

Docente: Adriano César Santana

Discente: Daniel Moraes dos Santos

Veículo autônomo utilizando visão computacional

1 - Descrição Projeto

O objetivo desse trabalho é a criação do protótipo de um carro de controle remoto, equipado com uma câmera, capaz de reconhecer placas de trânsito. Esse protótipo deverá fornecer a base para a futura adaptação e melhoria por outros alunos da disciplina, e tem como fim ser usado em projetos educacionais. Atualmente, é feito apenas o controle dos motores com base em eventos gerados pela detecção de objetos através da câmera.

O projeto foi desenvolvido como parte de uma disciplina que ensina técnicas de machine learning e visão computacional, no qual foi escolhido o método clássico de reconhecimento de objetos chamado HAAR-Cascade, que devido a sua simplicidade e eficiência, é ideal para ser embarcado em hardware como um Raspberry Pi.

2 - Desenvolvimento

2.1 - Preparação dos dados

A placa de trânsito 'PARE' foi escolhida para ser o objeto detectável. Para utilizar o HAAR cascade é necessário uma coleção de imagens positivas (o objeto que deseja se identificar) e um conjunto de imagens negativas (imagens que **não** possuem o objeto à ser detectado, e de preferência seja relacionado ao *background* onde o objeto normalmente é encontrado (no contexto de uma placa de trânsito, imagens de rua sem a placa))

2.1.1 - Imagens do objeto

Primeiramente deve-se coletar imagens do objeto a ser identificado, a quantidade inicial de imagens do objeto depende da característica do mesmo, uma placa de trânsito necessita de poucas imagens (foram utilizadas **8 imagens** neste trabalho), porém objetos mais complexos como faces, é necessário um grande número, com diversas características possíveis de fazer parte do objeto a ser detectado (fenótipos, iluminação, adereços etc).



d3f95ab11de8f7749f7d5eba43f5de89.png



placa-pare.png



Screenshot_5.png



TECFIL_IMG_04_02.png

2.1.2 - Recorte e fundo branco

Uma vez de posse das imagens brutas, é necessário recortar a imagem para obter apenas o objeto, e adicionar um fundo branco, passo importante para o próximo estágio de processamento.



pare_1.jpg



pare_2.jpg



pare_7.png



pare_8.png

2.1.3 - Criação de samples positivas

Com o intuito de gerar um grande número de imagens positivas utilizando um pequeno dataset, podemos utilizar as imagens e inserir as mesmas em background diferentes, aplicando transformações de escala e rotação para gerar variedade. Neste trabalho foram utilizadas 100 imagens de background, para as 8 imagens originais, gerando no total 800 samples positivas.



[0001_0093_0053_0279_0279.jpg](#)



0002_0092_0171_0164_0164.jpg



0007_0117_0118_0309_0309.jpg



0008_0371_0236_0100_0100.jpg

2.1.4 - Recorte samples positivas

Finalmente, deve-se recortar as samples criadas anteriormente para isolar apenas o objeto a ser identificado. Essas são as imagens positivas que serão utilizadas no treinamento do filtro.



[1691301010050.jpg](#)



1691301018493.jpg



1691301091654.jpg



1691301097737.jpg

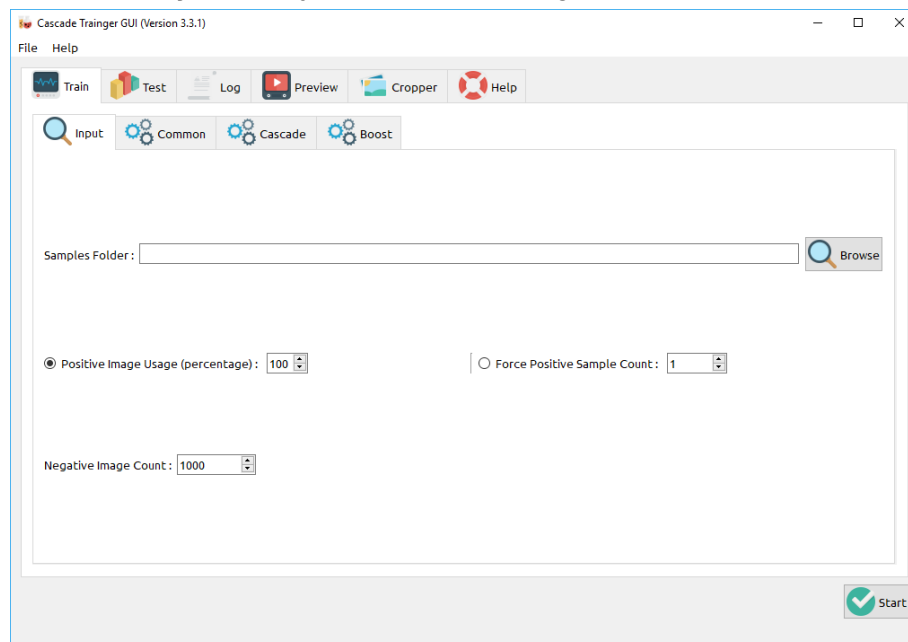
2.2 - Construção do modelo

Para este projeto foi escolhido a detecção de objetos através de filtros HAAR. O treinamento de um filtro Haar envolve criar um classificador para detectar objetos específicos usando características Haar, que são conjuntos retangulares de pixels para identificar variações de intensidade em imagens.

O processo começa com a coleta de imagens positivas e negativas do objeto em questão. As Características de Haar são então calculadas a partir dessas imagens. Um algoritmo de aprendizado de máquina é aplicado para selecionar as características mais relevantes e criar um classificador. O filtro resultante é aprimorado pela combinação de vários classificadores simples, visando a robustez. O classificador é testado em imagens não usadas no treinamento para avaliar seu desempenho e ajustes são feitos conforme necessário. O resultado final é um filtro Haar capaz de detectar padrões semelhantes do objeto em novas imagens.

2.3 - Treinamento do modelo

Para treinar o filtro que detecta a placa de pare, foi utilizado o software Cascade Trainer GUI, é um front-end para as ferramentas de treinamento do OpenCV, e simplifica o processo de treinamento de filtros Haar. O processo consiste em inserir as imagens positivas e negativas do objeto, organizadas em pastas separadas de acordo com a estrutura solicitada pelo software. O treinamento envolve a configuração de parâmetros, como estágios e tamanho do objeto, e a extração automática de características de Haar das imagens. O OpenCV por sua vez aprimora gradualmente o classificador, adicionando classificadores fracos em cada estágio e descartando características menos relevantes. Os resultados são avaliados em um conjunto de validação, e ajustes nos parâmetros podem ser feitos. Ao final, um arquivo XML é gerado para uso na detecção de objetos em novas imagens.



O Cascade Trainer GUI torna o processo de treinamento de filtros Haar mais acessível e automatizado, uma vez que as ferramentas do OpenCV são programas com interface de texto, e devido à ser um método antigo, novas versões do OpenCV não são mais distribuídas com as ferramentas de treinamento.

O processo de treinamento do filtro HAAR é extremamente lento, cada iteração demorou aproximadamente 3h em um processador i7 12700F. Apesar de ser uma característica negativa do método HAAR, uma vez treinado, a detecção utilizando o filtro é extremamente eficiente, sendo possível a implementação em diversos tipos de dispositivos de baixa performance.

2.4 - Ajuste fino do modelo

Ao longo do projeto foram gerados 4 filtros HAAR capazes de detectar a placa 'PARE'. Foi observado que com um número muito baixo de imagens positivas, a detecção é muito frágil, a medida que se aumenta o número de imagens positivas e negativas, a detecção se torna cada vez mais confiável.

Filtro Haar	Imagens Positivas	Imagens Negativas	Desempenho
Versão 1	100	1000	Péssimo
Versão 2	200	2000	Ruim
Versão 3	400	2000	Ruim
Versão 4	800	3500	Bom

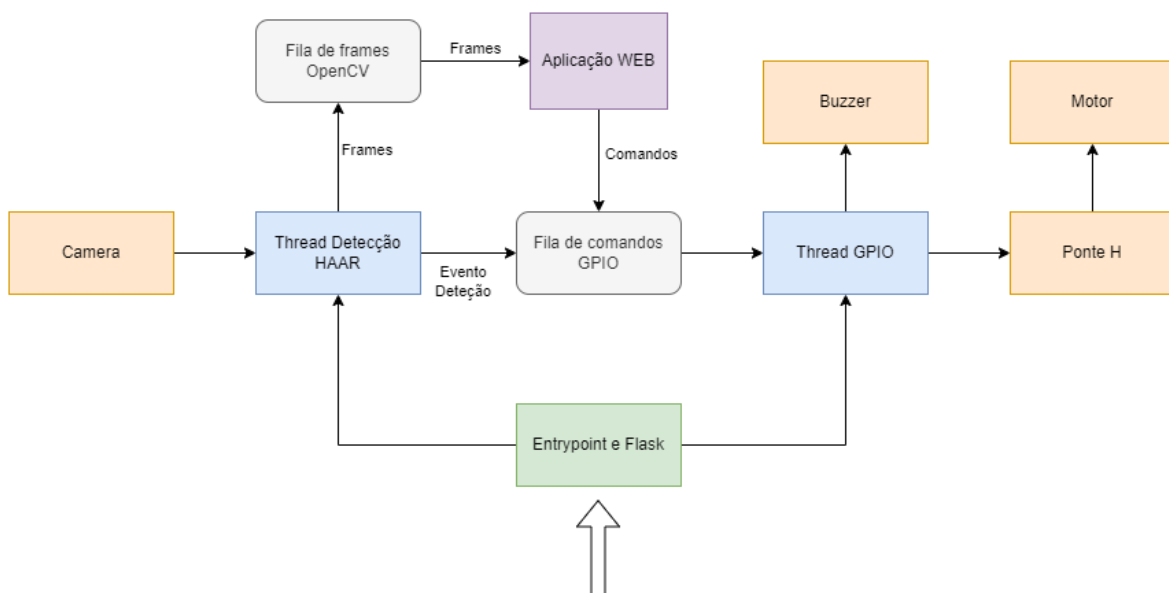
Os testes do filtro foram feitos utilizando a própria webcam do projeto, porém ligada em um computador desktop, em vez do raspberry pi, a fim de facilitar o teste dos filtros. É importante notar que condições diferentes de iluminação, câmera, e o fundo onde se encontra o objeto, irão influenciar o resultado, dessa forma, é importante entender a necessidade de um dataset maior caso o uso do filtro for mais diversificado, porém caso seja utilizado em condições mais controladas, um número relativamente baixo de imagens negativas e positivas é suficiente para gerar um resultado satisfatório.

2.5 - Implantação do modelo

2.5.1 - Software

Foi escolhida a linguagem python para a implantação do modelo, devido à rica biblioteca OpenCV disponível em python, como também essa ser a linguagem mais utilizada para interagir com a GPIO do raspberry pi.

A arquitetura do projeto é baseada em eventos e utiliza o multithreading para poder lidar com as atividades em tempo real necessárias para processar a imagem obtida através da câmera, como também controlar os motores.



Uma vez que com a utilização do filtro HAAR o objeto é identificado no feed, um evento é colocado em uma fila que é passado para a thread de controle, onde lá é possível tomar a decisão com base no que foi identificado. A implementação atual, interrompe o funcionamento dos motores assim que a placa é identificada, e emite um alerta sonoro.

Além disso, foi feito o uso do framework Flask para controlar os motores manualmente, além de poder visualizar o feed de vídeo.

2.5.2 - Hardware

Nessa primeira iteração do protótipo, optou-se por montar todos os componentes do carro em uma bancada, a fim de validar o funcionamento do mesmo antes de se desenvolver uma estrutura apropriada para que o mesmo possa funcionar em demonstrações.

Os materiais utilizados foram:

- 1x Webcam Logitech C970
- 1x Raspberry PI 3B+
- 2x Motores DC 6V de kits educacionais para robótica
- 1x Ponte H L298N
- 1x Conversor DC-DC LM2596
- 1x Buzzer Piezo Elétrico
- 1x Fonte alimentação 12v
- Suportes diversos impressos em impressora 3D FDM
- Conectores Wago
- Jumpers diversos




A ponte H é responsável por fazer a interface entre os motores e a GPIO do raspberry PI de maneira segura, e o conversor DC-DC reduz a tensão de entrada 12V para uma tensão de 5.2V, que alimenta os motores e o raspberry PI.

3 - Conclusão

O método de detecção HAAR, apesar de ser um método hoje considerado clássico, e por muitas métricas é considerado inferior a métodos que utilizam redes neurais profundas, ainda é utilizado amplamente por ser uma forma leve de se identificar objetos em uma imagem. Sua aplicação foi bastante eficaz no projeto proposto, uma vez que o computador embarcado empregado no projeto possui capacidade de processamento limitada.

Esse projeto se mostra também um recurso valioso para o ensino de conceitos de robótica, aprendizado de máquina, visão computacional, eletrônica e manufatura, uma vez que devido a sua simplicidade pode ser evoluído por alunos do curso de engenharia, e é um objeto lúdico e acessível para exposições para crianças e adolescentes.

4 - Fontes

1. Viola and Jones, "[Rapid object detection using a boosted cascade of simple features](#)"
2. https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html
3. https://docs.opencv.org/3.4/dc/d88/tutorial_traincascade.html
4. <https://amin-ahmadi.com/cascade-trainer-gui/>
5. <https://coding-robin.de/2013/07/22/train-your-own-opencv-haar-classifier.html>
6.  Detecting Faces (Viola Jones Algorithm) - Computerphile
7. <https://mememememememe.me/post/training-haar-cascades/>

Anexos

- Código fonte e datasets: <https://github.com/danielbibit/HAAR-Remote-Control>