Controle de Acesso por Leitor de Placa Veicular.

Ponto de Controle 3

Guilherme Marques Moreira da Silva
16/0029503
Programa de Engenharia de Software
Faculdade Gama - Universidade de Brasília
Brasília - DF, Brasil
guilherme@gmail.com

Vinícius Inacio Breda
16/0041511
Programa de Engenharia Eletrônica
Faculdade Gama - Universidade de Brasília
Brasília - DF, Brasil
vini.ibreda@gmail.com

I. Justificativa

Para obter uma maior praticidade e mais segurança na entrada de carros em estacionamentos com portões automáticos controlados por controle infravermelho, utilizaremos processamento de imagem para acionar abertura e o fechamento do portão.

O monitoramento da entrada é facilitado pelo envio das informações da entrada para a internet, garantindo mais segurança para quem entra na garagem.

II. Objetivos

Utilizando uma Raspberry Pi, o objetivo desse projeto é desenvolver um dispositivo de fácil manuseio e baixo custo, cuja a finalidade é controlar a abertura de portões utilizando uma placa veicular autorizada.

III. Requisitos

O sistema final deverá controlar a abertura do portão, com validações de funcionamento no próprio sistema, por meio de sensores, reconhecimento de placas por meio de imagem ao vivo e cadastramento e remoção de placas válidas.

A detecção deve ser feita com uma WebCam fixa no portão, o processamento da imagem será feito com OpenCV [1], em Python. Data, hora e placas detectadas deverão ser enviadas para um servidor para controle e as placas autorizadas poderão ser adicionadas e removidas do sistema para controle do acesso.

Um sensor infravermelho deve detectar se o comando para abrir o portão foi ou não enviado e uma confirmação de funcionamento será feita por meio de um LED, que deve acender se o comando for enviado com sucesso.

IV. Benefícios

As vantagens de se utilizar o controle de acesso são:

- Acionamento automático do portão sem a intervenção do motorista;
- Saber data e hora de entrada de todos os veículos cadastrados;
- O usuário não precisará comprar nenhum dispositivo para a abertura do portão, assim terá uma economia;
- O sistema sempre se manterá atualizado, pois quando o usuário trocar de veículo o mesmo é obrigado a atualizar os dados;
- Maior eficácia e agilidade ao entrar e sair;
- Maior segurança para o usuário, já que não dá chance de perder o controle do portão;
- Sem necessidade de controles extra para cada veículo, resultando em economia financeira;

V. Revisão Bibliográfica

Uma das maneiras mais utilizadas atualmente para a abertura e fechamento de portões é a eletrônica, que é uma solução considerada simples e funciona através de um acionamento automático, fazendo que o usuário não precise realizar esforço manual.

Para a abertura do portão é necessário utilizar um controle, o usuário aperta um botão para energizar o seu circuito interno. Enquanto o circuito é energizado, ele transmite um código que é enviado por um comando de radiofrequência (280, 292, 299, 315 e 433 Mhz) ou infravermelho para uma central. Essa central é um receptor que compara o código recebido com o armazenado em sua memória, autorizando o acionamento do motor. [2]

Para realizar o projeto vamos utilizar o funcionamento do portão eletrônico e acrescentar uma funcionalidade, em vez de todos os usuários terem que acionar o controle eles deverão cadastrar anteriormente a placa do seu automóvel e, com o auxílio de uma câmera e uma Raspberry Pi, o portão será aberto.

Existem alguns projetos semelhantes a esse, onde é necessário fazer a leitura de uma placa para que haja uma maior segurança na hora de autorizar uma pessoa a entrar no local. Algumas empresas prestam esse serviço são ROCKEN [4] e Tecnima [3]. O produto da ROCKEN serve como

autenticação em duas etapas para liberar o acesso ao prédio, porém tem a mesma premissa, controlar e monitorar o acesso ao estacionamento [4]. Já o da Tecnima funciona de forma semelhante ao que será feito no projeto, com reconhecimento de placa para liberação do acesso, porém disponibilizando fotos do motorista que conduz o veículo [3].

VI. Descrição de Hardware

Para a validação da ideia geral do projeto, uma webcam Logitech C270 [5] foi utilizada. Sua finalidade é a captura de uma imagem das placas dos veículos para que o reconhecimento seja feito. Este modelo foi definido por ser amplamente utilizado em aplicações com Raspberry Pi em diversos outros projetos, dando uma garantia de funcionamento para a aplicação.

O processamento de imagens será feito por meio de uma Raspberry Pi 3B, que enviará um sinal, à partir de um de seus pinos de propósitos gerais de entrada ou saída (GPIO) , a um transistor, que funcionará como uma chave, substituindo o botão físico presente no controle utilizado para abrir o portão da garagem.

O controle utilizado para testes será o PPA A02258 v1.9 [6], que funciona com radiofrequência, acionando a abertura do portão quando seus botões são pressionados. O circuito transistorizado substituirá o botão, funcionando de forma que, quando o GPIO for para nível lógico alto (3,3V), a corrente será liberada e poderá fluir entre as trilhas que antes estavam separadas pelo botão, removendo a necessidade de o botão ser pressionado fisicamente.

LEDs representarão o funcionamento do sistema, sendo um para quando a placa for reconhecida, um para quando não for reconhecida e um para quando a abertura do portão for iniciada, todos controlados pelos GPIOs da placa.

Um sensor PIR [7] funcionará como detecção da presença de um veículo. Se o sensor detectar movimento, uma imagem da webcam será capturada e o processo de detecção será iniciado.

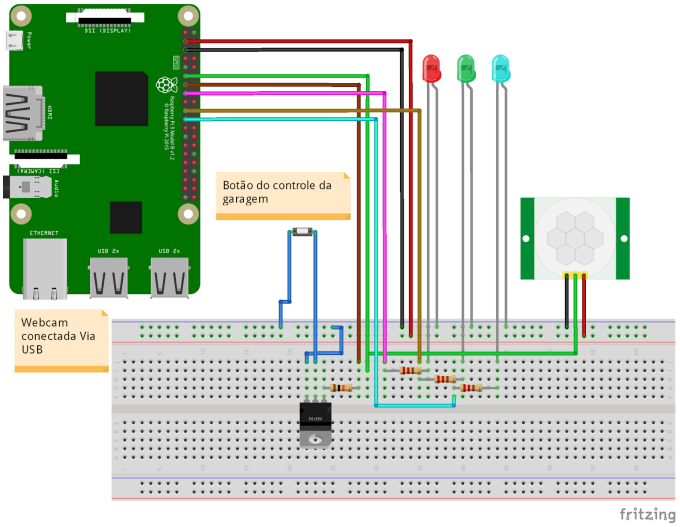

Figura 1. Esquemático de conexão dos componentes.

VII. Descrição de Software

Serão necessárias algumas bibliotecas para o funcionamento do software que controlará o sistema. A biblioteca Open Source Computer Vision (OpenCV) é a base para o processamento de imagens realizado pela biblioteca Open Source Automatic License Plate Recognition (OpenALPR) [8], dedicada à leitura de placas veiculares de diversas regiões do mundo, sendo placas brasileiras também reconhecidas por suas funções.

O reconhecimento do texto da placa é feito à partir de duas bibliotecas principais, a Leptonica [9] e a Tesseract [10], que também são ligadas à OpenALPR.

Utilizando as funções básicas do OpenCV, podemos extrair os frames capturados pela WebCam e transformá-los em imagens no formato PNG com a função imwrite(). A partir da imagem gerada, podemos realizar a detecção da placa com as funções da biblioteca OpenALPR que retorna os valores que estão de acordo com o padrão de placas brasileiro (ABC-1234).

A biblioteca OpenALPR é utilizada por meio de chamadas de sistema, tendo sua saída armazenada em uma string por meio de uma função que retorna o texto de saída de qualquer função do terminal. Por meio de manipulação de strings, é possível separar somente a parte correspondente à placa do carro.

A chamada realizada é a seguinte:

alpr -c br -p @@@#### -n 2 ./WebCamImage.png

que procura por placas brasileiras (-c br) com o padrão de três letras e quatro números (-p @@@####), apresentando dois resultados possíveis (-n 2) encontrados na imagem WebCamImage.png, que é obtida diretamente da WebCam e é salva ao ser detectado algum movimento no sensor de presença infravermelho PIR.

Podemos, assim, comparar o resultado das placas com uma planilha hospedada na nuvem contendo todas as placas permitidas na garagem, para que seja enviado o sinal para o Transistor MOSFET pelo GPIO, utilizando a biblioteca WiringPi [11], e o botão seja pressionado, acionando a abertura da porta.

Outra biblioteca também utilizada é a da Google, a Google Sheet API [12], que é uma API da Google para escrever e ler de planilhas. Essa API foi criada para uma conexão mais rápida e eficiente à ferramentas da Google como *Sheets, Document, Drive,* dentre outras funcionalidades fornecidas pela Google.

Para utilizar essa API é necessário *download* da biblioteca em Python [13] (Não está mais disponível a biblioteca em C++ porque foi depreciada pela Google).

Na utilização da biblioteca é necessário ter credenciais de OAuth 2.0 [14] da Google para que possa acessar em formato

Json. Essa credencial é feita acessando o *google developer* e gerando credenciais para usuário.

Outra funcionalidade é a utilização de código Python nos arquivos de C/C++[16]. Em vez de executar ele por terminal, que é uma tarefa extremamente lenta, é possível a utilização de objetos, funções e variáveis Python em código C++ que faz com que o desempenho desta função fique melhor e também a versatilidade com a mudança de valores de variáveis sem a necessidade de executar o programa Python o tempo todo.

Existe também questões de segurança relacionadas a esse projeto pois fazer chamadas de APIs em nuvem tem um custo, entretanto esse custo é aumentado por essa mesma API só ter suporte para poucas linguagens dentre elas Python a mais simples de se utilizar e que está ao alcance. Levando em conta esses custos teve-se a ideia de armazenar as placas na Raspberry e para isso teria que ter alguma segurança mesmo que mínima, então para que isso ocorra, para isso decidiu-se armazenar o hash todas as placas com SHA256 e para isso utilizou-se a biblioteca CryptoPP [15].

E por fim para facilitar a visualização e a manutenção do projeto foi feito uma estruturação do projeto. Essa estrutura é feita como mostra a **Figura 2** com uma pasta para os binários, uma para documentos como imagens, pdfs, textos, uma para includes de hpp, uma para os objetos e uma para os arquivos fonte. Junto com toda essa estrutura foi também feito um Makefile[17] para facilitar a compilação e a execução do projeto.

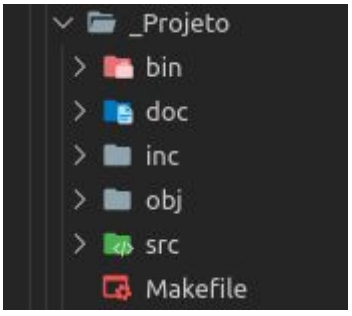

Figura 2.Estrutura de Pastas para o projeto C++..

VIII. Resultados Parciais (Ponto de Controle 2)

Os resultados obtidos, até o dia 30 de Setembro de 2019, demonstram que o projeto poderá ser concluído, necessitando somente de ajustes e aprimoramentos no código para que o reconhecimento de placas seja feito com uma imagem capturada pela webcam.

É possível realizar a captura de imagens ao vivo com a webcam, que são salvas em um arquivo PNG que pode ser lido pela OpenALPR. Esta, por meio de comandos de terminal, consegue reconhecer grande parte das placas veiculares brasileiras, com a necessidade de poucos ajustes e possibilidade de mais treinamentos de reconhecimento para maior precisão.

Para os próximos pontos de controle, tem-se por objetivo a implementação da biblioteca OpenALPR em código C++, sem a necessidade do uso de funções do terminal, bem como a implementação da planilha contendo o banco de placas permitidas, o circuito de controle do acionamento do portão e os LEDs de status do sistema.

IX. Resultados Parciais (Ponto de Controle 3)

Com o avanço no desenvolvimento, a integração dos componentes pode ser realizada. A biblioteca OpenALPR foi integrada ao sistema por chamadas de terminal, com sua saída sendo armazenada pela função exec(), presente no código. O sistema consegue detectar N placas, que, por manipulação de strings, podem ser separadas para comparação com a lista de placas presente no Google Sheets.

A comunicação com o Sheets é feita por meio de uma chamada API com o próprio API do Google Sheets, que retorna o conteúdo que desejamos corretamente, porém, em Python, não em C ou C++. Entretanto não é possível conectar-se a essa API com a internet da UnB , a *UnB Wireless*.

O sensor de presença detecta corretamente se algo cruza seu "campo de visão", enviando um sinal ao sistema para começar a análise da placa e sua detecção.

Devido à dificuldade de configurar a OpenALPR para melhorar sua precisão, a chamada de terminal retorna duas possíveis placas para o veículo, pois, ao realizarmos testes com diversas placas diferentes, verificamos que a placa correta, na maioria dos casos, está entre as duas primeiras placas reconhecidas pela biblioteca.

Para o próximo ponto de controle, será feita a otimização do código com mais funções como pipes e threads, para que o sistema funcione de forma mais ágil e a organização da estrutura do código para que ele se torne mais legível. Também tentaremos resolver o problema de conexão com a *UnB Wireless* para que o nosso sistema conecte-se mesmo com esse problema.

X. Resultados Parciais (Ponto de Controle 4)

Os avanços no projeto possibilitaram a comunicação com a API do Google Sheets [12], que possibilitou um armazenamento das placas permitidas de forma segura fora da Raspberry Pi. A segurança da informação obtida é garantida utilizando-se hashes, que codificam as placas e dificultam o acesso forçado ao sistema.

O sistema de ativação do portão, por meio do controle comum "hackeado", já está funcionando corretamente e será integrado ao sistema assim que o reconhecimento de placas for integrado completamente com a comunicação com a API, pois necessita da flag de liberação do portão para ser ativado. Esta flag só é ativada quando a comparação das placas reconhecidas com as placas da API é realizada com sucesso e, por isso, não pode ser implementado sem esta parte.

A implementação de otimizações de código, organização e facilidade de leitura foram feitas, faltando apenas otimizações de funcionamento do programa, com threads, pipes e outros.

As threads que serão utilizadas já foram definidas, faltando, somente, implementá-las.

Referências

[1] “OpenCV”. Disponivel em: <https://opencv.org/>. Acesso em: 30 de agosto de 2019.

[2] “Conserto de controle remoto de portão eletronico”. Disponivel em: <https://dicasdozebio.com/2013/03/19/tecnica-conserto-de-controle-remoto-de-portao-eletronico>. Acesso em: 30 de agosto de 2019.“

[3] “EVA - Leitor de Placas de veículos”. Disponivel em: <http://www.tecnima.com.br/eva.asp>. Acesso em: 30 de agosto de 2019.

[4] “Controle de acesso LPR leitor de placa”. Disponivel em: <https://www.rocken.com.br/leitor-de-placa/>. Acesso em: 30 de agosto de 2019.

[5] “C270 Videochamada de HD 720p plug and play”. Disponível em: <https://www.logitech.com/pt-br/product/hd-webcam-c270>. Acesso em: 30 de setembro de 2019.

[6] “Controle Remoto Ppa Tok Portão Alarme Cerca Elétrica 433mhz”. Disponível em: <https://produto.mercadolivre.com.br/MLB-805525634-controle-remoto-ppa-tok-porto-alarme-cerca-eletrica-433mhz-_JM>. Acesso em: 30 de setembro de 2019.

[7] “PIR Motion Sensor Tutorial”. Disponível em: <https://www.instructables.com/id/PIR-Motion-Sensor-Tutorial/>. Acesso em: 20 de outubro de 2019.

[8] “OpenALPR”. Disponível em: <http://www.openalpr.com/>. Acesso em: 30 de setembro de 2019

[9] “Leptonica”. Disponível em: <http://www.leptonica.org/>. Acesso em: 30 de setembro de 2019.

[10] “Tesseract OCR”. Disponível em: <https://github.com/tesseract-ocr>. Acesso em: 30 de setembro de 2019.

[11] “WiringPi”. Disponível em: <http://wiringpi.com/>. Acesso em: 30 de setembro de 2019.

[12] “Google Sheet API”. Disponível em: <https://developers.google.com/sheets/api/reference/rest>. Acesso em 30 de setembro de 2019.

[13] “Google Sheet API Tutorial”. Disponível em: <https://developers.google.com/sheets/api/quickstart/python>. Acesso em: 30 de setembro de 2019.

[14] “OAuth 2.0”. Disponível em: <https://oauth.net/2/>. Acesso em:1 de outubro de 2019.

[15] “Biblioteca CryptoPP”. Disponível em: <https://www.cryptopp.com/>. Acessado em: 7 de novembro de 2019.

[16] “Utilizando Python em C++”. Disponível em: <https://docs.python.org/2/extending/embedding.html>. Acessado em: 7 de novembro de 2019.

[17] “Make Documentação”. Discponível em: <https://www.gnu.org/software/make/manual/make.html> . Acessado em: 7 de novembro de 2019.