

Uma introdução ao OpenALPR

marco.mangan@gmail.com

2019-12-14

- 1 Visão Geral
- 2 OpenALPR
- 3 Privacidade
- 4 Oportunidades

Esta apresentação baseia-se no Capítulo 2 [1] do livro *Computer Vision and Imaging in Intelligent Transportation Systems*, editado por Loce, Bala e Triveti.

A maior parte do código está escrito em `C++`, recomendo o livro de Lippman et alli: *C++ Primer* [2] para os interessados na linguagem.

A leitura automática de placas de veículos é uma aplicação de visão computacional na identificação do registro de um automóvel com o uso de câmeras.

O recurso substitui o reconhecimento manual e o uso de dispositivos, como RFID, na segurança pública, estacionamentos, pedágios e controle de frota. O *Open Automatic License Plate Recognition* (OpenALPR) é um sistema escrito em C++, distribuído sob a *GNU Affero General Public License v3.0*, que pode ser utilizado a partir de aplicações em C#, Java, Node.js e Python, incluindo aplicações móveis.

Com o objetivo de incentivar e discutir seu uso, a palestra apresenta uma visão geral do reconhecimento de placas de veículo, do impacto na privacidade e das oportunidades de novas aplicações.

- ✓ Distribuições Linux e *Aplicações Livres*
 - Administração de Sistemas, Redes e *Segurança*
- ✓ *Desenvolvimento*, Banco de Dados e Web
 - Virtualização, *Big Data* e *Cloud Computing*
- ✓ Kernel, *Sistemas Embarcados* e *Internet das Coisas*
- ✓ *Empreendedorismo* e *Negócios*
- ✓ Comunidade e *Filosofia*

- Usuários
 - veículos privados
 - comerciais
 - públicos
- Provedores
 - logradouros privados
 - comerciais
 - públicos

Temas do sistema de transporte

- Aumento da segurança
- Aplicação de normas, leis e regulamentos
- Aumento da eficiência e redução de custo
- Aumento da proteção e conforto

- Aplicações na década de 1970
- Custos baixam após 1990
- Atual tendência de uso da Nuvem, Internet das Coisas e dispositivos móveis

- Placas de veículos são itens obrigatórios
- Identificador único em pedágios e estacionamentos
- Placas podem ser obtidas manualmente
- Problemas: clonagem, obstáculos, fumaça, intempéries e velocidade
- Complementa ou substitui: bilheterias, barreiras, identificação por rádio frequência

- Recuperação de veículos aumenta em 68% (EUA, 2012)
- Execução de prisões aumenta em 55% (EUA, 2012)
- Pedágio na Nova Zelândia (2009)
- Pedágio na Suécia (2007)

- Uso em estacionamentos, como sistema secundário
- Fotos de infrações (ex. excesso de velocidade)
- Sistema de Identificação Automática de Veículos (SINIAV)
- Placas do Mercosul

Placa do Mercosul



- OpenALPR
 - comercial ou aberto
 - diferenças em benchmarks
- AWS Rekognition
 - reconhecimento de objetos e caracteres em geral

OpenALPR

OpenALPR Agent







File Configure View Help

Cameras Services

axing
parking_lot

Delete Add Edit

Recent Plates


AEBX2K		us-mn 5s ago
FC1D8H		us-mn 5s ago
FB9J2E		us-mn <1 min ago
SE6H3G		us-mn <1 min ago
DAOC7L		us-mn <1 min ago
AD3U6V		

View Full Results

Resume

openALPR

parking_lot



FFS: 26.7 Motion: 98% Recognized: 14%

Resolution: 1280x720

Integração com Python

```
from openalpr import Alpr

alpr = Alpr("us", "/path/to/openalpr.conf", "/path/to/runtime_data")
if not alpr.is_loaded():
    print("Error loading OpenALPR")
    sys.exit(1)

alpr.set_top_n(20)
alpr.set_default_region("md")

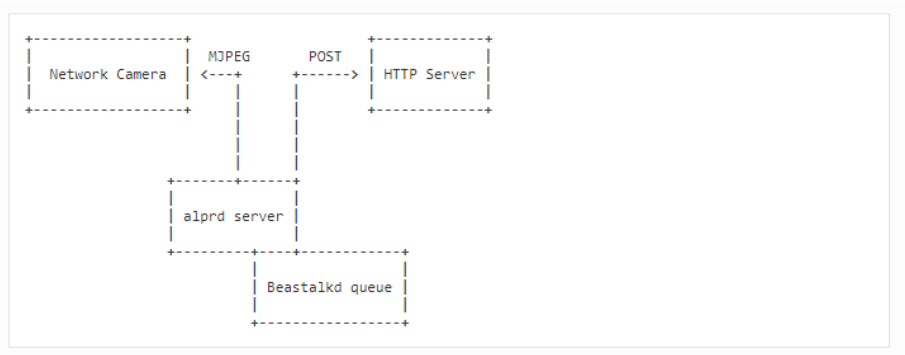
results = alpr.recognize_file("/path/to/image.jpg")

i = 0
for plate in results['results']:
    i += 1
    print("Plate #%" % i)
    print("    %12s %12s" % ("Plate", "Confidence"))
    for candidate in plate['candidates']:
        prefix = "-"
        if candidate['matches_template']:
            prefix = "*"

        print("    %s %12s%12f" % (prefix, candidate['plate'], candidate['confidence']))

# Call when completely done to release memory
alpr.unload()
```

Integração via HTTP



- Localizar e identificar texto em imagens
- Imagens de veículos apresentam distorções e ruídos
- Desafios: sombras, contraste baixo, variação na iluminação, rotação, inclinação, adesivos, oclusão, sujeira.

Exemplo de reconhecimento



Plate	Confidence (%)	Processing Time (ms)
PE3R2X	88.94%	152.94 ms

Fluxo de processamento geral

- Captura de imagem
- Localização da placa
- Segmentação de caracteres
- Reconhecimento de caracteres
- Localização de jurisdição

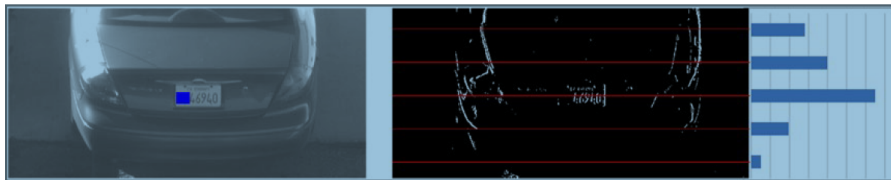
- Vídeo ou imagem
- Câmeras dedicadas ou genéricas
- Velocidade do veículo
- Local fixo ou móvel

Câmera dedicada



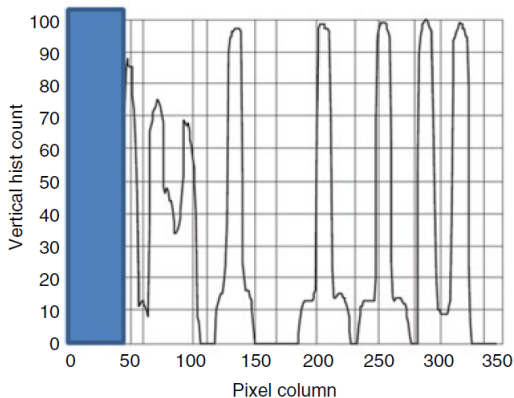
- Região de interesse separa veículos e contorno
- Reduz custo de processamento
- Métodos por cor, borda, textura
- Afetado por treinamento

Exemplo de localização



- Delimitação da área de caracteres individuais
- Afetada pela cor e textura da placa

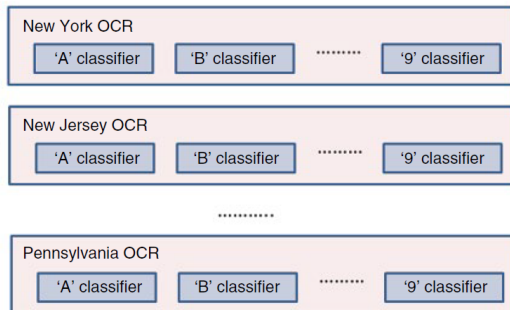
Segmentação por histograma



- Identificar dígito ou letra
- Afetado pela fonte e condições da placa
- Afetado por treinamento

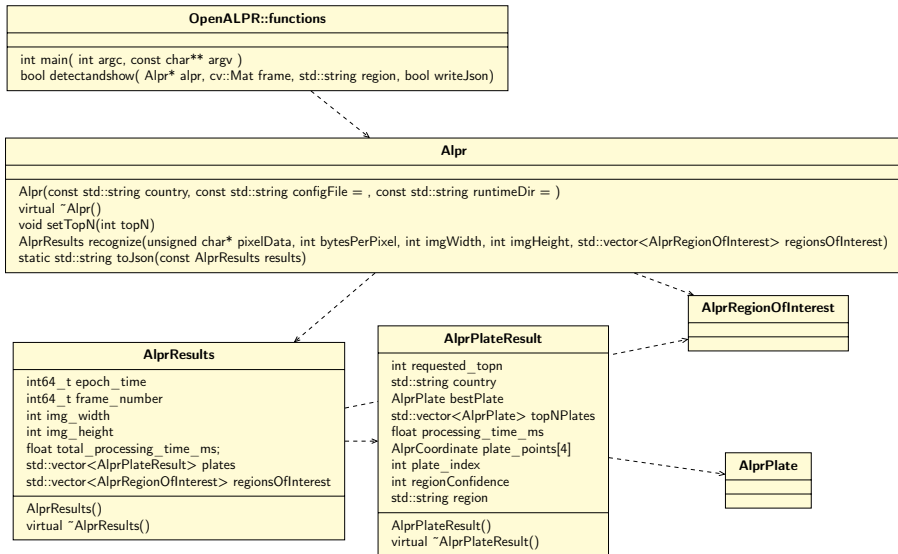
Reconhecimento de fontes diferentes

EDR5308



- Detection ([region?]`detector.cpp`)
- Binarization (`binarizewolf.cpp`)
- Char Analysis (`textdetection/characteranalysis.cpp`)
- Plate Edges (`edges/platelines.cpp` e `edges/platecorners.cpp`)
- Deskew (`licenseplatecandidate.cpp`)
- Character Segmentation (`ocr/segmentation/charactersegmenter.cpp`)
- Optical Character Recognition (`ocr/ocr.cpp`)
- Post Processing (`postprocess/postprocess.cpp`)

Fachada e aplicação

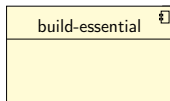
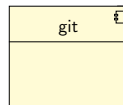
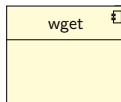
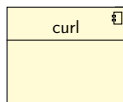
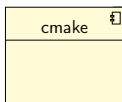
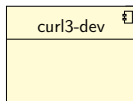
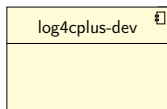
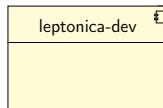
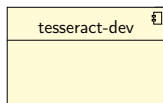
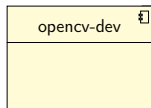


- Variantes de modelos (tamanho, cores)
- Revisões de modelos (Mercosul, amarelas, cinzas)
- Placas estrangeiras, colecionadores, militares, oficiais
- Identificar logotipos, texto e marcas ao redor da área de interesse que foi reconhecida com sucesso
- Permite a confirmação do registro em um cadastro

Principais elementos do sistema

- Processamento de imagens
- Reconhecimento de caracteres
- Registro de dados
- Comunicação em rede

Dependências do OpenALPR



Dependências indiretas

- build-essentials: g++, gcc, libc6-dev, make
- brew: homebrew, ruby
- tesseract: libpng, jpeg, libtiff, leptonica
- opencv: eigen, lame, x264, xvid, ffmpeg, ilmbase, openexr, gdbm, openssl[TLS], sqlite, xz, python, python@2, numpy, tbb

- 3000 imagens
- 200 imagens

- Reconhecimento de face
- Rastreamento na Rede
- Rastreamento de dispositivos
- Rastreamento de veículos

- Perseguição de jornalistas e ativistas
- Perseguição de condenados
- Perseguição de suspeitos sem antecedentes
- Falta de regulamentação
- Falta de *habeas data*

- Cobrança e operação em estacionamento privado (sem: cartão, chave, controle remoto)
- Pedágios e barreiras
- Controle de velocidade
- Controle de frota (garagem, localização)

- Transporte público: ônibus, táxi, lotação
- Sistema principal ou secundário
- Integrado com câmeras de segurança pública, segurança privada e câmeras abertas
- Cumprimento de horário
- Previsão de chegada e aviso de interrupções
- Menor custo de operação (sem RFID)
- Mesma câmera, diversos sistemas, diversas funções
- Google Transit e similares



Aaron Burry and Vladimir Kozitsky. Automated license plate recognition. In Robert P. Loce, Raja Bala, and Mohan Trivedi, editors, *Computer Vision and Imaging in Intelligent Transportation Systems*, chapter 2, pages 15–45. Wiley, 2017.



Stanley Lippman. *C++ Primer*. Addison-Wesley, 2012.