# Desenvolvimento de algoritmo para leitura e extração de caracteres de placas veiculares

Alexandre Pereira Vega
*Universidade Federal de Santa Catarina*
*Departamento das Engenharias*
Blumenau, Brasil
Email: alexandre.xeno@gmail.com

Aléxei Felipe Paim
*Universidade Federal de Santa Catarina*
*Departamento das Engenharias*
Blumenau, Brasil
Email: alexei.felipe@grad.ufsc.br

Beatriz Picinin Pinheiro
*Universidade Federal de Santa Catarina*
*Departamento das Engenharias*
Blumenau, Brasil
Email: biappinheiro@gmail.com

***Resumo*—O artigo aborda o desenvolvimento de um algoritmo para reconhecimento automático de placas utilizando técnicas de visão computacional e processamento de imagens. São destacados os desafios envolvidos na identificação de placas devido a fatores como velocidade, ângulo, qualidade de imagem e iluminação. O algoritmo é dividido em duas seções: manipulado por placa e manipulado por caractere. O artigo também aborda os benefícios do uso do Python e da biblioteca OpenCV para processamento e análise de imagens, apresentando resultados parciais do projeto. Por fim, são discutidos os potenciais benefícios do uso da tecnologia de reconhecimento de placas no monitoramento e gerenciamento de veículos.**

***Palavras chave*—LPR, OCR, Python, Visão Computacional**

***Abstract*—The article addresses the development of an algorithm for automatic license plate recognition using computer vision and image processing techniques. The challenges involved in identifying license plates due to factors such as speed, angle, image quality and lighting are highlighted. The algorithm is divided into two sections: board-handled and character-handled. The article also addresses the benefits of using Python and the OpenCV library for image processing and analysis, presenting partial results of the project. Finally, the potential benefits of using license plate recognition technology in vehicle monitoring and management are discussed.**

***Index Terms*—Computer Vision, LPR, OCR, Python**

## I. Introdução

A proposta deste projeto, que envolve a visão computacional, surgiu da necessidade de aplicar e integrar os conhecimentos adquiridos durante a formação acadêmica. Diante disso, foi concebida a ideia de realizar um estudo abrangente, seguido do desenvolvimento e implementação de um algoritmo de reconhecimento automático de placas veiculares.

De acordo com dados do Instituto Brasileiro de Geografia e Estatística referentes ao ano 2022, a frota de veículos automotores no Brasil atingiu aproximadamente 115 milhões de veículos registrados[1]. Diante desse cenário, surge a problemática de como podemos aproveitar os avanços tecnológicos para controlar e fiscalizar de forma mais eficiente os veículos, tanto em vias públicas como em espaços privados, como shoppings, condomínios e estacionamentos.

O reconhecimento da placa dos veículos automotores é uma aplicação que está inserida no contexto do reconhecimento de padrões, um ramo da ciência que se interessa pela descrição, classificação e também reconhecimento de objetos ou partes de uma imagem digital[2].

Em um mundo em constante evolução tecnológica, a demanda por métodos e ferramentas que facilitem o monitoramento e a gestão dos veículos é crescente. A aplicação de técnicas baseadas em Visão Computacional e tecnologias de reconhecimento de placas veiculares, que envolvem a extração e reconhecimento automático dos caracteres da placa a partir de uma imagem[3], pode proporcionar benefícios significativos nesse contexto.

O reconhecimento de placas veiculares apresenta desafios complexos devido a diversos fatores, como velocidade do veículo, ângulo de captura, qualidade da imagem, luminosidade, sombra, reflexo e estado da placa. Essas características específicas tornam o problema mais difícil de ser tratado, mesmo que a placa seja composta por caracteres com formas bem definidas[4].

A metodologia empregada neste artigo é a pesquisa bibliográfica, abrangendo diversos temas relacionados à programação, processamento digital de imagens, visão computacional, Reconhecimento Automático de Placas e extração de caracteres. Essa abordagem permite explorar e analisar as contribuições científicas existentes nessas áreas, a fim de embasar o desenvolvimento do seguinte projeto.

## II. Revisão bibliográficas

Nesta seção, serão apresentados os principais tópicos relevantes para o desenvolvimento do sistema, proporcionando uma compreensão aprofundada de seu funcionamento.

### A. Identificação de automóveis

As placas de automóveis no Brasil desempenham um papel fundamental na identificação e controle dos veículos em circulação. Cada placa é única e serve como uma espécie de identidade para o veículo, permitindo sua rastreabilidade e regulamentação pelas autoridades competentes.

No Brasil, o formato das placas de veículos segue um padrão estabelecido pelo Departamento Nacional de Trânsito (DENATRAN). Atualmente, utiliza-se o modelo conhecido como placa MERCOSUL, que substituiu o antigo padrão de placas alfanuméricas. Essas placas possuem uma combinação de letras e números, juntamente com uma identificação do estado ou do país, além de um código QR para facilitar a leitura e autenticação[5].

As placas de automóveis são usadas para diversos fins, desde a fiscalização de trânsito e aplicação de multas até o controle de veículos roubados e furtados. Além disso, elas são utilizadas em sistemas de pedágio, estacionamentos e outros locais onde é necessário identificar os veículos de forma rápida e precisa. As placas seguem um tipo de fonte específica chamada FE Engschrift, com altura de 65mm para veículos. Assim como ilustra a Figura 1


Fig. 1. Placa modelo MERCOSUL [5].

### B. Visão computacional

Não existe uma definição clara entre a fronteira de Processamento de Imagens e Visão Computacional. Podemos dizer que o processamento de imagens envolve a transformação de uma imagem de entrada em um conjunto de valores numéricos, que podem ou não formar outra imagem. Por outro lado, a Visão Computacional busca emular a visão humana, tendo também uma imagem de entrada, mas produzindo uma interpretação da imagem como um todo ou parcialmente. Conforme mencionado por Gonzalez[6], os processos de Visão Computacional geralmente começam com o processamento de imagens [7].

### C. Processamento digital de imagens

Segundo Gonzalez[6], "O campo do processamento digital de imagens se refere ao processamento de imagens digitais por um computador digital", o que nos dá uma breve dimensão do que se trata. No entanto, o processamento digital de imagens vai além dessa definição básica. Envolve a aplicação de algoritmos e técnicas para aprimorar, analisar e extrair informações valiosas de imagens digitais. Essas técnicas abrangem desde a correção de imperfeições e ruídos até a segmentação de objetos, extração de características e reconhecimento de padrões.

O processamento digital de imagens desempenha um papel fundamental em diversas áreas, como medicina, segurança, automação industrial e realidade virtual, contribuindo para aprimorar a qualidade de vida, a precisão diagnóstica e a eficiência dos sistemas automatizados[8, 9].

#### c.1) Segmentação de imagem

A segmentação de imagem é um desafio complexo que consiste em dividir uma imagem em unidades significativas, representando os objetos de interesse presentes nela. Embora a descrição dessa tarefa possa parecer simples, sua implementação é uma das mais difíceis. Isso se deve à diversidade de objetos, variações de iluminação, ruídos e outros desafios que podem estar presentes nas imagens[10].

Existem diferentes técnicas de segmentação de imagem, cada uma com suas características e aplicabilidades. Alguns tipos comuns de segmentação são:

- Segmentação baseada em limiarização: Nessa técnica, um valor de limiar é definido para separar os pixels da imagem em regiões de interesse. É útil para segmentar objetos com diferenças de intensidade bem definidas, como segmentar objetos em um fundo homogêneo[6].

- Segmentação baseada em regiões: Essa abordagem busca agrupar pixels que possuam características similares, como cor, textura ou intensidade, formando regiões coesas. É útil para segmentar objetos com características distintas, mesmo que não haja diferenças de intensidade nítidas[6].

- Segmentação por detecção de bordas: Essa técnica visa identificar as bordas ou contornos dos objetos na imagem. É útil para segmentar objetos com fronteiras bem definidas, como reconhecimento de objetos em imagens médicas ou detecção de objetos em visão computacional[6].

- Segmentação por agrupamento: Nesse método, os pixels são agrupados com base em similaridades de características, como cor, textura ou intensidade, sem a necessidade de um limiar pré-definido. É útil para segmentar objetos com distribuições de características complexas ou quando o número de regiões desejadas é desconhecido[6].

## III. Objetivos

A proposta do projeto é desenvolver um Algoritmo para a detecção e extração de caracteres de placas de carro padrão MERCOSUL. O algoritmo deve receber uma imagem, que contenha uma placa de carro padrão MERCOSUL, que será processada e retornará os caracteres que estão contidos na placa. Para desenvolver a proposta, os autores estabeleceram 4 objetivos a serem alcançados:

1) Algoritmo para extração da Placa: O algoritmo deve reconhecer o elemento da imagem que seja a placa de interesse. Esta parte do algoritmo deve retornar as coordenadas em que a região da placa se encontra.
2) Algoritmo para extração dos Caracteres: O algoritmo deve conseguir identificar as letras e números que estão contidas na placa. Ao mesmo tempo, em que ignore quaisquer elementos que não seja pertencente ao grupo alfanuméricos.
3) Implementar interface: De modo a facilitar a demostração do resultado, será adotada uma interface que consiga demonstrar as coordenadas da placa e os caracteres presentes nela.
4) Comparação: Para conseguir medir a eficiência do algoritmo desenvolvido, será utilizado outro algoritmo/programa de reconhecimento de placas de automóveis como referencial. Será verificado o quão próximo ambos conseguem realizar a análise, assim como qual obteve maiores acertos. Um banco de imagens, selecionado pelos autores, será utiliza como parâmetros para comparação.

Para o funcionamento do projeto, e suas principais características, o tópico seguinte irá abordar sobre o assunto.

## IV. Modelo

O algoritmo que está em desenvolvimento, pode ser dividido em duas seções: Extrair a Placa; Extrai caracteres. De modo a apresentar o resultado final, será adicionado uma interface para visualização da imagem examinada e os caracteres obtidos. Um arquivo externo será usado para comparar se a numeração da placa está presente ou não neste arquivo, simulando um controle de entrada e saída por reconhecimento. A sequência do funcionamento do projeto é exemplificado na Figura 2.

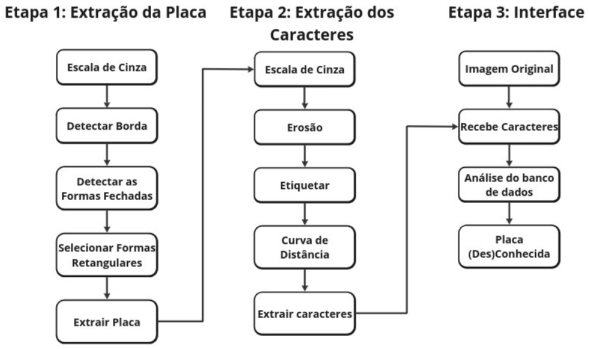


Fig. 2. Sequência de etapas do algoritmo

As etapas da Figura 2 serão descritas nos tópicos seguintes e em ordem.

### A. Etapa 1: Extração da Placa

#### a.1) Escala de Cinza

A escala de cinza ocorre quando a imagem não utiliza as camadas RGB (Red, Green, Blue), portanto, os valores do pixels da imagem pertencem a um intervalo de 0 a 255 ou de 0 a 1 em uma única camada. A utilização das camadas RGB pode ser de relevância dependendo do objetivo do projeto, um exemplo segundo Simon[11] é a utilização das camadas para a detecção de pele, auxiliando a identificar rostos e mãos. Para isso, Simon[11] diz aplicar a regra de Bayes, uma equação onde o valor pode ser atribuído como pele ou não pele. Para o presente projeto, é interessante a utilização da escala de cinza por ter menos camadas a serem analisadas.

#### a.2) Detecção de Borda

A prática de detecção de borda é utilizado recorrentemente em trabalhos relacionados a Visão Computacional, dentre as diversas técnicas, Gonzalez[6] destaca o algoritmo de Canny[12] como sendo mais complexo e de desempenho superior aos algoritmos de detecção de borda. De forma sucinta, a abordagem de Canny[12] se baseia em: Baixa taxa de erro; Os pontos de borda devem estar bem localizados; Resposta de um único ponto de borda.

Segundo Gonzalez[6], diferença entre as técnicas de detecção de borda pode ser visto entre as Figuras 3, 4 e 5. A Figura 3 é a imagem original, antes de sofrer as técnicas de detecção, a Figura 4 é o resultado obtido pelo algoritmo de Marr-Hildreth e a Figura 5 é a imagem obtida pelo algoritmo de Canny. É perceptível na Figura 4 alguns ruídos na imagem, enquanto na Figura 5 as bordas são mais contínuas e apresenta menos ruído.

Fig. 3. Imagem Original[6]

Fig. 4. Imagem Marr-Hildreth[6]

Fig. 5. Imagem Canny[6]

A primeira etapa do algoritmo, é a aplicação do Filtro Gaussiano na imagem, utilizando a Equação 1, fornecida por Solem[13], resultando em uma imagem suavizada. Os valores x e y seriam as coordenadas linha e coluna do pixel na imagem e o $\sigma$ é o desvio padrão da operação. A operação seguinte é o cálculo da magnitude pela Equação 2 por uma área (comumente 3x3) de pixels. O resultado obtido, é armazenado em outro pixel de coordenadas correspondentes. Ao mesmo tempo, em que é obtido os valores de Magnitude, é possível obter os valores da direção (ângulo) do gradiente pela Equação 3, fornecido por Gonzalez[6].

$$G_\sigma = \frac{1}{2\pi\sigma} e^{-(x^2+y^2)/2\sigma^2} \tag{1}$$

$$M(x,y)) = \sqrt{g_x^2 + g_y^2} \tag{2}$$

$$\alpha(x,y) = tg^{-1}\left|\frac{g_y}{g_x}\right| \tag{3}$$

O valor da Magnitude deve ser utilizado como valor de comparação aos valores de magnitude dos pixels vizinhos. Para analisar quais pixels devem ser utilizados na comparação, é utilizado a direção (ângulo) obtido pela Equação 3, fornecida por Gonzalez[6]. Os valores de ângulo podem pertencer a 8 intervalos distintos, conforme demostrado pela Figura 6. Se o resultado do ângulo estiver entre o intervalo de 67,5° à 112,5° o gradiente está apontando para a direita, portanto, os valores a serem comparados seria os pixels da direita e o da esquerda.

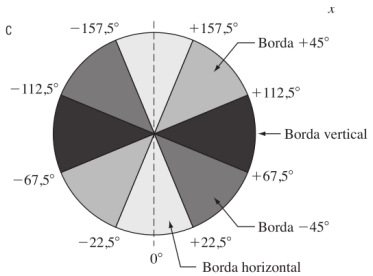


Fig. 6. Intervalos do Ângulo[6]

Para que o pixel analisado seja considerado um pixel de borda, seu valor de Magnitude deve ser superior aos dois pixels vizinhos utilizados na comparação. Caso contrário, o pixel não é um pixel de borda. A Figura 7 exemplifica o processo de análise. O píxel central deverá ser comparado ao pixel que o

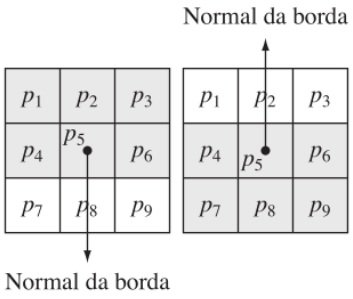


Fig. 7. Exemplos de direção de comparação[6]

vetor aponta e o pixel da direção oposta, em seguida, o seu valor de magnitude será comparado.

Mesmo com esse processo, alguns falsos positivos e falsos negativos pixels de borda podem surgir, de modo a corrigir a análise, o algoritmo de Canny[12] utiliza a limiarização por histerese. Esta limiarização utiliza um limiar baixo TL, e um limiar alto TH. Canny sugeriu que a razão do limiar alto para o baixo deve ser de dois out três para um[6]. A operação de limiarização pode ser vista como a criação de duas novas imagens de valor nulo e de mesmo tamanho da imagem a ser analisada, ambas imagens serão preenchidas conforme a Equação 4 e Equação 5, ambas fornecidas por Gonzalez[6]. Preenchida ambas as imagens com os novos pixels de borda, utilizamos a Equação 6, também fornecida por Gonzalez[6].

$$g_{NH}(x,y) = g_N(x,y) \geq T_H \tag{4}$$

$$g_{NL}(x,y) = g_N(x,y) \geq T_L \tag{5}$$

$$g_{NL}(x,y) = g_{NL}(x,y) - g_{NH}(x,y) \tag{6}$$

*a.3) Detectar Formas Fechadas*

Para verificar se a forma está fechada, deve-se utilizar o Seguidor de fronteira. Na Figura 8, temos um elemento de forma indefinida, sabe-se que seu contorno é os pixels de valor 1. Pode-se percorrer seu contorno a partir do pixel denominado $b_0$ na Figura 9 em um sentido horário, a partir do pixel denominado $c_0$, até encontrar um pixel de valor igual a 1. O processo se repete ao próximo pixel vizinho encontrado, conforme ilustrado na Figura 10. O processo se repete até que se encontre o primeiro pixel analisado, conforme é apresentado na Figura 11.

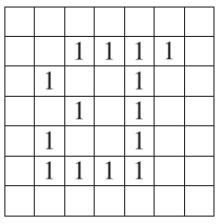

Fig. 8. Elemento em forma binária[6]

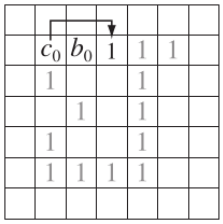


Fig. 9. Análise do Pixel Vizinho[6]

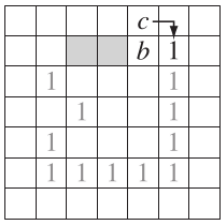


Fig. 10. Percorrendo os pixels de borda[6]

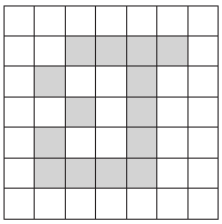

Fig. 11. Percorrido todos os pixels[6]

*a.4) Selecionar Formas Retangulares e Curva de Distância*

Ao aplicar a detecção de borda é possível obter a Assinatura de Distância ou Curva de Distância de um elemento. De forma sucinta, a Curva de Distância é a representação da distância dos pixels de borda do elemento até seu centroide, conforme é representado na Figura 12 ao analisar um círculo e um quadrado. Conforme é percorrido o pixel, seja pixel por pixel ou por ângulo, será montado um gráfico que representa o formato do elemento examinado, desta forma, é possível comparar as Curvas de diferentes elementos ao utilizar métricas de similaridade, obtendo a proximidade entre os elementos comparados.

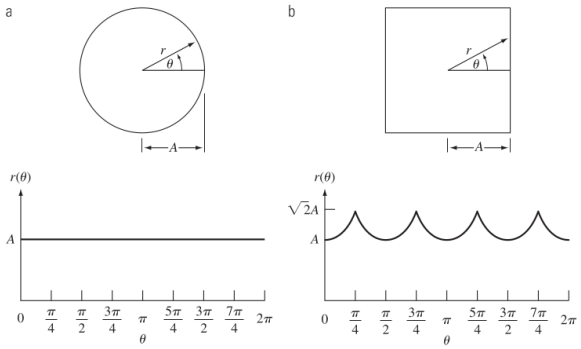


Fig. 12. Curva de Distância de um círculo e quadrado[6]

A métrica de similaridade é uma equação que calcula o percentual de semelhança entre duas imagens, para o presente projeto, se pretende a aplicação da Equação 7. Portanto, deve ser selecionado uma imagem de referência conforme os parâmetros necessários para o projeto. Como já foi obtido ambas Assinaturas de distância, estas serão utilizadas na Equação 7, podendo gerar valores no intervalo de -1 a +1. Quanto mais próximo de +1, mais semelhante serão as imagens, quanto mais próximo de -1, mais distintas são as imagens.

$$s(\mathbf{I}_1, \mathbf{I}_2) = \frac{\sum_{(u,v)\epsilon\mathbf{I}_1}[\mathbf{I}_1(u,v) - \bar{\mathbf{I}_1}][\mathbf{I}_2(u,v) - \bar{\mathbf{I}_2}]}{\sqrt{\sum_{(u,v)\epsilon\mathbf{I}_1}[\mathbf{I}_1(u,v) - \bar{\mathbf{I}_1}]^2 \sum_{(u,v)\epsilon\mathbf{I}_2}[\mathbf{I}_2(u,v) - \bar{\mathbf{I}_2}]^2}} \tag{7}$$

Portanto, para identificar o elemento que mais se aproxima do retângulo, será adotada a técnica de curva de distância. Por fim, será possível extrair as coordenadas em que a placa está localizada na imagem, encerrando a Etapa 1.

### B. Etapa 2: Extração dos caracteres

#### b.1) Escala de Cinza

A adoção da escala de cinza foi explicado anteriormente, na Etapa 1. Segue a mesma estratégia.

#### b.2) Erosão

O processo de erosão resulta na redução de pixels. Em casos em que uma imagem apresente muitos pontos isolados, representando ruídos, um processo de erosão simples pode eliminá-los. Apesar de útil para eliminação de ruídos, ele também reduz os pixels que compõe os elementos da imagem, portanto, o elemento pode acabar tendo seu volume ou espessura reduzidos. A Figuras 13 e Figura 14 exemplificam a erosão, a Figura 13 seria o elemento em sua forma original e a Figura 14 é o elemento após sofrido o processo de erosão, se repara que a diferença de espessura entre ambos, aumentou.

Fig. 13. J original[14]

Fig. 14. J após a erosão[14]

#### b.3) Etiquetar

Este processo é aplicado em uma imagem binária. Os elementos em uma imagem binária tem seu pixels atribuídos a valores de 1, e assim, os demais elementos também terão seus valores como 1. A ideia no processo de etiquetamento é atribuir valores diferentes entres os pixels de elementos diferentes, portanto, na Figura 15 temos uma imagem binarizada qualquer, com diversos elementos diferentes. Ao etiquetar, teremos a Figura 16, com os elementos em escalas distintas de cinza, essa é a representação que os valores dos elementos são distintos, sendo possível analisar os elementos de forma específica.

Fig. 15. Imagem Binária[14]

Fig. 16. Imagem Etiquetada[14]

#### b.4) Curva de Distância

Como foi explicada no tópico 4 da Etapa 1, a curva de distância, junto com a métrica de similaridade, servem para identificar imagens/elementos semelhantes. Na Etapa 2, a Curva de Distância será usado para comparar as letras/números encontradas na placa com outra imagem de referência que contenha as letras/números conforme o padrão MERCOSUL. Deste forma, será obtida as letras/números da placa, concluindo a Etapa 2.

### C. Etapa 3: Interface

Para demonstrar os resultados obtidos, uma interface será utilizado para representar a imagem a ser analisada e o resultado encontrado na Etapa 2. Após obter o resultado, será verificado se este está registrado ou não em uma lista externa, representando um controle de entrada e saída de veículos por reconhecimento.

## V. Ferramentas

Para o respectivo trabalho foi utilizado a linguagem de programação Python, incluindo algumas ferramentas da biblioteca OpenCV. O Python foi escolhido por apresentar algumas vantagens em relação a outras linguagens existentes, dentre quais podemos citar o fato de permitir ser executado nos sistemas operacionais Windows e Linux sem fazer alterações[13], ser uma linguagem que combina recursos, capacidades, com sintaxe de código muito clara, estando equipada com funcionalidade de biblioteca padrão grande e abrangente[15], e também levando em conta termos de facilidade, habilidade, desenvolvimento, interação com os usuários e paradigma de programação[16]. A biblioteca OpenCV fornece várias funções de processamento de imagem, bem como funções de análise de imagem e de padrões. OpenCV é uma abreviação para "Open Source Computer Vision Library" ou no português literal "Biblioteca de Visão Computacional de Código Aberto", que consiste em uma biblioteca com mais de 300 funções[17] voltada para visão computacional, se tratando de uma alternativa open-source para a comunidade, altamente recomendada para programadores que vão se aprofundar no campo da visão computacional. Isso se dá, principalmente, por ser capaz de criar aplicativos confiáveis, através de operações de processamento de imagem, como aprimoramento de imagem, remoção de ruído, detecção lateral, contorno e segmentação[16].

## VI. Aplicações

A leitura de placas de automóveis através da visão computacional têm sido uma demanda presente em diversas áreas da vida urbana, que vão desde o monitoramento de infrações em vias, identificar veículos roubados ou irregulares[18], controlar o acesso a estacionamentos, controle de entrada e saída apenas por veículos autorizados em condomínios, escolas[19], empresas etc. Desse modo, se mostra capaz de elevar a eficiência e eficácia do pagamento eletrônico de pedágio, a coleta de dados de movimentação de veículos, recuperação de carros roubados, identificação de veículos e infratores em investigações de excesso de velocidade, colisão, sequestro, roubo de carros e antiterrorismo[20]. Além de no cenário de cidades inteligentes, possuir um papel fundamental, como um facilitador chave de Sistemas de Transporte Inteligentes (ITS)[21]. Sendo constatado, que sua aplicação é responsável por um efeito dissuasor em uma ampla gama de ofensas, incluindo roubo de automóveis, violência, desordem pública e ofensas contra a propriedade[22].

## VII. Simulação

Nesta seção, serão apresentados os resultados parciais obtidos durante o desenvolvimento deste projeto.

Até o presente momento da entrega deste artigo, a equipe conseguiu desenvolver algumas funções para a realização do trabalho, como:

### A. Conversão para escala de cinza

No processo de conversão de uma imagem RGB para escala de cinza, ilustrado na Figura 17 , que claramente demonstra que o resultado obtido é coerente com o esperado.


Fig. 17. Processo de Conversão para escala de cinza .

Esse resultado pode ser verificado visualmente, onde a imagem em escala de cinza preserva as informações de intensidade da imagem original, mas sem as informações de cor. Os tons de cinza na imagem resultante representam a variação de intensidade da imagem original.

Uma métrica interessante para apresentar o resultado da conversão para escala de cinza é exibir o histograma, que mostra a distribuição das intensidades de pixels na imagem assim como mostra a Figura 18.

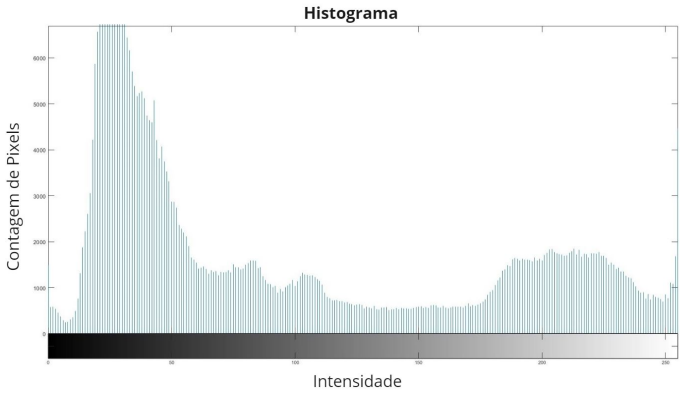


Fig. 18. Gráfico de Histograma.

O histograma é uma ferramenta que auxilia na definição dos limiares adequados para aplicar a detecção de bordas ou outras técnicas de processamento de imagem. Por exemplo, é possível identificar quais faixas de intensidade contêm as bordas mais relevantes e, assim, ajustar os limiares de forma mais precisa.

### B. Detecção de Bordas

O resultado parcial obtido para a função de detecção de bordas pode ser visualizado na Figura 19. A imagem revela as bordas identificadas pela função, destacando as transições de intensidade que indicam as possíveis fronteiras entre objetos na imagem original.

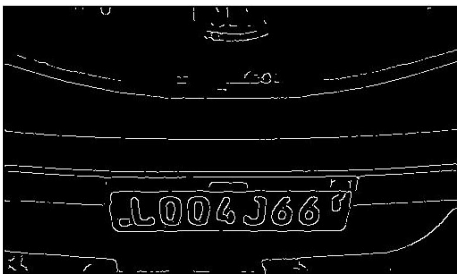

Fig. 19. Detecção de Borda

A métrica utilizada para verificar a validade do resultado obtido é a *Multiscale Structural Similarity*(SSIM)[23], para avaliação da qualidade de imagem. Essa métrica nos fornece uma medida de similaridade entre duas imagens, sendo neste contexto comparada a imagem de bordas gerada pela implementação com uma imagem de referência obtida por meio de bibliotecas prontas. O processo é ilustrado na Figura 20.

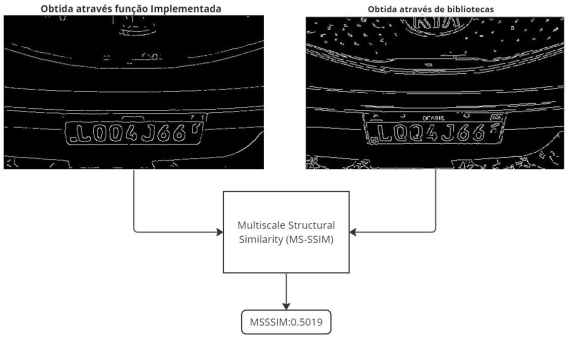


Fig. 20. Métrica de similaridade entre as imagens obtidas

O valor retornado para a métrica de similaridade é de 0.5019, o que indica que as imagens possuem aproximadamente 50% de similaridade. Esse resultado sugere que a função de detecção de bordas precisa ser aprimorada para obter melhores resultados.

## VIII. Conclusão

A área de Visão Computacional está em constante crescimento, sua adesão é tão ampla que consegue ser combinado com praticamente qualquer outra área de tecnologia. Quando combinada a Visão Computacional a outras tecnologias, como Inteligência Artificial, é possível alcançar resultados mais precisos e concluir operações cada vez mais complexas. O escopo deste projeto contribui na otimização vigilância no trânsito, ao automatizar o processo de reconhecimento de placas veiculares. E ainda, o projeto poderia ser implementada em estacionamentos inteligentes, para o controle automático de entrada e saída de veículos.

O desenvolvimento do algoritmo encontra o desafio ao ser optado a linguagem de programação Python. Os autores são familiarizados a utilizar o MATLAB, programa utilizado durante as aulas de Visão Computacional, e a mudança de linguagem permite a criação de oportunidades de erros ao programar. Para contornar esta barreira, os autores buscaram se aprofundar no estudo de Python, assim como na biblioteca OpenCV para Python, biblioteca recomendada nos artigos analisados.

Para trabalhos futuros, além do aperfeiçoamento do código, se planeja a aplicação do algoritmo em um caso real, medindo sua eficiência e requisitos necessários para sua aplicação. Espera-se que ao final da disciplina, não apensa seja concretizado os objetivos estabelecidos, mas que estimule os colegas ao estudo de Visão Computacional.

## Referências


[1] *Cidades - IBGE*. https://cidades.ibge.gov.br/brasil/pesquisa/22/0. Acesso em: 06 05 2023.

[2] Bruno Clemente Guingo. "Reconhecimento Automático de Placas de Veıculos Automotores". Tese de dout. Dissertação de Mestrado). Universidade Federal do Rio de Janeiro. Rio de Janeiro, 2003.

[3] Lele Xie, Tasweer Ahmad, Lianwen Jin, Yuliang Liu e Sheng Zhang. "A New CNN-Based Method for Multi-Directional Car License Plate Detection". Em: *IEEE Transactions on Intelligent Transportation Systems* 19.2 (2018), pp. 507–517. DOI: 10.1109/TITS.2017.2784093.

[4] Bruno Clemente Guingo, Roberto José Rodrigues e Antonio Carlos Gay Thomé. "Técnicas de Segmentação de Imagens, Extração de Características e Reconhecimento de Caracteres de Placas de Veículos". Em: *VII Simpósio de Informática e II Mostra Regional de Software Acadêmico*. Uruguaiana-RS, 2002.

[5] Conselho Nacional de Trânsito (CONTRAN). *Resolução Nº 729, DE 06 DE MARÇO DE 2018*. Portaria nº 729, de 06 de março de 2018. Disponível em: https://www.gov.br/infraestrutura/pt-br/assuntos/transito/conteudo-contran/resolucoes/resolucao7292018n.pdf. 2018.

[6] Rafael C Gonzalez e Richard E Woods. *Processamento de imagens digitais*. Editora Blucher, 2000.

[7] Maurıcio Marengoni e Stringhini Stringhini. "Tutorial: Introdução à visão computacional usando opencv". Em: *Revista de Informática Teórica e Aplicada* 16.1 (2009), pp. 125–160.

[8] Anderson Jesus, Eulanda Santos e Waldir Junior. "Reconhecimento de Placas Veiculares em Cenários Complexos utilizando o Método de Subespaço". Tese de dout. Nov. de 2021.

[9] Alexandre Barbosa, Bruno Silva e Juniro Bandeira. *PROCESSAMENTO DIGITAL DE IMAGENS PARA O RECONHECIMENTO DE PLACAS DE VEÍCULOS*. Faculdade de Balsas, 2017.

[10] Ogê Marques Filho e Hugo Vieira Neto. *Processamento digital de imagens*. Brasport, 1999.

[11] Prince J.D. Simon. *Computer Vision: Models, Learning, and Inference*. URL: http://www.computervisionmodels.com/.

[12] John Canny. "A Computational Approach to Edge Detection". Em: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-8 (6 nov. de 1986), pp. 679–698. ISSN: 0162-8828. DOI: 10.1109/TPAMI.1986.4767851.

[13] Jan Erik Solem. *Programming Computer Vision with Python*. 2012. URL: http://programmingcomputervision.com/downloads/ProgrammingComputerVision_CCdraft.pdf.

[14] Richard Szeliski. *Computer Vision: Algorithms and Applications 2nd Edition*. 2021. URL: https://szeliski.org/Book,.

[15] Saurabh Kapur. *Computer Vision with Python 3: Use the power of Python for real-time image processing and analysis*. 2017. ISBN: 1788299760.

[16] C E Widodo; K Adi; R Gernowo. "Medical image processing using python and open cv". Em: *Journal of Physics: Conference Series* 1524 (abr. de 2020), pp. 012003–012008. ISSN: 1742-6596. DOI: 10.1088/1742-6596/1524/1/012003.

[17] Gary Bradski; Adrian Kaehler. *Learning Opencv: Computer Vision with the Opencv Library*. 2008. ISBN: 0596516134.

[18] Vinıcius Campos Tinoco Ribeiro. "Automatic Brazilian Mercosur license plate recognition: an approach with synthetic images". B.S. thesis. Universidade Federal do Rio Grande do Norte, 2019.

[19] Fabio Augusto Cabral e Victor Hugo Pereira Machado. "Identificador de placa veicular: alternativa para segurança em escolas". Em: (2014).

[20] E. Barron; A. Slomowitz. "Massachusetts Should Facilitate – Not Inhibit – Law Enforcement Use of License Plate Data". Em: (set. de 2013), pp. 1–17.

[21] Soledad Pellicer, Guadalupe Santa, Andres L. Bleda, Rafael Maestre, Antonio J. Jara e Antonio Gomez Skarmeta. "A Global Perspective of Smart Cities: A Survey". Em: *2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. 2013, pp. 439–444. DOI: 10.1109/IMIS.2013.79.

[22] C. S. Koper; B. G. Taylor; D. J. Woods. "A randomized test of initial and residual deterrence from directed patrols and use of license plate readers at crime hot spots". Em: *Journal of Experimental Criminology* 9 (2 jan. de 2013), pp. 213–244. ISSN: 1572-8315. DOI: 10.1007/s11292-012-9170-z.

[23] Z. Wang, E.P. Simoncelli e A.C. Bovik. "Multiscale structural similarity for image quality assessment". Em: *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*. Vol. 2. 2003, 1398–1402 Vol.2. DOI: 10.1109/ACSSC.2003.1292216.