

# Processamento de Imagens

## Rastreio de Imagem

Igor Froehner<sup>1</sup>, Pedro Serpa<sup>1</sup>, Rafael Granza de Mello<sup>1</sup>,

<sup>1</sup>Universidade Estadual de Santa Catarina (UDESC)

R. Paulo Malschitzki, 200 - Zona Industrial Norte, Joinville - SC, 89219-710

[igoor.sf14, pedroserpah, rafaelgranzademello]@gmail.com

### 1. Introdução

Durante os últimos anos, a Formula 1 vem sendo uma modalidade de automobilismo muito inovadora, não só quando a tecnologia dos carros e equipes, mas também quanto a tecnologia envolvida nas transmissões de TV. No ano de 2020 as transmissões receberam uma nova funcionalidade que impressionou muitos fãs: em alguns momentos da corrida, quando a câmera estava no ponto de vista do piloto, um efeito visual mostra o carro da frente ressaltado por uma marcação na pista e uma informação em cima do carro – que pode ser o nome do piloto, a velocidade, etc – isso pode ser visto na Figura 1. Tal funcionalidade envolve conceitos interessantes de processamento de imagem, entre eles o objetivo desse trabalho: o **rastreio**.

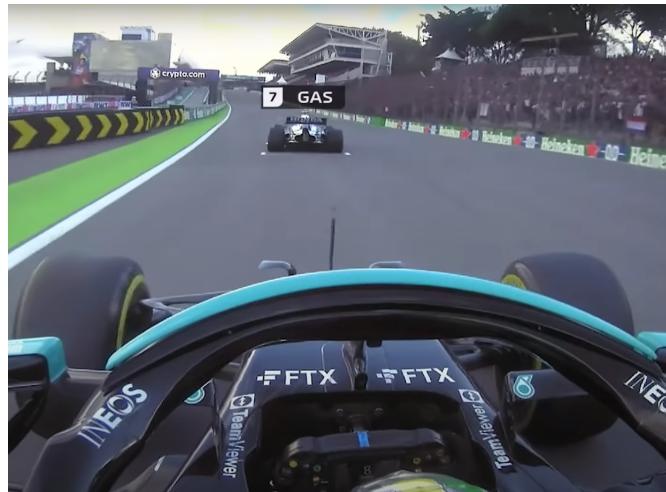


Figura 1. Exemplo F1

Sendo assim, este trabalho tem como objetivo realizar rastreio – que consiste de evidenciar o rastro – de um carro de formula 1 no ambiente de uma pista de corrida usando um método de *template matching*. Para tal, será necessário fazer um comparativo entre os métodos disponíveis na biblioteca OpenCV, e verificar qual melhor se aplicou ao cenário em questão. Após esse comparativo, o método deve ser aplicado, tendo como resultado um vídeo com o rastreio do objeto na cena.

#### 1.1. Introdução Conceitual

Como dito, rastreio é uma técnica de processamento de imagens que visa evidenciar o rastro de um objeto numa cena, dado uma sequência de imagens. Isso pode ser feito

usando diferentes métodos, tanto complexos quanto mais simples. Os métodos estudados nesse trabalho aplicam *template matching*, que consiste de pegar uma imagem inicial do objeto que se deseja rastrear e buscar equivalências nas imagens onde o objeto aparece. Tendo como resultado, depois de aplicado o método, uma região onde provavelmente o objeto deve estar [1].

Tendo em mente a abordagem principal do trabalho, resta o estudo dos métodos a serem testados. O método **CCOEFF** utiliza como seletor o coeficiente de correlação, isto é, nível da correlação é dado conforme o movimento das variáveis de cor e intensidade. Já o método **CCORR** leva em consideração a correlação cruzada. A correlação cruzada mede a similaridade entre duas ou mais séries. Por fim, também é testado o método **SQDIFF**, o qual mede a semelhança entre quadrados das variáveis, levando em conta cor e intensidade [3]. E para cada um desses métodos, nesse trabalho, foram testadas suas versões normalizadas.

## 2. Experimentos

O trabalho foi feito utilizando a linguagem de programação Python (3.8) e a biblioteca OpenCV (4.5.5.62), executado em uma plataforma Ubuntu Linux. Como base de dados para esta etapa foram extraídos 250 frames de uma parte de 10 segundos de um vídeo do youtube, que mostra os carros percorrerem a icônica curva "Eau Rouge" no circuito de Spa - Belgica [2]. O vídeo usado pode ser encontrado nesse link: <https://www.youtube.com/watch?v=hpvuu5MfaSk>. Abaixo é possível ver alguns exemplos de imagens as quais os métodos foram aplicados.

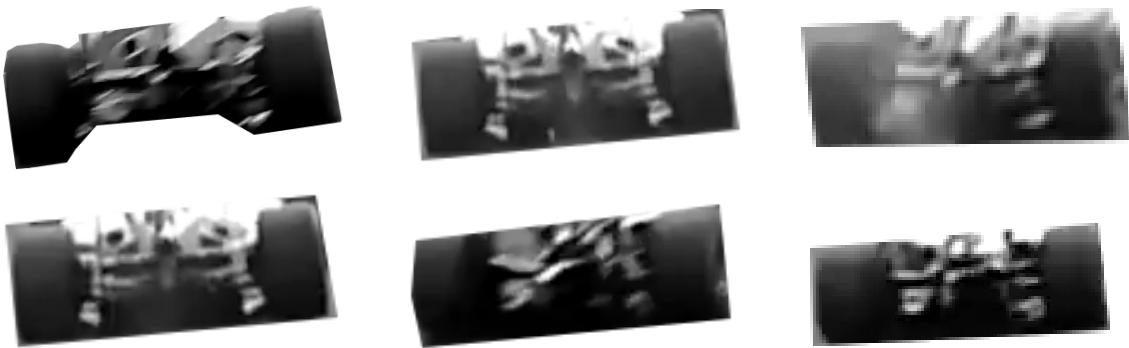


**Figura 2. Exemplos de Imagens**

É importante perceber que algumas características do vídeo selecionado não facilitam o uso da técnica de *template matching*, a câmera não é estática, o carro pode ficar em posições diferentes no decorrer da passagem e a distância do carro pode variar bastante. Tendo isso em mente, talvez algumas técnicas adicionais seriam necessárias para ter um resultado mais interessante.

Como a dificuldade do vídeo escolhido pode ser maior que os métodos de *template matching* disponíveis, foram usados mais que um template, pra ser mais exato foram usados 6 templates, que podem ser vistos na figura 3. Então, cada um desses templates foram aplicados nas imagens, usando os métodos escolhidos, e foi extraída uma confiança de que houve o acerto – no caso o melhor valor retornado do matching. Então, com esses resultados em mãos o melhor matching é aplicado na foto para que seja possível observar onde o objeto está.

Para isso, comparou-se os seguintes métodos de template matching disponíveis na biblioteca OpenCV(cv2):



**Figura 3. Exemplos de Imagens**

- cv.TM\_CCOEFF\_NORMED
- cv.TM\_CCORR\_NORMED
- cv.TM\_SQDIFF\_NORMED

Para analisar a precisão de cada um dos métodos de template matching, definiu-se uma limiar de 70% de acurácia e então calculou-se a Taxa de Verdadeiro Positivo (TPR) para cada uma dos métodos. Por último, foi usada uma tabela de confusão para clarificar os resultados e decidir qual é o melhor método para o problema proposto por este trabalho.

## 2.1. Análise da precisão de cada teste

Para análise de precisão foi utilizado uma métrica simples baseada em **acertos** e **erros**. Nessa análise não existe meio termo, cada *frame* será considerado de acordo como foram avaliados pelos algoritmos. Eles serão considerados errados se indicarem um local onde não há carro ou se não o encontrarem. Serão considerados corretos todos os outros casos.

Método	Acertos		Erros	
	Absoluto	%	Absoluto	%
cv.TM_CCOEFF_NORMED	10	59	7	41
cv.TM_CCORR_NORMED	8	47	9	53
cv.TM_SQDIFF_NORMED	3	18	14	82

**Tabela 1. Erros e acertos dos métodos utilizados**

Concluiu-se, portanto, com base nos resultados da tabela 1 que o método a ser utilizado seria cv.TM\_CCOEFF\_NORMED. Como esperado, a precisão não foi muito interessante dado a dificuldade das imagens usadas nos testes. Mas é possível analisar que em um vídeo tendo 60% de acerto é possível ter um resultado interessante.

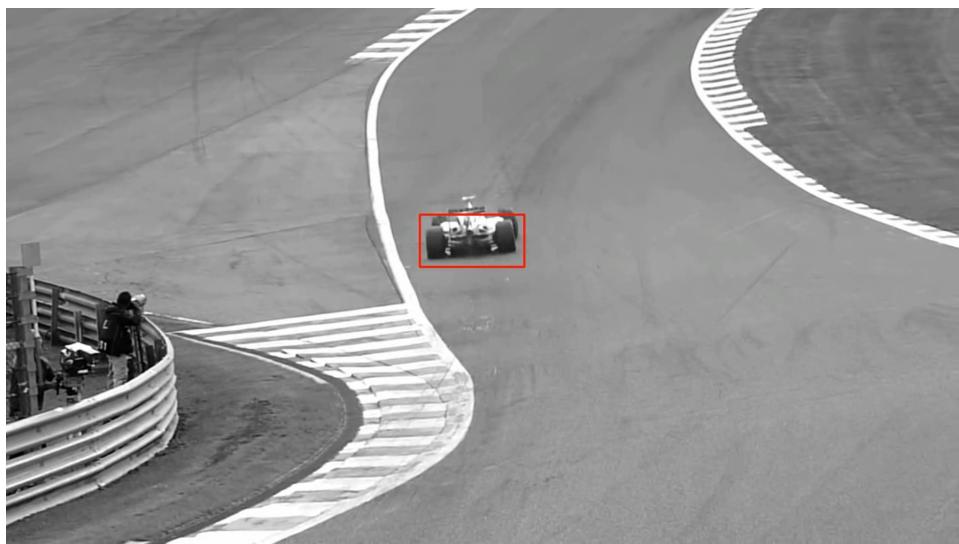
## 3. Resultados

A partir das informações obtidas na seção Experimentos, decidiu-se que seria utilizado o método cv.TM\_CCOEFF\_NORMED prosseguir com o estudo. Tendo isso em mãos, criou-se então um vídeo contendo a predição do trajeto de carros de Formula 1 percorrendo a curva Eau Rouge. Agora pode-se obter resultados da predição e acompanhar o

template matching aplicado em um vídeo verdadeiro. Para ter uma graduação da dificuldade o vídeo foi dividido em 3 partes: uma mais "fácil", na qual o cinegrafista dá um zoom na pista; uma "difícil", na qual o plano está mais aberto e o fundo fica mais detalhado; e uma versão completa, na qual quase todo o vídeo foi submetido ao método. Esses vídeos podem ser vistos respectivamente nesses links:

- [Fácil](#)
- [Difícil](#)
- [Completo](#)

Na figura 5 é possível ver um exemplo do resultado atingido, onde a resposta dada foi correta.



**Figura 4. Resultado obtido a partir da aplicação do cv.TM\_CCOEFF\_NORMED em um vídeo com templates**

Como esperado, também houveram casos de matching errados no vídeo gerado, mostrando que a acurácia do método não é correta. Principalmente em um caso onde o plano de fundo é um pouco mais complexo, com bastante diferença na intensidade das cores. Também é um problema a diferença de escala que o carro toma durante a passagem, ele começa grande no começo e vai diminuindo no decorrer do vídeo. Isso poderia ter sido mitigado pelo uso de diferentes templates, um com o carro maior e outro com o carro menor. Porém quando, muitos templates, e templates menores, foram usados, a acurácia dos métodos foi piorada. Então foi usado uma quantidade menor de imagens no set de templates.



**Figura 5. Erro na detecção do carro**

#### 4. Conclusão

O resultado foi interessante e deu para validar o trabalho, nos testes o método que teve melhor desempenho foi o CCOEFF, em sua versão normalizada. No cenário mais "fácil" o resultado foi relativamente bom, já nos outros cenários ficou mais complicado para que os métodos usados acertassem onde o carro se encontrava, pois como foi citado, quando o carro passa, ele fica em posições diferentes e sua escala diminui bastante no decorrer, ficando mais difícil seu reconhecimento. A técnica de usar mais que um template para tentar atingir um melhor matching se mostrou relevante, e trouxe um melhor resultado, porém templates muito pequenos deterioraram os resultados. Como uma validação o resultado foi interessante, mas para atingir um resultado melhor seria necessário usar mais técnicas além do template matching.

#### Referências

- [1] BRUNELLI, R. *Template matching techniques in computer vision: theory and practice*. John Wiley & Sons, 2009.
- [2] FORMULA1. Belgian grand prix - f1 race - circuit de spa-francorchamps: Formula 1®, 2020.
- [3] KAEHLER, A., AND BRADSKI, G. Learning opencv 3: computer vision in c++ with the opencv library, 2016.