8ª JORNADA CIENTÍFICA E TECNOLÓGICA DO IFSULDEMINAS
5º Simpósio de Pós-Graduação

# APLICAÇÃO DE VISÃO COMPUTACIONAL PARA RECONHECIMENTO DE PLACAS DE VEÍCULO BRASILEIRAS

**José A. B. GOMES[1]; João M. RIBEIRO[2]**

## RESUMO

A visão computacional e seus recursos surgiram de forma a trabalhar computacionalmente a capacidade humana do enxergar. O poder de visualizar, classificar e reconhecer coisas à volta, capacidade que os humanos e muitos animais têm naturalmente, vem sendo trabalhado, com sucesso, em computadores. Esse trabalho visa abordar técnicas de visão computacional em uma aplicação de reconhecimento automático dos caracteres alfanuméricos das licenças de veículos atualmente em circulação no Brasil.

**Palavras-chave:** ANPR; OpenCV; Python; Visão Computacional; Reconhecimento Automático de Placas de Veículos

## 1. INTRODUÇÃO

**A** Visão Computacional é uma ampla área de pesquisa e aplicações que envolvem desenvolver soluções que permitam, na medida do possível, que o computador ganhe a capacidade de reconhecer e interpretar imagens. Tais processos envolvem uma série definida de etapas, que podem envolver diferentes técnicas para extrair as informações desejadas de imagens digitais. A necessidade constante de criação de soluções que automatizam processos envolvendo o processamento de imagens têm criado pesquisas inovadoras no ramo, símbolos alfanuméricos presentes em imagens (RICE; NAGY; NARTKER, 1999).

Complementarmente Kwaśnicka e Wawrzyniak (2002) afirmam que dentre as aplicações que envolvem OCR, está o Reconhecimento Automático de Placas de Veículo (ANPR). Tais sistemas de reconhecimento para esta finalidade têm sido desenvolvidos e aprimorados desde a década de 70, sendo utilizados para diversas finalidades, como monitoramento de trânsito, identificação de veículos em situação irregular, controle de estacionamento, dentre outros.

## 2. MATERIAL E MÉTODOS

Na etapa inicial do desenvolvimento, foi utilizada exclusivamente a linguagem *Python*, que foi escolhida para a criação do algoritmo de reconhecimento. A linguagem *Python* foi adotada por se tratar de uma linguagem de fácil aprendizado, rápido desenvolvimento e sintaxe de alto nível, além de trabalhar de forma satisfatória com

[1,2] Instituto Federal de Educação, Ciência e Tecnologia do Sul de Minas Gerais – Campus Muzambinho/MG
Email: joseantonio1@hotmail.com.br; joao.ribeiro@muz.ifsuldeminas.edu.br

processamento de imagens. Na execução do algoritmo de reconhecimento, são cumpridos os seguintes passos lógicos:

**2.1 Imagem de Entrada**

A imagem de entrada é capturada pela câmera do *smartphone*, que pode ocorrer de duas maneiras: de forma normal pelo usuário ou, com o aparelho devidamente conectado ao computador, por um comando do próprio algoritmo, que captura a imagem de forma automática.

**2.2 Aplicação de Contraste**

A aplicação de contraste na imagem de origem serve para tornar os caracteres da imagem bem escuros, ou bem claros, dependendo do padrão de cores da placa. A aplicação de contraste vai ajudar a eliminar os tons mais claros e medianos. Nessa etapa, o contraste é acrescido em 1050%. A imagem então é convertida para tons de cinza, usando o comando padrão do *Open* CV. Uma inversão de cores é aplicada, e a imagem inversa também é utilizada nos próximos passos.


**FIGURA 1:** Aplicação de contraste
FONTE: desenvolvida pelo autor

**2.3. Aplicação de Filtro de Cor**

A aplicação do filtro de cor tem por objetivo eliminar mais partes da imagem que não são importantes, de forma a sobrar o mínimo de informação possível. Nessa etapa, somente são mantidos os dez tons mais escuros em RGB, ou seja, somente os tons entre RGB (0,0,0) e RGB (10,10,10). O filtro de cor pode ser caracterizado como um processo de erosão, uma vez que este filtro retira pixels não desejados da imagem original. A análise é feita *pixel* a *pixel*.

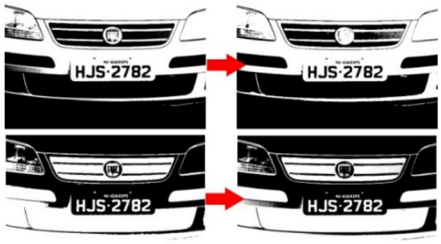


**FIGURA 2:** Antes e depois da aplicação do filtro de cor
FONTE: desenvolvida pelo autor

## 2.4. Segmentação de Grupos de pixels com proporção pré-definida

Nessa fase, os grupos isolados de *pixels* escuros são segmentados e classificados de acordo com a proporção entre altura e largura desses segmentos. Basicamente, foi observado em testes que os caracteres da licença ficam isolados em grupos de *pixels* escuros em relação ao fundo branco, e que tais segmentos apresentam uma proporção entre sua largura e altura bem definida.

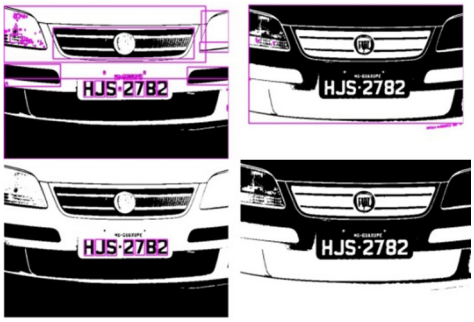


**FIGURA 3:** Imagens após a segmentação
FONTE: desenvolvida pelo autor

Foi estabelecido após uma série de testes, que embora o intervalo de valores para as proporções entre a altura e largura dos caracteres tenham ficado estáticos para qualquer situação, observou-se que esses valores devem ser estudados caso a caso, de acordo com as condições de captura (distância entre o veículo e a câmera, ângulo de captura, etc.). O estudo prévio fez-se por uma amostragem de fotos de exemplo, para que esses valores sejam ajustados até que se encontre um intervalo que permita mais precisão no reconhecimento dos caracteres.

## 2.5. Aplicação de Algoritmo OCR sobre as partes segmentadas

A etapa anterior gera uma série de imagens dos segmentos que possivelmente se tratam dos caracteres da licença do veículo. Assim, é necessário processar essas imagens em um algoritmo de OCR. No algoritmo, é utilizado o aplicativo *Tesseract OCR*, que pode ser instalado e utilizado nativamente com a linguagem Python.

## 2.6. Resultado do OCR

A etapa anterior gera como resultado uma cadeia de caracteres em ASCII que representa o que foi reconhecido pelo *Tesseract OCR*. Para a exibição do resultado final, é necessário aplicar algumas correções de caracteres ambíguos. Devido ao padrão de placas de veículos brasileiros ser composto por três caracteres alfa seguido de quatro caracteres

numéricos, é possível que sejam aplicados certos tratamentos que garantam que o resultado tenha essa característica.

## 3. RESULTADOS E DISCUSSÕES

De forma geral, o algoritmo apresentou bons resultados em imagens com boa iluminação e captura. Placas de motos foram menos reconhecidas que de carros, visto o seu formato diferenciado (em duas linhas).

Caso o local da captura seja fixo, garanta boa visão da placa e seja feito os ajustes das proporções dos caracteres, baseados em uma amostragem de imagens capturadas no local, é previsível que a taxa de acerto do algoritmo fique em torno de 80%, já que foi essa a taxa observada em testes realizados com capturas de boa qualidade.

## 4. CONCLUSÕES

Com a solução apresentada nesse trabalho, obteve-se resultado satisfatório, mas pode ainda ser melhorado, caso sejam adicionados métodos para reconhecimento, diferentes daqueles apresentados.

Por exemplo, algo que poderia melhorar bastante o reconhecimento seria um método para reconhecer o local da placa. O algoritmo desenvolvido não realiza esse reconhecimento, e segmenta os caracteres em si. Caso a área da licença fosse reconhecida, tratamentos específicos poderiam ser realizados naquela área reconhecida.

O uso da linguagem *Python* e *Java* para o desenvolvimento da aplicação apresentadas nesse trabalho permitem o uso em diferentes plataformas e devido a isso, testes em plataformas diferentes do Windows também podem evidenciar o multiuso da solução desenvolvida. Não apenas software, mas hardwares diferenciados também podem ser testados, como o *Raspberry PI*.

## REFERÊNCIAS

GONZALEZ, R.C.; WOODS, R.E. **Digital Image Processing** New Jersey: Pearson Pentrice Hall, 2008.

KWAŚNICKA, H., WAWRZYNIAK, B.: **License Plates Localization and Recognition in camera pictures**. Trabalho apresentado em: 3rd Symposium on Methods of Artificial Intelligence, 2002.

RICE, S.V.; NAGY, G; NARTKER, T.A. **Optical Character Recognition: An Illustrated Guide to the Frontier** New York: Kluwer Academic Publishers, 1999.