Deteção de sinais de trânsito

Carlos Frias[1], Nuno Marques[2]

Mestrado Integrado em Engenharia Informática e Computação

Faculdade de Engenharia da Universidade do Porto

[1]carlos.frias@fe.up.pt

[2]nuno.marques@fe.up.pt

Abstract— **This report aims to document the implementation of a computer vision application for the automatic detection of limit speed traffic signs on the road. The application was developed in C++ using OpenCV library for computer vision.**

Keywords **— Computer vision; Sign detection; OpenCV; Image Processing.**

I. Introdução

Este relatório pretende documentar a implementação de uma aplicação de visão por computador para deteção de sinais de trânsito de limite de velocidade em imagens obtidas na estrada, em vídeos e ainda utilizando a webcam do computador. A aplicação foi desenvolvida em C++, recorrendo à biblioteca OpenCV. O programa permite detetar o sinal de limite de velocidade em imagens estáticas, em vídeos previamente gravados e na captura de frames pela webcam. O programa permite também o reconhecimento do valor do limite de velocidade máxima. A figura 1 ilustra o resultado pretendido com a aplicação, primeiro isolar o sinal de trânsito e em seguida detetar o valor do limite de velocidade.

Figura 1:Imagem inicial, imagem segmentada pela cor vermelha, sinal detetado.

II. Especificação

O funcionamento do programa desenvolvido é o seguinte:

É apresentada ao utilizador um menu de opções como é apresentado na figura 2:

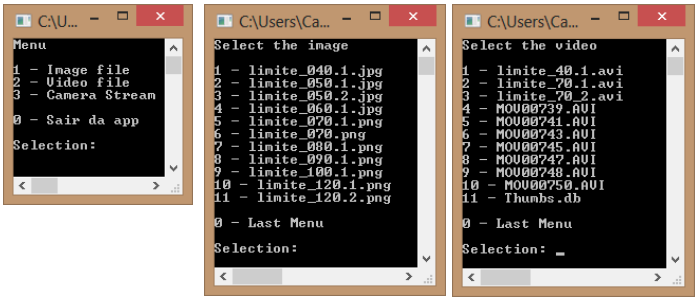

Figura 2: Menus de seleção de tipo de input, imagem, vídeo e webcam.

Neste menu o utilizador pode selecionar o tipo de dados do input de entre ficheiros de imagem, de vídeos gravados em disco e ainda a captação vídeo stream pela webcam do computador.

No desenvolvimento deste trabalho teve-se em conta os seguintes pressupostos simplificativos: todos os sinais estão colocados em posição vertical e frontal, isto é, o plano da imagem e o plano que contém o sinal são mais ou menos paralelos; para cada tipo de sinal de limite de velocidade, existe um ficheiro template contendo uma imagem dos valores numéricos de cada um dos sinais de limite de velocidade permitida na legislação portuguesa.

III. Detalhes da Implementação

A aplicação implementada percorre as seguintes etapas:

1. Escolha por parte do utilizador do tipo de dados de input a utilizar;
2. Seleção do ficheiro de imagem/vídeo a analisar;
3. No caso da escolha de vídeo/webcam há ainda a captação de um frame para analisar a presença de sinais de limite de velocidade;
4. Segmentação da imagem selecionada pela cor vermelha;
5. Deteção dos círculos presentes na imagem anterior utilizando a função “HoughCircles” do OpenCV;
6. Fazer o recorde da imagem interior ao círculo (sinal) detetado para obter os números;
7. Deteção das posições mínimas e máximas dos caracteres detetados na imagem anterior e do tamanho das letras;
8. Escalamento dos templates para o tamanho determinado anteriormente e deteção utilizando a

função “matchTemplate” do OpenCV usando o algoritmo de cross correlation normalizada;

9. Identificação do template que mais se aproxima da imagem obtida e consequente identificação da velocidade máxima.

Na primeira etapa é apresentada uma interface em modo de texto, como se pode observar na figura 2, para a seleção da imagem/vídeo ou camara stream obtida pela webcam. No caso do vídeo e da webcam, a frame a analisar é obtida quando o utilizador prime a tecla ‘c’. Após selecionar a imagem, o passo que se segue é a segmentação pela cor vermelha, uma vez que o sinal tem sempre um rebordo vermelho. Desta forma “elimina-se” o resto da imagem, ficando apenas os sinais de trânsito e/ou outros elementos vermelhos da imagem. Esta segmentação é realizada com a função: *segment_color(Mat &srcImageNameEqualized, Mat &srcImageNameNotEqualized)* que retorna nos seus argumentos a imagem segmentada equalizada e não equalizada. Na figura 3 pode observar-se o resultado da aplicação desta função:

Figura 3: imagem original e imagem obtida após segmentação pela cor.

À imagem segmentada é, em seguida, aplicado o algoritmo de deteção de circulos de Hough, através da chamada à função *hough_detection(Mat &image, vector<Vec3f> &circles)* que devolve no segundo argumento um vetor de objetos contendo os centros e raios dos círculos detetados. Na figura 4 pode observar-se o resultado obtido:

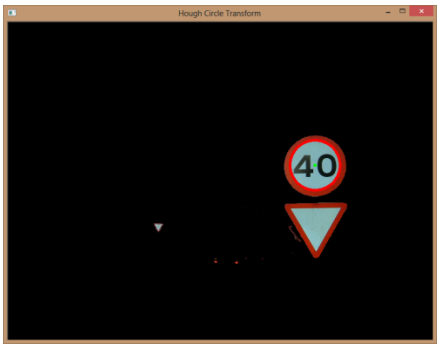

Figura 4: Imagem segmentada contendo os círculos detetados.

Após a deteção dos círculos, a cada um deles, desde que o seu raio seja superior a um determinado valor limite, é recortado o seu interior contendo as letras. Esse recorte é realizado da seguinte forma: a altura das letras é igual ao raio do círculo interior, portanto, com uma simples operação matemática pode-se concluir que a altura do recorte é igual ao raio do círculo e a largura do recorte é $2\sqrt{3}$ vezes o raio do círculo. O recorte da imagem com estas dimensões aproxima relativamente bem o espaço ocupado pelos números, no entanto, para garantir que o recorte pelos números é perfeito, é aplicada a função *Size_number get_size_numbers(Mat &img)* que efetua o seguinte procedimento: realiza uma segmentação pela cor preta, deixando os pixéis com cor branca nos números e com cor preta nos restantes. Em seguida, são “varridos” todos os pixéis da imagem, procurando o menor e maior valores na horizontal e na vertical onde surgem pixéis brancos, com uma certa margem de erro, encontrando assim as dimensões da região exata contendo números. A figura 5 mostra o recorte do interior do sinal e o refinamento posterior para ajustar perfeitamente aos números.

Figura 5: Recorte do interior do sinal, binarização da imagem e ajuste para o exato tamanho das letras.

Após determinação da altura e comprimento da região onde os números estão inscritos, resta fazer o escalamento dos templates existentes em disco para essas medidas e compará-los com a imagem obtida. Essa comparação é realizada com a chamada à função “*matchTemplate*” do OpenCV utilizando o algoritmo de comparação “*normalized cross correlation*” que nos pareceu ser aquele que apresenta melhores resultados. Após comparados todos os templates com a imagem, é determinado aquele que obtém um maior coeficiente de correlação.

Se existirem templates cujo coeficiente de correlação exceda 0.80, então destes é escolhido o template com maior coeficiente de correlação. Caso não se encontre um coeficiente de correçlação superior a 0.80, então considera-se que não foi possível detetar o valor da velocidade máxima.

Quando a velocidade limite é detetada, esse valor é apresentado em modo de texto e também é apresentado uma imagem do sinal correspondente, como se pode ver na figura 5.

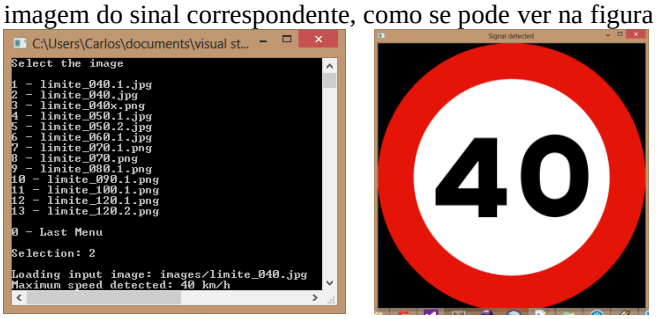

Figura 5: Ecrãs finais, modo de texto e imagem do sinal detetado.

IV. Conclusões e melhoramentos

A realização deste trabalho, permitiu elaborar um programa estável, para deteção de sinais de limite de velocidade, e ainda o aprofundamento de conhecimentos de processamento e análise de imagens e visão por computador, utilizando a biblioteca OpenCV. Algumas melhorias para este projeto

passariam pela possibilidade de deteção dos sinais por outros métodos, não apenas o template matching, como por exemplo, o algoritmo SURF ou a deteção por subtração de imagens. E ainda, a implementação de uma interface gráfica para tornar a aplicação mais apelativa.

Em suma, pensamos que o trabalho realizado apresenta qualidade e que atingiu claramente os objetivos mínimos que se propunha.

BIBLIOGRAFIA E REFERÊNCIAS

[1] Professores Jorge Alves da Silva e Luís Filipe Pinto de Almeida Teixeira, Página da disciplina de Visão por Computador, http://moodle.up.pt/course/view.php?id=707

[2] Documentação do OpenCV, http://docs.opencv.org/

[3] Template matching tutorial using OpenCV matchTemplate function, http://docs.opencv.org/doc/tutorials/imgproc/histograms/template_matching/template_matching.html

[4] Hough circle transform tutorial using OpenCV, http://docs.opencv.org/doc/tutorials/imgproc/imgtrans/hough_circle/hough_circle.html