

Reconhecimento de placas veiculares

Acesso ao Cine Drive-in e Comanda Eletrônica

Brenda Medeiros Santos
Faculdade UnB Gama
Gama-DF, Brasil()
brenda.eng.unb@gmail.com

Gabriela Conceição dos Santos
Faculdade UnB Gama
Gama-DF, Brasil
gabrielacsantos.engunb@gmail.com

Resumo – Este relatório apresenta o Projeto desenvolvido para a disciplina Sistemas Embarcados. O Projeto consiste na implementação de um sistema de reconhecimento de placas de veículos, usando o *Raspberry Pi*, para controle de comanda no Cine Drive-in de Brasília.

Palavras-chave – *Raspberry Pi*, reconhecimento de placas, *OpenCV*, *Tesseract*, Cine Drive-in, comanda eletrônica.

I. INTRODUÇÃO

O monitoramento de veículos tem se tornado cada vez mais necessário diante do aumento da frota e do fluxo destes nos centros urbanos. De acordo com a Agência Internacional de Energia, espera-se que em 2035 haja cerca de 1.7 bilhões de automóveis, o que representa aproximadamente o dobro da frota atual [1].

Os avanços tecnológicos têm possibilitado maior controle desse fluxo e monitoramento de veículos, nesse sentido destaca-se a identificação e reconhecimento das placas de carros que podem ter diversas funcionalidades, como: detecção de infratores, carros roubados, estudo de tráfegos e, principalmente, para permitir o acesso de veículos a determinados locais [2].

1.1 Cine Drive-in

Inaugurado em agosto de 1973, o Cine Drive-in de Brasília (Figura 1) é o último do país em funcionamento. Possui 15 mil metros quadrados de área asfaltada, capaz de acomodar 400 veículos em seu estacionamento, uma tela de concreto medindo 312 metros quadrados, sendo sua projeção feita com moderno projetor Digital Barco [3].

No Cine Drive-in, a transmissão do áudio do filme para o rádio dos carros é feita com um transmissor de FM. Caso o carro do usuário não possua som, basta acender o farolete de seu veículo e solicitar orientação do atendente [3].

Além disso, para o maior conforto e comodidade de seus usuários, o Cine Drive-in disponibiliza também um atendimento de lanchonete realizado no veículo por um garçom. Desse modo, ao ingressar no cinema o usuário recebe um cardápio com todos os alimentos e bebidas oferecidos e usa o farolete para solicitar o atendente [3].



Figura 1: Cine Drive-in de Brasília.

Sendo assim, perante a necessidade de aperfeiçoamento e automatização do serviço de atendimento prestado pela lanchonete do Cine Drive-in e com o intuito de se evitar eventuais equívocos nos pedidos realizados por seus usuários, propõe-se a construção e implementação de um sistema de reconhecimento de placas veiculares para comandas individualizadas.

Para cada carro será criada uma comanda que será controlada pelo usuário através do aplicativo que deverá ser instalado no celular. Tal aplicativo contabilizará os pedidos e somará os gastos para que o cliente possa efetuar o pagamento na saída, cuja identificação de comanda será feita através da leitura da placa na saída, momento em que a placa será retirada da lista de carros presentes no local.

II. REVISÃO BIBLIOGRÁFICA

2.1 Sistemas comerciais:

Aborda-se o cenário atual das tecnologias de reconhecimento de placas de automóveis e de comandas eletrônicas, serão apresentados alguns produtos desses segmentos, que são oferecidos por empresas privadas, com descrição de seu funcionamento e preço, caso essas informações sejam de domínio público.

2.1.1 Reconhecimento de Placas:

2.1.1.1 ALPR da Motorola: trata-se de um dispositivo móvel que fica acoplado a um servidor situado no interior do veículo, realizando o processamento de dados adquiridos e avisando somente quando uma placa, em específico, é reconhecida. Segundo a Motorola, o sistema está apto para reconhecer as placas mesmo em condições

climáticas e temporais adversas. De acordo com dados do fabricante, a implementação do sistema tem variação entre 12.250 a 18.700 dólares nos EUA, esse valor pode variar em virtude da quantidade de câmeras a serem utilizadas [4].

2.1.1.2 **3M:** em termos de requisitos, o dispositivo para a captura de placas da **3M** é bastante semelhante ao da Motorola. Um diferencial desse produto é possuir a tecnologia *Triple Flash*, cuja a função é suprimir os efeitos causados pelos faróis de outros carros ou do pôr do sol na captura da imagem da placa [5].

2.1.1.3 - **Coban:** construído com o intuito de realizar a captura de placas de veículos, a distinção entre o produto desenvolvido pela empresa Coban e os demais apresentados está no fato desse produto poder atualizar seu banco de dados via conexão *wireless* e, além disso, ser capaz de identificar a posição geográfica da placa de interesse. De acordo com o fabricante, a Coban, o seu produto está avaliado a partir de 9.500 dólares [6].

2.1.2. Comanda Eletrônica:

2.1.2.1 - **Altec:** a comanda eletrônica foi desenvolvida para ser utilizada, em bares e restaurantes, por garçons por meio de tablets, durante o atendimento aos usuários desses estabelecimentos. Segundo a empresa desenvolvedora, os diferenciais da comanda Altec com relação a outras oferecidas no mercado são: possuir uma interface amigável e de fácil aprendizado, apresentar funcionalidades em uma única tela, navegação por ícones, multi-funções em qualquer etapa do pedido, acessos rápidos às categorias e pratos, grupo de produtos customizáveis: cores e ícones para fácil visualização [7].

2.1.2.2 - **Pocket Cheff:** é um sistema de comanda eletrônica composto por um software completo de gerenciamento de restaurante que proporciona maior agilidade no atendimento, aumento da rotatividade de mesas e redução de falhas em pedidos, melhorando a rentabilidade do estabelecimento [8].

O Pocket Cheff foi desenvolvido para estabelecimentos de todos os tamanhos: de bares, lanchonetes, redes de fast food e restaurantes. Através da solução de comanda eletrônica, o garçom captura os pedidos que são enviados automaticamente para a cozinha por meio de iPods ou terminais touch screen. Este sistema para restaurantes oferece uma visão geral do status dos pedidos, controle diário do caixa e divisão de conta por pessoa [8].

2.1.2.3 - **Atendi:** foi criado para simplificar e agilizar o atendimento em estabelecimentos como bares, restaurantes e lanchonetes. A proposta desse sistema é trazer os melhores recursos tecnológicos em uma nova forma de apresentar e vender produtos. E termos de atendimento, o Atendi se destaca por: gerar comanda eletrônica, realizar o controle de caixa, apresentar o cardápio de forma digital, fazer pedidos através de tablets ou celulares, ou seja, trata-se de um frente de caixa completo com Sistema de Gestão Integrada (ERP) online [9].

2.2 Ferramentas para o processamento digital de imagens:

2.2.1 OpenCV

O OpenCV (Open Computer Vision) é uma biblioteca multiplataforma usada no desenvolvimento de aplicativos para

as áreas de Processamento de Imagens e Visão Computacional. Dispõe de vários algoritmos, onde pode-se destacar os de: segmentação, reconhecimento de faces, aprendizado de máquinas, filtragem de imagens, dentre outros [10].

A biblioteca detém interfaces com linguagens como C, C++, Python e Java, além de suportar sistemas operacionais Windows, Linux, Mac OS e Android. Além disso, ela dispunha de ferramentas de processamento de imagens e vídeos como filtros de imagem, calibração de câmera, reconhecimentos de objetos e análise estrutural tornando o processo de programação mais factível [10].

2.2.2 Tesseract

O Tesseract foi originalmente desenvolvido na Hewlett-Packard Laboratories Bristol e na Hewlett-Packard Co, Greeley Colorado, entre os anos de 1985 a 1994, com algumas mudanças, foi portado para Windows em 1996, além de alguns “C++zing” (upgrades) em 1998. Em 2005 foi liberado a comunidade pela HP e desde 2006 é então desenvolvido pela Google [11].

Tesseract-OCR (Optical Character Recognition) trata-se uma biblioteca de código aberto e seu objetivo é a leitura de textos e caracteres de uma imagem. Ela tem a capacidade de transformar a imagem de um texto em um arquivo .txt do mesmo. Com os avanços no âmbito da programação, essa biblioteca foi modificada, permitindo o funcionamento da mesma em programação Python. Por esse motivo, o nome desta biblioteca para a linguagem Python tornou-se “PyTesseract” [11].

O funcionamento do PyTesseract se deve a diversos arquivos externos, chamados de dicionários. Nestes arquivos há diferentes tipos de fontes de letras e diferentes combinações de palavras. Assim é possível que o programa faça a leitura de qualquer frase ou caractere que se encaixe nesses arquivos [11].

III. REQUISITOS

- Raspberry Pi 3 Model B;
- Câmera ou Webcam;
- Captura de imagem;
- Detecção de placa veicular;
- Reconhecimento de placa veicular;
- Cadastro da placa no sistema;
- Descadastro da placa no sistema mediante efetivação do pagamento;
- Aplicativo com descrição de cardápio para pedidos.

IV. DESENVOLVIMENTO E RESULTADOS

Diante das informações apresentadas, o objetivo deste projeto é a identificação da placa do veículo e o armazenamento de seus caracteres na lista de placas para a criação de comandas individualizadas a serem acessadas através de um aplicativo no celular.

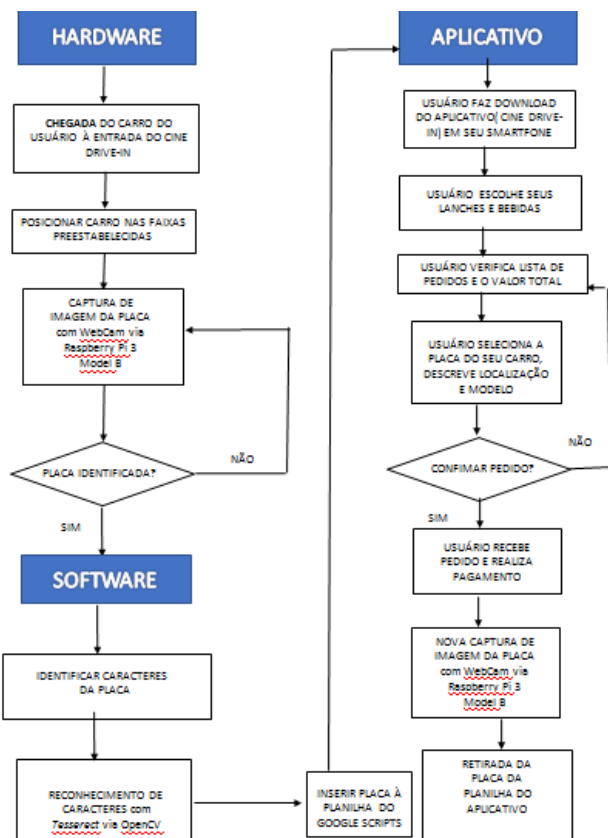
O veículo irá se posicionar na entrada, de modo que a câmera conseguirá visualizar a placa. Mediante pagamento das entradas, o atendente irá acionar o executável, momento em que será feita a captura da imagem, a identificação dos

caracteres e o reconhecimento de quais letras e algarismos estão na placa, para posterior cadastro da placa no sistema.

A ideia inicial é trabalhar com a identificação de veículos particulares brasileiros (com placas de fundo cinza claro e caracteres pretos). Podendo ser ampliada com modificações no sistema para outros modelos de placas de veículos.

De maneira resumida, as atividades do sistema serão a captura da imagem para identificação da placa do veículo, em seguida, deve ser feita a identificação dos 7 caracteres presentes na placa, que serão reconhecidos como letras e algarismos. Por fim, a placa lida pelo sistema será adicionada à lista de placas presentes no Cine Drive-in, que permite o acesso ao cardápio e às compras.

A seguir, têm-se o fluxograma simplificado do Sistema de Reconhecimento de Placas:



Fluxograma 1: dinâmica do funcionamento do Sistema de Reconhecimento de Placas.

4.1 Descrição de Hardware

O sistema de reconhecimento de placas será desenvolvido através do uso de um *Raspberry Pi 3*, uma câmera (Webcam conectada via USB). Segue abaixo descrições técnicas de cada componente utilizado no projeto:

A. Raspberry Pi 3 Model B: o Raspberry Pi 3 é a terceira geração Raspberry Pi (Figura 2), possui um processador Broadcom BCM2837 de 64 bits, clock de 1.2GHz, Wifi e Bluetooth 4.1 integrados. **Custo médio do componente:** R\$170 - R\$250.



Figura 2: Raspberry Pi 3 Model B.

B. WebCam Multilaser PLUG&PLAY 1.3MP MIC USB 2.0: possui resolução de 1.3MP para vídeo e 5.0MP para foto (via interpolação), LED flash; microfone USB, clip para fixar a webcam na tela LCD, compatível Windows 7/Vista/XP/2000, lente 2G de vidro, S/N Radio: 45dB, interface USB(Figura 3) **Custo médio do componente:** R\$50 - R\$100.



Figura 3: webcam modelo Plug&Play Multilaser.

4.2 Descrição de Software

Para o software, as atividades são a identificação da placa, o reconhecimento dos caracteres e o aplicativo que permitirá o acesso ao cardápio e aos pedidos. O processo de captura de imagens é feito através da função “*fswebcam*”, cada imagem é salva, analisada e em seguida substituída pela posterior (assim como os arquivos gerados durante a execução do código), para não haver acúmulo de arquivos desnecessários na memória. As Figuras 4 e 5 apresentam duas das imagens capturadas com o uso do *Raspberry Pi* sob diferentes condições de luminosidade.



Figura 4: placa 1 capturada pelo sistema.



Figura 5: placa 2 capturada pelo sistema.

Para uma melhor identificação dos caracteres da placa, foi utilizado o Código 1 em anexo, escrito em *Python*. O código realiza um processo de filtragem da imagem, aumentando seu brilho, contraste e realizando uma binarização da mesma. A medida de se utilizar o código citado foi tomada, pois este facilitou o uso do Tesseract para a conversão da imagem em texto sem erros. As Figuras 6 e 7, apresentam as imagens captadas pelo sistema após o tratamento do filtro e binarização.

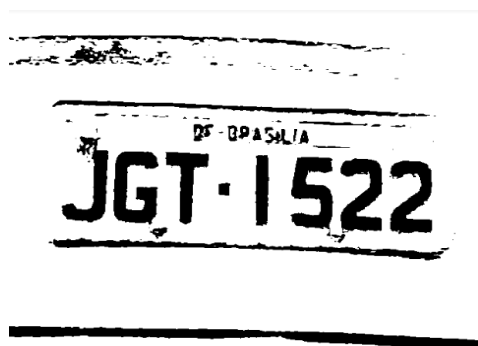


Figura 6: imagem da placa 1 captada pelo sistema após filtragem e binarização.



Figura 7: imagem da placa 2 captada pelo sistema após filtragem e binarização.

Enquanto as Figuras 8 e 9 mostram a leitura realizada pelo *Tesseract*, dessas placas, respectivamente. É possível notar a presença de espaçamentos inadequados nas placas lidas, dessa forma se fez necessário no Códigos 2 e 3 (em anexo) a utilização de uma condicional que substituísse os espaços para que as placas ficassem em seu formato correto.

```
6.jpg filtro.py placa.png tesse.py
pi@raspberrypi:~/Documents $ tesseract result.png stdout
Warning: Invalid resolution 0 dpi. Using 70 instead.
Estimating resolution as 579
Detected 31 diacritics
JGT 1 522
pi@raspberrypi:~/Documents $
```

Figura 8: leitura feita pelo *Tesseract* da imagem da placa 1.

```
6.jpg filtro.py out.png servidor.c
pi@raspberrypi:~/Documents $ tesseract result.png stdout
Warning: Invalid resolution 0 dpi. Using 70 instead.
Estimating resolution as 566
PBA 1453
pi@raspberrypi:~/Documents $
```

Figura 9: leitura feita pelo *Tesseract* da imagem da placa 2.

Após as placas terem sido lidas pelo *Tesseract*, a placa lida será adicionada a uma planilha (Figura 10), utilizando o código do Google Scripts (Figura 11). A planilha possui 3 colunas, em que a coluna A refere-se às placas adicionadas, a coluna B refere-se às placas que saíram do Cine Drive-In e por fim, a coluna D que corresponde às placas que ainda estão no local.

Há ainda, colunas que foram omitidas, que possuem as funções para permitir a atualização da lista de placas presentes no local. De maneira geral, trata-se de funções condicionais que excluem placas repetidas e que concatenam a lista de placas para que estas sejam utilizadas, posteriormente, no aplicativo.

← → ↻ <https://docs.google.com/spreadsheets/d/1FEpLiW-cwX8>

PLACAS ☆

Arquivo Editar Ver Inserir Formatar Dados Ferramentas Complementos

100% RS % .0 .00 123 Arial 10 B

	A	B	D
1	-JGT-1522	-PAR-1746	-JGT-1522
2	-RBI-1689	-PAI-3257	-RBI-1689
3	-JKJ-2016		-JKJ-2016
4	-PAR-1746		
5	-JGG-2779		-JGG-2779
6	-PAI-3257		
7	-IPW-5058		-IPW-5058
8	-JUT-7458		-JUT-7458
9	-JKL-4567		-JKL-4567
10	-GBI-1234		-GBI-1234
11	-PBA-1453		-PBA-1453
12			

Figura 10: planilha de placas detectadas gerada pelo código do Google Script.

https://script.google.com/d/1bHUIKSFA_HADennQcve5VnYs3Q0SciFi-Mcz8YwzvPCKZgUT7XsuaPqi/edit

laca

Editar Ver Executar Publicar Recursos Ajuda

Código.gs

```

1 var planilha = SpreadsheetApp.openById("1FEpLiW-cwX8x0Ue9sceixghVc-HtoL_5TMJE3RF16xs")
2 var sheet = planilha.getActiveSheet();
3
4 function doGet(e)
5 {
6   var lin, num_linha, final;
7   var num_linha = sheet.getLastRow();
8   var placas = e.parameter.info;
9
10  for (var i=1; i<num_linha; i++) {
11    lin = sheet.getRange(i,1); //verifica se a coluna 1 está vazia
12    if (lin.isBlank()){
13      final = i;
14      var celula = sheet.getRange(i,1);
15      celula.setValue(placas);
16      break;
17    }
18  }
19  return ContentService.createTextOutput("PLACA ADICIONADA!");
20 }

```

Figura 11: código do Google Script que gera a planilha com as placas detectadas.

4.2.1. Aplicativo

O aplicativo que possibilitará a realização dos pedidos, foi desenvolvido no APP INVENTOR[12], que apresenta ao usuário em sua tela inicial duas abas com o cardápio do Cine Drive-In, sendo uma primeira aba à esquerda com toda a disponibilidade de lanches (Figura 12) e uma segunda à direita contendo as bebidas disponíveis (Figura 13), ambas as abas apresentam os preços atualizados de acordo com o próprio cardápio do Cine Drive-In[3].



Figura 12: Tela inicial aba com a categoria “LANCHES”.



Figura 13: Tela inicial aba com a categoria “BEBIDAS”.

Nas duas categorias apresentadas na tela inicial do aplicativo, “LANCHES” e “BEBIDAS”, o usuário tem a opção de escolher a quantidade de cada pedido que deseja. Após escolher a quantidade, para adicionar o pedido à lista, ele precisará apertar no ícone “PEDIR”. Feito isso, este ícone passará a ser apresentado como “NA LISTA”, significando que o pedido já foi adicionado a essa.

A tela inicial do aplicativo ainda conta com um ícone “ADM” para uso exclusivo dos administradores do Cine Drive-in, o acesso ao ícone só é permitido mediante uma senha (Figura 14), que deve ser escolhida pelos administradores e nele consta uma lista com todos os pedidos realizados pelos usuários do cinema ao longo de cada sessão (Figura 15). À medida que os usuários vão finalizando seus pedidos a lista desse ícone vai sendo atualizada com eles.



Figura 14: solicitação da senha para acesso ao ícone “ADM”.



Figura 15: descrição de todos os pedidos solicitados sendo mostrada na tela “ADM”.

Para visualizar seus pedidos, o usuário terá que ir até o ícone “PEDIR” localizado no canto superior da tela inicial. Desse modo, uma segunda tela (Figura 16) lhe será apresentada, nela constará todos os pedidos feitos por ele e o valor total de seus pedidos, sendo este valor associado a sua placa, que pode ser selecionada conforme mostra a (Figura 17). Caso deseje alterar o pedido, basta ele apertar o ícone “VOLTAR” adicionar ou retirar algum pedido e repetir todos os procedimentos descritos anteriormente.

Pedidos

VOLTAR LISTA DE PEDIDOS

Produto: CHEESE DRIVE-IN
Quantidade: 3

Produto: CERVEJA
Quantidade: 2

TOTAL = R\$ 49.00

DADOS DO VEÍCULO:

PLACA DO CARRO: JKK-1522

LOCALIZAÇÃO

COR OU MODELO

CANCELAR FINALIZAR

Figura 16: Tela secundária apresenta lista de pedidos, total a pagar e placas que foram captadas pelo sistema.



Figura 17: seleção de placa para ser feita pelo usuário.

Ainda nessa segunda tela, constará os “DADOS DO VEÍCULO”, onde ele escolherá a placa respectiva ao seu veículo que foi cadastrada ao entrar. Terá ainda, a opção de descrever a localização e as características do seu carro a fim de facilitar a entrega, ou caso o usuário considere necessário fazer algum comentário acerca de sua localização.

Conforme foi dito anteriormente, após a finalização do pedido, este será enviado para uma planilha (Figura 18), através de um código executado no Google Scripts (Figura 19), que atualizará a lista com pedidos realizados e todas as suas respectivas informações, assim como estarão também na tela de ADM. Desse modo, os administradores possuem controle total sobre a ordem de pedidos realizados.

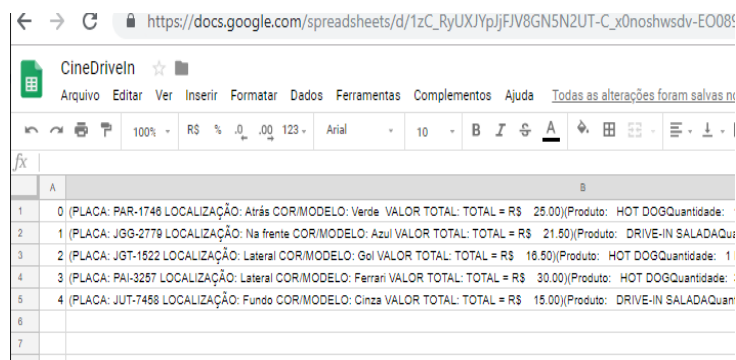


Figura 18: planilha de pedidos detectadas.

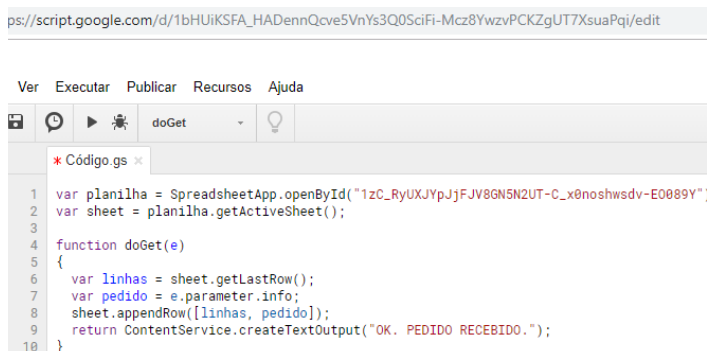


Figura 19: código do Google Scripts que permite a geração da URL para adição de dados dos pedidos.

Acerca das placas atualizadas no componente de seleção na tela de Pedidos, esta também é realizada através de um código executado no Google Scripts que permite a inserção após a leitura de novas placas à lista. Além disso, gera também uma URL com o texto da coluna referente às placas adicionadas. Essa função é realizada utilizando recursos da Google API que possibilita leitura de colunas e torna mais fácil a inserção dessas *strings* no aplicativo (Figura 20).



Figura 20: arquivo gerado pela função do Google API.

V. CONCLUSÃO

Diante do exposto, fica passível de perceber que o sistema de reconhecimento de placas implementado funcionou de modo eficiente, uma vez que apresentou excelentes resultados para uma mesma configuração de brilho, contraste e corte. Tais configurações foram suficientes para o reconhecimento correto de placas em situações distintas de luminosidade.

Com relação ao aplicativo, este apresentou conexão com o sistema de reconhecimento de placas veiculares, o que possibilitou o preenchimento das placas na planilha que atualizava-se a cada nova leitura de placa do sistema e, consequentemente, atualizava o componente do aplicativo. Além disso, de maneira análoga, o sistema de retirada de placas funcionou corretamente, permitindo que somente as placas presentes no local permanecessem na lista de placas do aplicativo.

O aplicativo funcionou sem apresentar qualquer falha. O único requisito para o funcionamento é a conexão com a internet. As planilhas atualizam-se automaticamente, o que faz com que o trabalho dos operadores do Cine Drive-In seja facilitado e cabe à eles, somente, a retirada dos pedidos que foram entregues e a compilação dos executáveis nos momentos de entrada e da saída, mediante pagamento de consumo.

VI. REFERÊNCIAS BIBLIOGRÁFICAS

- [1] Nascimento, Jean Dias do. "Detecção e reconhecimento de placa automotiva com baixo custo." (2012).
- [2] Leite, B.B., "Localização Automática de Placas de Veículos Automotores Particulares em Imagens Digitalizadas", Projeto Final, DEL/UFRJ, junho, 2003.
- [3] "Cine Drive-in Brasília - Um cinema fora de Série". Disponível em: <http://cinedrivein.com/>. Acesso em: 21 de outubro de 2018.
- [4] "ALPR - Motorola". Disponível em: <http://motorolasolutions.com> . Acesso em: 21 de outubro de 2018.
- [5] WEBER, Henrique; "Um protótipo móvel para detecção automática de placas veiculares brasileiras", UFRGS, 2013.
- [6] ALPR - CobanTech"; Disponível em: <http://www.cobantech.com/www/ALPR.html> . Acesso em: 21 de outubro de 2018.
- [7] "Comanda Eletrônica ALTEC". Disponível em: <https://www.altec.ws/solucoes/comanda-eletronica/> . Acesso em: 22 de outubro de 2018.
- [8] "Pocket Cheff". Disponível em: <http://www.cheffsolutions.com.br/produtos/pocket-cheff/>. Acesso em: 22 de outubro de 2018.
- [9] "Atendi". Disponível em: <http://www.atendi.com.br/solucoes#atendimento>. Acesso em: 22 de outubro de 2018.
- [10] Gary Bradski, Adrian Kaehler, "Learning OpenCV: Computer Vision with the OpenCV Library, Ed. 39; Reilly Media, Inc.
- [11] Smith, Ray. "An overview of the Tesseract OCR engine." Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on. Vol. 2. IEEE, 2007.
- [12] App Inventor; Disponível em: <http://ai2.appinventor.mit.edu/> . Acesso em: 22 de outubro de 2018.

ANEXOS

→ **Código 1:** escrito em *Python* realiza a filtragem e a binarização da imagem captada;

tratar_foto.py

```
import cv2
import numpy as np
from PIL import Image

# Open a typical 24 bit color image. For this kind of image there are
# 8 bits (0 to 255) per color channel
img = cv2.imread('placa.jpg')
# mandrill reference image from USC SIPI

s = 640
img = cv2.resize(img, (s,s), 0, 0, cv2.INTER_AREA)

def apply_brightness_contrast(input_img, brightness = 0, contrast = 0):

    if brightness != 0:
        if brightness > 0:
            shadow = brightness
            highlight = 255
        else:
            shadow = 0
            highlight = 255 + brightness
        alpha_b = (highlight - shadow)/255
        gamma_b = shadow

        buf = cv2.addWeighted(input_img, alpha_b, input_img, 0, gamma_b)
    else:
        buf = input_img.copy()

    if contrast != 0:
        f = 131*(contrast + 127)/(127*(131-contrast))
        alpha_c = f
        gamma_c = 127*(1-f)

        buf = cv2.addWeighted(buf, alpha_c, buf, 0, gamma_c)

    return buf

b = 60
c = 130

out = apply_brightness_contrast(img, b, c)

cv2.imwrite('out.png', out) #aumentar brilho e contraste

img = cv2.imread('out.png',0)
ret,thresh1 = cv2.threshold(img,72,255,cv2.THRESH_BINARY) #preto e branco

cv2.imwrite('result.png', thresh1) #resultado para tesseract
```

→

Código 2: escrito no *Google Apps Script* permite a criação de um link para a inserção de novas placas à planilha do aplicativo;

```
var planilha = SpreadsheetApp.openById("1FEpLiW-cwX8xOUe9sceixghVc-  
HtoL_5TMJE3RF16Xs");  
var sheet = planilha.getActiveSheet();  
  
function doGet(e)  
{  
  var lin, num_linha, final;  
  var num_linha = sheet.getLastRow();  
  var placas = e.parameter.info;  
  
  for (var i=1; i<num_linha; i++) {  
    lin = sheet.getRange(i,1); //verifica se a coluna 1 está vazia  
    if (lin.isBlank()){  
      final = i;  
      var celula = sheet.getRange(i,1);  
      celula.setValue(placas);  
      break;  
    }  
  }  
  return ContentService.createTextOutput("PLACA ADICIONADA!");  
}
```

→ **Código 3:** escrito no *Google Apps Script* permite a criação de um link para retirada de placas da planilha do aplicativo;

```
var planilha = SpreadsheetApp.openById("1FEpLiW-cwX8xOUe9sceixghVc-  
HtoL_5TMJE3RF16Xs");  
var sheet = planilha.getActiveSheet();  
  
function doGet(e)  
{  
  var lin, num_linha, final;  
  var num_linha = sheet.getLastRow();  
  var placas = e.parameter.info;  
  
  for (var i=1; i<num_linha; i++) {  
    lin = sheet.getRange(i,2); //verifica se a coluna 3 está vazia  
    if (lin.isBlank()){  
      final = i;  
      var celula = sheet.getRange(i,2);  
      celula.setValue(placas);  
      break;  
    }  
  }  
  return ContentService.createTextOutput("PLACA RETIRADA!");  
}
```

→ **Código 4:** escrito em **Linguagem C** realiza a **INSERÇÃO** de novas placas à planilha do aplicativo;

```
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>

int main(int argc, const char * argv[])
{
    char fp;
    char link[117];

    system("fswebcam placa.jpg");
    system("python3 tratar_foto.py");
    system("echo links2
https://script.google.com/macros/s/AKfycbwEAXjN8yF9P4wpAGKa4RY71-
luJnK8VjRZ_M5qFv7Q8cI0OqM/exec?info=>> resultado.txt");
    system("tesseract result2.png resultado2");

    system("cat resultado.txt resultado2.txt > result.txt");
    fp = open("result.txt", O_RDONLY);
    read(fp, &link, 117);

    close(fp);

    for(int i=10; i<117; i++){
        if(link[i]=='\n' | link[i]==' '){
            link[i]='-';
        }
    }

    system("rm resultado.txt resultado2.txt result.txt"); //apagar arquivos criados

    //printf("\nO link eh: %s\n\n", link);
    system(link);
    return(0);
}
```

→ **Código 5:** escrito em **Linguagem C** realiza a **RETIRADA** de placas da planilha do aplicativo, mediante pagamento por parte do usuário;

```
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>

int main(int argc, const char * argv[])
{
    char fp;
    char link[117];

    //system("fswebcam placa.jpg");
    system("python3 tratar_foto.py");
    system("echo links2
https://script.google.com/macros/s/AKfycbybu0nvWAmfHkNFBWvZl_0Vw37MF1OLvEtgKwI93_g2Gwx
6llo/exec?info=>> resultado.txt");
    system("tesseract result2.png resultado2");
}
```

```
system("cat resultado.txt resultado2.txt > result.txt");
fp = open ("result.txt", O_RDONLY);
read(fp, &link, 117);

close(fp);

for(int i=10; i<117; i++){
    if(link[i]=='\n' | link[i]==' '){
        link[i]='-';
    }
}

system("rm resultado.txt resultado2.txt result.txt"); //apagar arquivos criados

//printf("\nO link eh: %s\n\n", link);
system(link);
return(0);
}
```