polar Hough Transform research. **2012 IEEE International Conference on Computational Intelligence and Computing Research, ICCIC 2012**, p. 0–3, 2012.

BARBOSA, B. B. B.; SILVA, J. C. Interação Humano - Computador usando Visão Computacional. **Revista TECCEN**, v. 2, n. 1, p. 9–16, 2009.

BERNARDI, E. **Os sistemaas de indentificação veicular, em especial o reconhecimento automatico de placas**. [s.l.] Escola politecnica da Universidade de São Paulo, 2015.

BRADLEY, D.; ROTH, G. Adaptive thresholding using the integral image. **ACM J. Graph. Game Tools**, v. 12, n. 2, p. 13–21, 2007.

BUCH, N.; VELASTIN, S. A.; ORWELL, J. A Review of Computer Vision Techniques for the Analysis of Urban Traffic. **IEEE Transactions on Intelligent Transportation Systems**, v. 12, n. 3, p. 920–939, 2011.

BURGES, C. J. C. A Tutorial on Support Vector Machines for Pattern Recognition. **Data Mining and Knowledge Discovery**, v. 2, n. 2, p. 121–167, 1998.

CANNY, J. A Computational Approach to Edge Detection. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. PAMI-8, n. 6, p. 679–698, 1986.

CASTELAN, L. **Reconhecimento Automatico de Placas Veiculares (RAPV)**. [s.l.] Universidade Federal de Santa Catarina, 2009.

CHANG, C.; LIN, C. LIBSVM : A Library for Support Vector Machines. **ACM Transactions on Intelligent Systems and Technology (TIST)**, v. 2, p. 1–39, 2013.

CHEN, Z. X. et al. Automatic License-Plate Location and Recognition Based on Feature Salience. **IEEE Transactions on Vehicular Technology**, v. 58, n. 7, p. 3781–3785, 2009.

CONNELL, S. D.; JAIN, A. K. Template-based online character recognition. v. 34, n. August 1999, p. 1–14, 2001.

CORNETO, G. L. et al. A New Method for Automatic Vehicle License Plate Detection. v. 15, n. 1, p. 75–80, 2017.

D, R.; HOLI, G. **Historical document enhancement using Shearlet Transform and mathematical morphological operations**. 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI). **Anais**...2015

DAENGDUANG, S.; VATEEKUL, P. **Enhancing accuracy of multi-label classification by applying one-vs-one support vector machine**. 2016 13th International Joint Conference on Computer Science and Software Engineering (JCSSE). **Anais**...2016

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. **Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005**, v. I, p. 886–893, 2005.

DALIDA, J. P. D. et al. Development of Intelligent Transportation System for Philippine License Plate Recognition. **Proceedings of the International Conference - 2016 IEEE Region 10 Conference (TENCON)**, v. 1, p. 3766–3770, 2016.

DE QUEIROZ, L. C. **Estado da motorização individual no Brasil-Relatorio 2015**. Rio de Janeiro: [s.n.]. Disponível em: <http://www.observatoriodasmetropoles.net/download/automoveis_e_motos2015.pdf >.

DENATRAN. **CONTRAN - Sistema de Placas de Identificação de Veículos.** Brasilia, Brasil: [s.n.].

DU, S. et al. Automatic license plate recognition (ALPR): A state-of-the-art review. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 23, n. 2, p. 311–325, 2013.

DUDA, R. O.; HART, P. E. Use of the Hough transformation to detect lines and curves in pictures. **Communications of the ACM**, v. 15, n. 1, p. 11–15, 1972.

ELBAMBY, A.; HEMAYED, E. E. Real-Time Automatie Multi-Style Lieense Plate Deteetion in Videos. p. 148–153, 2016.

FACON, J. A Morfologia Matemática e suas Aplicações em Processamento de Imagens. **VII Workshop de Visao Computacional – WVC 2011**, p. 61–128, 2011.

GEHLER, P. V.; SCHÖLKOPF, B. An introduction to kernel learning algorithms. v. 12, n. 2, p. 181–201, 2009.

GONZALEZ, R. C.; WOODS, R. E. **Digital Image Processing (3rd Edition)**. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006.

GOU, C. et al. License Plate Recognition Using MSER and HOG Based on ELM. p. 217–221, 2014.

GOU, C. et al. Vehicle License Plate Recognition Based on Extremal Regions and Restricted Boltzmann Machines. **IEEE Transactions on Intelligent Transportation Systems**, v. 17, n. 4, p. 1096–1107, 2016.

HARALICK, R. M.; STERNBERG, S. R.; ZHUANG, X. Image Analysis Using Mathematical Morphology. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. PAMI-9, n. 4, p. 532–550, 1987.

HO, W. T.; LIM, H. W.; TAY, Y. H. **Two-Stage License Plate Detection Using Gentle Adaboost and SIFT-SVM**. 2009 First Asian Conference on Intelligent Information and Database Systems. **Anais**...2009

HUANG, G. BIN; ZHU, Q. Y.; SIEW, C. K. Extreme learning machine: Theory and applications. **Neurocomputing**, v. 70, n. 1–3, p. 489–501, 2006.

JAIN, N.; LALA, A. **Image segmentation: A short survey**. Confluence 2013: The Next Generation Information Technology Summit (4th International Conference). **Anais**...2013

JIAO, J.; YE, Q.; HUANG, Q. A configurable method for multi-style license plate recognition. **Pattern Recognition**, v. 42, n. 3, p. 358–369, 2009.

KAEHLER, A.; BRADSK, G. . **Learning OpenCV 3 Computer Vision in C++ with the OpenCV Library**. Sebastopol, CA: O'Reilly Media, 2016.

KNERR, S.; PERSONNAZ, L.; DREYFUS, G. Single-layer learning revisited: a stepwise procedure for building and training a neural network. In: SOULIÉ, F. F.; HÉRAULT, J. (Eds.). . **Neurocomputing: Algorithms, Architectures and Applications**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990. p. 41–50.

KULKARNI, P. et al. **License Plate Recognition: A review**. 2012 Fourth International Conference on Advanced Computing (ICoAC). **Anais**...2012

LAGANIÈRE, R. **Opencv 2 Computer Vision Application Programming Cookbook**. Birmingham, UK: Packy Publishing LTd, 2011.

LIN, X.; OTOBE, K. Hough transform algorithm for real-time pattern recognition using an artificial retina camera. **Opt. Express**, v. 8, n. 9, p. 503–508, 2001.

LIU, D.; YU, J. **Otsu Method and K-means**. Proceedings of the 2009 Ninth International Conference on Hybrid Intelligent Systems - Volume 01. **Anais**...: HIS '09.Washington, DC, USA: IEEE Computer Society, 2009Disponível em: <http://dx.doi.org/10.1109/HIS.2009.74>

MALLICK, S. **Learn Opencv**. Disponível em: <http://www.learnopencv.com/histogram-of-oriented-gradients/>. Acesso em: 1 jan. 2017.

MARIA, E. et al. O uso de geotecnologias em sistemas de transporte e organização urbana no brasil. **Mercator**, v. 13, n. 1, p. 143–152, 2014.

MASCARO, A. A. **Detecção de inclinação em imagens de documentos**. [s.l.] Universidade Federal de Pernambuco, 2010.

MATAS, J. et al. Robust Wide Baseline Stereo from Maximally Stable Extremal Regions. **British Machine Vision Conference 2002**, p. 384–393, 2002.

MATAS, J.; GALAMBOS, C.; KITTLER, J. Progressive Probabilistic Hough Transform. **Procedings of the British Machine Vision Conference 1998**, p. 26.1-26.10, 1998.

MATAS, J.; GALAMBOS, C.; KITTLER, J. Robust Detection of Lines Using the Progressive Probabilistic Hough Transform. **Computer Vision and Image Understanding**, v. 78, n. 1, p. 119–137, 2000.

MERCER, J. Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations. **Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences**, v. 209, n. 441–458, p. 415–446, 1909.

MILGRAM, J.; CHERIET, M.; SABOURIN, R. "One Against One" or "One Against All": Which One is Better for Handwriting Recognition with SVMs? **Tenth**

International Workshop on Frontiers in Handwriting Recognition, p. 1–6, 2006.

MUKHERJEE, D. A Fast Two Pass Multi-Value Segmentation Algorithm based on Connected Component Analysis. **CoRR**, v. abs/1402.2, 2014.

NEJATI, M.; MAJIDI, A.; JALALAT, M. License Plate Recognition Based On Edge Histogram Analysis and Classifier Ensemble. p. 16–17, 2015.

NETO, E. C. et al. Brazilian vehicle identification using a new embedded plate recognition system. **Measurement: Journal of the International Measurement Confederation**, v. 70, p. 36–46, 2015.

NIBLACK, W. **An Introduction to Image Processing**. `Prentice- ed. New York, NY, USA: [s.n.].

OLAGUE, G. **Evolutionary Computer Vision**. Proceedings of the 9th Annual Conference Companion on Genetic and Evolutionary Computation. **Anais**...: GECCO '07.New York, NY, USA: ACM, 2007Disponível em: <http://doi.acm.org/10.1145/1274000.1274121>

OTSU, N. A Threshold Selection Method from Gray-Level Histograms. **IEEE Transactions on Systems, Man, and Cybernetics**, v. 9, n. 1, p. 62–66, 1979.

PAN, M.-S.; YAN, J.-B.; XIAO, Z.-H. Vehicle license plate character segmentation. **International Journal of Automation and Computing**, v. 5, n. 4, p. 425–432, 2008.

PAPAGEORGIOU, C.; POGGIO, T. Trainable system for object detection. **International Journal of Computer Vision**, v. 38, n. 1, p. 15–33, 2000.

PEDRINI, H.; SCHWARTZ, W. **Análise de imagens digitais: Principios, algoritmos e aplicaçoes**. Sao Paulo: [s.n.].

PRABHAKAR, P.; ANUPAMA, P.; RESMI, S. R. **Automatic vehicle number plate detection and recognition**. 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT). **Anais**...2014

PREWITT, J. M. . object Enhancement and Extraction. **Picture processing psychopictorics**, 1970.

ROBERTS, L. GI. Machine perception of three-dimensional solids. **PhD Thesis**, n. January 1963, p. 159–197, 1965.

ROY, A. C.; HOSSEN, M. K.; NAG, D. **License plate detection and character recognition system for commercial vehicles based on morphological approach and template matching**. 2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT). **Anais**...2016

RYCHETSKY, M. **Algorithms and Architectures for Machine Learning Based on Regularized Neural Networks and Support Vector Approaches.** Shaker ed. Shaker Verlag GmbH, Germany: Berichte Aus Der Informatik, 2001.

SARFRAZ, M.; AHMED, M. J.; GHAZI, S. A. Saudi Arabian license plate recognition system. **2003 International Conference on Geometric Modeling and Graphics, 2003. Proceedings**, p. 36–41, 2003.

SAUVOLA, J.; PIETIKÄINEN, M. Adaptive document image binarization. **Pattern Recognition**, v. 33, n. 2, p. 225–236, 2000.

SCHAPIRE, R. E. et al. Boosting the margin: A new explanation for the

effectiveness of voting methods. **Annals of Statistics**, v. 26, n. 5, p. 1651–1686, 1998.

SHAWE-TAYLOR, J.; CRISTIANINI, N. **Kernel Methods for Pattern Analysis**. New York, NY, USA: Cambridge University Press, 2004.

SILVA, F. A. et al. Aplicação do pré-processamento de imagens para otimização do reconhecimento de padrões na detecção de deficiência nutricional em espécies vegetais. **Anais XVII Simpósio Brasileiro de Sensoriamento Remoto - SBSR**, n. 2010, p. 6381–6388, 2015.

SMOLA, A J.; SCHÖLKOPF, B. A tutorial on support vector regression. **Statistics and Computing**, v. 14, p. 199–222, 2004.

SOBEL, I. An isotropic 3 by 3 image gradient operator. **Machine Vision for three-demensional Sciences**, v. 1, n. 1, p. 23–34, 1990.

SOILLE, P. **Morphological Image Analysis: Principles and Applications**. 2. ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2003.

SU, A. et al. Efficient rotation-invariant histogram of oriented gradient descriptors for car detection in satellite images. **IET Computer Vision**, v. 10, n. 7, p. 634–640, 2016.

SUBHADHIRA, S. et al. License plate recognition application using extreme learning machines. **2014 Third ICT International Student Project Conference (ICT-ISPC)**, p. 103–106, 2014.

SUGIHARTO, A.; HARJOKO, A. **Traffic sign detection based on HOG and PHOG using binary SVM and k-NN**. 2016 3rd International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE). **Anais**...2016

SUZUKI, S.; BE, K. Topological structural analysis of digitized binary images by border following. **Computer Vision, Graphics, and Image Processing**, v. 30, n. 1, p. 32–46, 1985.

TASKIRAN, M.; CAM, Z. G. **Offline signature identification via HOG features and artificial neural networks**. 2017 IEEE 15th International Symposium on Applied Machine Intelligence and Informatics (SAMI). **Anais**...jan. 2017

TOBERGTE, D. R.; CURTIS, S. **Algorithms for image prcessing and computer vision**. [s.l: s.n.]. v. 53

VALVENY, E. **Detector Basado en HOG/SVM**. Barcelona, Catalunya, SPAIN: [s.n.].

VAPNIK, V. N. **Statistical Learning Theory**. [s.l.] Wiley-Interscience, 1998.

VIOLA, P.; JONES, M. J. Robust Real-Time Face Detection. **International Journal of Computer Vision**, v. 57, n. 2, p. 137–154, 2004.

WANG, Y. et al. **Adaptive binarization: A new approach to license plate characters segmentation**. 2012 International Conference on Wavelet Analysis and Pattern Recognition. **Anais**...2012

WU, B. F. et al. Degraded License Plate Recognition system for town buses on highway. **2013 10th IEEE International Conference on Networking, Sensing and Control, ICNSC 2013**, p. 446–450, 2013.

WU, H. H. P. et al. License plate extraction in low resolution video. **Proceedings - International Conference on Pattern Recognition**, v. 1, p. 824–827, 2006.

YANG, G. **License plate character recognition based on wavelet kernel LS-SVM**. 2011 3rd International Conference on Computer Research and Development. **Anais**...2011

YE, Q.; GAO, W.; ZENG, W. Color image segmentation using density-based clustering. **Acoustics, Speech, and Signal …**, v. 3, p. III-345-8, 2003.

ZAHEDI, M.; SALEHI, S. M. License plate recognition system based on SIFT features. **Procedia Computer Science**, v. 3, p. 998–1002, 2011.

ZEMOURI, E. T.; CHIBANI, Y.; BRIK, Y. **Restoration based Contourlet Transform for historical document image binarization**. 2014 International Conference on Multimedia Computing and Systems (ICMCS). **Anais**...2014

ZHANG, T. Y.; SUEN, C. Y. A Fast Parallel Algorithm for Thinning Digital Patterns. **Commun. ACM**, v. 27, n. 3, p. 236–239, 1984.

ZHAO, H. et al. **License plate recognition system based on morphology and LS-SVM**. 2008 IEEE International Conference on Granular Computing. **Anais**...2008

ZHENG, D.; ZHAO, Y.; WANG, J. An efficient method of license plate location. **Pattern Recognition Letters**, v. 26, n. 15, p. 2431–2438, 2005.

ZHENG, K. et al. License plate detection using Haar-like features and histogram of oriented gradients. **IEEE International Symposium on Industrial Electronics**, p. 1502–1505, 2012.

APÊNDICE A – Matriz de confusão Letras

Ground Truth \ Classificação	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
A	110	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
B	0	119	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0
C	0	0	222	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	1	0	243	0	0	0	0	0	0	0	0	0	0	29	0	11	0	0	0	0	0	0	0	0	0
E	0	0	0	0	124	3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
F	0	0	0	0	2	77	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
G	0	0	0	1	0	0	95	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	71	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	0	77	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
J	0	0	0	0	0	0	0	0	0	61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
K	0	0	0	0	0	0	0	0	0	0	55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L	0	0	0	0	0	0	0	0	0	0	0	68	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M	0	1	0	0	0	0	0	3	0	0	0	0	68	2	0	0	0	0	0	0	0	0	1	0	0	0
N	0	0	0	0	0	0	0	1	0	0	0	0	3	53	0	0	0	0	0	0	0	2	0	0	0	0
O	0	0	0	9	0	0	0	0	0	0	0	0	0	0	81	0	3	0	0	0	0	0	0	0	0	0
P	1	0	0	0	0	2	0	0	0	0	0	0	0	0	0	69	0	0	0	0	0	0	0	0	0	0
Q	0	0	0	3	0	0	0	0	0	0	0	0	0	0	5	0	48	0	0	0	0	0	0	0	0	0
R	1	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	96	0	0	0	0	0	0	0	0
S	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	77	0	0	0	0	0	0	0
T	0	0	0	0	0	0	0	2	0	0	1	0	0	0	0	0	0	0	0	91	0	0	0	0	0	0
U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	46	1	0	0	0	0
V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	49	2	0	1	0
W	0	0	0	0	0	0	0	0	0	0	0	0	2	3	0	0	0	0	0	0	0	0	46	0	0	0
X	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	68	0	0
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	74	0
Z	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	50

Fonte: Elaborada pelo autor (2016)