UNIVERSIDADE ESTADUAL DO RIO GRANDE DO SUL
UNIDADE UNIVERSITÁRIA EM PORTO ALEGRE
CURSO SUPERIOR DE TECNOLOGIA EM AUTOMAÇÃO INDUSTRIAL

LÍGIA PAULA CASTRO DE MIRANDA

ESTUDO DA VIABILIDADE DA CONSTRUÇÃO DE DISPOSITIVO DE BAIXO CUSTO PARA O RECONHECIMENTO DE PLACAS VEICULARES

PORTO ALEGRE
2021

LÍGIA PAULA CASTRO DE MIRANDA

ESTUDO DA VIABILIDADE DA CONSTRUÇÃO DE DISPOSITIVO DE BAIXO CUSTO PARA O RECONHECIMENTO DE PLACAS VEICULARES

Monografia apresentada como requisito parcial para obtenção do título de Tecnólogo em Automação Industrial na Universidade Estadual do Rio Grande do Sul.

Orientador: Prof. Me. Emerson Fernandes da Cunha.

PORTO ALEGRE

2021

C512c Miranda, Lígia Paula Castro de.

ESTUDO DA VIABILIDADE DA CONSTRUÇÃO DE DISPOSITIVO DE BAIXO CUSTO PARA O RECONHECIMENTO DE PLACAS VEICULARES/ Lígia Paula Castro de Miranda – Porto Alegre, 2021.

46f.

Orientador: Prof. Me. Emerson Fernandes da Cunha.

Trabalho de Conclusão de Curso (Graduação) – Universidade Estadual do Rio Grande do Sul, Curso Superior de Tecnologia em Automação Industrial, Unidade de Porto Alegre, 2021.

LÍGIA PAULA CASTRO DE MIRANDA

ESTUDO DA VIABILIDADE DA CONSTRUÇÃO DE DISPOSITIVO DE BAIXO CUSTO PARA O RECONHECIMENTO DE PLACAS VEICULARES

Monografia apresentada como requisito parcial para obtenção do título de Tecnólogo em Automação Industrial na Universidade Estadual do Rio Grande do Sul.

Orientador: Prof. Me. Emerson Fernandes Cunha

Aprovado em: 22 / 01 / 2021

BANCA EXAMINADORA

Orientador: Prof. Me. Emerson Fernandes Cunha
Universidade Estadual do Rio Grande do Sul – UERGS

Prof. Dr. Marcos Eufebio Mallqui Espinoza
Universidade Estadual do Rio Grande do Sul – UERGS

Prof. Dr. Andre Borin Soares
Universidade Estadual do Rio Grande do Sul – UERGS

PORTO ALEGRE

2021

Dedico este trabalho a minha família, namorado, amigos, professores e colegas que me auxiliaram durante esta trajetória.

AGRADECIMENTOS

Agradeço a minha família, namorado, professores, amigos e colegas que me apoiaram e auxiliaram até o presente momento. Todos tiveram uma parcela de contribuição durante o meu desenvolvimento acadêmico e pessoal. Além de todos os citados agradeço a Universidade por ter concedido a oportunidade de estudar gratuitamente e me proporcionar momentos dos quais sempre irei lembrar, apesar de todas as dificuldades durante a trajetória acadêmica. O aprendizado adquirido nesta trajetória foi de grande valia.

RESUMO

Com o decorrer dos anos a aplicação de sistemas de reconhecimento óptico de caracteres (OCR) cresceu consideravelmente. A aplicação desse sistema pode ser verificada em diversas áreas, tais como na segurança, educação e a inclusão de pessoas com necessidades especiais. Neste projeto é realizado o estudo da viabilidade do reconhecimento de placas veiculares por meio de equipamento acessível e de baixo custo, pois no mercado existem equipamentos que possuem custos elevados que giram em torno de R$ 8.500,00, assim como existem também planos mensais que custam em torno de R$ 2.000,00(com apenas uma câmera). Em comparação com o sistema comercial de R$ 8.500,00 a economia do sistema proposto gira em torno de 66,85% até 74,5%, e com a aplicação do Raspberry Pi pode resultar em uma economia ainda maior, por volta de 85,84%. O reconhecimento de placas veiculares em alguns locais ainda é feito manualmente, por exemplo, em um condomínio é designado a uma pessoa verificar as câmeras e registrar manualmente a placa do veiculo que acessa o local, o que de fato poderia ser realizado por meio de um programa, o qual identificaria a placa e salvaria a imagem para o operador. Este estudo visa à redução de falhas e também do trabalho manual, para isso será elaborado um sistema via algoritmo de reconhecimento de caracteres.

Palavras-chaves: OCR. Reconhecimento de Placas. Reconhecimento de Caracteres.

ABSTRACT

Over the years the application of optical character recognition (OCR) systems considerably. The application of this system can be verified in several areas, such as safety, education and inclusion of people with special needs. In this project, the feasibility study of vehicle license plate recognition through accessible and low-cost access is carried out, as there are no equipment in the market that have high costs, which are around R$ 8,500.00, as well as monthly plans that cost around R$ 2,000.00 (with only one camera). In comparison with the commercial system of R$ 8,500.00 the savings of the proposed system revolves around 66.85% to 74.5%, and with the application of the Raspberry Pi it can result in an even greater savings, around 85,84%. The recognition of vehicle license plates in some locations is still done manually, for example, in a condominium a person is assigned to check the cameras and manually register the license plate of the vehicle that accesses the location, which in fact could be accomplished through a program, which would identify the card and save the image for the operator. This study aims to reduce failures and also manual work, for that a system will be elaborated via character recognition algorithm.

Keywords: OCR. License Plate Recognition. Character Recognition.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

OCR	*Optical Character Recognition* (Reconhecimento Óptico de Caracteres)
RGB	*Red, Green, Blue*
UERGS	Universidade Estadual do Rio Grande do Sul
OpenCV	*Open Source Computer Vision Library*
HD	*High Definition*
FullHD	*Full High Definition*
CPU	*Central Process Unit* (Unidade Central de Processamento)

SUMÁRIO

1. INTRODUÇÃO

O processo de reconhecimento de placa veicular envolve diversas etapas que vão desde a identificação até a extração dos dados, porém o custo elevado de equipamentos associados acabam tornando inviável a aplicação em alguns locais. É empregado na atualidade um método com abordagem trabalhosa e manual nas vias públicas e locais privados. Esse processo ocorre da seguinte forma: um setor responsável pelo monitoramento recebe as imagens capturadas dos veículos e identifica as placas. Propor uma possibilidade para a redução de custos no processo de reconhecimento de placas veiculares é o objetivo deste projeto.

Existem alguns métodos para identificação veicular já conhecidos, sendo subdivido em três métodos diferentes. O primeiro seria a instalação de um dispositivo, por exemplo, RFID (Identificação por radiofrequência) em cada automóvel para que então ocorra o reconhecimento, porém devido ao custo associado e a necessidade de instalação unitária este método acaba sendo oneroso. Outra opção seria a utilização de equipamentos ópticos na faixa do infravermelho, entretanto mesmo com a alta precisão esta opção acaba sendo descartada devido aos seus custos elevados em função dos equipamentos utilizados. A melhor opção, é aquela a qual apresenta um custo mais baixo e com boa precisão, é obtida com a utilização de câmeras convencionais, sem a necessidade de equipamentos com alto custo. O valor resultante desta economia pode, por conseguinte, ser aplicado na melhoria de softwares.

1.1 PROBLEMÁTICA

A dificuldade no processo de reconhecimento de placa veicular é verificada em diversas ocasiões. Isso se deve ao método aplicado em alguns ambientes, tais como estacionamentos ou vias públicas. A forma com que este processo ocorre em alguns locais acaba sendo manual, porém devido ao tempo de verificação da placa e a possibilidade de falha humana esta forma acaba sendo desvantajosa. O tema deste projeto é a proposição de um meio de reconhecimento de placas veiculares com aporte financeiro inferior aos métodos e equipamentos convencionais oferecidos comercialmente.

1.2 HIPÓTESE

Como hipótese considera-se que é possível desenvolver um algoritmo para reconhecimento de placas veiculares sem a utilização de redes neurais na localização da placa, contendo aprendizagem profunda apenas no reconhecimento dos caracteres, no caso no wrapper da biblioteca Tesseract. Assume-se que por meio da linguagem python e a biblioteca OpenCV (Biblioteca de código aberto) o algoritmo possa ser executado de forma mais simples e clara.

1.3 OBJETIVOS

Este trabalho tem como intuito o estudo da melhoria, no que tange a redução de custo no processo de reconhecimento de caracteres em placas veiculares. A possibilidade de substituir sistemas embarcados e equipamentos com custos elevados, assim como trabalho manual por um programa e uma câmera de baixo custo, possibilita que o emprego desta tecnologia possa ser utilizado de forma mais abrangente, superando o limitante em relação ao investimento requerido.

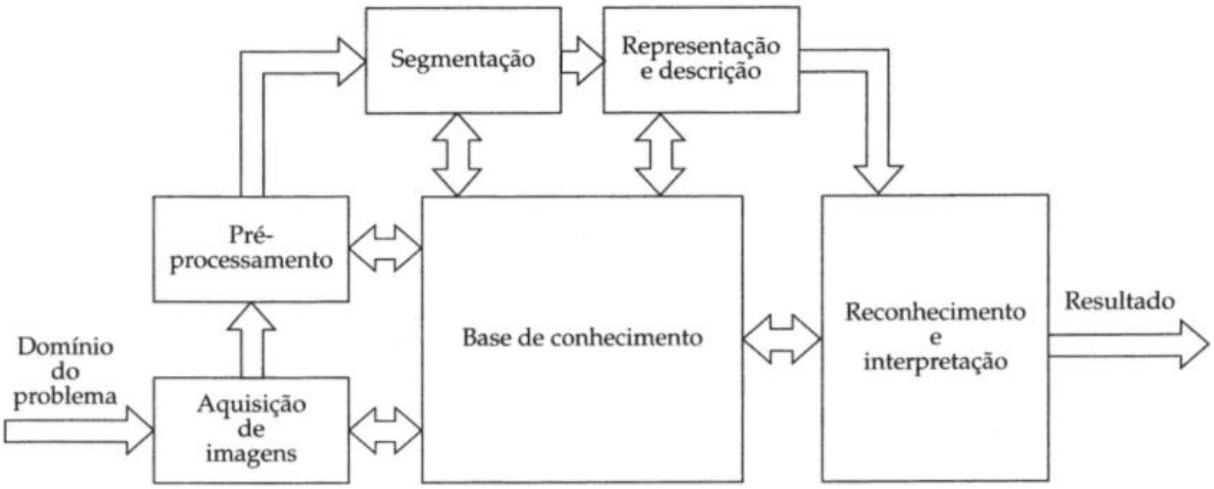
2. REFERENCIAL TEÓRICO

2.1 PROCESSAMENTO DE IMAGENS

O processamento de imagens utiliza diversas técnicas, dependendo das características da imagem a ser processada e da aplicação. O processamento de imagem pode ser dividido em três etapas: pré-processamento, realce e classificação. A primeira etapa é o pré-processamento onde os dados sofrem calibração radiométrica (visa a correção de degradações), ocorrendo remoção de ruído e correção de distorções; o realce melhora a qualidade da imagem e a classificação atribui classes aos objetos presentes na imagem (INPE, [S.d.]).

Além da visão micro de processamento subdividido em três, na visão macro podemos notar mais etapas durante o processamento, sendo elas representadas na Figura 1:

Figura 1 – Fluxograma de Processamento de Imagens.



Fonte: GONZALEZ, C. Rafael; WOODS, E. Richard. 2011

Conforme fluxograma, a aquisição da imagem para análise é primordial para podermos de fato realizar o pré-processamento. Segundo Woods (2011), dois itens são essenciais para obtenção da imagem sendo eles dispositivos físicos. Para ESQUEF e ALBUQUERQUE (2013, p. 3), são sintetizados da seguinte maneira os itens de Woods:

> O primeiro é um dispositivo físico que deve ser sensível ao espectro de energia eletromagnético, como por exemplo ao espectro de raio-x, luz ultravioleta, visível, ou infravermelha. Este dispositivo transdutor deve

> produzir em sua saída um sinal elétrico proporcional ao nível de energia percebido. O segundo, chamado de digitalizador, é um dispositivo que converte o sinal elétrico analógico produzido na saída do sensor em um sinal digital.

O espectro eletromagnético citado pelos autores é caracterizado pela distribuição da intensidade da radiação eletromagnética com relação ao seu comprimento de onda ou frequência. Portanto constitui-se por ondas eletromagnéticas com ampla faixa de comprimentos de onda e frequências de oscilação. As regiões de espectro eletromagnético podem ser verificadas na Tabela 1.

A frequência é caracterizada como o número de oscilações de onda, em um certo período de tempo. Ondas são perturbações que se propagam pelo espaço sem transporte de matéria, apenas de energia, como por exemplo raios-x.

Tabela 1- Regiões do Espectro Eletromagnético

Espectro de Radiação Eletromagnética				
Região	**Comp. Onda (Angstroms)**	**Comp. Onda (centímetros)**	**Frequência (Hz)**	**Energia (eV)**
Rádio	$> 10^{9}$	> 10	$< 3 \times 10^{9}$	$< 10^{-5}$
Micro-ondas	10^{9} - 10^{6}	10 - 0.01	3×10^{9} - 3×10^{12}	10^{-5} - 0.01
Infra-vermelho	10^{6} - 7000	0.01 - 7×10^{-5}	3×10^{12} - 4.3×10^{14}	0.01 - 2
Visível	7000 - 4000	7×10^{-5} - 4×10^{-5}	4.3×10^{14} - 7.5×10^{14}	2 - 3
Ultravioleta	4000 - 10	4×10^{-5} - 10^{-7}	7.5×10^{14} - 3×10^{17}	3 - 10^{3}
Raios-X	10 - 0.1	10^{-7} - 10^{-9}	3×10^{17} - 3×10^{19}	10^{3} - 10^{5}
Raios Gama	< 0.1	$< 10^{-9}$	$> 3 \times 10^{19}$	$> 10^{5}$

Fonte: Observatorio Educativo Itinerante (2010)

Após a aquisição da imagem podem ser verificadas muitas imperfeições. Contudo, existem diversas maneiras de corrigi-las. Alguns exemplos de métodos de ajustes são: realce de bordas, melhoras no brilho e no contraste, correção de iluminação irregular, redução de ruídos, entre outros (Gomes, 2006).

O pré-processamento é a etapa em que a imagem adquirida é analisada e a

partir desta análise ocorre a correção de defeitos da imagem. Portanto para que as próximas etapas sejam bem-sucedidas este estagio de melhoramento da imagem é essencial. Segundo Augusto (2012, p. 58) se “o procedimento de captura for realizado de forma cuidadosa, em condições corretas, não se tornam necessárias muitas operações de correção nas imagens adquiridas”.

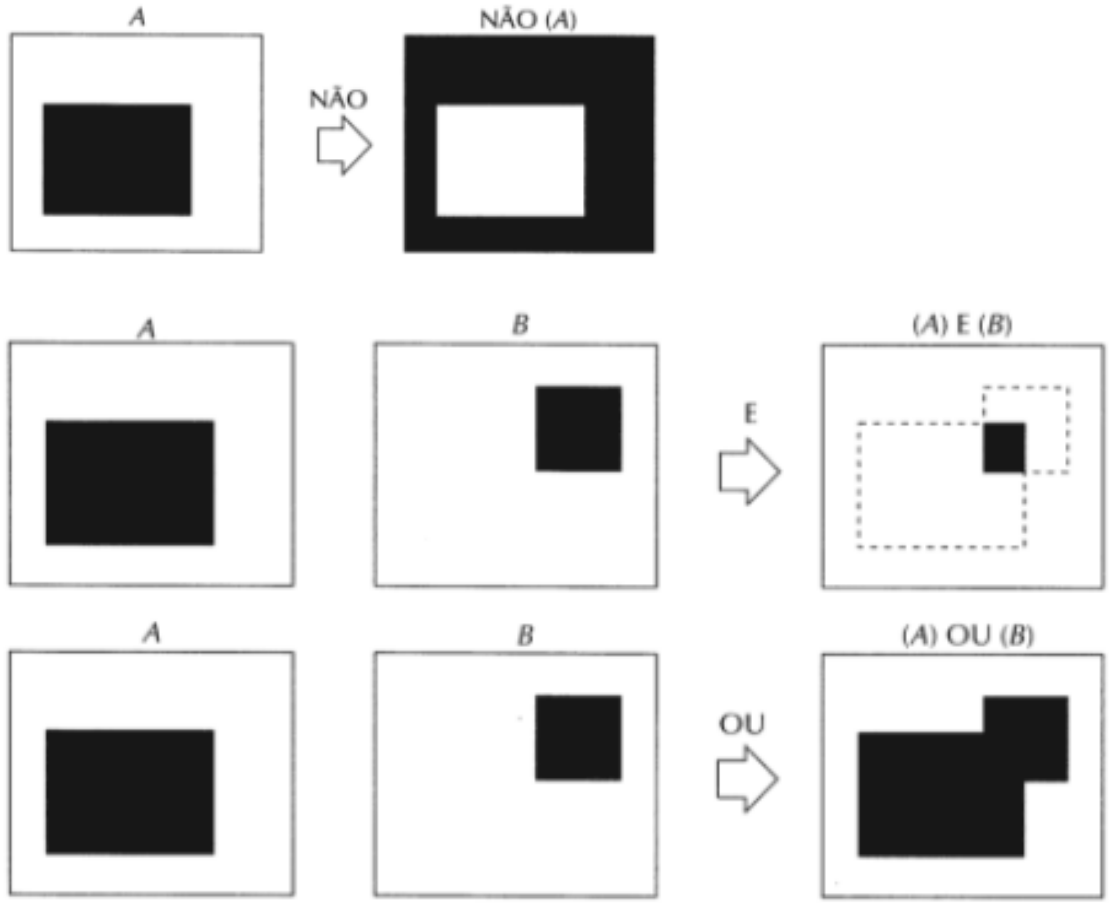
Na etapa de segmentação o propósito é a divisão da imagem em partes para a sua análise. A imagem resultante é binária, desta forma temos pixels pretos e brancos para verificação. Os pixels brancos representam o objeto de interesse; já os pixels pretos representam a parte que não é de interesse, o fundo (Augusto, 2012). Nessa etapa são delimitadas regiões de interesse por isso é considerada o momento mais critico, caso haja distorções presentes que não foram eliminadas o resultado final do processamento será insatisfatório.

> Na área de detecção e reconhecimento de imagens não basta simplesmente representar uma imagem com diferentes cores ou graduações de cinza. Também é necessário identificar regiões e estabelecer subdivisões na imagem em sua unidade básica (pixel) para que possa ser interpretada de acordo com uma finalidade específica. A identificação de regiões ou segmentos (não sobrepostos) na imagem é chamada de segmentação. MANTONELI. ([S.d], p. 25).

No pós-processamento as principais imperfeições residuais da segmentação são corrigidas. Podem ser feitos diversos procedimentos nesta etapa, tais como: a eliminação de objetos os quais não se deseja extrair informações, separação de objetos que se conectam e o agrupamento para formação de objetos complexos. Para a realização destes procedimentos são necessárias operações lógicas e morfológicas.

Operações lógicas são executadas pixel a pixel em imagens binárias, por meio da imagem de entrada que gera uma imagem de saída. Caracterizada por ser uma operação pontual, pois analisa pixel a pixel. Segundo Gonzalez & Woods (2011) as principais operações lógicas em processamento de imagens são a interseção (AND), o complemento (NOT) e a união (OR), a partir dos quais combinações podem ser feitas para formar qualquer outra operação lógica. Na Figura 2 constam imagens que elucidam as operações citadas retirados do livro Gonzalez & Woods (2011), onde a figura geométrica preta indica um e a figura geométrica branca indica zero.

Figura 2 – Operações lógicas.

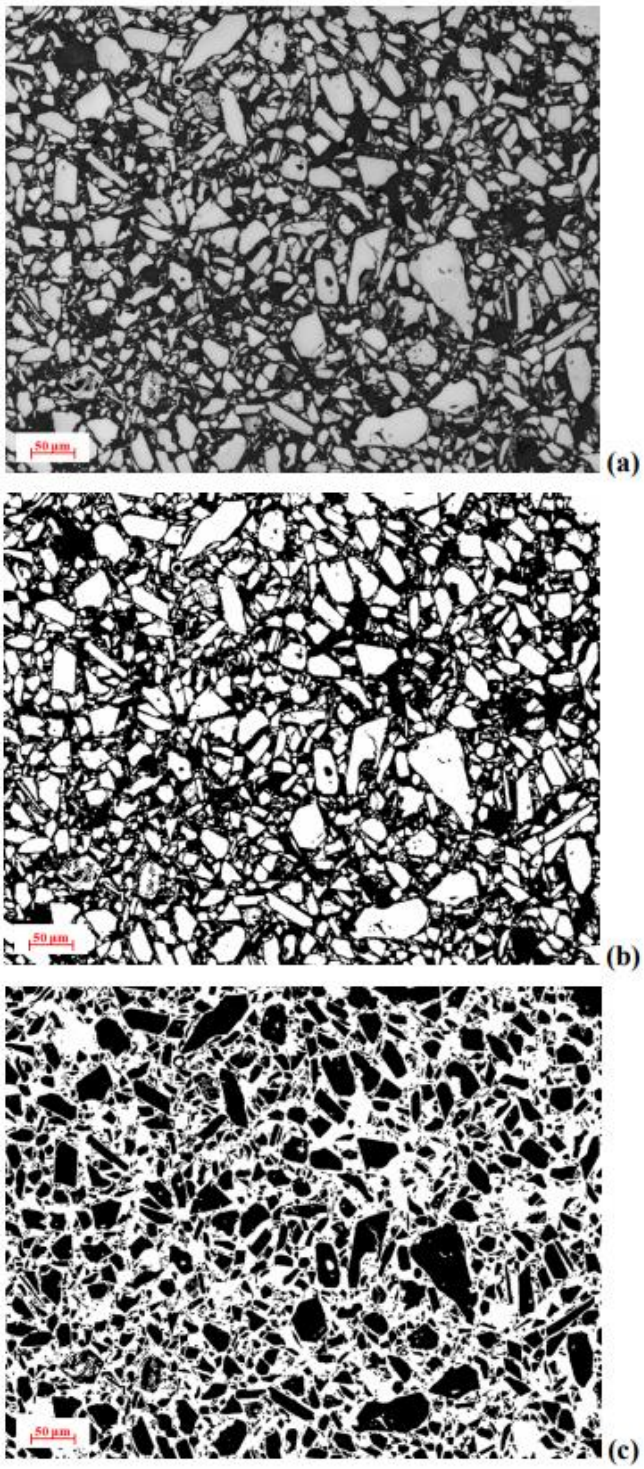


Fonte: GONZALEZ, C. Rafael; WOODS, E. Richard. 2011

As operações morfológicas diferem das operações lógicas, pois são realizadas localmente. Pode-se exemplificar da seguinte forma: um pixel da imagem de saída é função do valor dos pixels numa vizinhança da imagem de entrada. Em imagens binárias, um pixel será preservado ou invertido em função de ter certo número de vizinhos diferentes ou iguais em posições específicas de sua vizinhança e as posições desses pixels são definidas por uma espécie de máscara, denominada elemento estruturante. O elemento estruturante consiste numa janela / "kernel" que “varre” a imagem de entrada, preservando ou invertendo o pixel central da janela de acordo com a função aplicada (PUC-Rio, MAXWELL, [S.d]).

Como exemplo na Figura 3 pode ser verificada uma operação NOT, a mais utilizada na etapa do pós processamento. A imagem (a) é a imagem original; a imagem (b) é a imagem binária da segmentação da fase clara; a imagem (c) é a imagem resultante da aplicação de um NOT.

Figura 3 – Operação morfológica.



Fonte: AUGUSTO, S. Karen. 2011.

Após a realização do pós-processamento é realizada a extração de atributos, e por meio desta etapa pode-se realizar medidas na imagem. Por meio dessas

medidas são gerados dados quantitativos para o objetivo final. A extração é dividida em dois grupos parâmetros de medição: a medida de campo, onde o campo todo é considerado, por exemplo, área total de um objeto que resulta em apenas um valor por medida; já a medida de região é o outro parâmetro de medida, o qual inclui grupos de tamanho e forma.

Com a extração de atributos da imagem concluída podemos realizar o reconhecimento e classificação/interpretação. No processo de reconhecimento é atribuído um rotulo ao objeto, baseado na informação fornecida. Por exemplo, a identificação de um caractere “A” requer a associação do que foi fornecido para aquele caractere com rótulo “A”. Na interpretação é atribuído um significado a um conjunto de entidades rotuladas, como por exemplo, o código postal é descrito como uma cadeia de cinco números e posteriormente um hífen e mais três números (ALVES, F. Neide, 2003).

2.2 CONVERSÃO E CLASSIFICAÇÃO DE IMAGENS

As imagens binárias que possuem tons de cinza têm apenas um único canal de cor, sendo assim seu processamento é mais rápido do que o de imagens com diversas cores. A conversão de imagens coloridas para a escala cinza é relativamente simples, utilizando a intensidade das cores vermelha, verde e azul para conversão. A conversão da escala cinza para binária ocorre de forma mais elaborada, onde é definido um limite na faixa de 0 a 255 do pixel e se estiver acima deste limite é atribuída a cor branca, e abaixo dele a preta; este limite pode ser definida via algoritmo ou manualmente. Converter imagens na escala cinza ou binária para a colorida é impossível se quiser recuperar exatamente os valores originais, pois quando a cor vermelha, verde e azul se converte em um único canal, a informação se perde (NASCIMENTO; SOUZA, 2014). Caso não se queira recuperar exatamente os valores originais é possível, como ocorre em filmes preto e branco que são restaurados.

2.3 FUNDAMENTAÇÃO TEÓRICA

A seguir serão elencadas as tecnologias e os conceitos que são

imprescindíveis à compreensão do trabalho.

2.3.1 Filtro Sobel

É um método de detecção de bordas tanto horizontais quanto verticais separadamente em uma imagem em escalas de cinza. Ocorre uma transformação, denominando como filtragem da imagem pixel a pixel, onde a conversão da imagem de RGB (representa o método de cores utilizado nos monitores, que fazem utilizam as cores Vermelho, Verde e Azul para compor imagens coloridas no computador) para escala cinza. Esse processo é utilizado no processamento de imagens, onde por meio dela pode-se detectar contornos.

Os operadores de Sobel fornecem os efeitos de diferenciação e de suavização, os quais constituem uma característica atrativa dos operadores de Sobel. Tais efeitos destacam mais as bordas, mas também aumentam o ruído da imagem (MATURANA, 2010). Os ruídos são caracterizados por serem as informações indesejadas em uma imagem, pode ocorrer devido às falsas informações dentro da imagem ou pela variação de brilho.

2.3.2 Filtro Canny

O método Canny, assim como o Sobel, é um filtro para detecção de bordas. Seu tempo de processamento é mais lento do que o Sobel, porém possui maior precisão. É caracterizado por ser mais robusto à presença de ruídos, desta forma possui maior qualidade em seus dados processados.

2.3.3 Filtro Gaussiano

Ao trabalharmos com o processamento digital de imagens consequentemente acabamos tendo que trabalhar com ruídos (variação aleatória, que diverge da realidade, de brilho e cor nas imagens digitais), e para diminuirmos estes ruídos a utilização do filtro gaussinano acaba sendo uma solução. Este filtro borra ou desfoca a imagem o que auxilia no processo de amenização de ruído. A utilização deste método de suavização de imagem pode ser encontrada no pré-processamento (é o

processamento inicial dos dados da imagem para remoção de ruídos, ajuste de distorções geométricas e calibração radiométrica da imagem).

O filtro gaussiano de passa-baixa reduz a amplitude do espectro, assim como atenua as altas frequências o que ocasiona a suavização da imagem. O espectro é a distribuição da intensidade de uma radiação em função de uma grandeza característica, como a energia, o comprimento de onda, a massa ou a frequência.

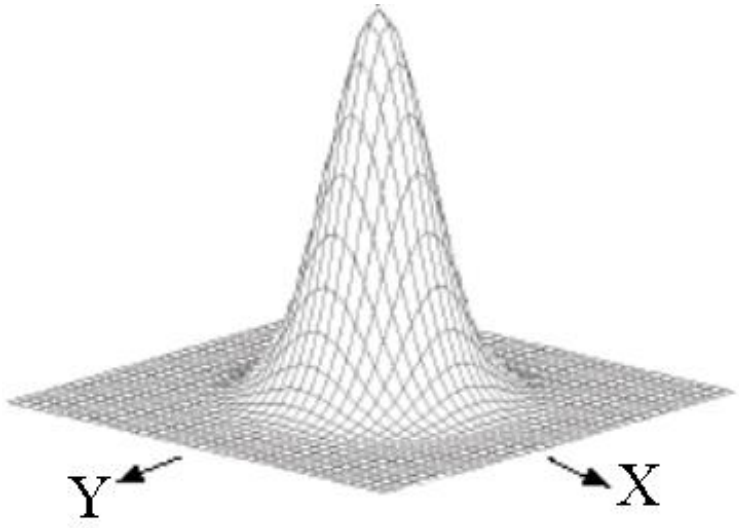
A expressão matemática Gaussiana é:

$$G(x,y) = G(x) \,.\, G(y)^t = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Fonte: Autor (2020)

A curva gaussiana é de uma dimensão, porém a imagem é bidimensional, logo temos o eixo x e y, sendo eles respectivamente a distância na origem do eixo horizontal e a distância da origem no eixo vertical. Desta forma é necessário multiplicar x e y conforme equação para obtermos o resultado referente a imagem. O desvio padrão (σ) é caracterizado pelo desvio padrão da distribuição de Gauss dos pixels que compõe a imagem.

Figura 4 – Gráfico da Curva gaussiana bidimensional.



Fonte: Autor (2020)

O resultado da aplicação da formula bidimensional (Figura 4) é uma superfície com contornos sendo círculos concêntricos, sendo que o σ define a abertura do sino formado pela curva.

2.3.4 Classificador Haar

O Classificador Haar ou *Haar Cascade* é um método de detecção de objetos eficaz. É baseada em aprendizado de máquina onde a função é treinada a partir de muitas imagens positivas e negativas (OpenCV, [S.d]). Em seguida, é usado para detectar objetos em outras imagens. Sua função pode ser aplicada por meio da biblioteca OpenCV.

2.3.5 OCR

O reconhecimento óptico de caracteres ou OCR (Optical Character Recognition) é um processo utilizado para reconhecer caracteres a partir de um arquivo de imagem, podendo ser escaneado, impresso ou escrito à mão. Desta forma pode-se verificar que é possível converter diferentes tipos de arquivos digitalizados em documentos com dados pesquisáveis ou editáveis, ou seja, convertem imagens de texto em texto real (ISLAM, Norman; ISLAM, Zeeshan. NOOR, Nazia, 2016). Este tipo de reconhecimento analisa o documento e compara seus caracteres com fontes armazenadas em seu banco de dados e/ou reconhece características típicas de determinado caractere. Portanto, OCR é um programa que reconhece caracteres, extraindo o conteúdo de uma imagem e convertendo em texto (IFRS, 2018).

Segundo Chaudhuri, em torno de 1870 surgiram os primeiro relatos sobre o reconhecimento óptico de caracteres, por meio da criação do scanner de retina desenvolvido por Charles R. Carey, de Boston, Massachusetts. Esse *scanner* era um sistema de transmissão de imagem que utilizava um mosaico de fotocélulas. Em 1890, Nipkow inventou o scanner sequencial, o qual analisava a imagem linha por linha, sistema semelhante ao usado na televisão moderna (CUNHA, C. Renata, 2018).

Em 1900, o russo cientista Tyurin, por meio de experimentos bem sucedidos realizados com OCR mostrou a aplicação para auxilio de deficientes visuais. Após diversos experimentos durante as primeiras décadas neste ramo se sucederam.

Em torno de 1940 por meio do desenvolvimento de computadores digitais pode-se obter a versão moderna de OCR. Uma década após esta tecnologia já

estava disponível comercialmente.

Os sistemas comerciais que surgiram entre 1960 e 1965 foram denominados como a primeira geração de OCR, esta versão era caracterizado pela leitura de um formato restrito de letras, o que com o decorrer do tempo foi aprimorado e pode reconhecer diversos tipos de formatos de letras. Em meados da década de 1960 e inicio da década de 1970 a segunda geração, o marco desta geração foi reconhecimento de caracteres escritos a mão, porém com limitação a poucas letras. Já a terceira geração surgiu em meados de 1970, nesta geração os principais desafios foram os documentos de baixa qualidade e grande conjunto de caracteres impressos e manuscritos.

Os sistemas comerciais que começaram a ser disponibilizados na década de 1950 tiveram apenas o aumento das vendas após 1986. Este fato foi causado devido ao custo do hardware ter baixado ao longo desta época. Atualmente os sistemas OCR já esta incluso em pacotes de software, o que gerou grande aumento na utilização desta tecnologia.

2.3.6 Deep Learning

Aprendizagem profunda ou *Deep Learning* é o aprendizado de máquina. Esse aprendizado ocorre por meio de redes neurais profundas. Algoritmos que aplicam esse método são capazes de realizar análise de dados não estruturados sem que haja algum tipo de pré-processamento ou supervisão. A aplicação pode ocorrer para a realização de reconhecimento de imagens, objetos, face, entre outras aplicações.
.

2.3.7 Tesseract

É uma biblioteca de funções utilizada para a realização de reconhecimento óptico de caracteres, pode ser utilizado para a detecção de textos contidos em imagens ou vídeos. Possui suporte a Unicode (UTF-8) e pode reconhecer diversos idiomas.

Essa biblioteca possui interface em C++, desta forma o wrapper (são classes que adaptam uma interface para outra, fazendo com que duas interfaces incompatíveis se comuniquem) pytesseract é utilizado quando aplicado, por

exemplo, em python.

2.4 PESQUISAS RELACIONADAS

A seguir são analisados trabalhos publicados onde são abordadas questões relacionadas com as tecnologias de registro de placas veiculares por meio do reconhecimento com uso de OCR.

2.4.1 Detecção e Reconhecimento

A solução proposta no estudo “Detecção e Reconhecimento Automático de Placas Veiculares” foi a utilização de software programa utilizando bibliotecas OpenCV para manipulação de imagens e Tesserract para o reconhecimento de caracteres. Foram citados dois filtros mais comuns para detecção de placa, sendo eles o Sobel e o Canny. Após análises verificou-se que o Canny é um pouco mais lento, porém mais preciso, desta forma ele foi utilizado. Associado a este programa foi utilizado uma câmera comum para capturar imagens. Após análise de 400 imagens verificou-se que a taxa de precisão de reconhecimento de placas veiculares brasileiras obteve uma taxa de acerto de 60% (NASCIMENTO; SOUZA, 2014).

2.4.2 Reconhecimento Automático De Placas Automobilísticas

De forma semelhante em alguns pontos ao trabalho citado anteriormente (NASCIMENTO; SOUZA, 2014), o trabalho “Reconhecimento Automático De Placas Automobilísticas” utilizou as mesmas bibliotecas para a manipulação de caracteres e reconhecimento de placa, assim como o mesmo hardware. O que difere é a utilização de classificadores Haar em cascata para detecção da placa. Os dados referentes a detecção giram em torno de 59,7% a noite e 98,4% de detecções durante o dia (PAES, S. L. Alexandre, 2017).

2.4.3 Reconhecimento de Placas de Veículos

Neste projeto é abordado o reconhecimento de placas via programação, onde

assim como os trabalhos citados anteriormente não foi utilizado Deep Learning (aprendizado estruturado profundo, aprendizado hierárquico ou aprendizado de máquina profundo). De forma sucinta, é aplicado o filtro Gaussiano, diferentemente da publicação de Nascimento (2014), que utilizou o filtro Canny. Assim como citado por Nascimento (2014), são utilizadas as bibliotecas OpenCV e Tesserract. Esta publicação relata que a exatidão no reconhecimento pode girar em torno de 80% a 90% (BARBOSA, Alexandre, 2017).

3. METODOLOGIA

É possível, com a utilização de um software em linguagem python, bibliotecas, filtros para conversão de cores, limiarização e desfoque executar o reconhecimento de placas veiculares. Como pode-se realizar isso, é apresentado abaixo.

3.1 ESTUDO DA VIABILIDADE DO RECONHECIMENTO DE PLACAS VEICULARES

Esta pesquisa possui cunho bibliográfico e qualitativo, no qual após leitura de trabalhos de conclusão, especialização e livros percebeu-se que existe a necessidade de elaboração do sistema proposto para melhoria de processos pertinentes ao dia a dia.

Com base nas pesquisas realizadas foi evidenciado que o método de Aprendizagem profunda (Deep Learning) seria uma aplicação complexa e com necessidade de maior tempo hábil para estruturação, sendo descartada tal solução.

A solução encontrada foi a utilização da linguagem python com a aplicação da biblioteca OpenCV, a qual possui diversas funções, tais como, suavização de imagem, detecção de bordas, reconhecimento de caracteres, entre outras. A linguagem foi escolhida devido a sua clareza e objetividade nas funções. A biblioteca de código OpenCV é amplamente utilizada nesta linguagem por ser uma programação de código aberto. Para o reconhecimento optico dos caracteres de entrada foi utilizado a biblioteca pytesseract.

Além do programa e linguagem escolhido, o filtro aplicado a imagem é de extrema importância, pois ele ajuda a eliminar ruídos oriundos do processamento de imagem. Por meio das pesquisas foi verificada a grande utilização de três filtros: Sobel, Gaussiano e Canny. O filtro Sobel é o mais utilizado, porém considerado inferior ao método Canny; entretanto o método Canny exige maior capacidade de processamento (NASCIMENTO; SOUZA, 2014). Ao invés de utilizar os filtros Canny e Sobel optou-se por detectar bordas por linhas de comandos OpenCV (findContourns, cv2.retangle) e não utilizando filtros, desta forma não fora exigido tanta capacidade de processamento. Fora realizada a filtragem da imagem com o

filtro Gaussiano, o qual reduz o ruído, segundo BARBOSA (2017) ele obteve boa resposta na sua aplicação, além disso não exige tanto esforço computacional.

A limiarização ou binarização(como é chamada trivialmente) foi utilizada para fazer a separação entre as classes “claro” e “escuro” no caso de a propriedade ser o brilho, ou seja, transforma uma imagem definida a níveis de cinza em uma imagem definida a níveis preto e branco.

O processamento de imagem ocorre da seguinte forma: a imagem é fornecida, após é feito o pré-processamento da imagem, neste processo fora aplicado o filtro de escala cinza(gray scale), posteriormente limiarização e por fim o filtro gaussiano para o desfoque da imagem.

A próxima etapa é a segmentação da imagem. A utilização da função de contorno de bordas nesta etapa é a técnica aplicada para localizar padrões em uma imagem. Nesse caso o intuito é localizar o retângulo da placa para extrair os dados desta região.

No pós-processamento ocorre o processo de localização das regiões que contém texto para então ser segmentada e extraída a informação. Ocorre então posteriormente o reconhecimento e classificação dos caracteres para serem, por fim, exibidos na tela.

As tecnologias disponíveis no mercado possuem custos elevados que giram em torno de R$ 7.000,00(sem a CPU) e muitas possuem planos mensais com valores aproximados de R$ 2.000,00. A proposta visa um algoritmo claro e simplificado com seus custos não ultrapassando R$ 3.000,00 (incluso CPU).

3.2 ABRANGÊNCIA DA PESQUISA

Este projeto é voltado para todo e qualquer local que necessite de controle de acesso de veículos por meio do reconhecimento de placa veicular. Quer seja estacionamentos, vias públicas, condomínios, e demais locais que se queira aplicar tal método para maior agilidade no processo.

3.3 MÉTODO APLICADO

Durante as pesquisas foram encontradas diversas soluções de construção de

algoritmos para OCR. Existem algoritmos complexos que envolvem *Deep Learning* e outros de certa forma mais simplificados que utilizam filtros específicos. Voltado para a área de reconhecimento de ótico e caracteres temos uma gama de filtros, os quais podem ser escolhidos conforme avaliação do usuário. Os filtros mais citados foram os de detecção de borda Sobel e Canny, e o filtro de redução de ruído o Gaussiano. Neste estudo fora utilizado para melhoria da imagem o filtro Gaussiano devido a seu método simplificado e bom desempenho. Os filtros para detecção de bordas Sobel e Canny não foram utilizados, pois foram utilizados comando próprios de localização de área do OpenCv, desta forma não fora exigido tanto poder computacional.

3.4. INSTRUMENTOS E EQUIPAMENTOS UTILIZADOS

Os principais componentes utilizados neste trabalho foram um notebook (Tabela 2) para execução de software e uma Câmera (Tabela 3). O notebook utilizado possui sistema operacional Windows 10 e Linux, para execução deste programa foi utilizado o Linux. A câmera utilizada deve possuir boa resolução, no caso, resolução *High Definition* (HD) e *Full High Definition*(FullHD), a resolução utilizada foi de 1920 x 1080 pixels e 1280 x 720 pixels. Referente, a resolução da câmera alguns celulares possuem resolução HD e FullHD indicada e podem ser utilizados para testes, neste sistema foi utilizado um celular com esta resolução e também uma Câmera.

Tabela 2- Dados Notebook:

Processador	Intel Core i7
Modelo processador	Core i7-7500U
Memória RAM	4GB DDR4
Armazenamento	HD 1TB
Sistema Operacional	Windows 10 Home / Linux
Outros recursos	Webcam integrada
	Microfone integrado
Bateria	Bateria 43 Wh
	Fonte Adaptadora AC 60 W

Tabela 3- Dados Câmera:

Resolução	1920 x 1080 (Full HD) / 1280 x 720 (HD)
Conexão	USB 2.0

4. DESENVOLVIMENTO

Por meio dos estudos das publicações acadêmicas pode-se verificar que os dados obtidos pelo algoritmo foram retirados da imagem fornecida. A imagem adquirida fora capturada por uma câmera para posteriormente ser analisada.

A partir da imagem adquirida é realizado o processamento e após o processamento da imagem o dado obtido, no caso a placa veicular, é exibida na tela. O resultado do processamento da imagem foi comparado com a placa veicular real e em todos os casos ele reconhece os caracteres, porém o acerto dos caracteres nem sempre é de 100%, independente do método aplicado.

Na Figura 5 consta a imagem capturada. Para realizar o reconhecimento de uma placa de automóvel é necessário submetê-la inicialmente ao processo de pré–processamento (UFF, [S,d]). Primeiramente, pode-se notar que a imagem possui mais de um canal de cor, por exemplo, imagens RGB contém três canais de cores. Entretanto deve ser convertido para o canal em escala de tons de cinza. Isso auxilia na melhoria do processamento, pois com apenas um terço da informação o processamento é mais rápido e demandamos menos tempo e energia do equipamento.

Figura 5 – Vídeo capturado.

Fonte: Autor (2020)

Existem diversos métodos de conversões de um canal de cor para outro disponíveis no OpenCV. Um dos métodos mais implementados conforme indicado no site de tutoria OpenCV é utilizar a função cv2.cvtColor(input_image, flag). Onde input_image é a imagem adquirida, flag é o tipo de conversão no caso a função cv2.COLOR_BGR2GRAY é a mais utilizada e indicada na seção de processamento de imagens. Portanto utilizando a linha de comando img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) pode ser realizada a transformação para a escala cinza, reduzindo assim os canais de cores manipulados. A aplicação da função BGR2GRAY pode ser verificada na Figura 6.

Figura 6 – Vídeo com aplicação da função BGR2GRAY

Fonte: Autor (2020)

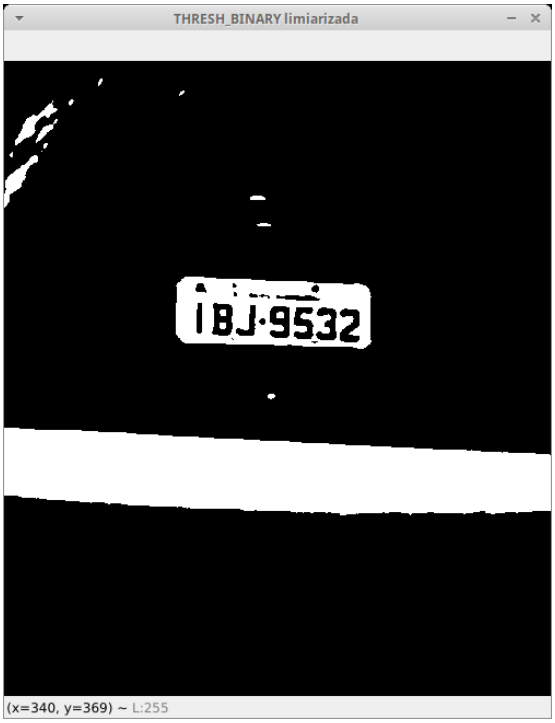
Com a imagem em tons de cinza pode ser realizado o processo de limiarização, ele nos auxilia na segmentação da imagem. Com ela pode-se trabalhar com dois limites de cores, branco e preto. A função retorna dois parâmetros; o primeiro parâmetro é o limiar definido, o segundo é a imagem após a limiarização.

> Se o valor do pixel for maior do que um valor limite, é atribuído um valor (pode ser branco), caso contrário, é atribuído outro valor (pode ser preto). A função usada é cv2.threshold. O primeiro argumento é a imagem de origem, que deve ser uma imagem em tons de cinza. O segundo argumento é o valor limite que é usado para classificar os valores de pixel. O terceiro

> argumento é o maxVal que representa o valor a ser dado se o valor do pixel for maior que (às vezes menor que) o valor limite (OpenCV, 2013).

O segundo argumento será definido pelo limite de preto, para no terceiro parâmetro ser utilizado o limite máximo de 255 (branco). Desta forma o comando: ret, img = cv2.threshold(img, 70, 255, cv2.THRESH_BINARY) irá executar a limiarização. Onde ret e img respectivamente é o primeiro e segundo parâmetros retornados da função de limiarização. Na Figura 7 pode ser verificada a aplicação da limiarização.

Figura 7 – Vídeo com aplicação da função THRESH_BINARY



Fonte: Autor (2020)

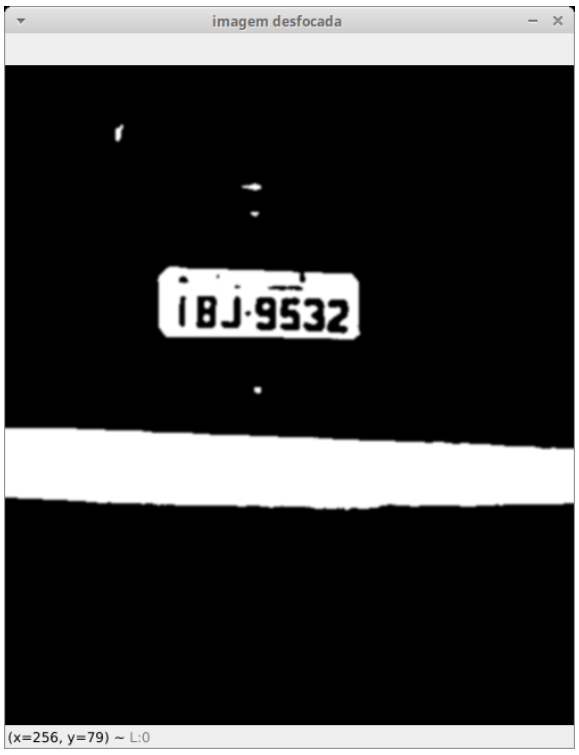
Além de transformar os canais de cores e converter por meio da limiarização, a aplicação de filtros auxilia para um melhor resultado no processo. Em suma, existem diversos filtros, porém o filtro gaussiano foi escolhido, por não exigir tanto poder computacional e resultado satisfatório. Os filtros sobel e canny atuam como um filtro passa-alta, já o gaussiano é passa-baixa. Filtros passa-baixa tem melhor resposta no domínio frequência; grande parte dos filtros de suavização é fundamentada na convolução - operação matemática, que “multiplica” duas matrizes de mesma dimensão que podem ter tamanhos diferentes. A resultante será uma terceira matriz com mesma dimensionalidade. Logo, como é baseado na convolução o filtro age como passa-baixa no domínio das frequências, portanto remove

frequências especiais depois de certo valor.

> Um filtro passa-baixa atenua as frequências altas e mantém as frequências baixas inalteradas. O resultado no domínio espacial é equivalente ao de um filtro de suavização; já que as altas frequências bloqueadas correspondem a mudanças nítidas de intensidade, ou seja, aos detalhes de escala fina e ruído na imagem do domínio espacial. Um filtro passa-alta, por outro lado, detecta a borda no domínio espacial, porque as bordas contêm muitas frequências altas. As áreas com um nível de cinza bastante constante consistem principalmente em frequências baixas e, portanto, são suprimidas (USP, [S.d]).

O filtro gaussiano é utilizado por meio da função cv2.GaussianBlur () e na Figura 8 é verificado o efeito do desfoque. No caso foram utilizados os parâmetros dados no tutorial do site OpenCv : imagemdesfocada = cv2.GaussianBlur(img, (5, 5), 0). Sendo respectivamente o kernel/matriz 5 x 5 e o desvio padrão 0. O desvio padrão denominado foi 0, portanto será calculado a partir do tamanho do kernel.

Figura 8 – Vídeo com aplicação da função GaussianBlur



Fonte: Autor (2020)

Com a imagem desfocada conseguimos melhorá-la, reduzindo desta forma os ruídos. Consequentemente com a suavização da imagem podemos obter um melhor resultado na localização dos contornos. O objetivo em questão é a localização de

contorno da placa, existem diversos métodos de localização, porém o menos complexo é utilizar a função findContours. Portanto a linha de comando para esta execução será: contornos, hierarquia = cv2.findContours(imgdesfoc, cv2.RETR_TREE, cv2.CHAIN_APPROX_NONE). Com o método de aproximação de contorno CHAIN_APPROX_NONE, todos os pontos de fronteira são armazenados. O parâmetro RETR_TREE recupera todos os contornos e cria uma lista completa de hierarquia familiar.

Para verificação sobre a existência de algum contorno que não foi identificado no processo anterior é testado com a função cv2.arcLength, que possibilita analise de existência de contorno retangular com a função cv2.rectangle. A síntese da função para localizar contorno retangular é cv2.rectangle (imagem, ponto_início, ponto_final, cor, espessura). Após verificações da área e obtendo o campo desejado, no caso a região da placa é realizado o recorte do local.

Após todos estes processos realizados no vídeo é obtida a imagem da placa, com a imagem da placa recortada (Figura 9) suas letras são segmentadas. No caso é aplicado os as funções BGR2GRAY (Figura 10), THRESH_BINARY (Figura 11) e GaussianBlur (Figura 12). Sendo assim a imagem é ajustada para obter um melhor resultado no reconhecimento.

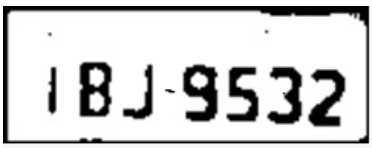
Figura 9 – Imagem da placa capturada no vídeo.

Fonte: Autor (2020)

Figura 10 – Imagem da placa com aplicação COLOR_BGR2GRAY.

Fonte: Autor (2020)

Figura 11 – Imagem da placa com aplicação THRESH_BINARY.



Fonte: Autor (2020)

Figura 12 – Imagem da placa com aplicação GaussianBlur.

Fonte: Autor (2020)

Por meio do tesseract é possível realizar a leitura dos caracteres que estão embutidos na imagem. Como fora utilizado o python, será utilizado o wrappers pytesseract, e por meio da função pytesseract.image_to_string a leitura simples da imagem e converte para texto. Desta forma se obtém a leitura da placa de forma mais simplificada e automatizada.

5. RESULTADOS

Inicialmente fora realizado estudo sobre a viabilidade da elaboração de um algoritmo para a detecção de placa e reconhecimento de caracteres. Pode-se concluir que é possível interpretar dados, porém como não é utilizado redes neurais na localização da placa o programa terá que ser adaptado, pois poderão ocorrer falhas.

Neste estudo foi utilizado um notebook, conforme pesquisa custa em torno de R$ 1.800,00, entretanto existem modelos mais baratos disponíveis no mercado. Além disso, foi utilizada uma câmera no inicio, em sites de vendas pode ser encontradas câmeras que custam em torno de R$ 160,00, porém após foi utilizado celular para os testes. Deve ser usada preferencialmente uma câmera com boa resolução.

Tecnologias similares são comercializadas com custo aproximado de R$ 7.000,00 (Figura 13). Além disso, existem sistemas embarcados com planos mensais de no mínimo R$ 2.000,00 com contrato de no mínimo 24 meses. Conforme pesquisas o sistema possui custo relativamente baixo em comparação com valores que constam no mercado.

Tabela 4- Tabela com valores do sistema:

Sistema utilizando câmera:	
Software	R$ 200,00
Notebook Samsung X41	R$ 1.800,00
Cabo USB 2.0	R$ 18,00
Câmera Usb Live Stream Alta Resolução - 1920 x 1080p	R$ 160,00
Total:	R$ 2.178,00

Fonte: Autor (2020)

Tabela 5- Tabela com valores do sistema:

Sistema utilizando câmera de celular:	
Software	R$ 200,00
Notebook Samsung X41	R$ 1.800,00
Celular Asus Zenfone Selfie - 16GB - 1920 x 1080p / 1280 x 720p	R$ 800,00
Cabo USB 2.0	R$ 18,00
Total:	R$ 2.818,00

Fonte: Autor (2020)

Tabela 6- Tabela com valores do sistema (sugestão para aplicação em Raspberry):

Sistema utilizando Raspberry Pi3 B:	
Software	R$ 200,00
Kit Raspberry Pi3 B, Fonte, Case, Dissipadores, Hdmi e Câmera	R$ 510,00
Tela Touch Lcd 7” + Suporte 1024x600 Para Raspberry Pi3/4	R$ 664,00
Total:	R$ 1.374,00

Fonte: Autor (2020)

Tabela 7- Tabela com valores do sistema (tecnologia disponível comercialmente):

Tecnologia Disponível Comercialmente: Câmera Reconhecimento Placas Veiculares	
Câmera Reconhecedora de placas 1.3 Megapixels	R$ 300,00

Placa PCI	R$ 400,00
Cd rom com manual e software em inglês	R$ 6.170,00
Cabo Conector D9 com 4 BNC	R$ 90,00
Fonte	R$ 40,00
Computador ICC IV2546KM19 Intel Core I5 3.20 ghz 4GB HD 120GB SSD Kit Multimídia Monitor LED 19,5	R$ 1.500,00
Total:	RS 8.500,00

Fonte: Autor (2020)

Figura 13 – Tecnologia Disponível Comercialmente:

Fonte: https://produto.mercadolivre.com.br/MLB-1231889068-cmera-reconhecimento-placas-veiculares-intertraff-walzcam-_JM#position=20&type=item&tracking_id=d14f2825-731e-4179

Com relação aos itens do sistema embarcado não foram obtidas mais

detalhes sobre os componentes, foram fornecidos apenas os valores referentes a câmera e a licença mensal do software, que custam respectivamente R$ 250,00 e R$ 1.750,00.

A Tabela 4 e a Tabela 5 especificam os gastos do sistema utilizando um computador o qual possui um processador com um ótimo desempenho; já na Tabela 6 é sugerido a aplicação por meio do Raspberry Pi, com este estudo e pesquisa, fora verificado que a utilização dele fora viável e pode ser utilizado em projetos futuros. Na Tabela 7 é ilustrado os valores de equipamento disponível comercialmente.

Com este estudo nota-se que é viável utilizar um algoritmo aplicando as funções opencv para reconhecimento de placa veicular. As tecnologias disponíveis no mercado conforme pesquisas são de custo elevado. Neste projeto é demonstrado que por meio de um algoritmo que não exige tanto poder computacional e uma câmera não tão sofisticada, pode-se realizar o reconhecimento de caracteres de placas veiculares.

6. ANÁLISE

Por meio de teste realizados com o algoritmo pode-se verificar que não ocorre 100% de assertividade. Nota-se que ele reconhece todos os números, e com relação as letras contidas na placa ocorre divergência, pois a letra “I” o algoritmo interpreta como o número “1”. Na Tabela 8 é verificada a taxa de acertos em cada placa, a taxa de acertos de todas as letras e números e mediana de acertos.

Tabela 8- Tabela com análise dos resultados das placas:

PLACA	RESULTADO	Acertos
MEC 7672	MEC 7672	100%
IBJ 9532	1BJ 9532	86%
OSD 2284	OSD 2284	100%
IXG 4309	1XG 4309	86%
IKM 0555	1KM 0555	86%
PNO 3894	PNO 3894	100%
Acerto total (letras/números):		93%
Mediana de acertos:		93%

Fonte: Autor (2020)

7. CONCLUSÃO

Por meio deste estudo pode-se concluir que existem várias formas para a realização de reconhecimento óptico de caracteres, porém alguns são extremamente complexos. Contudo existem maneiras mais claras e simplificadas para aplicação de OCR, como a citado neste estudo de viabilidade. Foi abordado a aplicação de algoritmo via linguagem python utilizando bibliotecas OpenCV e Pytesserract, tendo aplicação de filtros de conversão em escala cinza, limiarização e desfoque como ferramentas.

A aplicação citada reconhece os caracteres da imagem, porém não possui 100% de assertividade, pois o ambiente pode interferir. O processamento ocorre de forma clara e simplificada sem exigir tanto processamento como com a aplicação de outros métodos, tais como os que envolvem aprendizagem profunda e que exigem CPU com alta capacidade de processamento.

Nota-se que o valor do sistema proposto no estudo pode ocasionar uma economia que gira em torno de 66,85% até 74,5% em relação ao valor comercial de R$ 8.500,00. Além disso, para trabalhos futuros é sugerida a aplicação do sistema via Raspberry Pi que conforme demonstrado pode resultar em uma economia ainda maior, por volta de 85,84% sobre a mesma opção comercial usada como referência neste estudo.

Para projetos futuros o aprimoramento para aplicação em diferentes ambientes seria um fator a ser explorado, além disso, inserir condições no programa para diferenciar a letra “I” e o número “1” é uma tarefa que irá auxiliar na melhoria do reconhecimento. Contudo a forma estudada, conforme já citado, e sua estrutura sugerida poderia ser mantida, visto que conforme estudos tiveram resultados positivos, mas que podem ser melhorados.

Neste projeto pode-se notar que algumas disciplinas cursadas auxiliaram na compreensão de alguns processos sendo elas: as disciplinas de programação para criação de algoritmo para execução do programa de reconhecimento de caracteres; eletrônica digital para o conhecimento das portas lógicas e operações; instrumentação para medição e delimitação de parâmetro de área na qual se deseja trabalhar.

REFERÊNCIAS

ALVES, F. Neide. **ESTRATÉGIAS PARA MELHORIA DO DESEMPENHO DE FERRAMENTAS COMERCIAIS DE RECONHECIMENTO ÓPTICO DE CARACTERES**. Monografia (Mestrado em Engenharia Elétrica), Centro de Tecnologia e Geociências (CTG) - UFPE, Pernambuco, 2003.

AUGUSTO, S. Karen. **Identificação Automática do Grau de Maturação de Pelotas de Minério de Ferro**. Dissertação (Mestrado em Engenharia de Materiais e de Processos Químicos e Metalúrgicos), PUC-Rio, Rio de Janeiro, 2012.

BARBOSA, Alexandre. **Processamento Digital De Imagens Para O Reconhecimento De Placas de Veiculos.** Monografia (Graduação), Unibalsa – Curso de Sistemas de Informação, Maranhão, 2017. .

CHAUDHURI, Arindam. *In*:_. **Optical Character Recognition Systems** – 1ª ed. Nova York: Springer International Publishing, 2017.

CUNHA, C. Renata. **ESTUDO COMPARATIVO SOBRE FERRAMENTAS DE RECONHECIMENTO ÓPTICO DE CARACTERES**. Dissertação (Graduação), Instituto Federal de Minas Gerais – Curso de Ciência da Computação, Minas Gerais, 2018.

ENSINA AI. **Detectando objetos com métodos clássicos - OpenCV (cascades).** São Paulo, 2018. Disponível em: https://medium.com/ensina-ai/detectando-objetos-com-m%C3%A9todos-cl%C3%A1ssicos-opencv-cascades-440e29913b1b. Acesso em: 28 junho 2020.

ESQUEF, A. Israel; ALBUQUERQUE, P. Marcelo. *In: Centro Brasileiro de Pesquisas Físicas.* **Processamento Digital de Imagens**. Rio de Janeiro, 2013. https://www.maxwell.vrac.puc-rio.br/21365/21365_6.PDF. Acesso em: 12 de agosto 2020.

GONZALEZ, C. Rafael; WOODS, E. Richard. *In:_.* **Digital Image Processing**. 3ª ed. rev. São Paulo: Pearson Education, 2011.

IFRS. **Ferramentas OCR – entenda o que são e sua relação com a acessibilidade.** Bento Golçalves, 2018. Disponível em: https://cta.ifrs.edu.br/ferramentas-ocr-entenda-o-que-sao-como-funcionam-e-qual-sua-relacao-com-a-acessibilidade/. Acesso em: 7 novembro 2020.

INPE. **Teoria: Processamento de Imagens.** São Paulo, [S.d.]. Disponível em: http://www.dpi.inpe.br/spring/teoria/realce/realce.htm/. Acesso em: 28 junho 2020.

ISLAM, Norman; ISLAM, Zeeshan. NOOR, Nazia. A Survey on Optical Character Recognition System. **ournal of Information & Communication Technology,** Nova York, volume X, ed.2, pág. 1-2, dezembro de 2016.

MANTELI, Sylvio. **Introdução à Visão Computacional**. São Paulo, [S.d]. Disponível em: http://www.inf.ufsc.br/~aldo.vw/visao/segmentos.pdf. Acesso em: 01 agosto

2020.

MATURANA, S. Patrícia. **ALGORITMOS DE DETECÇÃO DE BORDAS IMPLEMENTADOS EM FPGA.** Dissertação (Mestrado em Engenharia Elétrica), Universidade Estadual Paulista Júlio Mesquita Filho, São Paulo, 2010.

MRA GRUPO ÁLAVA. **Qual a diferença entre uma imagem multiespectral e uma hiperespectral?** Espanha, 2019. Disponível em: http://www.mra.pt/industria/actualidade/qual-a-diferenca-entre-uma-imagem-multiespectral-e-uma-hiperespectral/. Acesso em: 28 junho 2020.

NASCIMENTO, P. Verônica; SOUZA, C. S. Antonio. **Detecção e Reconhecimento Automático de Placas Veiculares.** Monografia (Graduação), Instituto Federal da Bahia – Curso de Análise e Desenvolvimento de Sistema, Bahia, 2014.

Observatorio Educativo Itinerante. Hipertextos do OEI - Galaxias e o Universo - O Espectro Eletromagnetico. 2010. Disponível em: http://www.if.ufrgs.br/oei/cgu/espec/intro.htm. Acesso em: 01 agosto 2020.

PAES, S. L. Alexandre. **RECONHECIMENTO AUTOMÁTICO DE PLACAS AUTOMOBILÍSTICAS.** Monografia (Graduação), UFRJ – Curso de Engenharia Eletrônica e de Computação, Rio de Janeiro, 2017.

PEDRINI, Helio; SCHWARTZ, R. Wiliam. *In:_*. **Análise de Imagens Digitais: Princípios, Algoritmos e Aplicações.** – 1ª ed. São Paulo: THOMSON Learning, 2008.

PETROU Maria, PETRO Costas. *In*: *British Library*. **Image Processing, The Fundamentals** – 2ª ed. Inglaterra: John Wiley & Sons Ltda, 2010.

QUEIROZ, E. R. José; GOMES, M. Hermam. Introdução ao Processamento Digital de Imagens. **Revista de Informática Teórica e Aplicada - RITA,** Campina Grande, volume XIII, ed.2, pág. 11-42, julho de 2006.

UFF. **Binarizacao Por Otsu e Outras técnicas Usadas na Detecção de Placas**. Rio de Janeiro, [S.d]. http://www.ic.uff.br/~aconci/OTSUeOutras.pdf. Acesso em: 12 de agosto 2020.

USP. **Filtro de frequência**. São Paulo, [S.d]. http://homepages.inf.ed.ac.uk/rbf/HIPR2/freqfilt.htm. Acesso em: 8 de novembro 2020.

ANEXO A – Comandos de filtros OPENCV para reconhecimento

As linhas de comando abaixo representam os filtros utilizados no algoritmo. Inicia com a conversão da imagem para escala cinza, posteriormente a imagem em tons de cinza é limiarizada. Com a imagem já limiarizada é possível desfocar a imagem.

```
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)  #Transforma para escala cinza

ret, imagemlimi = cv2.threshold(imgcinza, 70, 255, cv2.THRESH_BINARY)
#  Transforma a imagem cinza em imagem limiarizada(imagem com duas cores,
#limite mínimo de preto e máximo de branco 255. função retorna dois parâmetros; o
#primeiro parâmetro é o limiar definido, o segundo é a imagem após a limiarização.

imgdesfoc = cv2.GaussianBlur(imagemlimi, (5, 5), 0) #Comando para desfocar,
#reduzir ruídos da imagm. Definido kernel (matriz) 5 , 5e limite mínimo de 0(com
#desvio padrão 0 o valor será calculado a partir do kernel).
```

ANEXO B – Remoção de Caracteres

Com as linhas de comando abaixo é possível remover caracteres indesejado na leitura da placa, diminuindo erros que podem ocorrer na leitura.

```
def removerChars(text):
    str = "()?#¨&_{}+:;!%^`~<>*¬\=,.'ºª/»-@‘"
    for x in str:
        text = text.replace(x, '')
    return text
```