# *Detector de placas de trânsito*

*Um sistema capaz de detectar placas e informar a situação em que se encontra veículos fiscalizados*

Ítalo Rodrigo Moreira Borges
Universidade de Brasília - UnB
Brasília-DF, Brasil
italrmb@gmail.com

Marcos Adriano Nery de Abrantes
Universidade de Brasília - UnB
Brasília-DF, Brasil
marcosadrianonery@gmail.com

***Resumo*— Usando câmera embarcada e a raspberry pi 3, o sistema permite detectar as placas para fins de segurança e consulta.**

***Keywords—Embarcada, detectar as placas, segurança , consulta, OpenCV, Tesseract e ALPR.***

## I. JUSTIFICATIVA

As frotas de veículos vem aumentando a cada ano, de acordo o departamento de trânsito do Distrito Federal (Detran - DF) no ano 2007 havia 964.534, já em 2017 verificou-se 1.716.878, observa que houve quase um aumento de 100% da frota em apenas 10 anos, porém esse aumento está ficando cada vez mais acentuado. Em fevereiro de 2018 o Detran-DF registrou 1.726.148 veículos no distrito federal, percebe-se que em apenas 2 meses houve um aumento de aproximadamente 2.000 frotas[1].

De acordos com os números documentados pela entidade que regulamenta o transporte terrestre do DF, constata-se que será difícil controlar e gerenciar os veículos por sistemas de monitoramento com grande eficiência.

A extração automática do conteúdo das placas de automóveis a partir de imagens viabiliza uma vasta quantidade de aplicações. Para agentes de segurança dos centros urbanos, por exemplo, permite identificar veıculos roubados ou irregulares, monitorar vias para multar precisamente motoristas infratores e controlar o acesso a estacionamentos [2].

Tendo em vista a problemática, esse projeto tem como solução o auxílio aos agentes de órgãos regulamentadores de trânsito, seja em uma simples patrulha ou nas "Blitz" realizada pelos mesmos. Com o projeto em questão pretende-se realizar a leitura de uma dada placa presente, feita a conversão desses dados para um banco de dados, e a partir desse banco de dados será realizado um processo onde será verificado a condição do veículo se o mesmo está irregular ou não assim o agente presente somente será alertado se o automóvel se encontrar em caráter não permissível de circulação. Podendo-se verificar os veículos de forma instantânea para o auxílio de infrações, perseguições, roubo de carro, placas clonadas e toda uma gama de irregularidades que podem vir a tornar não admissível a circulação do veículo.

## II. OBJETIVO

### *A. Auxiliar o agente de trânsito no desempenho de suas funções*

Este é o principal, pois a partir deste podemos alcançar os objetivos secundários abaixo. Sendo por via do projeto tornar mais rápido e eficiente as patrulhas e "Blitz" realizadas pelos órgãos responsáveis. E alertar em tempo real o agente quanto a condição do veículo, o que permite um gerenciamento e uma rápida ação.

*a) Auxiliar nas infrações*

O órgão regulamentador de trânsito poderá utilizar os dados em um banco de dados para analisar presumíveis infrações.

*b) Automatizar uma ação efetiva quanto a possível incidente com veículos roubados*

Como há a possibilidade de consultar o banco de dados, se torna eficiente a verificação de veículos quanto a serem furtados, de uma forma autônoma tornando mais fácil e eficiente a caracterização do fato.

*c) Atuar com precisão quanto a quais veículos parar*

Quem também ganha é o motorista, sendo que os motoristas em situação regular não serão parados pois o agentes responsáveis não mais agirão por suposições e instinto.

*d) Toda a população tende a ser beneficiada*

Além de que uma fiscalização mais eficiente tende a coibir os roubos e promover uma maior segurança. Assim toda a população será beneficiada em termos de que o condutor irá procurar sempre deixar seu

carro regular, e também auxiliado de uma correta implementação uma possível diminuição de infrações.

## III. Requisitos

Como o mínimo necessário para o projeto ser desenvolvido temos:

- Raspberry pi 3 modelo b;
- Fonte de energia para Raspberry pi 3 modelo b;
- Módulo de Câmera;
- Carro;
- Suporte para Raspberry;
- Cooler;
- Caixa para envolver e proteger a Raspberry pi 3 modelo b;
- Conseguir o reconhecimento de todos os possíveis algarismos presentes nas placas do sistema nacional;
- Ter uma câmera capaz de detectar os dígitos da placa a luz do dia na corrente aplicação;
- Tempo máximo para reconhecimento da imagem e alerta deve ser menor do que o tempo de passagem do veículo pela Blitz;

## IV. Benefícios

Tornar possível aos estudantes responsáveis entender e aplicar o conteúdo apresentado em sala, a saber da disciplina de Sistemas embarcados, tornando possível aos envolvidos segundo a proposta da disciplina entender de forma a serem capazes de replicar tudo aquilo que se tem expectativa que os estudantes sejam capazes de cumprir.

Quanto acerca do projeto visa de uma forma eficiente e também em acordo com os requisitos da disciplina apresentar uma possível forma de gerenciar os veículos, buscando alternativamente criar-se um auxílio ao órgão regulamentador de trânsito que o uso que tornaria possível a estes uma identificação autônoma de um veículo suspeito. Atuando de forma breve e pontual.

E ainda promover por meio de uma fiscalização mais eficiente uma maior segurança e coibir os roubos. Beneficiando toda a população em termos de que o condutor irá procurar sempre deixar seu carro regular. Auxiliando também na prevenção de acidentes com a retirada de veículos irregulares.

## V. Descrição de Hardware

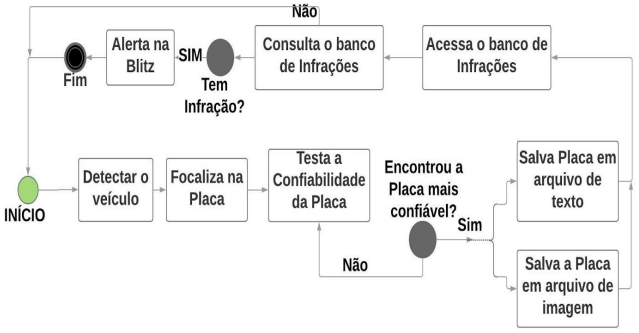
Por enquanto serão usados os seguintes hardware listados na tabela 1. Percebe-se a simplicidade no quesito hardware, isso se deve que a aplicação exige uma concentração maior no software.

Tabela 1 - Materiais utilizados

| | Materiais | Fabricante |
|---|---|---|
| 01 | Raspberry PI 3 modelo B | Raspberry |
| 01 | WebCam | - |
| 01 | Teclado-Mouse | |
| 01 | Display | LANDZO |
| 01 | conversor HDMI-VGA | - |
| 01 | Fonte de Alimentação 5V - 3A | - |
| 01 | Cooler | |

## VI. Descrição de Software

Para que este projeto possa a vir ser realizado, é necessário a utilização da biblioteca OpenCV (Open Source Computer Vision Library) [3]. Esta responsável pelo processamento das imagens que serão capturadas.




Fluxograma 1 - Funcionamento do Projeto

### *A.* Detecção e reconhecimento de algarismos

O algoritmo montado conta com o uso da biblioteca OpenCV, do software tesseract-OCR e do ALPR.

- O OpenCV: O OpenCV (Open Source Computer Vision Library) é uma biblioteca de software de visão computacional e aprendizado de máquina em código aberto.[3]
- O Tesseract é um mecanismo de reconhecimento de texto, do inglês optical

character recognition (OCR) de código aberto, disponível sob a licença Apache 2.0. Pode ser usado diretamente ou (para programadores) usando uma API para extrair texto impresso de imagens. Suporta uma ampla variedade de idiomas.[4]

- OpenALPR é uma biblioteca de *reconhecimento de placa de licença automática de* código aberto escrita em C ++ com ligações em C #, Java, Node.js, Go e Python. A biblioteca analisa imagens e fluxos de vídeo para identificar as placas de veículos. A saída é a representação de texto de qualquer caractere da placa de licença.[5]

*B.* Tirar e Salvar Fotos

Para a geração de imagens foi-se utilizado as funções presentes na biblioteca OpenCV, estas permitem a geração e apresentação das imagens na tela. Com as funções dessa mesma biblioteca realiza-se a abertura de uma câmera via comunicação serial universal (USB), as imagens são atualizadas continuamente em um intervalo que permite a visualização na tela sendo essa projeção realizada por essa mesma. As imagens são salvas e sobrepostas pelas seguintes, o salvamento das imagens é necessário para que nesse meio tempo a função raiz da biblioteca ALPR possa realizar o processamento sobre a imagem. As Threads presente em videoRead.cpp *thread_camera* e a *thread_grava* são responsáveis por isso.

Na figura 01 tem-se a interface atual, colocou-se um fundo preto para melhor visualização dos dados, que mais a frente serão explicados. Toda essa construção é feita apartir da OpenCV que também permite escrever dados sobre imagens.


Figura 01 - Captura e amostragem da câmera

*C.* Detecção da placa e dígitos presentes

A partir da imagem salva a função nativa da biblioteca ALPR na Thread *thread_alpr* presente no código videoRead.cpp realiza os seguintes passos, segundo [6]:

- Detecção: Encontra potenciais regiões de placas.
- Binarização: Converte a imagem da região da placa em preto e branco.
- Análise Char: Encontra "bolhas" de tamanho de caracteres na região da placa.
- Bordas da placa: Encontra as arestas / forma da matrícula.
- Deskew: Transforma a perspectiva em uma visão direta com base no tamanho ideal da placa de licença.
- Segmentação de Personagem: Isola e limpa os caracteres para que possam ser processados individualmente.
- OCR: Analisa cada imagem de caractere e fornece várias letras / confidências possíveis.
- Pós-processamento: Cria uma lista das principais possibilidades de placas com base nas confidências de OCR.Também realiza uma correspondência de Regex nos modelos de região, se solicitado.

O ALPR na última fase apresenta por padrão dez possíveis resultados da leitura feita sobre a imagem, e apresenta junto a cada leitura o nível de confiabilidade da mesma, como pode ser visto na figura 02. Os dados apresentados são salvos em um arquivo de texto para o próximo passo que seria manipular e peneirar esses dados para a obtenção da correta placa.

```
2304  plate0: 10 results
2305      - NNX380I    confidence: 91.5825
2306      - NNX38OI    confidence: 88.4206
2307      - NNX38QI    confidence: 85.6413
2308      - NNX38DI    confidence: 85.5214
2309      - NNX3G0I    confidence: 83.5119
2310      - NNX3B0I    confidence: 82.9642
2311      - HNX380I    confidence: 82.8582
2312      - MNX380I    confidence: 82.8206
2313      - NMX380I    confidence: 82.7267
2314      - NNX38GI    confidence: 82.6341
```

Figura 02 - Valores fornecidos pelo ALPR

*D.* Processamento dos dados fornecidos pelo ALPR

Tendo os dados salvos em um arquivos de texto, o primeiro passo é a correta manipulação do mesmo, as funções utilizadas para tal pertencem a biblioteca POSIX. Na figura 01 pode ser visto na primeira linha de texto a linha atual que está sendo lida do arquivo. Nessa linha pode ser visto ainda o formato acerca de cada

escrita feita pelo ALPR. Também pode ser visto o nível de confiança da leitura em questão. A peneira de dados se dá em alguns passos, em primazia pega-se um total de 60 leituras, dentre as 20 primeiras é adquirida a de maior confiança. Depois é se resetado o nível de confiança para 85, e então é se lido mais 20 amostras dentre essas 20 é se pego a com maior confiança e isso se repete por mais uma vez. Ao fim é testado quais algarismos é comum entre os 3 requeridos, e assim é obtido dentre 60 amostras a mais válidas dentre as tais. Na última linha é possível verificar esse valor em questão, todo esse processamento é realizado pela Thread *thread_ler_dados* presente em videoRead.cpp.

### *E.* Verificação da placa

A partir dos algarismos obtidos será verificado no banco de dados qual a situação que o veículo se encontra. Se o mesmo se encontra irregular, logo será informado de qual natureza é a irregularidade.

.

### *F.* Alerta ao agente

O agente será alertado por meio de um dispositivo, onde aciona assim que for apresentado a situação de cada veículo que passou pela Blitz.

## VII. Resultados

As bibliotecas OpenCV, tesseract-ocr e ALPR alem de outros softwares foram instalados com sucesso. Isto permitiu que a leitura de uma placa veicular padrão brasileira fosse lida com sucesso como apresentado.

Foi-se realizado a concepção de dois códigos implementados em C++ o que permitiu que houvesse uma correta abertura da câmera, salvamento de imagens e análise destas assim foi possível alcançar o objetivo principal que é determinar quais os dígitos que compõem uma placa veicular.

## VIII. Conclusão

Neste ponto de controle, foram criados códigos em C++ o que respeita o escopo da disciplina e realizado a instalação das biblioteca OpenCV, Tesseract-OCR, ALPR e outros diversos cujo qual os anteriores dependem para seu correto funcionamento. Até este momento atendeu-se grande parte do objetivo proposto. Restando ainda a verificação quanto a situação do veículo.

Todo o escopo deste foi realmente desafiador. Gastou-se uma grande porção de tempo no trabalho, e grande parte deste tempo foi-se gasto na instalação das bibliotecas que apresentam uma grande dificuldade em sua compilação e instalação em especial o OpenCV para a Raspberry 3.

A dupla preocupou-se em adquirir conhecimentos necessários para a utilização das bibliotecas e nesse sentido fica citado o [7] que disponibiliza um curso gratuito nesse sentido o que permitiu um grande avanço no uso da biblioteca OpenCV. Verificou-se com sucesso a possibilidade real de um projeto na Raspberry PI 3.

Todo o trabalho realizado até o presente momento oportunizou a dupla um desenvolvimento fantástico e a mesma pode constatar o quão poderosas são estas bibliotecas e a Raspberry PI 3 consegue ser. O avanço quanto ao entendimento acerca de processamento de imagens foi algo que o trabalho em questão melhorou com bastante destaque.

## IX. Anexos

### *A.* Códigos

https://github.com/marcosadrianonery/SistemasEmbarcados/tree/master/2_PCs/Codigos_PC_03

## Referências


[1] Estatísticas do Trânsito, site: http://www.detran.df.gov.br/o-detran/estatisticas-do-transito.html. Acessado em: 03/04/2018

[2] Corneto, G., da Silva, F. A., Pereira, D. R., de Almeida, L. L., Artero, A. O., Papa, J. P., de Albuquerque, V. H. and Sapia, H. M. (2017). A new method for automatic vehicle license plate detection, IEEE Latin America Transactions 15: 75 – 80.]

[3] OpenCV library, acesso em 06/06/2018. http://projectabstracts.com/ list-of-projects-on-image-processing

[4] Tesseract-OCR library, acesso em 06/06/2018. https://github.com/tesseract-ocr/

[5] OpenALPR library, acesso em 06/06/2018. https://github.com/openalpr/openalpr

[6] OpenALPR, acesso em 06/06/2018. https://github.com/openalpr/openalpr/wiki/OpenALPR-Design

[7] OpenCV para iniciantes, acesso em 06/06/2018. https://courses.learnopencv.com/p/opencv-for-beginners