# Classificador Cascade para Detectar Bola em Competições de Robótica

Gabriel Martins de Miranda e Fernanda Amaral Melo

***Index Terms*—classificador cascade, rastreamento**

## I. Introdução

Um grande problema em competições de futebol é rastrear a bola. O padrão branco e preto é facilmente confundido com as outras entidade do campo, como gol e linhas. Através de casamento de padrões e aprendizado de máquina, baseado na técnica de boosting, é apresentado um método robusto de detecção de bola capaz de ser processado num fluxo de vídeo em tempo real.

## II. Fundamentação teórica

O classificador cascade é um método de aprendizado de máquina baseado em features, em que atualmente classificam-se em haar ou lbp, e em estágios, o que define seu nome de cascata. Sua detecção ocorre de forma binária, ou seja, ou o objeto de interesse se encontra na imagem, ou não. É treinado a partir de um grande set de imagens positivas ou seja, do objeto, e negativas, qualquer imagem em que o objeto definitivamente não esteja presente.

O objetivo do classificador é gerar um arquivo capaz de definir um conjunto de classificadores fortes em cada estágio do cascade, sendo os classificadores fortes a soma ponderada das melhores features do estágio, sendo cada uma dessas features um classificador fraco.

Para sumarizar o funcionamento do algoritmo, é feita uma separação em quatro tópicos, features haar, imagens integrais, adaboosting e cascading.

- Haar features
  As features utilizadas são definidas como kernels capazes de detectar características. Quando convolucionados com uma imagem, recebem um valor único definido em função da feature utilizada e da posição em que ela se encontra. Para cada feature haar, seu valor é dado pelo produto interno entre os pixels da feature, sendo so brancos definidos com o valor 1 e os pretos com o valor -1, e os pixels da imagem em nível de cinza. Em Figure 1 é possível ver a configuração de possíveis features haar utilizadas.

Em Figure 2 pode-se ver dois resultados de alto valor para duas features haar em determinada posição na imagem. Cada uma delas representa um classificador fraco.

Ainda assim, numa pequena janela 24x24 pixels podem existir mais de 160 mil features haar. Para facilitar o processo de obtenção dos classificadores fracos dentro todas as outras, uma abordagem de imagens integraise é usada.

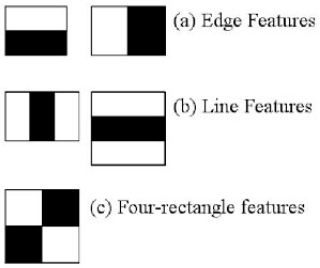


Fig. 1. Algumas features utilizadas no haar, com dois, três e quatro retângulos.

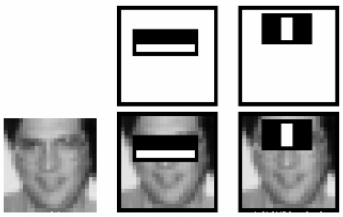

Fig. 2. Algumas features de alto valor em convolução com uma face.

- Imagens Integrais
  Cada pixel na imagem integral é a soma acima e à esquerda na imagem original. Este tipo de vizualização da imagem permite rápido cálculo dos valores das features haar quando estão na fase de convolução do treinamento. Na figura 3 pode-se ver um exemplo de imagem integral para uma entrada 3x3 contendo apenas pixels de valor unitário.

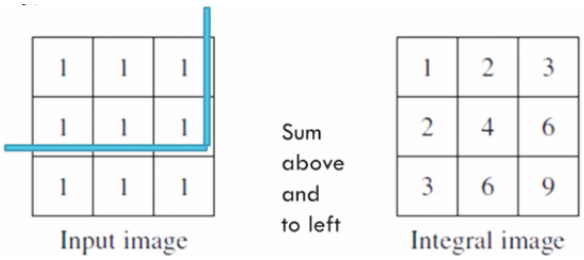


Fig. 3. Imagem integral para uma três por três unitária.

- Adaboost
  Uma técnica muito utilizada em aprendizado de máquina é o boosting, sendo comumente utilizada em aprendizado supervisionado e na definição de classificadores fortes a partir dos fracos. O Adaboost, ou adaptative boosting, é uma variação desta técnica. Através dele, é possível encontrar as melhores features haar dentre todas as outras

cento e sessenta mil features. A Figura 4 mostra a seleção das melhores features e a construção de um classificador forte a partir delas.

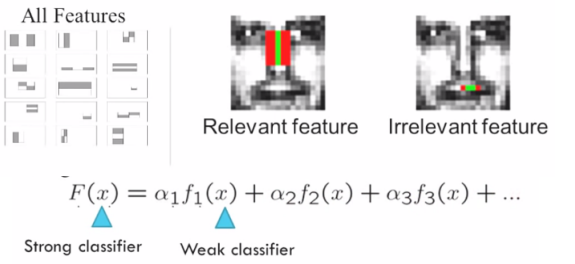


Fig. 4. Seleção das melhores features e construção do classificador.

- Cascading
  Para realizar uma computação dos classificadores de forma mais rápida e eficiente, um modelo de cascata é utilizado. Neste modelo, apenas features mais simples são utilizadas nos estágios iniciais. Com isto, padrões com chance baixa de serem classificadores já são descartados com baixo custo computacional. Aos classificadores simples que passaram, padrões mais complexos são usados a partir destes nos estágios mais avançados. Com isto, são usadas features complexas apenas em áreas na imagem com maior chance de serem o objeto de interesse. Na Figura 5 pode-se ver um fluxograma da etapa de cascading. As features que possuem maior chance de serem classificadores passam para o estágio do algoritmo seguinte.

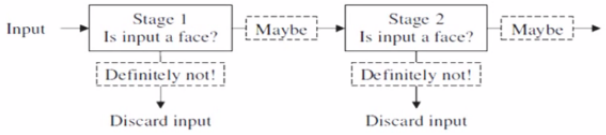


Fig. 5. Etapa de cascading.

Pode-se ver um sumário do algoritmo na Figura 6. Como entrada, temos um grande set de imagens positivas e negativas. O algoritmo é treinado com Adaboost e são selecionadas as melhores features, com limiares e pesos bem definidos, formando classificadores fortes. Após a definição das features e de suas posições na subjanela, os cálculos são usados em imagens de teste para detectar o objeto.

## III. Implementação teórica

Diante do método, o principal objetivo é auxiliar um robô de baixo custo a rastrear uma bola de futebol oficial em competições. Para treinamento do método, foi utilizada a biblioteca OpenCV em sua versão 2.4.12. Com a função opencv_createsamples, temos como entrada uma listagem de todas as imagens positivas utilizadas, no caso 1100, sendo que para cada uma é definida uma região de interesse, ou

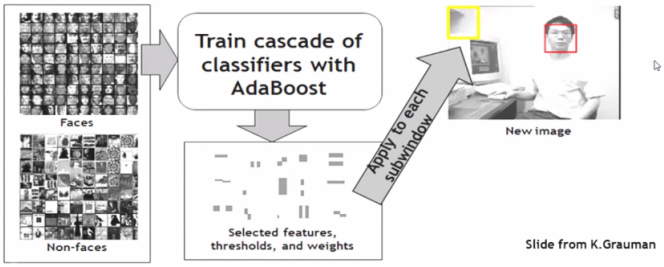


Fig. 6. Sumário do método.

seja, a região onde o objeto de interesse se encontra. Para isto, foi usada a bola de futebol nas mais diversas situações, seja parada, se movendo, com baixa e alta luminosidade e nas mais variadas distâncias. A única restrição imposta pelo método é que as images tenham o mesmo aspect ratio. Como negativas, foram usadas 1700 images, sendo que para estas não há restrição. Com opencv_traincascade é executado o treinamento em si. Para executar o treinamento de forma mais eficiente, habilitou-se o modo de utilização do opencv com multi-core, e usando um computador com processador i7 de primeira geraçao e 4GB de RAM conseguimos executar o treino em até seis horas, tempo bem restrito se comparado à execução comum, que demora em média uma semana. Ao fim do processo, é gerado um arquivo xml especificando todos os classificadores e também as diversas opções utilizadas. Dentre estas opções, usamos tipo de estágio BOOST, tipo de feature HAAR, taxa mínima de acerto em torno de 9,95%. Máxima taxa de alarme false em 0.4 e quinze estágios. Os últimos parâmetros são para o Adaboost. A Figura 7 mostra os robôs utilizados para a competição em cuja bola foi treinada.

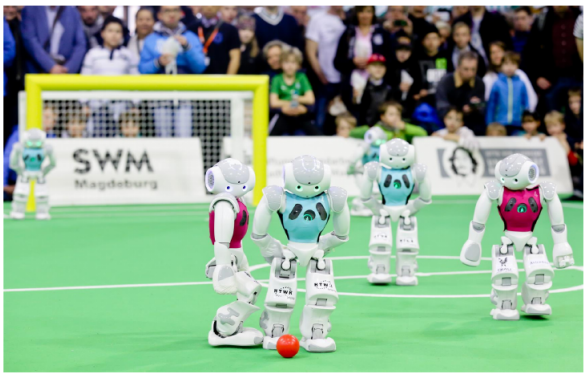


Fig. 7. Robôs NAO na robocup 2015. Antigamente a bola era vermelha.

## IV. Análise

Para validação do código, fez-se uma métrica de falsos positivos e falsos negativos. Para o caso de falsos positivos, o algoritmo encontrou uma bola onde na verdade não havia bola na imagem. Já para os falsos negativos, o método nada achou, porém havia uma bola. A Tabela 1 a seguir exemplifica os resultados para um total de 200 imagens usadas.

| Falso positivo | Falso negativo | Acerto(%) |
|---|---|---|
| 1 | 26 | 86,5 |

TABLE I
RELAÇÃO ENTRE FALSOS POSITIVOS E FALSOS NEGATIVOS.

## V. CONCLUSÕES

Através do uso do algoritmo a detecção de bolas melhorou de forma significativa, porém ainda é um problema o fato de o processamento dentro do robô ser muito lento, já que os resultados foram testados em nossas máquinas com processadores muito mais potentes que os usados pelo nao, que usa um Intel Atom de apenas um núcleo. Sendo assim, diversas otimizações são necessárias para tornar o código mais rápido e robusto.

[1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, 2001, pp. I-511-I-518 vol.1.