João Filipe Modesto Fornelos

Nº41617

Extração Semi-Automática de Informação Desportiva a partir de Vídeo

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática

Orientador: Teresa Romão, Profª. Auxiliar,
Universidade Nova de Lisboa

Coorientador: Nuno Correia, Prof. Catedrático,
Universidade Nova de Lisboa

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Março, 2018

Automatic Generation of Sports Videos Highlights

À minha família.Obrigado.

Agradecimentos

Primeiramente gostaria de agradecer aos meus orientadores neste projeto, Professora Doutora Teresa Romão, Professor Doutor Nuno Correia e Doutor Pedro Centieiro, por terem estado sempre disponíveis para me auxiliar em todas as questões que foram surgindo ao longo do desenvolvimento do projeto, desde o seu início até ao momento da entrega final.

Queria também agradecer à Faculdade de Ciências e Tecnologia, Universidade de Nova de Lisboa, por me ter dado a possibilidade de estudar nesta instituição e de poder concluir os meus estudos, num ambiente muito amigável e positivo.

Finalmente agradecer à minha família que sempre me apoiou em tudo, ao longo deste percurso, que chega agora ao fim. Obrigado.

Resumo

Os eventos desportivos nos dias de hoje, são bastante populares, sendo visualizados pelo mundo inteiro nos mais variados dispositivos. Porém se o utilizador não tiver possibilidade de assistir ao evento em direto, e tendo em conta a velocidade com que a informação se difunde nos dias de hoje, a probabilidade de o utilizador saber como correu o evento (resultado, lances polémicos) antes de ter a possibilidade de o visualizar em diferido é muito grande. É nestas situações que os *highlights* possuem uma grande importância, pois a tendência natural do utilizador não será visualizar o evento em diferido. O que os vários utilizadores pretendem é sim visualizar as partes do evento com maior relevância.

Com este projeto pretende-se gerar, de forma automática, uma base de dados associada a uma transmissão televisiva de um evento desportivo, mais especificamente de um jogo de futebol, recorrendo a várias características que se possam extrair através do próprio vídeo. Pretende-se combinar o processamento dos vários elementos visuais, de modo a produzir ficheiros XML com a informação associada a cada *frame* e momento do jogo. Para além do resumo, há a intenção de dar ao utilizador a possibilidade de executar pesquisas específicas sobre o jogo, como todos os lances que ocorreram numa das balizas, detetar jogadores, detetar certos elementos e lances do jogo entre outros. Como existem imensos desportos o foco principal deste projeto, será o futebol, porém pretende-se alargar os algoritmos e métodos a desenvolver, posteriormente a outros eventos. Os resultados obtidos foram bastante favoráveis, com percentagens de precisão que variam entre os 70% e 88%.

Palavras-chave: Resumo de vídeo, Processamento imagem, Pesquisa em vídeo

Abstract

Sports events these days are quite popular, being viewed all over the world on the most varied devices. However, if the user is not able to watch the live event and considering the speed with which the information spreads these days, the probability of the user knowing what happened at the event (result, controversial bids) before he has the possibility to view it in deferred mode is very big. It is in these situations that the highlights are very important, because the natural tendency of the user will not be to visualize the event in deferred. What the various users want is to see the parts of the event with greater relevance.

With this project, we intend to create a database that represents every moment of the game, in an automatic way, using several possible features to extract through video. It is intended to combine the processing of emotions, sound, text, image and video to produce XML files with the information associated with each frame. In addition to the summary, it is intended to give the user the ability to perform specific searches on the event, such as all the moments that occurred on one of the goal-mouths, detect players, detect certain elements and game moments, among other elements. As there are numerous sports the focus of this project, will be football, however it is intended to extend the algorithms and methods, later to other events. The results obtained were very favorable, with percentages of accuracy varying between 70% and 88%.

Keywords: Video Highlights, Image Processing, Video Search

Conteúdo

Índice de Figuras

Índice de Tabelas

1

Introdução

Este capítulo contém uma pequena descrição do contexto em que se enquadra esta dissertação e, posteriormente explica-se as motivações do projeto e desafios propostos a abordar. Finalmente define-se o problema da forma mais pragmática possível de forma a facilitar a compreensão do trabalho desenvolvido.

1.1 Motivação

Nos dias de hoje se uma pessoa pretende ver um evento desportivo, caso não o veja em direto, muito raramente pretenderá visualizar o evento completo em diferido, pois tendo em conta a velocidade com que a informação circula pelos vários meios de comunicação (televisão, rádio, redes sociais), a probabilidade de já se saber o desfecho do encontro é muito elevada. Por outro lado, mesmo que a pessoa tenha assistido ao evento em direto, é normal gostar de rever os lances mais importantes que ocorreram ao longo do mesmo.

Nestas duas situações casos o pretendido não será visualizar o evento por inteiro, mas sim os momentos mais interessantes e relevantes para o utilizador.

De modo a resolver estas questões surgiram os chamados *highlights*, vídeos curtos nos quais é possível observar de forma resumida o que ocorreu de maior relevo nesses eventos desportivos.

Porém atualmente grande parte desses *highlights* são gerados "à mão", ou seja, ao longo do jogo são marcados vários pontos de jogadas/momentos de relevo e no final é feita uma compilação dos mesmos. Normalmente a primeira versão que é apresentada logo após o jogo não é uma versão final, sendo modificada e só umas

horas depois, ou mesmo no dia seguinte, é que surge uma nova versão com melhor qualidade, que realmente retrate o que ocorreu no jogo.

Essa componente já foi implementada no trabalho anteriormente desenvolvido [1], e agora pretende-se estender o projeto documentando não só os melhores momentos dum jogo, mas também qualquer outro momento que o utilizador pretenda visualizar, através da criação de uma base de dados, em XML, com um conjunto de características extraídas dos vários frames processados.

1.2 Contexto

Esta dissertação vem na continuação de um projeto anterior do aluno Guilherme Fião de título "*Automatic Generation of Sport Video Highlights Based on Fan's Emotions"* [1], no qual é apresentado um modelo de geração de vídeos de resumo tendo como base jogos de futebol, onde eram utilizados dados como o som durante o jogo e as emoções expressas pelas várias pessoas ao longo do mesmo. Os dados dessas duas componentes foram combinados num algoritmo para que posteriormente fosse produzido um vídeo de *highlights* com um grau de qualidade elevado.

Os dados relativos às emoções foram obtidos através da plataforma WeFeel, onde os utilizadores podem partilhar remotamente as suas emoções ao longo da transmissão do evento desportivo.

Esta deteção é feita através de uma aplicação móvel que funciona como um comando de emoções onde o utilizador seleciona a emoção que está a sentir num momento do evento e que posteriormente é mostrada na televisão em conjunto com as emoções de outros utilizadores da aplicação, em tempo real, de modo a evitar que o utilizador tenha de consultar o *smartphone* distraindo-se do jogo que está a ver.

Como hoje em dia as redes sociais estão presentes no dia-a-dia de grande parte dos indivíduos, cada vez que ocorre um evento de grande relevo, a probabilidade de ocorrer um elevado número de publicações sobre tal acontecimento, numa plataforma como *Twitter*, *Facebook* ou *Instagram*, é muito elevada.

O que se pretende com este projeto é complementar o trabalho já desenvolvido, utilizando um conjunto de informação que se encontra presente nos vários *frames* do jogo. Ou seja, pretende-se aplicar uma série de algoritmos de processamento de imagem de modo a caracterizar cada *frame* do jogo.

Após esse processamento, o utilizador terá, sobre o formato XML, uma base de dados com um conjunto de informações relativo a cada momento do jogo. A partir desse ficheiro, o utilizador poderá realizar um conjunto de *queries* que o auxiliem a compreender e detetar os vários lances/momentos que ocorreram durante os vários períodos de jogo.

Estas informações poderão ser muito úteis não só para o utilizador adepto, mas também para os treinadores das várias equipas, pois poderão selecionar momentos mais relevantes, de modo a auxiliar o seu trabalho na análise dos vários jogadores e momentos que ocorreram no evento. Por outro lado, poderá auxiliar as produções televisivas detetando de forma automática momentos que pretendam observar.

1.3 Definição do Problema

Tendo em conta o que já foi implementado pelos autores [1,28], é possível gerar vídeos, onde os momentos mais relevantes foram efetivamente detetados podendo mesmo fazer-se uma comparação com os gerados pelas produções televisivas.

Dados das emoções e do som permitem gerar vídeos de *highlights* com uma grande precisão pois é de certa forma fácil detetar momentos relevantes, tendo em conta os valores obtidos através dessas duas componentes.

De referir que, no projeto desenvolvido, os dados das emoções nem sempre estão disponíveis para análise, ficando então como único elemento possível de utilizar o som.

Por exemplo, no caso do som, cada vez que existe um lance perigoso a tendência do público que está a assistir ao jogo no estádio é reagir através de festejos (aplausos, gritos de euforia ou assobios) podendo esses momentos ser facilmente detetados se for feita uma análise do som ao longo do tempo.

Outro caso em que se poderá detetar momentos importantes de um jogo é analisando a forma como os comentadores estão a falar, uma vez que na presença de lances perigosos, a tendência é terem um discurso mais acelerado e com entusiasmo, comparativamente com os momentos mais calmos do jogo.

Relativamente aos dados das emoções, quando existe um maior número de publicações relacionadas com o jogo, normalmente deve-se a um acontecimento de relevo e que pode ser marcado como um momento de *highlights*. No caso de haver

um número menor de publicações isso significa que não ocorreu nenhum acontecimento de grande relevo. Na aplicação WeFeel é aplicado um conceito semelhante, os utilizadores partilham emoções uns com os outros, ao longo do jogo e com base nesses dados são atribuídas emoções a cada momento, onde estas sejam detetadas.

Se combinarmos estes dois componentes, som e emoções, assumindo que os dados das emoções se encontram disponíveis, é possível criar vídeos curtos e com grande qualidade, onde se encontram representados os melhores momentos do jogo em questão.

Então, tendo em conta que já é possível produzir vídeos com qualidade no trabalho já desenvolvido, qual o próximo passo neste projeto?

É certo que já se produzem vídeos de resumo que representam os melhores momentos do jogo, com o trabalho já realizado, mas pretende-se ir mais além e caracterizar não só os momentos selecionados pelas emoções e som, mas também os vários frames que serão processados pelos algoritmos de imagem.

Ou seja, através de processamento de vídeo/imagem pretende-se retirar *features* que, de certa forma, caracterizam cada momento do jogo, que posteriormente serão utilizadas para a criação de uma base de dados que represente o jogo, e que posteriormente poderá ser utilizada para gerar os mais variados tipos de *highlights* e descrições do que ocorreu no jogo.

Uma das possíveis aplicações da combinação dos vários dados seria, por exemplo: imaginemos que, no minuto 44 do jogo Portugal-França, se detetou um grande valor de felicidade na aplicação *WeFeel*. Neste caso e para complementar esta informação retirada das emoções dos utilizadores que estão a ver o jogo, serão processados uma série de algoritmos sobre esse período de modo a obter dados como em que zona do terreno foi, se foi golo de Portugal ou França, se não foi golo, que tipos de zoom de câmara ou *shots* ocorreram nesse período, ou até que jogadores ou quantos jogadores estiveram envolvidos no lance.

Com todos os dados obtidos dos algoritmos de processamento de imagem será posteriormente criado um ficheiro XML, onde para cada *frame* ou conjunto de *frames*, serão atribuídos um conjunto de características retiradas através dos vários algoritmos de vídeo/imagem de modo a caracterizar o momento ou *frame* analisado.

Pretende-se que os resultados obtidos sejam guardados num ficheiro XML, pois trata-se de um formato normalizado, bem estabelecido e com suporte nos

mais variados tipos de sistemas e arquiteturas atualmente existentes. Com este formato, a análise dos dados e interface de apresentação podem ser separados e dar uma certa flexibilidade ao utilizador. Os resultados podem ser apresentados em ambiente web, mobile, desktop entre outros, dado o modelo XML utilizado com base de dados.

Em suma, atualmente o sistema consegue identificar e produzir resumos de um determinado evento futebolístico, porém é proposto ir mais ao detalhe e caracterizar cada momento presente em toda a extensão do evento desportivo, tendo em conta os vários descritores presentes em cada *frame*, como cor dominante, zona do terreno, tipos de *shot* utilizados, tipo de lance, que equipa participou, ou que jogadores estiveram envolvidos. Com esses dados o utilizador poderá gerar uma base de dados de componentes visuais e realizar pesquisas sobre esses dados.

A forma como posteriormente esses valores são aplicados fica ao critério do utilizador, ou seja, a aplicação fornece um conjunto de resultados em formato XML que poderão ser utilizados nos mais variados tipos de aplicações.

Todos estes processamentos devem ser realizados da forma mais automatizada possível de modo a retirar o esforço manual que é usado hoje em dia tanto na documentação dos jogos como na geração dos vários vídeos de resumo.

1.4. Principais contribuições

Para a implementação deste projeto definem-se as seguintes contribuições:

- **Implementação de um conjunto de algoritmos de processamento de imagem e vídeo.** Pretende-se extrair um conjunto de *features* capazes de caracterizar cada momento do jogo.
- **Teste dos vários algoritmos implementados.** Como existe uma dependência entre os vários algoritmos a desenvolver, é necessário garantir que os algoritmos de mais baixo nível possuem o desempenho esperado.
- **Criação de uma interface de teste.** Implementação de uma interface que permita ao utilizador gerar o ficheiro XML relacionado com o jogo que pretende analisar, bem como realizar um conjunto de *queries* extraindo também os vários resultados obtidos, também em formato XML.
- **Análise aos ficheiros gerados**. É necessário garantir o correto funcionamento da interação entre a interface e os ficheiros gerados. Pretende-se garantir que o conteúdo presente no vídeo efetivamente aparece no ficheiro XML gerado.
- **Otimizações de desempenho.** Como é esperado desenvolver um grande número de algoritmos, é também uma componente importante minimizar o tempo que todo o processo demora.

2

Trabalho Relacionado

2.1 Introdução

Neste tópico serão apresentados alguns conceitos teóricos básicos necessários para compreender os algoritmos e métodos estudados sobre processamentos de imagem e vídeo. Serão também apresentados um conjunto de temas desenvolvidos por outros investigadores que possuem características em comum com o trabalho que se pretende desenvolver.

2.1.1 Histogramas

O histograma (figura 2.1), também conhecido por distribuição de frequências, é um agrupamento de dados em classes, onde são contabilizados o número de ocorrências de cada classe. O número de ocorrências de uma determinada classe recebe o nome de frequência absoluta. O objetivo passa por apresentar os dados de uma maneira mais concisa e que permita extrair informação sobre os vários valores.

No processamento de imagem é muito comum utilizar histogramas para representar a distribuição de cores (RGB ou outros) e valores de luminosidade para uma determinada imagem.

No caso das cores, o histograma representa no eixo dos X os valores dos pixéis (0 a 255), podendo esse valor referir-se a uma determinada cor ou luminosidade dependendo do que o utilizador pretende analisar, enquanto no eixo dos Y encontra-se representado o número de pixéis na imagem [7]. Com este conceito encontrou-se uma maneira diferente de processar o conteúdo da imagem. Através do histograma é possível constatar valores associados ao contraste, brilho, variação ao longo da imagem. Praticamente todas as ferramentas de processamento de

imagem oferecem suporte para a utilização de histogramas, tal como o OpenCV [7], que será utilizado no desenvolvimento deste projeto.

Em termos de aplicações práticas o histograma desempenha um papel importante no processamento de imagem/vídeo, sendo muito útil na deteção de mudanças/transições de cenas, através da variação repentina dos valores de cores, luminosidade, dum *frame* para o outro. São também utilizados para a identificação de pontos de interesse numa imagem em que, para cada um desses pontos são associados um conjunto de histogramas com informações relativas às *features* que se pretende observar. São também bastante utilizados para a deteção de certos objetos pertencentes ao *background* ou *foreground* da imagem.

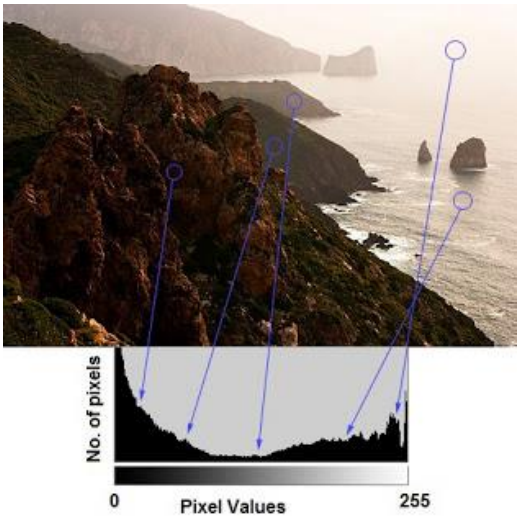


Figura 2.1 – Exemplo de possível aplicação de histograma retirado de [7].

2.1.2 *Background Subtraction*

Quando se pretende analisar certos objetos que se encontram num determinado *frame* ou conjunto de *frames*, uma das técnicas mais utilizadas é o *background subtraction* [9].

Devido à sua simplicidade, e como os vários *shots* utilizados ao longo de um vídeo muitas vezes estão fixos, em alguns pontos, esta técnica é utilizada nos mais variados tipos de aplicações, com o maior destaque para as aplicações na área da segurança.

Para esta técnica poder ser utilizada o primeiro passo é estabelecer o background que será utilizado, o chamado *background model*. Em seguida, realiza-se uma subtração entre o *frame* atual e o *background model* e como resultado dessa operação é possível obter todos os objetos que não pertencem ao *background mo-*

del, sendo também possível detetar movimento e fazer deteção individual de certos objetos/pessoas (Figura 2.2).

Normalmente o *background model* consiste na parte estática ou parcialmente estática de uma determinada cena/momento de um vídeo, e que permanece nesse estado ao longo de um certo período.

Quando se pretende realizar o *background extraction* existem dois grandes tipos de localizações onde este processo necessita de pequenas alterações: localizações *indoor*, ambientes com pouco movimento e onde a variação da luminosidade é relativamente reduzida, e localizações *outdoor*, locais onde ocorrem variações que podem influenciar o processo de extração de uma forma não desejada, como as variações de luminosidade, alterações causadas pelo vento, movimento de árvores, ocorrência de chuva. Nestas situações estas pequenas alterações podem ser detetadas como parte do *foreground* de forma errada, em vez de pertencerem ao *background*.

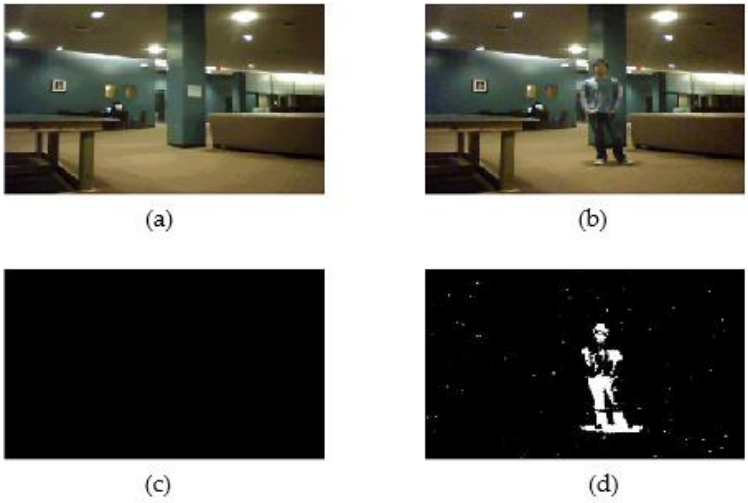


Figura 2.2 – (a) *Background model*, (b) nova imagem a analisar, (c) subtração de (a) pelo *background model*, (d) resultado de subtração de (b) por (a) retirado de [9]

Obstáculos ao Background Subtraction

Tendo em conta a técnica anteriormente apresentada, consideremos o seguinte exemplo: um carro pretende estacionar num lugar apropriado. Na implementação do ponto 2.1.2 seria necessário escolher um momento que represente o *background model* do parque de estacionamento, de maneira a ser possível distin-

guir os vários elementos do *foreground*, porém, após o carro estacionar este continuaria a fazer parte dos elementos de *foreground* o que não seria o mais desejado pois este encontra-se parado e não deverá mover-se nos próximos momentos, ou seja, passaria a fazer parte do *background* e não do *foreground* [9].

Com base no exemplo descrito é possível observar que a descrição inicial do *background subtraction* não é suficiente para representar os vários elementos do *foreground* e os seus respetivos movimentos. A solução proposta em [2] apresenta um modelo onde seriam criados *scene models*, onde é possível encontrar vários níveis de *foreground* e *background*, nos quais quando um objeto não apresentar movimento durante um certo período vai descendo para níveis inferiores até chegar ao nível mais baixo, correspondente ao *background*. Com esta implementação é possível resolver problemas semelhantes ao descrito no exemplo.

Uma outra questão, com igual relevância, relaciona-se com pequenas oscilações de um *frame* para o outro. O exemplo mais simples de observar este acontecimento num cenário *outdoor* de uma árvore, onde passa uma brisa, movendo ligeiramente as folhas da árvore. Com o modelo descrito até este ponto [2] todas essas pequenas oscilações seriam detetadas como novos objetos pertencentes ao *foreground*, o que não é verdade. Para resolver esta questão sugeriu-se que se implementasse um esquema de "flutuação de pixéis", onde estas pequenas oscilações são detetadas e ignoradas pelo sistema quando se pretende identificar elementos pertencentes ao *foreground*.

Com estas duas funcionalidades descritas já será possível obter melhores resultados quando se utiliza a técnica de *background subtraction.* Claro que estas duas novas funcionalidades representam uma maior complexidade para a implementação, porém o OpenCV já fornece bibliotecas bem estruturadas que suportam todo o processo descrito.

2.1.3 *Edge Detection*

Edge detection é uma técnica de processamento de imagem e vídeo onde é possível determinar pontos em que a luminosidade muda repentinamente [8]. Mudanças repentinas em imagens geralmente refletem elementos distintos e importantes no cenário, como a descontinuação da profundidade (transição entre o objeto e o *background*), descontinuação da orientação da superfície, mudança das propriedades do material ou variações na iluminação da cena.

Esta técnica é bastante usada para a extração de *features* duma determinada imagem/vídeo. A sua utilização reduz significativamente a quantidade de dados a serem processados e descarta informação que é considerada menos relevante, ainda que preservando importantes propriedades estruturais de uma imagem.

Quando se realiza este esquema de deteção um dos primeiros problemas a ter em atenção são as falsas bordas/limites, criadas por ruído na imagem (provenientes da digitalização, compressão ou através do próprio processo de captura da imagem). Este tipo de problema poderá ser resolvido facilmente utilizando algumas técnicas como *erosion* ou *dilation*, escolhendo o processo mais adequado para o problema.

Este princípio de *edge detection* possui várias aplicações [8], como deteção de linhas verticais, linhas horizontais, círculos ou semicírculos, entre outros. Uma das aplicações mais conhecidas designa-se *Canny Edge detection* [8], que será utilizada em alguns algoritmos apresentados mais à frente.

2.1.4 Transformação de Hough

A técnica Transformação de Hough [8] é um método de processamento de imagem, variante do *Edge detection*. Inicialmente esta técnica tinha como principal objetivo a deteção de linhas, mas foi posteriormente estendida para suportar outras formas geométricas como círculos, elipses e outras formas geométricas de estrutura simples.

Nos dois pontos seguintes serão abordadas as duas variações com maior relevo para o trabalho a desenvolver.

Deteção de Linhas

Esta técnica tem como objetivo detetar todas as possíveis linhas, tanto verticais como horizontais presentes numa determinada imagem (Figura 2.3). Numa primeira fase é aconselhado aplicar um processo de *Edge detection* como anteriormente foi apresentado, retirando da imagem elementos não necessários para a transformação de Hough, por exemplo *Canny detection*. Em seguida é necessário saber qual o sistema a utilizar para a expressão linear e, neste caso, utiliza-se um sistema de coordenadas polares com a seguinte expressão:

$$r = x\cos\theta + y\sin\theta$$

Para cada ponto da imagem é possível detetar todas as linhas que passam nesse ponto tendo como base a expressão:

$$r\theta = x\theta.cos\theta + y\theta.sin\theta$$

Após a aplicação dessa expressão é gerada uma onda sinusoidal. Se para cada ponto existir um outro ponto onde as suas curvas se intersetam no plano e possuem o mesmo r e θ, então pode-se afirmar que pertencem à mesma linha.

A linha pode ser detetada através do número de interceções que existe entre curvas. Quanto maior for esse número, maior será o número de pontos que a linha possuí. Geralmente, é possível definir um *threshold* que determina o número mínimo de interceções para que seja detetado como linha. Quando o número de interceções excede o *threshold* selecionado, então é possível indicar que se trata de uma linha com os parâmetros $(\theta, r\theta)$ do ponto de interseção.

(a) (b)

Figura 2.3 – (a) Imagem a ser processada, (b) imagem após o processo de Deteção de Linhas, retirado de [8]

Deteção de Círculos

Num espaço 2D um círculo é definido pela seguinte expressão:

$$(x-a)^2 + (y-b)^2 = r^2 \ (1)$$

Em que (a, b) é o centro do circulo e r é o raio. É com base nesta fórmula que o princípio de Hough foi criado, ou seja, num determinado ponto (x, y) é possível saber, para um determinado valor de r, se existe um círculo na imagem tendo como centro esse ponto.

Inicialmente aplica-se um processo semelhante ao utilizado para as linhas, ou seja, um processo de *Edge detection* como o *Canny* para excluir informação não necessária, reduzindo assim o número de valores que poderão ser utilizados para detetar círculos ou semicírculos. Posteriormente verifica-se se os pontos nesse raio formam um círculo ou não como é possível observar na Figura 2.4.

Figura 2.4 – Exemplo do processamento de Deteção de Círculos retirado de [8]

2.2 Emoções

A partilha de emoções é uma componente importante no dia-a-dia do ser humano. Com o avanço tecnológico surgiram novas maneiras de partilhar emoções através das várias redes sociais. Cada vez que um ser humano pretende reagir a algo com uma emoção, poderá recorrer a plataformas *online* como *Facebook*, *Twitter*, *Instagram* de modo a partilhar essa emoção com os seus amigos/seguidores. Essas emoções permitem determinar a forma como cada indivíduo reage a um determinado estímulo. No contexto deste projeto é interessante analisar o comportamento e as emoções de um grupo de indivíduos durante um evento desportivo.

Com a utilização em larga escala destas plataformas de redes sociais, de cada vez que ocorre um evento desportivo ou de grande interesse para a população, em geral, o volume de *tweets* ou *posts* relacionados com o mesmo é enorme. Estes meios tornaram-se uma fonte essencial para a partilha de emoções entre os vários cidadãos.

Esta situação ocorreu no Campeonato do Mundo de Futebol de 2014, onde houve uma partilha de 672 milhões de *tweets* durante o evento, e 35.6 milhões de *shares* durante um único jogo [3]. Existe um grande volume de informação a circular e é possível utilizar algumas dessa informação para saber o que está a acontecer sem ser necessário ver o jogo.

No caso de eventos desportivos, se for detetado um grande volume de publicações durante um certo período, é muito provável que algo de importante tenha ocorrido nesse evento. Ou seja, se for possível arranjar um modelo de modo a classificar os vários tipos de emoções partilhadas, seria possível e interessante utilizar esses valores de modo a estabelecer uma ligação entre as publicações e os pontos mais relevantes que ocorreram ao longo do evento.

Por exemplo, um adepto de Portugal está a ver o jogo Portugal-Brasil e no minuto 60 Portugal marca um golo, entretanto, esse adepto resolver fazer uma publicação a demonstrar a sua alegria, partilhando a sua emoção perante o golo de Portugal. Da mesma forma que este adepto faz a publicação, milhares de pessoas fazem o mesmo e é possível observar que nesse instante ocorreu um número elevado de publicações, marcando esse momento como um momento relevante. Essa marcação é feita ao segundo, pois, como se sabe muitos momentos poderiam ocorrer se a marcação fosse feita ao minuto.

Para tal ser possível, primeiro é necessário criar modelos que agrupem os vários tipos de emoções partilhadas. De modo a ser possível representar as emoções e as relações entre elas, surgiu o modelo da Roda das Emoções de Robert Plutchik [4].

Este modelo considera que existem oito emoções primárias: raiva, medo, tristeza, nojo, surpresa, curiosidade, aceitação e alegria. Cada uma dessas emoções representa um nível às quais são atribuídos uma cor e um conjunto de emoções secundárias. Trata-se de um modelo simples e de fácil utilização através do modelo de cores apresentado.

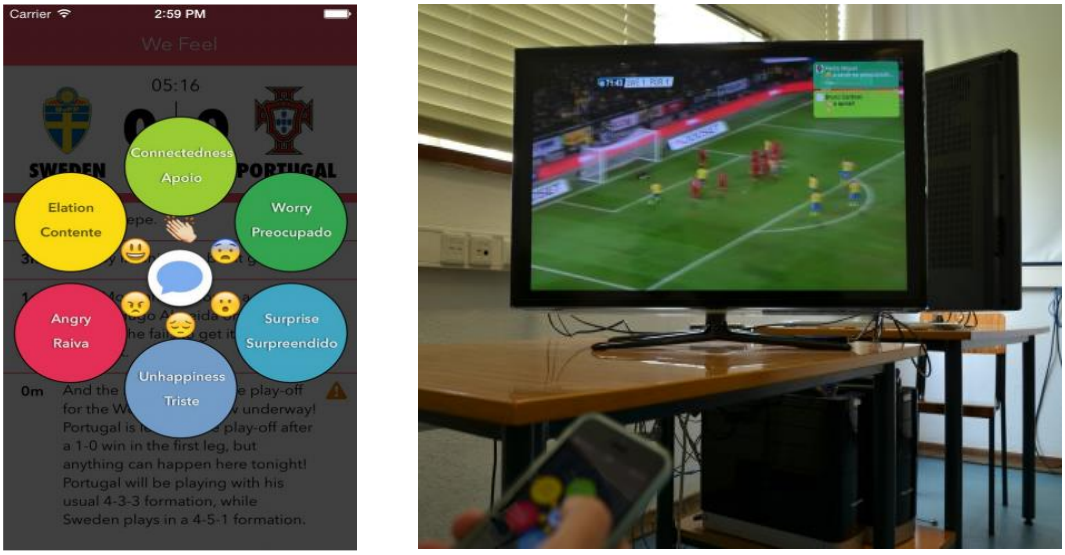
Para a representação e partilha das emoções existem aplicações móveis que permitem aos utilizadores realizar essa partilha de forma fácil e rápida, durante as transmissões televisivas de eventos desportivos. Uma delas é o *WeFeel* [28].

2.2.1 *WeFeel*

WeFeel [30] é um sistema utilizado para partilhar emoções durante a transmissão em direto de um evento desportivo. Utiliza uma aplicação móvel como controlador, através da qual os utilizadores podem partilhar as suas emoções. Este sistema utiliza uma adaptação da ferramenta de *self-assesment* CAAT [5] para a seleção da emoção, tornando a partilha de emoções mais simples e rápida (Figura 2.5.a). No *WeFeel* o sistema CAAT, que se baseia na roda de emoções de Plutchik

[4], foi adaptado e simplificado, considerando apenas as emoções mais sentidas pelos adeptos [30].

As várias emoções partilhadas são depois mostradas no ecrã da televisão em *real-time*, não sendo necessário o utilizador consultar o *smartphone* para saber que outras emoções foram partilhadas, distraindo-se do jogo que está a visualizar (Figura 2.5.b).



(a) (b)

Figura 2.5 – (a) Interface da aplicação *WeFeel* [30] (b) Ambiente da aplicação *WeFeel* [30]

2.3 Áudio

O áudio é uma das componentes mais importantes do vídeo. No contexto deste trabalho apresenta uma importância ainda maior. No início todas as transmissões de eventos desportivos eram feitas através da rádio, com a utilização única do som/áudio, que permitia a quem estava a ouvir saber o que estava a acontecer através dos relatos dos comentadores. Só mais tarde é que a transmissão começou a possuir suporte para imagens/vídeo. Por isso é fácil aceitar que o áudio possui um grande grau de importância quando se pretende analisar um determinado evento desportivo, neste caso, um jogo de futebol.

Normalmente, em cada jogo de futebol existem vários emissores como os comentadores, os cânticos e reações dos adeptos, a própria comunicação entre os

jogadores no campo e os apitos do árbitro. Com base nesses emissores, é possível analisar as situações onde estes se manifestam e retirar informações que auxiliam a identificar certos acontecimentos num jogo de futebol [1,28]. Por exemplo, cada vez que ocorre um lance de golo a reação do público, que está a assistir, tenderá a ser de ânimo e exaltação, enquanto a tendência dos comentadores será produzir um discurso mais acelerado e num tom elevado. No caso de um lance de golo ou de uma falta perigosa, o árbitro deverá utilizar o seu apito para marcar o acontecimento. Por outro lado, o público, se for um lance contra a equipa que apoia, terá tendência para assobiar, se for um lance a favor, festejar.

Estes tipos de manifestações podem ser usados para marcar acontecimentos ao longo do jogo, tal como foi desenvolvido no projeto de [1,28].

O áudio torna-se assim uma componente importante, não só para o processo de geração automática dos vídeos de resumo, mas também como forma de caracterizar cada momento do jogo. Nos dois pontos seguintes, serão apresentadas duas técnicas desenvolvidas que utilizam o processamento do áudio para a extração de momentos importantes durante um jogo de futebol.

2.3.1 Deteção do Apito

Os vários sons durante um jogo de futebol normalmente são complexos e difíceis de distinguir, porém o apito do árbitro diferencia-se bastante e torna-se fácil de detetar [20]. Com base no trabalho de Dian, Yi-Ping e Binh [11] num jogo ouve-se o apito nas seguintes situações:

- Início e fim do jogo ou da primeira parte.

- Ocorrência de uma falta ou fora-de-jogo.

- Ocorrência de golo, canto, livre ou lançamento de linha lateral

Cada uma destas situações representa possíveis momentos importantes para o jogo. Para a deteção do som do apito determinou-se que este ocorre nas frequências (3500Hz – 4500Hz), durante normalmente um período superior a 1 segundo, o que torna o som distinto e de fácil deteção.

Porém Dian, Yi-Ping e Binh [11] determinaram três possíveis limitações que interferem diretamente nos resultados obtidos com este método. Primeiro quando o árbitro realiza um apito de curta duração (inferior ao 1 segundo definido) este pode não ser detetado tendo em conta o ruído existente no estádio, o que poderá levar à não deteção de um acontecimento importante. Segundo, num momento em

que os adeptos estejam eufóricos e os comentadores estejam a produzir um discurso mais acelerado, estes podem sobrepor-se ao apito. Para além disso, também é frequente os adeptos produzirem assobios, que podem ser facilmente confundidos com o apito do árbitro tendo em conta todo o ruído. Por último, defendem que nem sempre o apito do árbitro possui as frequências de 3500Hz a 4500Hz, podendo ter valores diferentes consoante o ambiente envolvente, por exemplo, locais com água onde o som se propaga de maneira diferente.

Para a resolução destes problemas, Dian, Yi-Ping e Binh [11], propuseram a inclusão de dois *thresholds*, no modelo anterior [20]. O primeiro *threshold*, definia o valor mínimo a partir do qual é considerado um assobio/apito do árbitro, o segundo representa o número mínimo de *frames* vizinhos de um determinado *frame* onde foi detetado o assobio e que necessitam de conter tal valor de modo a confirmar que se trata efetivamente de um assobio corretamente detetado.

Para além disso, os valores mínimos e máximos de frequências foram ajustados consoante o desporto em análise. Por exemplo o futebol manteve-se nos 3500Hz-4500Hz, mas no caso da natação esse valor teve de ser ajustado para 2800Hz-3200Hz.

2.3.2 Deteção de Entusiasmo

Dian, Yi-Ping e Binh [11] defendem que para além dos apitos do árbitro, os adeptos e comentadores também podem ser alvo de estudo, de modo a retirar dados interessantes para o processamento do vídeo.

Defendem que existem três situações relevantes para a análise:

1. Os adeptos apoiam a sua equipa de forma mais efusiva e os comentadores começam a falar mais alto;
2. A voz dos comentadores muda de tom;
3. Os comentadores falam mais rápido e com menos pausas no discurso.

Através dessas três características realizaram um processo de análise e extração dos vários momentos relevantes do jogo, semelhante ao apresentado no tópico da deteção do apito. Os resultados foram significativamente melhores e com uma maior precisão quando comparados com o método anterior.

2.4 Texto

O texto presente em vídeos/imagens poderá representar uma fonte de informação bastante importante, dependendo do contexto, pois muitas vezes encontra-se diretamente relacionada com o vídeo/imagem em questão [2]. Os primeiros trabalhos relacionados com o processamento textual surgiram no âmbito da deteção de palavras/caracteres em documentos simples de texto. Porém, essa abordagem foi posteriormente adotada e modificada para ser possível utilizar em imagens e vídeos. Ao adaptar as bases desta técnica para imagens/vídeo surgiram obstáculos que não ocorriam no processamento de documentos de texto, devido ao maior número de variantes quando se compara uma imagem com texto fixo.

Entre os principais obstáculos destacam-se os seguintes: o facto de o *background* não ser fixo como em documentos de texto, a existência de várias resoluções de imagem, algumas delas bastante reduzidas, e o contraste entre o texto e outros elementos da imagem. Devido a estas questões foi necessário dividir o processamento de texto em dois grandes grupos [2]. Um primeiro grupo onde se encontram as legendas e informações de vídeos, adicionados manualmente sobre as imagens, não fazendo parte da cena em questão, por exemplo, painéis de resultados desportivos ou a legendas de filmes/séries. O segundo grupo representa o texto presente na imagem/cena, que não foi adicionado numa fase posterior à captura da imagem, como por exemplo: o texto em cartazes, vestuário, escrito nas paredes, ou papéis presentes na cena, no fundo, que pertençam ao ambiente da imagem.

Enquanto o primeiro tipo de texto apresenta uma posição com pouca variação, o que o torna mais fácil de processar, o segundo tipo, por ser mais imprevisível, representa um maior custo para a sua deteção e compreensão.

No projeto em questão, uma das componentes interessantes de implementar passaria pela análise da tabela de resultado e tempo presente em todos os jogos de futebol, pois como se encontra numa posição estática variando só os valores textuais mostrados, entraria assim na primeira categoria apresentada. Através dessa análise será possível obter dados relativos ao resultado do encontro, bem como estabelecer uma relação entre o *frame* que se está a analisar e o tempo retirado da análise textual, associando assim cada *frame* a um exato momento do jogo. Uma das aplicações deste conceito será apresentada no tópico seguinte com o conceito de OCR.

2.4.1 Optical Character Recognition

OCR é uma técnica de conversão e extração dos carateres presentes em imagens, textos escritos ou impressos [6]. É bastante utilizada como forma de obtenção de dados de formulários preenchidos por clientes, leitura de passaportes e cartões de identidade, entre outros. Através dessa extração é possível converter toda a informação retirada para uma plataforma informatizada de maneira a ser possível consultar e iterar sobre os dados obtidos. OCR é usado também em áreas como a inteligência artificial, *pattern recognition* e *computer vision* [6].

As primeiras versões criadas necessitavam que fosse fornecida uma base de dados com os vários modelos dos carateres que se pretendia reconhecer, bem como as várias variações tendo em conta o tipo de fonte escolhido. Atualmente, os sistemas mais avançados já são capazes de detetar os vários tipos e formatos de carateres com grande precisão, não sendo, portanto, necessário fornecer qualquer tipo de informação ao sistema.

Pretende-se implementar uma variação deste conceito no trabalho a desenvolver, para a deteção textual do painel com o resultado e o tempo do jogo.

Com esta proposta seria possível, a cada *frame* atribuir um tempo de vídeo bem como um tempo do jogo e o correspondente resultado.

2.5 Imagem/Vídeo

Analisar os vários *frames* dum jogo de futebol é uma das componentes mais importantes para a extração de informação, para a criação de vídeos de resumo. Ao longo desta secção serão apresentados um conjunto de algoritmos, já desenvolvidos e devidamente testados, que compartilham algumas semelhanças com o projeto desenvolvido.

Estudar e analisar cada um destes algoritmos tornou-se bastante importante, numa primeira fase, pois forneceu um conjunto de ideias e conceitos base bastante importantes no projeto desenvolvido, apesar de nem todos os algoritmos terem sido implementados muitas das ideias base são as mesmas.

Decidiu-se separar os vários algoritmos em três grandes grupos, *Low-Level*, *Mid-Level* e *High-Level*. Com esta separação é possível analisar os algoritmos e técnicas com graus de complexidade diferentes duma forma mais organizada.

2.5.1 Low-Level

Neste tópico serão apresentados um conjunto de algoritmos que analisam os aspetos mais básicos presentes em cada *frame* dum jogo de futebol. Estes métodos são bastante importantes pois muitos deles são utilizados como base por algoritmos de *features* mais avançadas, como serão apresentados nos tópicos *Mid-Level* e *High-Level*. Cada uma das características analisadas neste tópico não varia de campo para campo de jogo. Ou seja, são características estáticas.

2.5.1.1 Classificação dos tipos de *shots*

Durante um jogo de futebol as câmaras responsáveis pela transmissão televisiva do evento possuem várias posições diferentes. Esta *feature* pode ser aproveitada como uma forma de caracterizar cada imagem apresentada, atribuindo a cada momento um determinado tipo de *shot*. Segundo o defendido em [17] e de forma complementar em [14], existem durante um jogo quatro tipos de shots possíveis:

- C*lose-up Shot*: é realizado zoom muito aproximado a um jogador ou grupo de jogadores (Figura 2.6.a);

- *Out-Of-Field Shot*: ocorre quando existem imagens do público ou em redor do campo de jogo (Figura 2.6.b);

- *Medium Shot:* poderá mostrar o campo reduzidamente, ou apresentar um zoom maior comparado com o *Long Shot* (Figura 2.6.c);

- *Long Shot:* mostra uma vista geral do campo (Figura 2.6.d);

Figura 2.6 – (a) *Close-up Shot* (b) *Out-of-Field Shot* (c) *Medium Shot* (d) *Long Shot*, retirado de [17]

A classificação do tipo de *shot* que ocorre em cada *frame* é um processo importante e fundamental para a implementação de certos algoritmos de *Mid-level* e *High-level*, como será referido em mais detalhes nas respetivas especificações.

Para conseguir obter a diferença entre os vários tipos é fundamental observar o número de pixéis que possuem o valor correspondente ao relvado. Quando um determinado *frame* representa um número de pixéis baixo é fácil identificar que se trata de um *Out-Of-Field Shot* ou um *Close-up Shot*, pois em ambos os casos a quantidade de terreno de jogo que aparece é muito inferior quando comparados com os outros dois tipos de *shots*.

De modo a tornar este processo mais simples e eficaz tanto [20] como [17] defendem que deve ser feita uma divisão em cada *frame* de 3:5:3 (Figura 2.7). Como normalmente os jogadores e bola encontram-se no centro da imagem esta divisão torna-se interessante para auxiliar a identificação dos tipos de *shots*. Por exemplo no caso de um *Close-up Shot* o jogador em questão irá ocupar vários espaços dessa divisão sendo facilmente detetado que a relva não se trata do elemento dominante do *frame*.

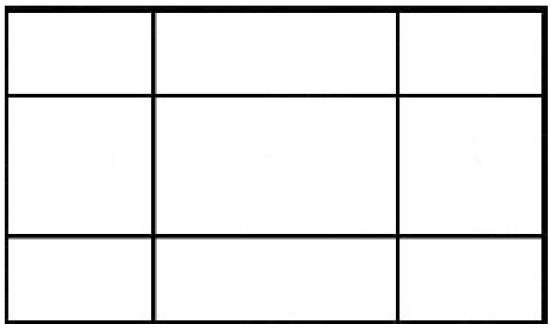


Figura 2.7 – Divisão do *frame* em 3:5:3, retirado de [17]

Apesar da divisão de *frame* ser interessante e ajudar nalguns casos, não consegue resolver o problema de diferenciar os *Medium Shots* e *Long Shots*, pois não existem fatores significativos que permitam identificar claramente qual o tipo de *shot* que está a ser mostrado, para além da diferença da quantidade de zoom utilizada.

Numa tentativa de resolver este problema os investigadores responsáveis pelo documento em [20], apresentaram a seguinte solução. No primeiro passo trata-se de retirar do *frame* a maioria dos pixéis que não pertençam ao relvado. Em seguida, são analisados os *frames* que apresentem cor verde como cor dominante e utilizado um *threshold* para determinar os limites. Como num *Long Shot* os jogadores encontram-se representados em poucos pixéis, estes são maioritariamente retirados da imagem, logo no final deste processo um *Long Shot* terá uma imagem praticamente toda verde, em contraste com o *Medium Shot* onde os jogadores são

representados logo, o valor de verde será menor, sendo possível diferenciar os dois tipos de cenas.

Relativamente à diferença entre um *Close-up Shot* e *Out-Of-Field Shot*, estes são facilmente diferenciados pois num *Out-Of-Field*, a cor predominante em todas as divisões da imagem é sempre diferente do verde do terreno de jogo, são sempre imagens do público presente no estádio, ou então do banco de suplentes de cada equipa.

2.5.1.2 Identificação da Zona do Terreno

Um campo de futebol pode ser dividido em 12 zonas diferentes, 6 para cada lado como se pode observar na Figura 2.8. Estas zonas foram escolhidas segundo os autores [12], para ser possível detetar transições ou mudanças numa jogada, ou seja, quando a câmara de jogo mostra uma zona diferente representa uma alteração na corrente de jogo, como um contra-ataque, uma defesa ou um lance de perigo com o jogador a entrar na grande área.

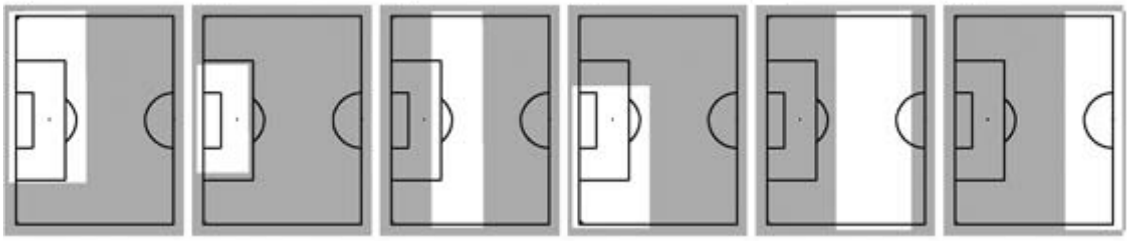


Figura 2.8 – 6 Zonas diferentes presentes em cada metade do campo, retirado de [12]

Cada uma destas zonas são possíveis de localizar através da análise das linhas e formas presentes na zona de jogo, como as linhas laterais, linhas de grande área ou, área do meio-campo. Segundo a proposta apresentada pelos autores [12], é necessário calcular e avaliar o resultado de cinco descritores para ser possível determinar qual a zona em que a câmara se encontra no momento, sendo esses descritores os seguintes [12]:

- Orientação das linhas detetadas. A cada linha é atribuída uma direção de modo a ser possível verificar a localização da câmara pois todos os campos de futebol, devido às regulamentações do desporto possuem as mesmas linhas, em posições semelhantes;

- Existência das linhas de canto, que permite, se forem detetadas, reduzir as possibilidades de localização da câmara aos quatro cantos do campo;

-Existência de linhas e círculo de meio-campo;

-Existência de semicírculo e linhas de grande área;

- Número de pixéis pertencentes à zona de campo. É definido um *threshold*, que permite associar cada valor obtido a um conjunto de zonas.

Após avaliar estes cinco descritores, deverá ser possível associar cada imagem a uma das doze zonas anteriormente definidas. Poderá haver alguns casos em que seja mais complicado associar uma zona à imagem, nomeadamente em imagens localizadas perto do meio campo pois torna-se difícil definir a diferença entre já estar de um lado do campo ou do outro, devido à falta de descritores auxiliares como linhas laterais.

Todo o processo de extração de linhas, círculos e outras formas utiliza, como já foi apresentado anteriormente, o processo de extração de Hough.

Em suma o processo utilizado para localizar as zonas do terreno é o seguinte: extrair linhas e outras formas da imagem, calcular e analisar os valores dos vários descritores, atribuir uma das doze zonas definidas ao *frame* em questão.

No algoritmo desenvolvido, como será apresentado no capítulo 3, optou-se por utilizar 5 zonas em vez das 12 como implementado em [12], pois era uma opção que fazia mais sentido tendo em conta a solução desenvolvida.

2.5.1.3 Deteção dos Postes das Balizas

A câmara de jogo, normalmente, segue sempre a posição onde a bola se encontra e cada vez que a baliza aparece no *frame* representa um lance de possível perigo para a equipa que está a defender e uma ocasião para marcar, da equipa que está a atacar.

O principal ponto que diferencia a técnica utilizada em [15] e [23], na geração de resumos de jogos de futebol, é a utilização dos postes das balizas como principal ponto de referência. Devido às leis do jogo, cada campo tem obrigatoriamente de possuir duas balizas com comprimentos regulamentados e sempre na mesma posição no campo.

De modo a localizar a baliza em cada *frame,* é realizado um processo de *Edge detection* chamado *Top-Hat Transform*, seguido de uma conversão do resultado de uma imagem de cores para uma imagem binária. Posteriormente, é necessário avaliar todos os *frames* candidatos que possam conter uma baliza.

Figura 2.9 – Resultado ideal do processo de Deteção de Balizas, retirado de [15]

Após o processo inicial de remoção dos elementos não necessários, como foi apresentado no tópico 2.5.1.1, existem quatro tipos de *shots* durante um jogo de futebol. Com base no trabalho proposto por [15], cada vez que ocorre uma possibilidade de remate normalmente são apresentados *shots* de *Long* ou *Medium Shot*, nos quais é possível encontrar a baliza.

Com base neste facto, foi desenvolvido um segundo algoritmo de modo a detetar a presença da baliza num determinado *frame* (Figura 2.9), quando ocorre um desses dois tipos de *shots*. Ou seja, o sistema desenvolvido analisa se o *frame* representa um *Long* ou *Medium Shot* e se a baliza está nesse *frame*. Caso se verifique as duas condições, realiza o mesmo processo para N *frames* seguintes, sendo N um *threshold* definido. Se o número de *frames* seguintes exceder N, então poderá tratar-se de um lance de ataque. O próximo passo do algoritmo define se o lance deu origem a um golo ou não. Para tal é necessário observar o que ocorre sempre que existe um golo de uma equipa.

Normalmente após um golo, a tendência das transmissões televisivas é mostrar num período entre 30 a 120 segundos reações dos jogadores e treinadores das equipas, bem como as reações dos adeptos que se encontram nas bancadas. Ou seja, uma sequência de *shots Close-up* e *Out-Of-Field*, logo o algoritmo analisa os *frames* seguintes, de modo a avaliar o número de *frames* que apresentam um destes dois tipos de *shots*. Mais uma vez os valores limites e mínimos são definidos por *threshold*.

2.5.1.4 Deteção da Área de Penálti

A grande área é uma das zonas mais importantes num campo de futebol, pois é onde se encontram as balizas de jogo. Como para uma equipa marcar, necessita de inserir a bola na baliza, torna-se uma zona do campo interessante para analisar. Ahmet e Murat [17] desenvolveram um modelo de extração e resumo dos vários

momentos durante um jogo de futebol. Numa das suas abordagens resolveram detetar a presença da grande área em cada *frame*.

Nesse processo limitaram a pesquisa à procura por três linhas paralelas, em *frames* do tipo *Medium* e *Long Shot*, representadas pela linha de grande área, linha de pequena área e linha final de campo. Para a deteção das três linhas inicialmente realizam um processo de extração da informação desnecessária, um processo de *Edge detection*, seguido de um algoritmo de transformação de Hough, como já foi apresentado anteriormente.

No caso do *frame* possuir as três linhas paralelas, poderá afirmar-se que se trata de um *frame* que contém a grande área. Posteriormente é realizado um processo semelhante ao Deteção de Balizas onde são analisados os *frames* seguintes de modo a verificar se ocorreu um lance de golo ou não.

2.5.2 *Mid-Level*

Neste tópico serão apresentados um conjunto de algoritmos estudados que utilizam algumas funcionalidades dos algoritmos *Low-Level* como base de implementação. Apresentam um grau de complexidade superior aos anteriores

2.5.2.1 Deteção da Bola

Durante um jogo de futebol a bola é o centro da atenção para todos os elementos do jogo, sejam jogadores, treinadores, árbitro ou adeptos, e é um dos objetos mais interessantes para analisar num ambiente de processamento de jogos de futebol.

Segundo o estudo de Samuel e David [16], é possível identificar e localizar a bola num determinado *frame* recorrendo ao seguinte processo. Segundo os autores, a bola possui três características únicas que possibilitam essa localização:

- Existência de um grande contraste entre a bola e o relvado;

- Dimensões reduzidas que podem ser estimadas utilizando uma relação com os tamanhos dos jogadores. Com este processo pode-se remover elementos que não sejam a bola da imagem;

- A bola é redonda e possui sempre as mesmas dimensões, que se encontram regulamentadas nas leis do jogo.

Utilizando estas três características é possível processar o *frame* com o objetivo de remover os elementos não relacionados [16], obtendo só os candidatos com o formato de bola. Posteriormente é aplicado, a cada um desses candidatos, um processo de dilatação e, todos os que apresentarem formas circulares, altura e largura similares a uma bola, utilizando uma comparação com os tamanhos dos jogadores, são marcados como tal. De referir que este processo deverá ser aplicado a *frames* do tipo *Long* e *Medium Shot*, pois a probabilidade de a bola estar no *frame* é maior. *Frames* do tipo *Out-of-Field* não faz sentido avaliar pois são imagens do público e treinadores que não contêm a bola, os *frames* do tipo *Close-up* dependem muito da forma como a produção televisiva realiza o zoom sobre os jogadores, sendo que nesses momentos a bola não é o foco principal.

2.5.2.2 Deteção e Identificação de Jogadores

Identificar os jogadores que aparecem em cada *frame* seria uma das componentes mais interessantes de implementar no trabalho em questão, pois permitiria saber quais os jogadores envolvidos numa determinada jogada. Para além disso, permitiria realizar vídeos de resumo mais específicos como, por exemplo, todos os lances em que um determinado jogador aparece.

Para realizar esse objetivo foram sinalizadas três possíveis abordagens com a finalidade de identificar os jogadores presentes em cada *frame*.

Uma primeira abordagem [20] seria aplicada quando ocorrem *shots* do tipo *Close-up*. Primeiramente, seria necessário haver uma base de dados com as faces de todos os jogadores envolvidos no jogo, suplentes incluídos, algo não muito difícil de obter pois essa informação encontra-se normalmente nas fichas de jogo. Após ter esse conjunto de informações, seria realizado um processo de f*ace detection* seguido de outro processo de *face recognition*, em cada *frame*, de modo a identificar quais os jogadores presentes nesse momento.

A segunda abordagem teria como objetivo a leitura dos caracteres presentes nas camisolas e calções dos jogadores, utilizando uma técnica semelhante à apresentada no ponto 2.4, técnica OCR [6]. Tal como na primeira abordagem, este processo necessita de imagem com boa qualidade, por isso, não será possível utilizar o reconhecimento de texto em todos os tipos de *frames* e mais uma vez só seria utilizado em *frames* do tipo *Close-up Shot*.

Em alguns casos também será possível aplicar estas duas abordagens a lances *Medium Shot*, porém fica muito dependente da quantidade de zoom aplicada, bem como a qualidade e resolução da imagem filmada pelas produções.

Numa última abordagem [21], passaria por utilizar um processo de identificação dos jogadores baseado na cor dos equipamentos. Esta abordagem seria aplicada a momentos onde os *shots* são do tipo *Long* e *Medium* (exceto alguns casos particulares), pois é difícil, ou praticamente impossível, realizar um processo de reconhecimento facial ou de leitura dos caracteres das camisolas, devido à pequena dimensão que os jogadores apresentam nesses *frames*. No final deste processamento não seria possível identificar individualmente cada jogador presente no *frame*, mas seria possível saber quantos elementos de cada equipa estão presentes no mesmo.

Frames do tipo *Out-Of-Field* não são alvo de estudo pois não possuem informação relevante para a identificação dos vários jogadores.

Quando se aplica um destes processos a um conjunto de *frames* é necessário ter em conta que normalmente só em *frames Close-up* é que existe um número reduzido de jogadores, nos outros casos existe um elevado número de informação a ser processada o que pode representar um elevado custo computacional.

2.5.2.3 Deteção do Árbitro

Os árbitros são elementos importantes que fazem parte dum jogo de futebol. Sempre que ocorre algum lance importante, como um golo, fora-de-jogo ou falta, são chamados para o assinalar, seja pelo apito ou com algum gesto corporal. Ou seja, analisar a presença do árbitro nos vários *frames* poderá contribuir positivamente para a geração dos resumos dos vários jogos.

Por regra, os vários árbitros usam equipamentos que os distinguem facilmente dos vários jogadores de campo [17], por isso, existindo essa informação, podem ser utilizados algoritmos como o *domain color region detection* para fazer a deteção do individuo. Por outro lado, cada vez que um árbitro intervém num jogo a tendência das produções televisivas é realizar um zoom, focando a imagem no árbitro. Sempre que exista um lance onde o árbitro intervém com alguma relevância, ocorre um conjunto de *shots* do tipo *Close-up*, *Out-Of-Field* e por vezes *Medium Shot* [17]. Neste último caso poderá ser mais difícil detetar o árbitro pois as suas dimensões poderão ser demasiado pequenas.

No processo implementado por Ekin e Tekalp [17] para a deteção de lances com árbitro são realizadas duas projeções binárias (horizontal e vertical) dos *frames* em que possivelmente o árbitro está presente, utilizando a cor distinta do seu equipamento como componente da transformação. Após as transformações, são realizadas comparações ao nível do rácio entre a área marcada como a posição do árbitro e o tamanho do *frame*, rácio de altura e largura (se corresponde a medidas capazes de representar um humano) e análise do padrão da distribuição dos valores resultantes.

Após confirmar a deteção do árbitro, esse *frame* ou conjunto de *frames* poderão ser marcados como um momento relevante para os vídeos de resumo. Claro que este processo não é o mais fiável para a geração dos momentos para o resumo do jogo, pois fica dependente da produção televisiva mostrar um plano do árbitro sempre que ocorre um lance relevante algo que nem sempre pode acontecer.

2.5.3 *High-Level*

Nos dois tópicos anteriores foram apresentados uma série de algoritmos e métodos para a extração de *features* e momentos importantes, com a finalidade de gerar vídeos de resumo com qualidade, analisando e documentando cada momento do jogo. Neste tópico serão apresentadas três situações, bastante importantes, que ocorrem num jogo de futebol. Cada uma dessas situações possui um grau de complexidade bastante elevado, utilizando os algoritmos e técnicas anteriores como base.

2.5.3.1 Deteção de Lances de Bola parada

Segundo os autores de [16], existe uma relação entre a posição da bola e as linhas do terreno. Perante esta afirmação decidiram classificar os de lances de bola parada em três tipos: cantos, livres e penáltis. Para a deteção dum lance de bola parada é feita uma análise durante um curto período, avaliando se os vários jogadores se encontram parados, uma característica dos lances de bola parada.

O método desenvolvido inicia-se com um processo de extração da cor do relvado, seguido de um processo de procura na imagem pela bola do jogo. No caso de a bola estar localizada num dos cantos do campo, trata-se de uma situação de canto.

Para distinguir os outros dois tipos de lance é realizada uma procura por linhas paralelas utilizando a transformação de Hough no caso de não ser possível extrair as linhas o lance analisado trata-se dum livre, caso contrário é um lance de penálti, pois a grande área foi detetada e a bola está dentro dessa região.

2.5.3.2 Deteção do Fora-de-jogo

Um dos lances mais polémicos e que na maioria das vezes gera mais discussão entre os adeptos é o fora-de-jogo. No modelo proposto em [16] definem duas possíveis abordagens para a deteção.

A primeira passa por analisar cada *frame* separadamente e quando existe uma mudança drástica na trajetória da bola verificar a posição de cada jogador. Porém, indicam que este processo não é nada simples, pois primeiro, é difícil perceber quem passou a bola, e segundo, dizem que é difícil até para o humano, poder distinguir e averiguar o fora-de-jogo quando existe um grande grupo de jogadores junto.

Numa segunda abordagem, decidiram não verificar *frame* a *frame*, mas estabelecer uma janela temporal, realizando *tracking* de todos os jogadores que possivelmente poderão estar fora-de-jogo, verificando também a posição da bola, num ambiente 3D criado pelos investigadores de [16]. Para além disso, executa um processo de verificação dos shots que se seguem ao potencial lance de fora-de-jogo, pois após ser identificado um lance desse tipo existe, normalmente, um período onde o árbitro auxiliar é mostrado com a bandeirola na mão.

2.5.3.3 Deteção de Golo

Marcar um golo é talvez o evento mais importante num jogo de futebol. Fazer a deteção de golos é importante não só para a produção de vídeos resumo, mas também para a descrição de cada *frame*.

Como já foi apresentado é possível detetar lances de golo utilizando algoritmos presentes nos pontos *Low-Level* e *Mid-Level*, combinando um algoritmo de extração de *features* num determinado *frame* com a análise de N *frames* seguintes. Cada vez que ocorre um lance de golo, existe sempre um período de 30-120 segundos de festejos, onde ocorre um conjunto de *Close-up* e *Out-Of-Field Shots.* Portanto, se for combinado um conjunto de algoritmos de processamento com a análise dos *shots* seguintes, será possível detetar a ocorrência de golos num jogo.

No documento [16], é apresentado um método que ainda não foi descrito e que possui o mesmo objetivo. Nesse sistema é realizado uma transformação circular de Hough, onde a posição da bola é analisada e verifica-se se já passou a linha de baliza ou não. Após obter todos os lances candidatos é feita uma seleção por rede neuronal, onde todos os candidatos a lance de golo são avaliados e excluídos os lances em que não ocorre tal acontecimento.

2.6 Discussão

Ao longo do capítulo 2 foram apresentados um conjunto de algoritmos e técnicas para extração de *features* de vídeos e imagens. Relativamente aos tópicos 2.2 e 2.3, já possuem uma versão implementada no trabalho [1], não sendo necessário efetuar qualquer alteração, pois trata-se de algoritmos que já se encontram devidamente implementados e testados.

Relativamente ao ponto 2.4 pretende-se fazer uma análise do local do *frame* que contém o pequeno placar com o tempo e resultado do jogo. A partir dessa leitura será possível associar cada *frame* a um segundo do jogo determinado a partir desse processamento.

No ponto 2.5 é apresentado um conjunto de algoritmos de processamento de imagem organizados em três níveis de complexidade. No primeiro, apresentam-se todos os algoritmos estáticos que não variam de jogo para jogo, como os tipos de *shots*, leitura das linhas do campo. Nos outros dois níveis, apresentam-se algoritmos mais complexos que utilizam como base conceitos dos algoritmos *Low-Level*.

Utilizando muitos dos conceitos apresentados neste capítulo é possível desenvolver um conjunto de algoritmos de processamento de imagem complementando assim o trabalho já desenvolvido.

3

Extração Semi-Automática de Informação Desportiva a partir de Vídeo

Neste capítulo será feita uma descrição do projeto desenvolvido, desde as funcionalidades da aplicação, a descrição da interface desenvolvida, bem como uma descrição da arquitetura e modelo escolhido para o desenvolvimento de todo o projeto. Serão utilizadas componentes visuais como imagens e diagramas UML para facilitar a compreensão de toda a organização do sistema.

3.1 Funcionalidades

Com o projeto desenvolvido, pretende-se dar ao utilizador todas as funcionalidades necessárias para ser possível gerar o maior e mais preciso volume de dados relacionados com um determinado jogo de futebol.

Dotou-se o sistema de um conjunto de funções e mecanismos de modo a ser possível extrair um conjunto de informações só utilizando as várias imagens presentes numa transmissão de um evento desportivo.

Mais concretamente decidiu-se criar mecanismos que extraiam a seguinte informação dum determinado *frame*:

- Identificar tipos de câmaras utilizadas;
- Localizar a zona do campo;

- Reconhecer jogadores de campo;
- Implementar o reconhecimento e identificação facial de jogadores;
- Identificar e reconhecer textualmente o placard que contem o tempo e resultado do jogo.

Estes foram os cinco principais tópicos em que o trabalho se focou e que serão explicados em maior pormenor mais à frente neste relatório.

Cada algoritmo gerado foi testado e otimizado para obter os melhores resultados possíveis, mas também de forma a suportar as várias variantes que podem ocorrer ao longo dum jogo de futebol, como mudanças de luminosidade, localizações de câmaras, entre outros.

Após ser possível realizar um processamento completo e bem estruturado, decidiu-se criar métodos para guardar a informação gerada, sob o formato de ficheiros XML, que posteriormente foram utilizados como forma de consulta para a criação e geração das várias *queries* que se podem executar tendo em conta os dados extraídos.

Criou-se um sistema automatizado diminuindo, ao máximo, o trabalho do lado do utilizador, de maneira a ser possível processar qualquer vídeo. Nesta aplicação, é necessário ter o vídeo do jogo, um conjunto de imagens faciais dos vários jogadores envolvidos, duas imagens que representem os equipamentos das duas equipas e a localização, na imagem do vídeo, do placard com o resultado e tempo de jogo.

Estes *inputs* são o mínimo necessário para que na fase de processamento seja possível obter resultados extremamente precisos, para além de permitirem por exemplo, atribuir a cada jogador detetado um nome e uma equipa, como a cada momento um resultado e um tempo de jogo, ou a cada câmara um tipo e ainda determinar que zona do campo está a ser mostrada a cada segundo.

Pretendeu-se dar ao utilizador, a partir de uma interface criada utilizando a *framework Qt*, uma maneira de interagir com todas as funcionalidades da aplicação. Poderá gerar novos processamentos, visualizar e extrair toda a informação gerada.

Existe a possibilidade de o utilizador criar a sua própria aplicação independente do sistema e utilizar os ficheiros gerados como uma base de dados, e realizar as diversas aplicações que entender, com base na informação obtida.

Como referido, todos os dados são guardados em ficheiros, ficheiros XML, pois trata-se de uma tecnologia universal e bem estabelecida.

Foram também desenvolvidas *queries* pré-definidas, com o objetivo de demonstrar uma possível aplicação dos dados gerados, onde o resultado obtido poderá ser também extraído para um ficheiro XML.

Em suma, com o sistema desenvolvido permite realizar um processamento de imagem global sobre um determinado jogo gerando uma base de dados, à qual poderão ser executadas um conjunto de *queries* (através da interface criada e apresentada no subcapítulo 3.6), permitindo assim filtrar toda a informação gerada.

Ao juntar-se as funcionalidades referidas com o projeto anteriormente desenvolvido [1], de processamento de áudio e emoções, é possível desenvolver um sistema que aborda todas as componentes presentes num vídeo de um jogo de futebol completando assim a componente em falta no projeto anterior.

3.2 Arquitetura do Sistema

Nesta secção serão apresentados um conjunto de diagramas UML, de forma a clarificar a organização e arquitetura da solução desenvolvida, tendo em conta o problema e objetivos propostos.

3.2.1 Diagramas de componentes

Como referido anteriormente, este projeto vem em continuação de uma tese anteriormente desenvolvida [1]. No trabalho até agora desenvolvido foram abordadas as componentes relativas às emoções e som que se verificam durante um jogo de futebol, como se pode observar na Figura 3.1 (componentes a vermelho).

Com o trabalho desenvolvido pretende-se analisar também a componente da imagem, extraindo um conjunto de informações visuais relevantes que ocorrem durante um jogo de futebol.

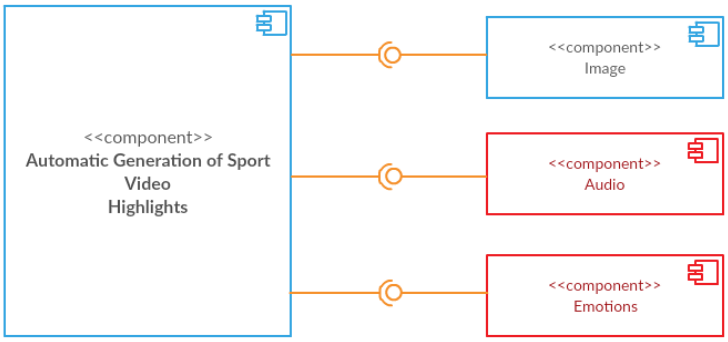


Figura 3.1 – Diagrama de componentes do sistema, visão geral.

De forma a explorar a componente da imagem foram selecionadas cinco características relevantes, que se conseguissem extrair a partir de cada um dos *frames*. No diagrama de componentes, Figura 3.2, é possível verificar o modelo de arquitetura, escolhido na implementação dos algoritmos desenvolvidos.

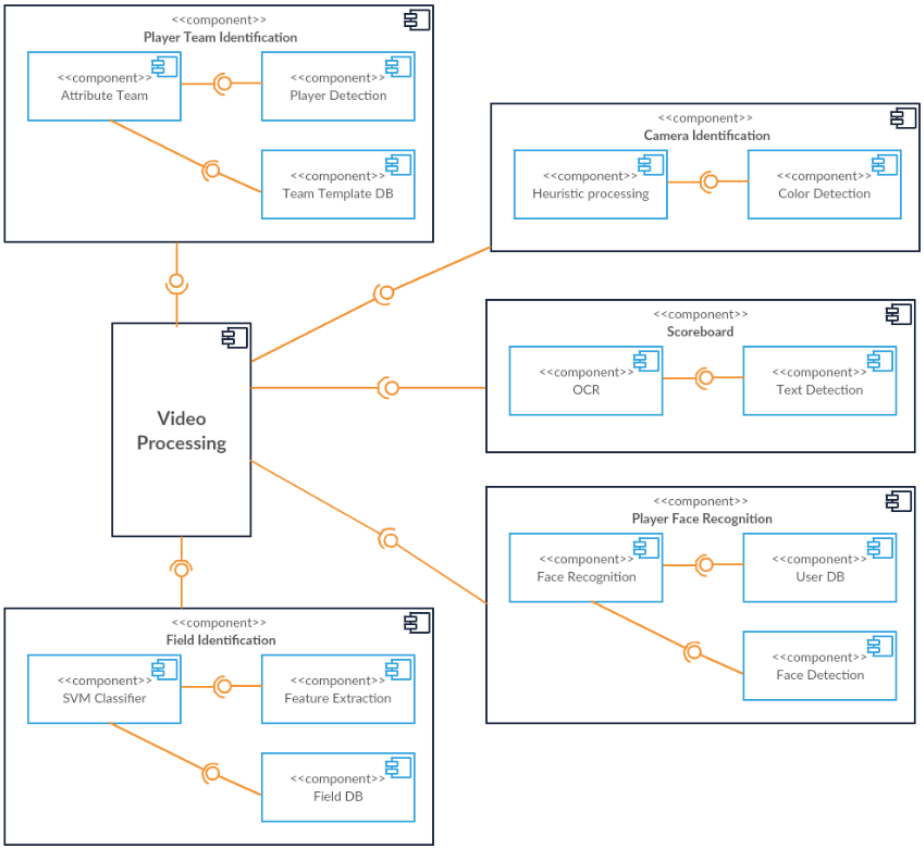


Figura 3.2- Diagrama de componentes dos vários algoritmos implementados.

3.3.2 Casos de Uso

O diagrama de casos de uso descreve as várias funcionalidades do sistema, focando-se principalmente na perspetiva do utilizador. Deverão ser descritas as interações entre o utilizador e o sistema, bem como entre os vários componentes. No projeto desenvolvido destacam-se os seguintes casos de uso:

Search Frame

O utilizador do sistema pretende visualizar a informação de um determinado *frame* e, para pesquisa pelo id do *frame* e após o sistema processar o pedido disponibiliza ao utilizador a informação extraída do *frame* pretendido.

Submit Video, Submit Players DB, Submit Team Templates

O utilizador pretende processar um vídeo, e para tal, necessita de inserir um conjunto de informações necessárias para o processamento ser possível, ou seja, o vídeo do jogo a ser analisado, as faces dos jogadores presentes e duas imagens que representem o equipamento de cada equipa.

Get Global XML File, Get Query XML File

Após a conclusão do processamento global de um vídeo onde se processa cada *frame* documentando a mesma, ou no caso de uma *query* ao sistema, o utilizador poderá entender extrair essa informação para um ficheiro, de modo a ser possível utilizar tal informação noutro projeto ou sistema (daí o mecanismo de extração para ficheiros XML).

See Field Distribution

O utilizador consegue visualizar através da distribuição de pontos sobre uma imagem dum campo de futebol os valores em percentagem, do número de vezes com que cada uma das 5 zonas foi detetada ao longo do vídeo processado.

Search Query

Para além da informação geral que é fornecida ao utilizador, este poderá querer certos pontos mais específicos, por exemplo só visualizar *frames* que mostrem o público, por isso executará uma *query* através de elementos da interface para atingir esse objetivo.

Add New Field Images

De modo a melhorar a performance do classificador das zonas do campo o administrador poderá querer incluir mais dados, neste caso, imagens.

Replace Field Zone Classifier

Após a inclusão de novas imagens é necessário gerar um novo classificador e, para tal, remove-se o classificador até então utilizado, permitindo assim ao sistema gerar uma nova versão.

Change Face Recognizer

No sistema, através de mudanças ao nível do código, existe a possibilidade de escolher o tipo de identificador facial a utilizar. Com o objetivo de testar a performance para uma determinada situação, o administrador poderá escolher o que for mais conveniente.

Process Each Frame

Como sistema pretende-se que seja possível para cada frame extrair um conjunto de informações definidas e relevantes para o projeto em questão.

Save Results

Pretende-se que exista um mecanismo, a ser utilizado pelo serviço quando solicitado pelo utilizador, que guarde toda a informação extraída pela aplicação, não sendo, portanto, necessário processar o vídeo sempre que se executa o programa.

Display UI, UI Interaction

É necessário fornecer um ambiente visual para que o utilizador e administrador possam visualizar e interagir com o sistema, daí o serviço fornecer tais funcionalidades.

Read/Write Classifiers

Sempre que existe a entrada de novos dados por parte do administrador é necessário criar classificadores com os novos dados. Por outro lado, se não existirem novos dados é necessário carregar os classificadores com a informação existente.

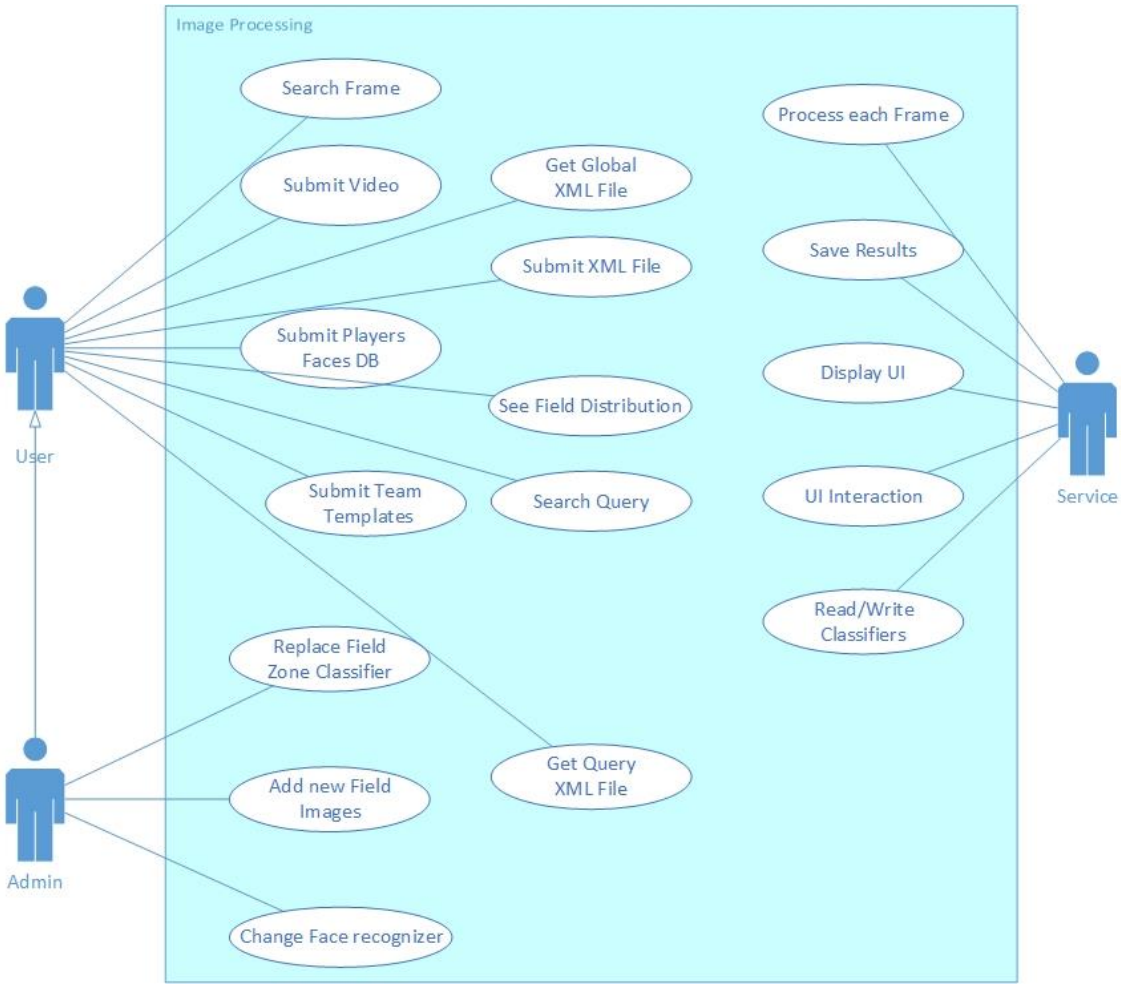


Figura 3.3- Diagrama de casos de uso da aplicação desenvolvida.

3.2.3 Diagrama de Classes

Com o diagrama de classes pretende-se demonstrar como foi elaborada a arquitetura do projeto em termos de código, neste caso, em termos de estrutura para a tecnologia C++ (Figura 3.4).

Frame.cpp, OperationsUtils.cpp, Location.cpp

Classes auxiliares responsáveis por métodos e objetos auxiliares criados ao longo do projeto.

FrameInfo.cpp, CloseUpFrame.cpp, LongMidFrame.cpp, OutOfField.cpp

Classes objeto dos 3 tipos de *frame* que estendem a classe *FrameInfo*, responsáveis por armazenar as várias informações extraídas para cada tipo de *frame*.

FieldFeatures.cpp

Classe responsável pela atribuição de um tipo de câmara a cada *frame*, bem como pela deteção automática do tipo de verde dominante no vídeo em análise.

SVMClassifier.cpp

Classe responsável pela criação do classificador SVM que deteta a zona do campo para uma determinada imagem, bem como por extrair as várias características da base de dados de imagens utilizadas pelo classificador e extratores.

PlayersFeatures.cpp

Classe responsável pela deteção facial e identificação de jogadores, bem como a distinção e atribuição de uma equipa para todos os elementos detetados num determinado *frame*.

Scoreboard.cpp

Classe responsável pela extração, reconhecimento e identificação textual dos placards com o tempo e resultado.

FrameFeatures.cpp

Classe responsável pela divisão da imagem necessária para a classificação do tipo de câmara a atribuir.

FeaturesExtraction.cpp

Classe objeto com os vários métodos necessários para extrair informação de cada *frame* associada a cada um dos cinco componentes desenvolvidos.

ProcessVideo.cpp

Classe responsável pelo processamento global, ou seja, processa cada *frame* e gera o seu respetivo objeto com toda a informação associada. Responsável por fazer todo o processamento necessário, de modo a produzir o ficheiro XML final.

XMLParser.cpp

Classe responsável pela leitura e escrita dos vários ficheiros XML utilizados ao longo do projeto.

UIDataConnect.cpp

Classe responsável por gerar e iterar os dados necessários a ser utilizados pelas várias componentes da interface.

MainWindow.cpp

Classe responsável pela criação e interação dos vários componentes da interface visual desenvolvida.

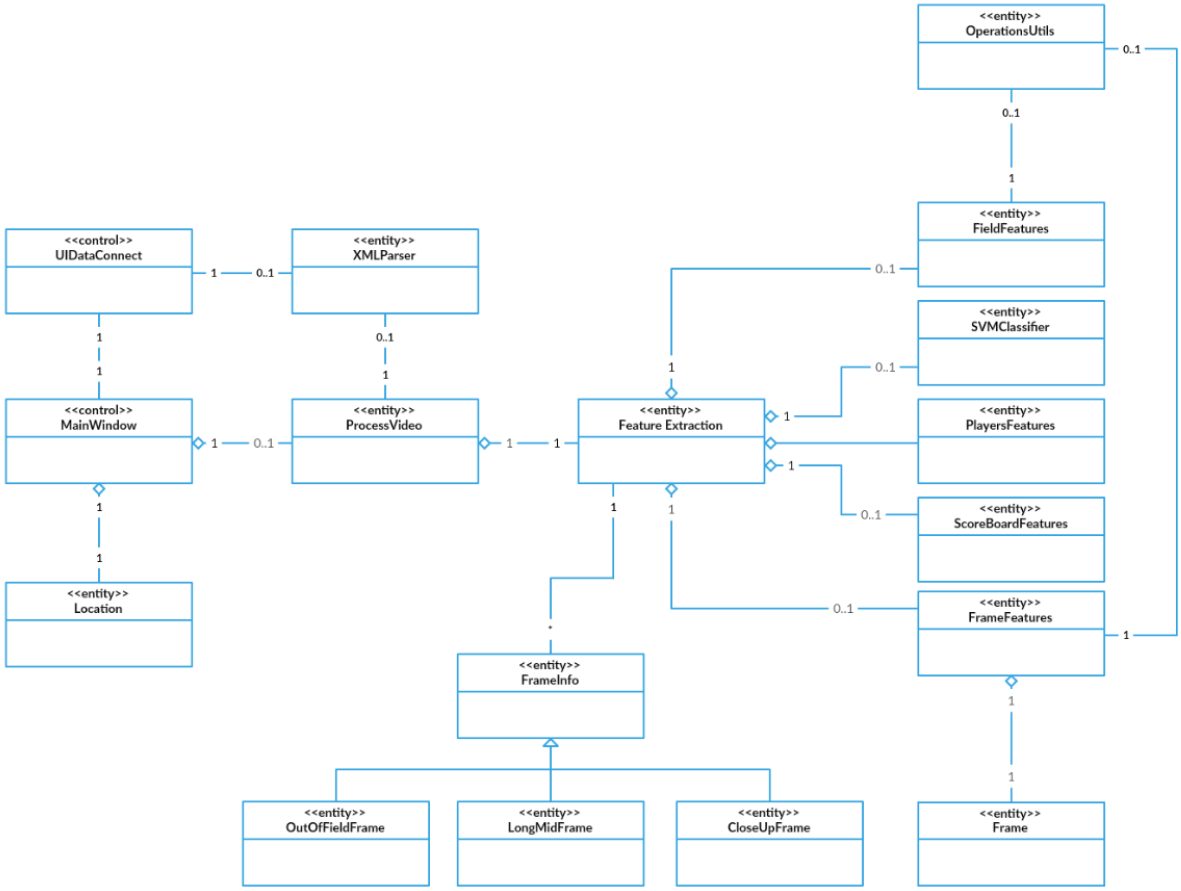


Figura 3.4- Diagrama de classes C++ do projeto desenvolvido.

3.3 Algoritmos

Neste tópico serão abordados os vários algoritmos desenvolvidos para a extração de informação, a partir dos vários *frames* dum jogo de futebol. Foram selecionados 5 algoritmos que se demonstraram mais eficazes e que poderiam ser aplicados em qualquer contexto durante um jogo, independente de luz, tipo de relvado ou jogadores envolvidos.

3.3.1 *Camera Shot*

Descrição

Durante a transmissão de um evento desportivo existem vários planos de imagem mostrados aos espectadores. Mais concretamente, num jogo de futebol muitas vezes são mostrados planos de jogadas, bem como planos dos vários elementos do jogo, como treinadores, jogadores e público. Ou seja, existe um conjunto de informação relevante que seria interessante extrair a partir de cada *frame* que está a ser mostrado ao espectador.

Tendo em conta essas características, decidiu-se criar um algoritmo de classificação do tipo de câmara para cada *frame*. Foram estabelecidas três categorias com características diferentes.

- *Long/Mid Shot* - caracteriza-se por mostrar ao espectador um panorama geral do campo ou de uma determinada jogada. Permite acompanhar todas as movimentações dos jogadores envolvidos.
- *Close-up Shot* - *frame* caracterizado por um momento de zoom a um jogador, ou grupo de jogadores, tendo como objetivo mostrar ao espectador as suas reações perante um determinado lance ou momento do jogo.
- *Out-of-field Shot* - pretende-se mostrar as reações que o público ou os bancos das equipas estão a ter durante o jogo.

Após estabelecidas as categorias, foi necessário encontrar uma característica presente na imagem que as diferenciasse facilmente. Foi então escolhido a percentagem de relvado presente em cada *frame*, no ponto de vista de processamento de imagem está a ser analisado a percentagem de verde presente em cada *frame*, pois, por regra, todos os relvados de jogos de futebol apresentam cores semelhantes.

Com o cálculo, em percentagem, de verde presente, tornou-se possível atribuir uma categoria a cada *frame* analisado, Figura 3.5. Por exemplo, lances *Out-of-Field* são caracterizados por conter uma percentagem de verde muito baixa, entre os 0% e 5%, o que os distingue facilmente dos restantes.

No caso de *Long/Mid Shot* e *Close-up*, foram necessários mais alguns cálculos para além da percentagem geral de verde. Nesses casos decidiu-se fazer uma divisão da imagem em 3:5:3, Figura 3.6 b), dividindo assim cada *frame* em 9 zonas pois, normalmente, quando ocorre um momento de *Close-up,* os jogadores em questão estão centrados no ecrã, ou então ocupam um grande espaço vertical, causando uma grande obstrução do relvado.

No caso de *Long/Mid Shot,* a parte de cima do ecrã está ocupada com o público onde não existem zonas verdes, sendo possível detetar facilmente através da divisão da imagem.

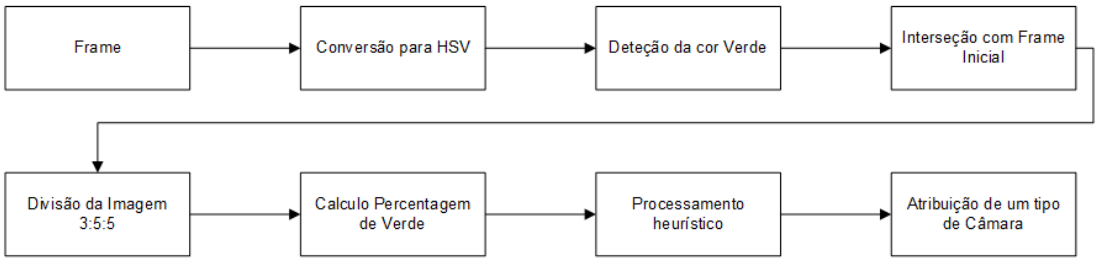


Figura 3.5 – Esquema da implementação do algoritmo de atribuição do tipo de câmara.

O algoritmo implementado, utiliza como base os conceitos e trabalhos desenvolvidos, apresentados na secção 2.5.1.1.

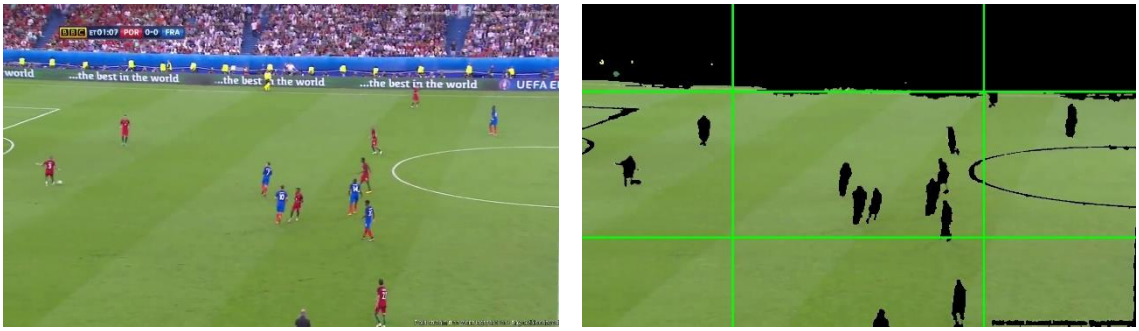
Implementação

Relativamente à implementação realizada, como pré-requisitos do processo é necessário ter um *frame* do jogo, num esquema de cores RGB ou HSV. O *frame* a analisar é convertido para um esquema de cores HSV (caso ainda não esteja nesse formato), pois trata-se de um modelo mais adequado para a deteção de uma determinada cor, uma vez que apenas é necessário variar em duas componentes, H e S, em comparação com o modelo RGB onde é necessário trabalhar com as três componentes. Após a conversão do esquema de cores, são definidos intervalos para o *Hue*, e *Saturation* que caracterizam a cor verde. Neste caso, convém estabelecer parâmetros corretos para a deteção funcionar independentemente da presença

de sombras ou diferenças de luminosidade. No caso do problema em questão, os melhores resultados foram obtidos com os intervalos [35;80] para o *Hue* e [90,255] para *Saturation*. Porém, tendo em conta que cada jogo poderá ser jogado a horas diferentes e em locais variados, onde as cores do relvado podem variar, foi implementado um mecanismo de deteção de cor automático, ou seja, após se detetar que um *frame* é do tipo *Long/Mid Shot*, utilizando os valores anteriormente definidos, é calculado o histograma de cores nesse *frame* e extraído o valor de verde dominante. Tendo em conta esse valor calculado, o intervalo de [35;80] é alterado, tornando assim o processo de deteção de cor adaptável à tonalidade de verde dominante presente no vídeo em análise.

Com os intervalos de cor estabelecidos pode então aplicar-se a função de deteção de cor, no caso da biblioteca OpenCv *inRange*, seguida dum processo de erosão, removendo alguns pontos não relevantes para o processamento. Como resultado obtém-se uma matriz binária onde os pontos que estiverem dentro dos intervalos, neste caso forem verdes ou similares, são considerados 1 e os restantes 0. De seguida, realiza-se um processo de *bitwise_and*, entre a imagem original e a máscara gerada pela função *inRange* e erosão, transformando, na imagem original, os pixéis não verdes em 0, Figura 3.6 b).

Após esse processamento é executada a divisão da imagem 3:5:3 e calculadas para cada zona da divisão as respetivas percentagens de pontos com valores diferentes de 0. A partir desses valores é aplicado um processo heurístico com o fim de atribuir uma categoria ao *frame* analisado.



(a) (b)

Figura 3.6 – (a) *Frame* original a processar, (b) *Frame* após a deteção do relvado onde é possível ver a divisão 3:5:3 efetuada.

3.3.2 *Field Location*

Descrição

Identificar em que zona do campo estão jogadores ou uma determinada jogada está a ocorrer é uma característica interessante a ser analisada do ponto de vista deste projeto. Ou seja, saber que num determinado instante que o foco do jogo está, numa das áreas do campo, poderá ajudar a compreender os vários acontecimentos ao longo do jogo.

Tendo em conta esse objetivo, decidiu-se implementar um processo de identificação de zonas do campo. Utilizou-se uma divisão do campo em 5 zonas, como está presente na Figura 3.7. No algoritmo estudado na subsecção 2.5.1.2, são utilizadas 12 zonas diferentes, porém no projeto desenvolvido, não houve necessidade de criar tantas divisões no campo, pois não se perdia informação relevante para o que se pretendida fazer neste projeto.

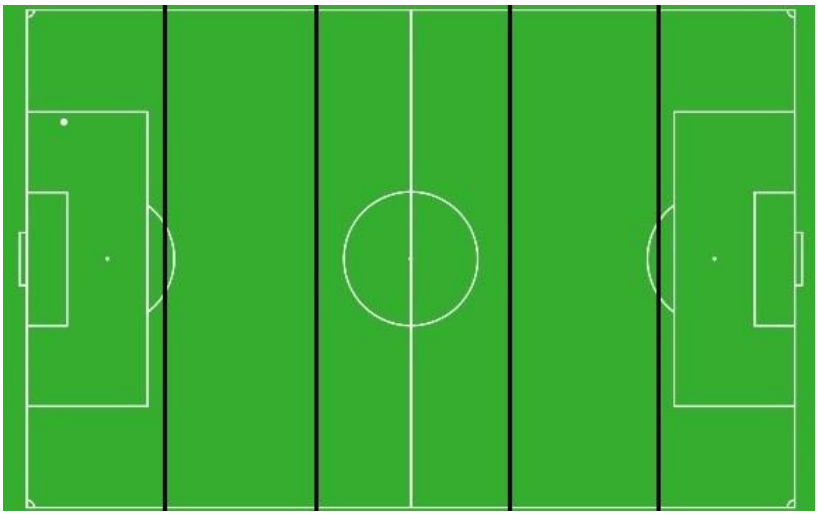


Figura 3.7 – Divisão do campo efetuada para o algoritmo apresentado.

Para realizar a identificação da zona do campo decidiu-se trabalhar apenas com *frames* do tipo de *Long/Mid Shot*, onde é possível obter uma visão mais alargada do campo permitindo atribuir uma das cinco zonas, enquanto nos outros dois tipos (*Close-up* e *Out-of-Field*) não existe informação suficiente que permita fazer uma correta atribuição.

Após estabelecido o tipo de *frames* que podem ser alvos de avaliação, numa primeira abordagem, decidiu-se estudar a deteção e localização das linhas e semi-

círculos do campo, utilizando técnicas como transformação de Hough, combinada com um processo heurístico atribuindo uma determinada zona.

Porém, este procedimento levantou várias questões na fase de avaliação, pois existia um grande número de falsos-positivos. Fatores como a quantidade de zoom presente no *frame* em questão, apesar de ser *Long/Mid Shot*, e a dificuldade de detetar os semicírculos presentes no campo levaram a maus resultados. Para além disso, o processo heurístico também não seria o mais indicado, pois nesta situação, esperava-se algo mais flexível e não tão fixo como uma heurística. Tendo em conta estes problemas esta primeira abordagem foi deixada de parte.

Numa segunda abordagem, Figura 3.8, decidiu-se utilizar um processo de *Machine Learning,* mais concretamente a um classificador *Support Vector Machine* (SVM), onde para uma dada imagem seria possível obter a respetiva zona do campo. Para tal foi necessário recolher um total de 250 imagens, 50 para cada zona do campo como base de treino para o classificador.

Num primeiro momento, optou-se por simplesmente carregar as imagens em formato RGB para o classificador e posteriormente fornecer também a imagem a classificar no formato também RGB. Este processo também se revelou pouco eficaz pois, para além de lento e muito dispendioso, não apresentava os melhores resultados.

Posteriormente verificou-se que, em vez de se carregar as imagens na íntegra para o classificador, seria mais interessante e com melhor performance, carregar sim características que representassem as imagens, combinou-se um processo de *Bag of Words,* onde o vocabulário utilizado foi gerado com o auxilio dum extrator de *keypoints* do tipo *Surf*, os resultados gerados foram carregados como base de treino para o classificador SVM.

O dicionário gerado pelo *Bag of Words* foi indicado como base de treino para o classificador SVM, substituindo as imagens anteriormente carregadas. Para além deste processo ser mais rápido revelou-se o mais eficaz sendo escolhido para realizar a atribuição da zona do campo.

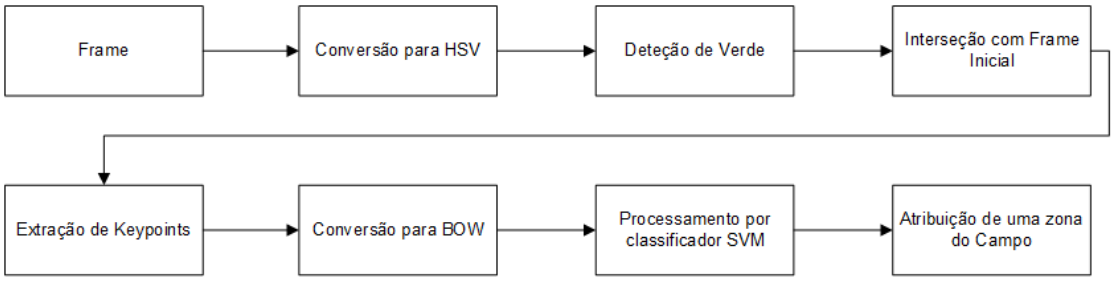


Figura 3.8 - Esquema da implementação do algoritmo de localização do campo.

Implementação

Como pré-requisitos deste processo é necessário existir uma base de dados para o classificador. Neste caso, como já foi referido existem 250 imagens, 20 para cada uma das 5 zonas do campo. A cada uma dessas imagens é aplicado um processo de deteção de verde semelhante ao processo de deteção do tipo de câmara, pois elementos como público ou jogadores não são relevantes para a identificação das zonas, logo podem ser agrupados, neste caso passam a ter valor 0 nos seus pixéis.

Posteriormente extrai-se os *keypoints* da imagem recorrendo ao detetor *SURF*, que por sua vez são inseridos para o descritor de *Bag of Words*. Após este procedimento ser aplicado a todas as imagens base é possível gerar o chamado dicionário que caracteriza cada uma das zonas.

Com o dicionário gerado é fornecido como base de treino para o classificador SVM, juntamente com a informação das *labels*, sendo esta uma matriz com os valores de entrada correspondentes a uma zona do campo numerada de 1 a 5. Como o processo é sequencial existe uma relação direta entre as *labels* e as entradas do dicionário.

Em termos de configuração dos parâmetros do SVM utilizou-se os valores por defeito.

Com o SVM carregado, utilizando a função *SVM.train*, poderá então iniciar-se o processo de previsão, no OpenCV *SVM.predict*, para qualquer *frame* desejado, para tal é necessário verificar se o frame é um *Long/Mid Shot* e posteriormente aplicar o processo de deteção de características similar ao aplicado aquando do carregamento de dados para o SVM.

De referir que o processo de geração de dados das imagens base só e realizado uma vez, e o seu resultado é guardado num conjunto de ficheiros que são carre-

gados quando se pretende utilizar o classificador, reduzindo assim o volume e tempo de processamento até ser possível obter um resultado concreto.

3.3.3 *Player Team Recognition*

Descrição

Dos vários elementos presentes num jogo de futebol, os jogadores são talvez os elementos de maior importância. Do ponto de vista deste projeto, analisar o número de jogadores de cada equipa, presente num determinado *frame*, poderia ser uma informação interessante a extrair.

Com base nesse objetivo, decidiu-se implementar um algoritmo com o intuito de detetar os vários jogadores presentes no campo e atribuir a sua respetiva equipa, em *frames* do tipo *Long/Mid Shot*, Figura 3.9.

Numa primeira abordagem decidiu-se utilizar um detetor e classificador de pessoas, em OpenCV chamado *Pedestrian Detection*, utilizado para tentar identificar os vários jogadores presentes num determinado *frame*. Só que rapidamente essa abordagem falhou, pois, o tamanho dos jogadores presentes no *frame*, é muito reduzido para ser possível obter resultados consistentes e corretos.

Após uma primeira abordagem falhada, optou-se por um outro procedimento em que para cada *frame* do tipo *Long/Mid Shot* são detetados os vários possíveis jogadores, através da combinação do processo de deteção do relvado, já apresentado, com uma deteção de contornos.

Esta combinação é possível pois com o resultado proveniente da deteção do relvado os vários elementos do jogo ficam assinalados no *frame* resultante como vultos, Figura 3.10 a). Ao aplicar uma deteção de contornos (bem parametrizada), torna-se possível obter os contornos dos corpos dos jogadores e as suas respetivas posições no campo. Os vários elementos do público e *staff* como não estão sobre o relvado, não interferem no processo de deteção, exceto nalguns casos particulares.

Após se obter os vários jogadores, é aplicado um processo de comparação de histogramas com a finalidade de agrupar os vários jogadores em 2 ou 3 equipas, no caso de aparecer o árbitro ou elementos do *staff*. Com as divisões feitas realiza-se uma comparação entre um *template* do equipamento de cada equipa e os vários grupos com a finalidade de atribuir um nome de equipa a cada agrupamento.

De referir que existem algumas situações onde este processo poderá falhar, por exemplo, situações onde um jogador aparece numa das extremidades do campo e junto aos adeptos, nesse caso é confundido como parte das claques. Situações onde os jogadores estão todos muito próximos uns dos outros, tornando-se muito difícil só com um plano de imagem distinguir os vários elementos presentes. E por fim, lances onde apesar de ser um *frame Long/Mid Shot* a câmara apresentada tem um grande zoom sobre o relvado, que leva a que certas linhas sejam detetadas como jogadores. Porém, tirando estas situações referidas, o conceito base do algoritmo apresenta um bom grau de sucesso, como será discutido na secção 4 deste relatório.

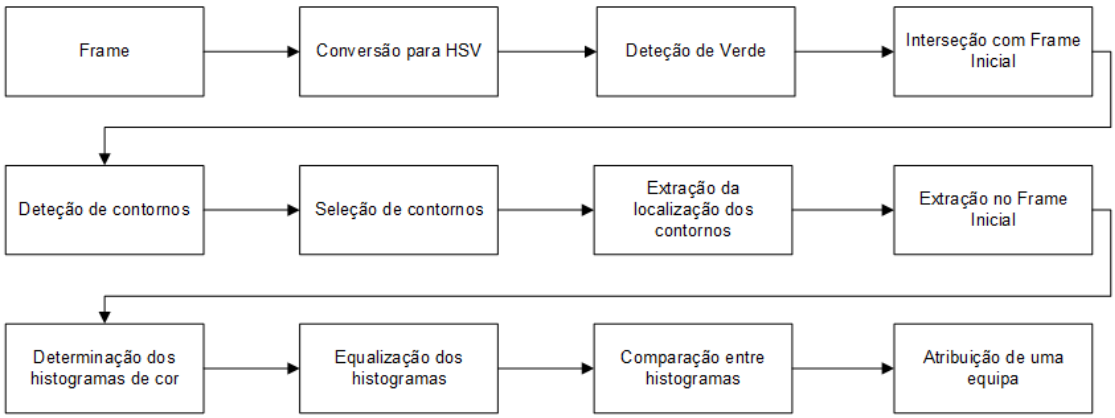


Figura 3.9 - Esquema da implementação do algoritmo de identificação da equipa dos vários jogadores e árbitros.

O algoritmo apresentado foi idealizado, desenhado e implementado de raiz no âmbito deste trabalho de Mestrado.

Implementação

Como foi referido anteriormente, este algoritmo aplica-se a *frames* do tipo *Long/Mid Shot*s. Após aplicar-se o processo apresentado na subsecção 3.3.1 e tendo determinado a categoria da imagem resultante, é aplicado um processo de *blur* e dilatação, no OpenCv *GaussianBlur* e *dilate* respetivamente, de modo a ser possível obter contornos mais definidos. Isto porque todos os pontos não verdes possuem valores 0 ou seja existem formas que representam os jogadores de campo que podem ser detetadas.

Para a deteção de contornos utilizou-se a função *findContours*, a partir da qual se obteve como resultado um vetor de estruturas *Rect*, que posteriormente

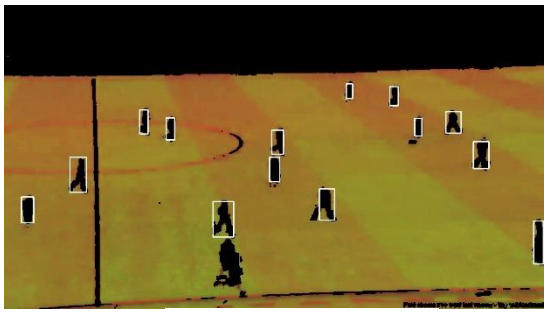
foram alvo de um processamento analítico com a finalidade de remover duplicados e falsas deteções, através da verificação das dimensões e área ocupada pelos vários *Rect* detetados.

Com os valores resultantes desse processamento, foi feita uma extração do *frame* inicial na localização dos vários *Rect*, de modo a obter um objeto *Mat* com as cores originais de cada jogador. Para cada *Mat* foi calculado o seu histograma RGB, função *calcHist* em OpenCV e aplicado um processo de equalização, função *equalizeHist*. Com todos os histogramas calculados foi então realizada uma comparação entre os vários histogramas, *compareHist* em OpenCV, com a finalidade de agrupar os vários jogadores, tendo em conta uma possível divisão por equipas.

Após os grupos estarem definidos, é feita novamente uma comparação entre duas imagens que representam os equipamentos de cada uma das equipas e os vários grupos detetados. Com isto torna-se possível atribuir uma equipa a cada grupo de jogadores. Posteriormente agrupam-se em 2 ou 3 grupos, no caso de haver jogadores das duas equipas e outros elementos como árbitros e *staff*, que partilhem o mesmo nome de equipa, obtendo no final o total de jogadores correspondentes a cada equipa.

Para os árbitros ou situações de erros na deteção, é estabelecido um *threshold,* determinado através do experimento e adaptação, para que um jogador possa ser atribuído a uma determinada equipa. Elementos que não atinjam esse limite são atribuídos à categoria *Staff/Referee/Missmatch*.

(a)



(b)

Figura 3.10 – (a) *Frame* a processar pelo algoritmo, (b) resultado parcial do algoritmo de identificação dos jogadores.

3.3.4 *Player Face Identification*

Descrição

Seria interessante para o contexto deste projeto saber quais os jogadores que aparecem num determinado momento do jogo. Por isso foi decidido analisar e criar métodos necessários para cumprir esse objetivo, Figura 3.11.

Numa primeira abordagem pretendeu-se detetar os jogadores presentes em todo o tipo de *frames*, porém rapidamente foi percetível que tal não seria fiável pois em momentos em que a câmara de jogo se encontra muito distante do campo é impossível realizar qualquer processo de identificação facial. Por isso, ficou definido aplicar este processo só a *frames* do tipo *Close-up* onde será possível obter melhores e mais fiáveis resultados pois as faces dos jogadores são mais percetíveis.

Porém, tendo em conta o público presente nos vários estádios durante os jogos de futebol, existem algumas situações onde o reconhecimento facial poderá falhar, pois poderá identificar um elemento do público como sendo um possível jogador por aparecer na imagem quando ocorre um *Close-up*. Ou seja, apesar de ser uma abordagem interessante e importante para o objetivo deste projeto existem algumas questões que podem dificultar a performance do algoritmo.

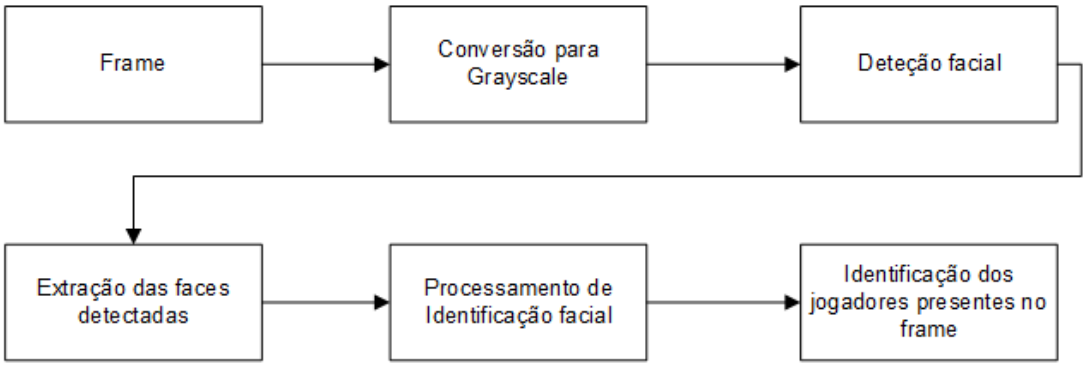


Figura 3.11 - Esquema da implementação do algoritmo de identificação facial dos vários jogadores.

Implementação

Em termos de implementação da deteção facial, como parâmetros iniciais é necessário restringir os tipos de *frame* a somente *Close-up Shot*. Após essa restrição, cada *frame* é convertido para esquema de cores *Grayscale* e redimensionado

para 75% do seu tamanho original, para efeitos de performance. Posteriormente é utilizado um classificador facial, *CascadeClassifier* no OpenCV, utilizando o ficheiro XML *haarcascade_frontalface_alt2.xml* fornecido também pela biblioteca OpenCV, para detetar as várias faces presentes no *frame* em análise.

Numa fase inicial do desenvolvimento, ainda foi considerada e testada uma implementação que incluía também a deteção de faces de perfil, utilizando o ficheiro "haarcascade_profileface.xml" e um novo *CascadeClassifier* com esse XML, só que os resultados não se verificaram melhores com a introdução dum grande número de falsos positivos o que levou a só se implementar o modelo de deteção facial frontal.

Após obter-se o resultado, todas as faces detetadas são transmitidas para um processo de identificação facial.

Para ser possível executar este processo corretamente é necessário criar uma base de dados com as faces e nomes dos jogadores que participam no jogo em análise. Foram recolhidas 4 imagens faciais de cada jogador, que posteriormente foram carregadas para um modelo de identificação facial, juntamente com o nome do jogador associado, OpenCV *FaceRecognition*. Em termos do algoritmo utilizado pelo modelo, foram testados os três algoritmos mais utilizados, *LBPHFaceRecognizer*, *FisherFaceRecognizer* e *EigenFaceRecognizer*, sendo o algoritmo de Fisher o escolhido por apresentar melhores resultados na identificação das faces dos jogadores. No entanto as 3 implementações podem ser executadas no projeto.

No final do processamento para cada *frame Close-up Shot* foi possível obter o nome de cada jogador presente, porém existe sempre a possibilidade de um falso positivo como foi referido anteriormente.

Num cenário ideal, seria possível obter os melhores resultados se não existisse a presença do público no fundo dos *frames*, removendo assim muito "ruído" da imagem, e se existissem mais câmaras que pudessem ser analisadas pois nem sempre é possível obter a face do jogador com o ângulo ideal para executar a deteção.

(a) (b)

Figura 3.12 – (a) *Frame* a processar pelo algoritmo facial, (b) Faces detetadas no *frame* processado em *grayscale*.

3.3.5 *ScoreBoard Processing*

Descrição

O placard com o resultado e o tempo corrente de jogo é um elemento importante para quem está a visualizar um evento desportivo. A partir desse placard é possível saber qual o resultado num determinado momento, quais são as equipas que estão a jogar e também saber em que ponto do jogo se encontra, através do tempo mostrado.

No âmbito deste trabalho, torna-se importante analisar a informação presente nesse tipo de placard. No caso dos jogos de futebol, por exemplo, é possível saber quando ocorreu um lance de golo, basta verificar quando ocorreu uma mudança na componente do resultado, ou seja, o momento em que o resultado passa de 0-0 para 1-0, sabendo assim que ocorreu um lance de golo. Outra abordagem, passaria por associar o tempo do jogo corrente ao tempo corrente do vídeo em análise. Normalmente existem diferenças entre os dois tempos, atribuindo assim para cada *frame* os dois tempos diferentes.

Tendo em conta estas situações foi decidido realizar uma análise ao placard de tempo e resultado (Figura 3.13).

Para ser possível fazer uma correta extração da informação presente nesses tipos de placards, utilizou-se ferramentas de segmentação de imagem e de reconhecimento textual, *Tesseract OCR* fornecido em *opensource* pela Google.

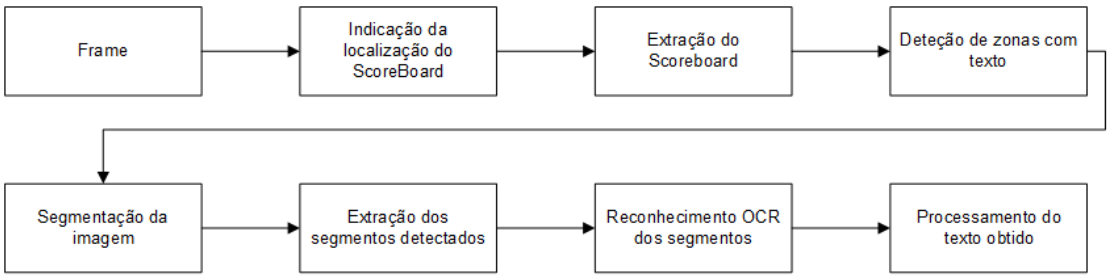


Figura 3.13 - Esquema da implementação do algoritmo de reconhecimento textual OCR implementado.

Implementação

De modo a ser possível realizar o processamento é necessário fornecer a localização do placard em cada *frame* do vídeo analisado. Como a sua localização, de forma geral, não muda ao longo do vídeo, exceto certas situações em que ocorre a sua omissão, por exemplo, lances de repetição, basta indicar uma vez a sua posição e aplicar essas coordenadas aos restantes *frames* a analisar.

Numa fase inicial ainda foi discutido realizar uma deteção automatizada, só que devido ao grande volume de painéis publicitários existentes, em redor do campo desportivo, o número de falsos positivos poderia ser muito elevado. Por isso decidiu-se recorrer a um *input* por parte do utilizador, onde este fixa a localização do placard.

Após a indicação da posição é feita uma extração do *frame* em análise obtendo numa estrutura *Mat* do OpenCV o conteúdo do placard. É então necessário aplicar um conjunto de processos de segmentação da imagem, de modo a separar os vários componentes textuais, facilitando assim o processamento realizado pelo OCR.

Após obter o *Mat* com o placard, são realizadas um conjunto de funções de transformação morfológica, no OpenCV morphologyEx, com *kernels* adaptados ao tamanho da imagem, seguindo-se de um processo de *threshold*, função *threshold* em OpenCV, removendo algo ruído causado por brilho ou iluminação presente na imagem.

Posteriormente é feita uma deteção de contornos, função *findContours*, detetando as possíveis localizações de texto (Figura 3.14 b)). Os resultados provenientes dessa deteção são posteriormente processados de forma heurística: se o contorno detetado estiver preenchido mais que 45% então trata-se de uma componente textual.

Cada um dos contornos detetados como possível texto é então extraído da imagem inicial e transmitido para uma função de reconhecimento textual, utilizando a ferramenta Tesseract OCR. Esta função, que recebe o contorno sobre a forma de *Mat*, converte essa matriz para *grayscale*, e executa uma função de *threshold*, removendo qualquer ruído ainda existente na imagem. Posteriormente é então aplicado um processo de reconhecimento utilizando o *Tesseract OCR* para a deteção de palavras.

Após se processar cada contorno detetado, é possível obter sob a forma de *array* de *strings* os valores resultantes do OCR presentes no placard. Esses resultados são alvo de um processo de avaliação para determinar se o placard estava efetivamente presente na imagem ou era omisso e, no caso de efetivamente estar a ser mostrado aos espetadores, separar a componente resultado e tempo.

Numa primeira fase deteta-se se o placard que está a ser ou não mostrado, faz-se uma pesquisa no array de *strings* pelo caracter ':', pois é um caracter que se encontra sempre presente na componente do tempo. Se nenhuma *string* contiver esse caracter então o placard encontra-se omisso, caso contrário o placard está presente no *frame* podendo proceder-se à análise seguinte. Nesta fase, após obter-se a localização do caracter, essa *string* é distinguida como contendo a componente do tempo, as restantes são atribuídas à componente do resultado. De modo a obter uma componente textual idêntica ao que aparece no *frame*, é feita uma ordenação das *strings* tendo em conta a posição horizontal de cada contorno que estão associadas.

No final torna-se possível extrair a informação presente no placard caso este esteja presente no *frame* em análise.

(a)

(b)

(c)

Figura 3.14 – (a) Resultado a processar, (b) segmentação da imagem, (c) conversão da imagem para binário.

3.4 XML

Após a conclusão do processamento dos vários algoritmos sobre os diversos *frames* do vídeo, os resultados obtidos são guardados num ficheiro XML onde é possível observar para cada *frame* os vários parâmetros calculados.

À semelhança das estruturas de dados utilizadas, os parâmetros de cada *frame* variam consoante o tipo de câmara *Shot* atribuído. Enquanto em *frames Out-of-field* existe um volume de informação reduzida que pode ser extraída, o mesmo não acontece nos outros dois tipos de frames, *Close-up* e *Long/Mid Shot*.

É a partir do XML gerado, Figura 3.15, por defeito designado *Result_XXX.xml*, sendo XXX um número aleatório, que posteriormente são realizadas as várias *queries* permitidas por este projeto. Assim evita-se ter de se processar o vídeo sempre que se executa uma *query* nova.

A estrutura do ficheiro pode ser encontrada na Figura 3.15, onde se pode ter uma noção de como foi mapeado cada objeto para o ficheiro XML.

```
<? xml version="1.0" encoding="utf-8"?>
  <video>
            <frame>
            ...
            </frame>
            <frame>
                        <id>92</id>
                        <time>46</time>
                        <gameTime>ET00:43</gameTime>
                        <gameResult>POR 0-0 FRA </gameResult>
                        <type>2</type>
                        <location>4</location>
                        <numberOfPlayers>
                                    <team>5</team>
                                    <team>4</team>
                                    <team>1</team>
                        </numberOfPlayers>
                        <teamNames>
                                    <teamN>Portugal</teamN>
                                    <teamN>France</teamN>
                                    <teamN>Referee/Staff/Missmatch</teamN>
                        </teamNames>
            </frame>
            <frame>
                        <id>93</id>
                        <time>46</time>
                        <gameTime>ET00:43</gameTime>
                        <gameResult>POR 0-0 FRA </gameResult>
                        <type>2</type>
                        <location>5</location>
                        <numberOfPlayers>
                                    <team>7</team>
                                    <team>3</team>
                        </numberOfPlayers>
                        <teamNames>
                                    <teamN>Portugal</teamN>
                                    <teamN>France</teamN>
                        </teamNames>
            </frame>
            <frame>
                        <id>94</id>
                        <time>47</time>
                        <gameTime>ET00:44</gameTime>
                        <gameResult>POR 0-0 FRA </gameResult>
                        <type>1</type>
                        <playerNames>
                                    <name>Ricardo Quaresma</name>
                                    <name>Renato Sanches</name>
                        </playerNames>
                        <playerRect>
                                    <rect>
                                                <x>560</x>
                                                <y>174</y>
                                                <w>26</w>
                                                <h>26</h>
                                    </rect>
                                    <rect>
                                                <x>716</x>
                                                <y>118</y>
                                                <w>77</w>
                                                <h>77</h>
                                    </rect>
                        </playerRect>
            </frame><frame>
            ...
            </frame>
  </video>
```

Figura 3.15 – Representação da organização do ficheiro XML gerado no processamento geral.

3.4.1 XML *Query*

No projeto desenvolvido existe a possibilidade de o utilizador executar *queries* aos resultados obtidos. Porém, de modo a não restringir o output dessas *queries* a um resultado em vídeo, decidiu-se produzir um ficheiro XML onde são guardados os identificadores de cada *frame* e tempo, em segundos, do vídeo em análise (Figura 3.16). Com esta abordagem pretende-se dar ao utilizador a possibilidade de ser o próprio a escolher como aplicar os resultados obtidos, não ficando assim restringido ao vídeo que possivelmente seria gerado.

Com os valores de tempo e identificador de *frame* o utilizador poderá construir o seu próprio vídeo removendo os momentos que achar não relevantes.Por convenção quando gerado um XML de *query* ele terá o nome *Query_TIPO_DE_QUERY_XXX.xml*, sendo TIPO_DE_QUERY uma *string* que representa o tipo de *query* criada e XXX um número aleatório gerado.

De referir que as *queries* criadas para o projeto servem de exemplo, pelo que existe a possibilidade de fazer outro tipo de questões, sendo que para tal o utilizador utilize os dados presentes no XML noutro tipo de aplicação.

```
<? xml version="1.0" encoding="utf-8"?>
<queryResult>
        <type>Frame_Type</type>
        <frame>
                <id>0</id>
                <time>0</time>
        </frame>
        <frame>
                <id>1</id>
                <time>0</time>
        </frame>
        <frame>
                <id>2</id>
                <time>1</time>
        </frame>
        <frame>
                <id>3</id>
                <time>1</time>
        </frame>
        <frame>
                <id>4</id>
                <time>2</time>
        </frame>
        <frame>
                ...
        </frame>
</queryResult>
```

Figura 3.16 – Representação da organização do ficheiro XML gerado quando se executa uma *query*.

3.5 Video Highlight

No projeto anterior [1] foram gerados vídeos de *highlights* que utilizavam os dados de emoções e som de forma a ser possível detetar os momentos mais relevantes que ocorrem ao longo dum jogo de futebol.

Tendo em conta que o foco principal deste projeto foi inteiramente a componente visual do vídeo, ou seja, processamento de imagem, surgiu a possibilidade de utilizar algumas das componentes visuais extraídas e gerar também um vídeo de *highlights* só utilizados esses dados visuais.

Em termos de representação dos vários acontecimentos que ocorrem ao longo dum jogo, elementos como o som e emoções são muito mais eficazes que a imagem. A partir da imagem não é possível, de forma automática, distinguir perante dois lances elegíveis qual o mais relevante de ser incluído num vídeo curto que represente o jogo, enquanto que se combinar os valores de emoções com os valores do som torna-se possível realizar essa distinção com mais precisão.

Utilizando como base os conceitos presentes no trabalho de David Menotti e Samuel Araújo [16], decidiu-se utilizar a componente de *Camera Shot* (3.3.1) para gerar os vídeos de resumo.

Segundo os autores, percorre-se os vários *frames* do vídeo atribuindo a cada um tipo de câmara, à semelhança do implementado no algoritmo descrito na seção 3.3.1. Seguidamente, realizava-se uma procura, em pequenos intervalos, 20 em 20 *frames* ou 30 em 30 *frames* por exemplo, consoante a forma como o algoritmo de *Camera Shot* foi aplicado e a qualidade do vídeo analisado, calculando a percentagem de câmaras *Close-up* e *Out-of-Field* presentes nesses intervalos. Repete-se esse processo de forma iterativa localizando os momentos onde a percentagem ultrapassa um determinado limite.

Nos testes realizados detetou-se o melhor limite com o valor de 75% de *Close-up* e *Out-Of-Field* e 25% de *Long/Mid Shot* por intervalo.

A ideia do algoritmo é interessante uma vez que, normalmente, os momentos mais importantes durante um jogo são acompanhados por um curto período onde os produtores televisivos mostram reações dos jogadores e treinadores para além do público presente no estádio. Essa característica permite detetar os pontos de maior relevância, utilizando o algoritmo, e gerar o vídeo de resumo à semelhança do que ocorre com os dados das emoções e som.

Porém, existem algumas limitações detetadas que afetam a qualidade do vídeo gerado, como a dificuldade de escolher um intervalo em detrimento do outro, ou seja, quando dois momentos apresentam valores que estão dentro do limite definido. Porém, não existe mais nenhum elemento que indique que um intervalo poderá ser mais importante que o outro.

Por outro lado, em jogos de maior importância, como finais de competições desportivas, é normal as produções focarem-se mais nas reações do público e dos jogadores mais importantes para o jogo, o que poderá alterar a qualidade do vídeo produzido, pois o volume de *frames* com a atribuição de *Close-up* e *Out-of-Field* sobe substancialmente.

Para além disso em lances de faltas, existe normalmente um grande número de *Close-up Shot* seja pela entrada da equipa médica ou mesmo com o objetivo de mostrar as reações dos jogadores envolvidos. Apesar de serem momentos relevantes nem sempre são os mais importantes que o utilizador pretender visualizar num vídeo curto que represente o jogo.

Em suma, o algoritmo mostrou-se interessante, com resultados bastante realistas, onde lances de golo e alguns momentos mais importantes, como lances de perigo ou faltas merecedoras de cartão, foram efetivamente detetados e incluídos no vídeo de resumo. No entanto pelas razões anteriormente referidas, utilizar as emoções e som revelaram-se dados mais fiáveis, tendo em conta o objetivo de produzir um vídeo mais curto e só com os momentos mais importantes do encontro.

Se o objetivo for detetar todos os momentos de relevo, bem como momentos de faltas e sanção disciplinar por parte do árbitro então a geração do vídeo baseada neste algoritmo revela-se bastante eficaz.

3.6 Interface

Com o objetivo de o utilizador poder interagir com o sistema e utilizar todas as funcionalidades anteriormente descritas, foi criada uma interface para a aplicação utilizando a *framework* Qt, em C++. Através dos vários ecrãs o utilizador poderá visualizar e obter a informação que entender e lhe for mais conveniente a cerca do vídeo. De seguida, será feita uma apresentação dos vários componentes presentes em cada ecrã.

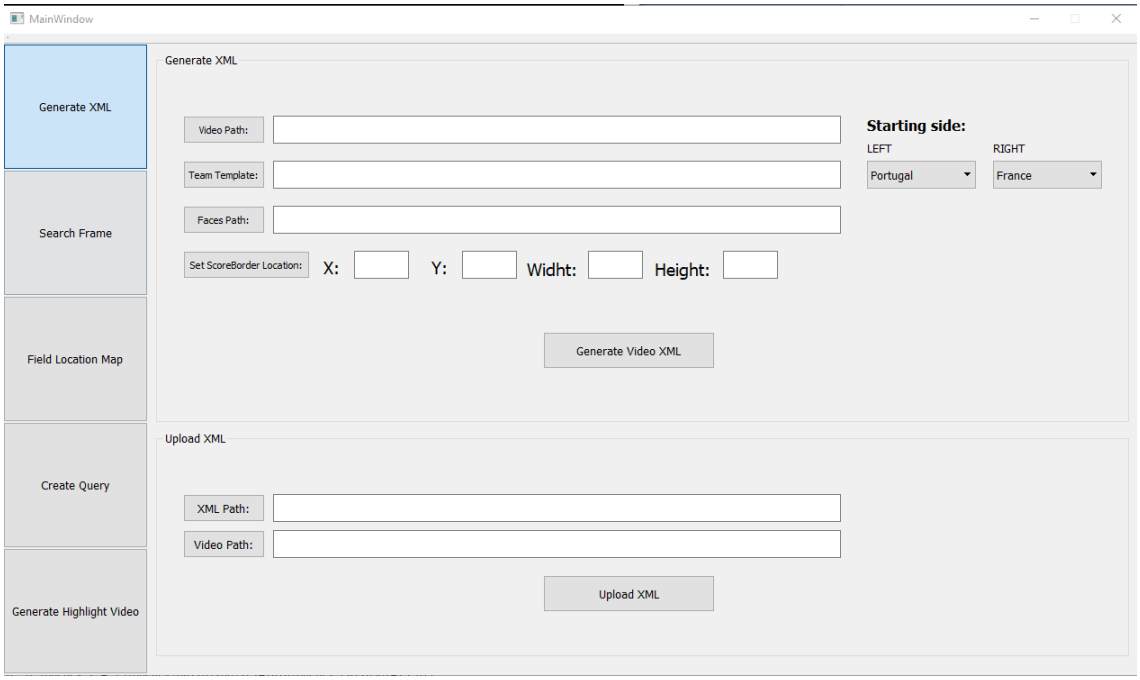


Figura 3.17- Interface responsável por criar o processamento de vídeo para um determinado jogo, ou de realizar o upload de ficheiros XML.

O ecrã principal da aplicação desenvolvida encontra-se dividida em duas partes (Figura 3.17):

- *Generate XML*, onde o utilizador necessita de inserir o vídeo a processar, duas imagens *template* que representem os equipamentos de cada equipa (onde o nome do ficheiro é o nome da equipa correspondente), localização para uma pasta com as caras dos vários jogadores envolvidos no jogo, seleção do lado do campo onde cada equipa inicia o jogo e finalmente, a localização do placard com o resultado e tempo do jogo.

Upload XML, o utilizador indica o local para o ficheiro XML e vídeo que pretende analisar e carregar para a aplicação, neste caso assume-se que o utilizador já

processou anteriormente o vídeo pretendido. Esta funcionalidade poderá ser utilizada em situações onde já se tenha gerado, anteriormente um ficheiro XML.

Existe na aplicação um ficheiro carregado automaticamente para o utilizador poder verificar as funcionalidades da aplicação sem ter necessidade de processar um vídeo, para fins de demonstração de funcionalidades.

Figura 3.18 – Interface responsável por mostrar a informação extraída para cada *frame* processado.

A partir da interface presente na Figura 3.18 pretende-se dar ao utilizador uma forma de visualizar para cada *frame* todos os detalhes extraídos, bem como fornecer um conjunto de ferramentas para ser possível selecionar um determinado *frame* e visualizar também os seus dados associados.

Com base na Figura 3.19 pretende-se dar ao utilizador uma noção, em percentagem, do número de vezes que uma determinada zona do campo foi detetada pelo algoritmo aplicado.

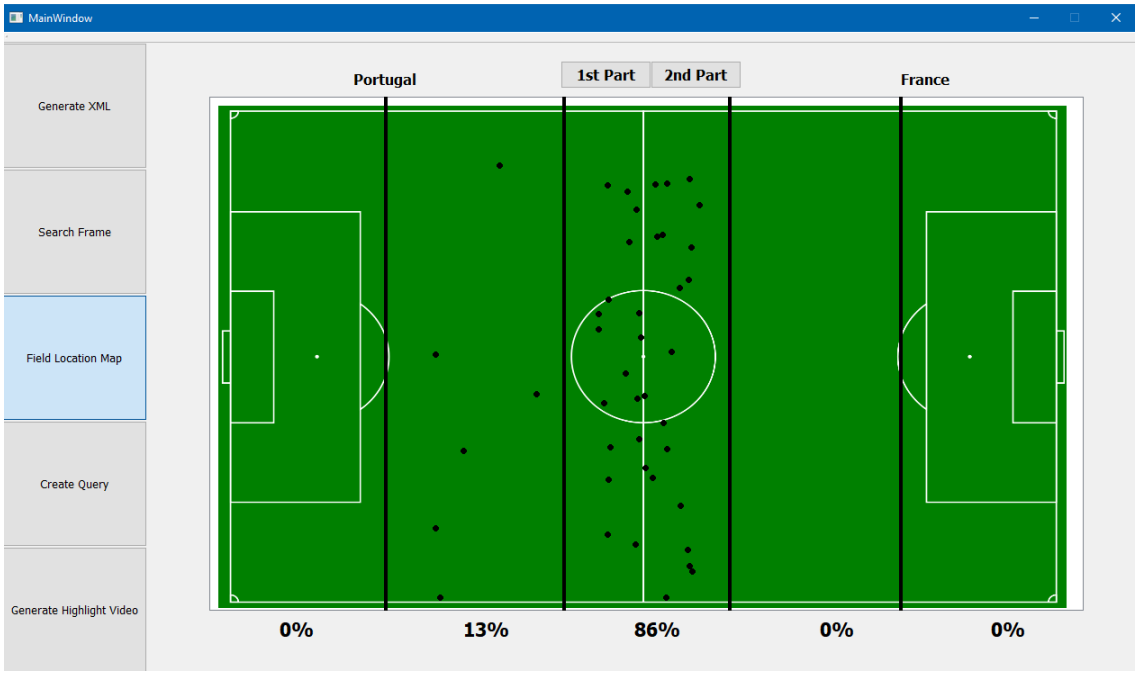


Figura 3.19 – Interface onde se pretende mostrar a distribuição das várias zonas do campo detetadas.

Por exemplo, no caso da figura 3.19, que representa o processamento dos 50 primeiros *frames* dum jogo de futebol, considerando que o processamento geral é executado neste caso de 25 em 25 *frames* e tendo em conta a qualidade do vídeo analisado (50 FPS), pode verificar-se facilmente que a maioria dos *frames* detetados foram identificados no centro do relvado, o que corresponde à realidade, uma vez que se trata dos primeiros momentos de jogo onde a bola e os jogadores se encontram no centro do campo à espera que se inicie o jogo.

A partir dos dois botões *1st Part* e *2nd Part*, o utilizador poderá ver a diferença na distribuição das zonas do campo nos primeiros 45 minutos do jogo e posteriormente nos segundos 45 minutos.

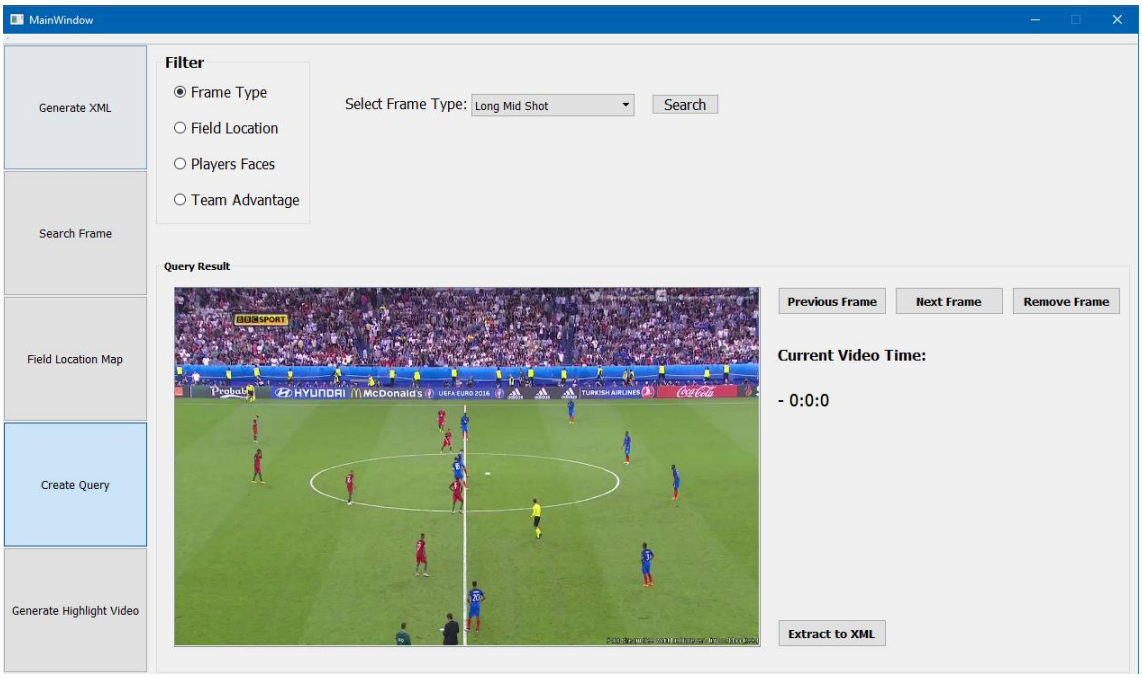


Figura 3.20 – *Create Query*, com a interface presente na figura pretende-se dar ao utilizador a possibilidade de realizar algumas *queries* pré-definidas.

Selecionaram-se 4 *queries* que o utilizador pode fazer, visualizando os respetivos *frames* onde as características selecionadas ocorrem e a partir das quais pode extrair essa informação para um ficheiro XML. Nesse ficheiro encontra-se o id de cada *frame* bem como o momento correspondente, em segundos.

Com a interface presente na Figura 3.20, o utilizador pode executar as seguintes *queries*:

- *Frame Type* – O utilizador seleciona um dos 3 tipos base de *frame* (*Long/Mid Shot, Close-up e Out-of-Field*).
- *Field Location* – Selecionando uma das 5 zonas em que o campo se encontra dividido o utilizador obtém os *frames* correspondentes.
- *Players Faces* – É fornecida uma lista com os nomes dos vários jogadores de campo e, através de uma *checklist* o utilizador escolhe os jogadores que pretende analisar, caso tenham sido detetados.
- *Team Advantage* – Neste ponto o utilizador pretende visualizar *frames* onde uma determinada equipa possui vantagem numérica em relação ao adversário.

Estas foram as *queries* exemplo definidas, porém se o utilizador pretender, recorrendo ao ficheiro XML gerado, poderá fazer outro tipo de combinações e *queries,* obtendo outro tipo de resultados com base na informação extraída.

Por exemplo, combinar deteção de determinadas zonas com modificações na componente do resultado do placard, permitindo saber o percurso executado antes dum lance de golo, ou utilizar o reconhecimento facial a seguir a detetar-se um lance de golo identificando quem esteve envolvido nesse golo.

Essas possibilidades podem ser exploradas consultando o ficheiro gerado, através doutro tipo de aplicações criadas pelos vários utilizadores que possam interpretar ficheiros XML.

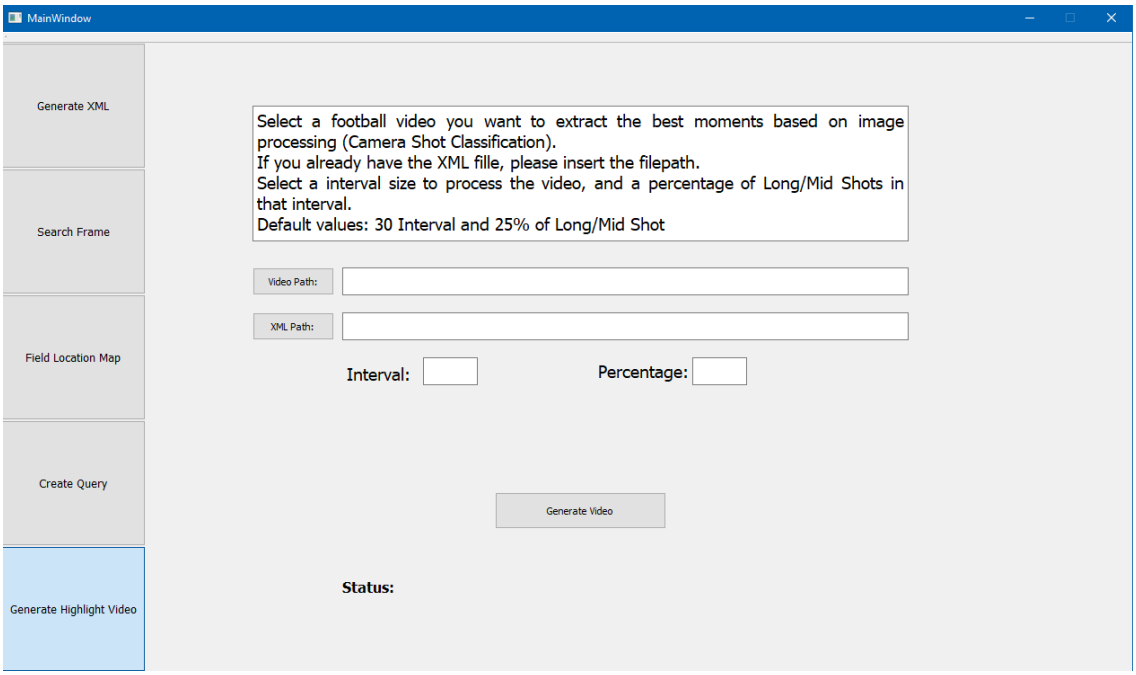


Figura 3.21 – *Generate Highlight Video,* pretende-se dar a possibilidade de gerar vídeos de resumo.

Através da interface apresentada na figura 3.21, pretende-se que o utilizador possa aplicar o algoritmo apresentado na secção 3.5, gerando o vídeo de resumo para um determinado vídeo fornecido. Para além do vídeo, no caso de já ter sido processado anteriormente, o utilizador poderá fornecer o ficheiro XML resultante desse processamento, bem como os valores do tamanho do intervalo a analisar e a percentagem de *frames Long/Mid Shot*, que serão utilizados no algoritmo de geração do vídeo de resumo.

4

Análise de Resultados

Neste capítulo será efetuada uma análise aos vários algoritmos desenvolvidos, bem como ao processamento global em termos de tempo e precisão. Pretende-se avaliar se a implementação realizada apresenta os melhores resultados possíveis. Para cada algoritmo serão analisados os seus resultados para um conjunto de *frames*. Pretende-se verificar a precisão da informação extraída por cada um dos algoritmos implementados. Com esta análise será possível validar todo o projeto desenvolvido.

Todos os resultados foram obtidos utilizando o computador Asus, modelo ROG GL702VM com as seguintes características:

- OS: Windows 10
- Processador: 2.6 GHz Intel Core i7 (4 Cores)
- RAM: 16 GB
- Storage: 250 SSD + 1 TB HDD 7200 rpm

Em relação à qualidade dos vídeos analisados apresentaram todos uma frequência de 50 FPS e uma qualidade de 720p (1280x720). O vídeo principalmente

utilizado como base de teste foi o jogo entre Portugal e França, no Euro 2016, porém outros vídeos foram também utilizados como base de comparação.

4.1 Análise de algoritmos

Nesta secção será realizada uma análise e escrutínio de resultados para cada um dos algoritmos desenvolvidos. Pretende-se discutir os vários aspetos positivos e a melhorar para cada um dos algoritmos, tendo em conta os vários resultados.

Camera Shot

Analisando 100 e 200 *frames* retiradas do inicio do jogo entre Portugal e França no Euro 2016, como se pode verificar na Tabela 4.1 e 4.2, os resultados obtidos foram bastante positivos, com taxas de precisão de 88%. Com a avaliação heurística existe sempre uma margem de erro associada, basta os valores estarem um ponto percentual acima ou a baixo de um limite para ser atribuída uma categoria diferente. Porém, apesar disso, a performance do algoritmo foi elevada, o que é bastante importante porque este algoritmo é utilizado como base nos outros processos desenvolvidos.

Na tabela 4.3 analisou-se novamente 100 *frames*, do mesmo jogo, num momento onde seria esperado detetar um número maior de *Close-up* e *Out-of-Field*, e os resultados encontram-se dentro do esperado à semelhança das Tabelas 4.1 e 4.2

Tabela 4.1: Resultados Processamento do tipo de Câmara, 100 *frames*.

Tipo de Câmara	***Close-up***	***Out-of-Field***	***Long/Mid Shot***
Esperado	16	0	84
Resultado	14	0	86

Tabela 4.2: Resultados Processamento do tipo de Câmara, 200 *frames*.

Tipo de Câmara	***Close-up***	***Out-of-Field***	***Long/Mid Shot***
Esperado	28	2	170
Resultado	25	2	173

Tabela 4.3: Resultados Processamento do tipo de Câmara, 100 *frames*.

Tipo de Câmara	*Close-up*	*Out-of-Field*	*Long/Mid Shot*
Esperado	32	7	61
Resultado	29	7	64

Field Location

Utilizando os dados presentes na Tabela 4.4, onde são analisados 84 *frames* do tipo Long/Mid Shot, pode verificar-se que os resultados obtidos pelo classificador SVM, atribuindo uma zona do campo a cada *frame*, foram bons com altas taxas de precisão. Porém, devido à dificuldade de diferenciar algumas zonas do campo, por exemplo, entre a zona 2 e zona 4 (Figura 3.7) não existe muito que as diferencie, e mesmo comparando com a zona 3 é difícil identificar características que as identifiquem claramente. Devido a estas questões será normal, nalguns casos, o classificador atribuir erradamente as zonas 2, 3 e 4.

Estas questões poderão ser resolvidas com a introdução de novas imagens base ou até mesmo pela criação de um mecanismo de treino onde o próprio utilizador verifica as atribuições feitas pelo classificador. No fundo, realizar um processo de treino intensivo de modo a aumentar a precisão do classificador implementado.

Apesar dos casos anteriores estarem relacionados com a dificuldade de diferenciar algumas zonas do campo, o classificador apresenta resultados com taxas elevadas e que permitem saber como um determinado jogo está a correr, atribuindo a cada frame uma zona do campo.

Na Tabela 4.4 é possível obter os valores gerais onde se pode concluir que apesar de existirem alguns casos onde a zona é mal atribuída, o algoritmo apresenta bons resultados e que de futuro, com um maior treino do classificador só existe espaço para melhorar.

Tabela 4.4: Resultados Processamento do tipo de zonas, 84 *frames*.

Zonas	1	2	3	4	5
Esperado	8	6	70	0	0
Resultado	4	6	68	2	4

Player Team Recognition

A partir dos dados presentes nas Tabelas 4.5, 4.6 e 4.7 que foram retirados dos *frames* presentes nas imagens 4.1, 4.2 e 4.3 respetivamente, poderá observar-se que, apesar de não se ter obtido um resultado totalmente correto, obtiveram-se resultados bastante próximos e dentro do esperado. Existiram casos onde os resultados atingiram os 100%, porém, foram selecionados estes conjuntos de forma a demonstrar algumas limitações detetadas.

Quando se analisam os dados do algoritmo implementado é necessário ter em atenção situações, como a sobreposição entre jogadores, com elementos do público e também com algumas linhas do campo, em casos onde existe um grande zoom no *frame*. Nessas situações esses jogadores não são considerados, sendo removidos do algoritmo através da análise das áreas e dimensões dos contornos obtidos, por isso é que em casos como na Tabelas 4.5 e 4.6 o número de jogadores detetados é menor do que o número de jogadores presentes no *frame*.

Para além dessas situações, existem casos onde não se consegue atribuir uma equipa ao jogador, casos como o guarda-redes ou em momentos onde o equipamento do jogador não se encontra muito visível, o que leva a um aumento de elementos pertencentes ao agrupamento de *Referee/Staff/Missmatch*.

Como se pode observar na tabela 4.7, existiram 2 valores adicionais na categoria *Referee/Staff/Missmatch*. Esse fato poderá dever-se à situação onde ocorre sobreposição de jogadores ou então devido ao jogador se apresentar de lado o que poderá levar a uma incorreta identificação do mesmo, pois torna-se difícil realizar uma comparação entre o equipamento do jogador e a imagem do mesmo devido às pequenas dimensões que apresenta.

Tendo em conta o algoritmo implementado e estas situações particulares os resultados obtidos tiveram taxas de precisão elevadas (84%) e dentro dos parâmetros esperados.

Figura 4.1- *Frame* analisado pelo algoritmo de Identificação de jogadores, tabela 4.5.

Tabela 4.5: Resultado da deteção de jogadores aplicado à Figura 4.1.

Número de Jogadores	Portugal	França	*Referee/Staff/Missmatch*
Esperado	6	8	1
Resultado	6	7	1

Figura 4.2- *Frame* analisado pelo algoritmo de Identificação de jogadores, tabela 4.6.

Tabela 4.6: Resultado da deteção de jogadores aplicado à Figura 4.2.

Número de Jogadores	Portugal	França	*Referee/Staff/Missmatch*
Esperado	6	4	3
Resultado	4	3	3

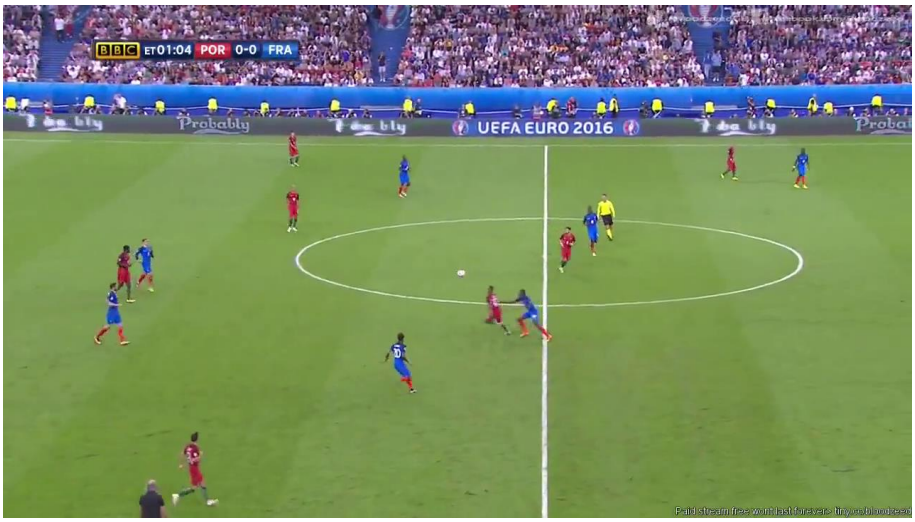


Figura 4.3- *Frame* analisado pelo algoritmo de Identificação de jogadores, tabela 4.7.

Tabela 4.7: Resultado da deteção de jogadores aplicado à Figura 4.3.

Número de Jogadores	Portugal	França	*Referee/Staff/Missmatch*
Esperado	7	7	2
Resultado	5	2	4

Player Face Identification

Ao analisar os resultados presentes nas Tabelas 4.8 e 4.9 referentes ao algoritmo de reconhecimento e identificação facial implementado, é possível concluir que o processo de reconhecimento facial apresentou uma performance positiva, detetando a grande maioria das faces presentes nos vários *frames* analisados. Re-

lativamente ao processo de identificação facial, o mesmo ficou um pouco aquém do esperado. Este processo poderá ser melhorado no futuro fornecendo uma base de dados de faces dos jogadores, para o identificador, mais rica.

No projeto desenvolvido foram fornecidas para cada jogador 4 imagens das suas faces. Se esse número for estendido para, por exemplo, 100 imagens de cada jogador com certeza que os resultados irão melhorar. Como em todos os classificadores o processo inicial de treino é muito importante para uma melhor performance do algoritmo.

Tabela 4.8: Resultado da deteção facial a um conjunto de *frames*.

Face Detection	Presentes	Reconhecidas
Resultados	53	44

Tabela 4.9: Resultado da identificação facial a um conjunto de *frames*.

Face Identification	Correto	Errado
Resultados	24	20

Scoreboard Processing

Para uma correta análise dos resultados obtidos pelo algoritmo de extração de informação presente no placard de resultados e tempo, decidiu-se realizar uma divisão dos resultados obtidos em três níveis:

- Nível 1 – Toda a informação presente no placard foi corretamente retirada e dividida em tempo e resultado;
- Nível 2 – Texto extraído com sucesso, porém existe um caracter que foi mal identificado, ou então um caracter que pertence à componente do tempo foi atribuído ao resultado; por exemplo, Tempo (01:1) e Resultado (5 POR 0-0 ESP), ou seja, o carater 5, que correspondia ao tempo 01:15, foi detetado como parte do resultado;
- Nível 3 – Situações onde o placard está presente, mas por algum motivo é considerado omisso.

Com esta classificação torna-se mais claro que os resultados obtidos foram bastante positicos, com taxas de precisão em que todo o texto foi corretamente identificado (nível 1), na ordem dos 70%. Porém alguns *frames* foram classificados níveis 2 e 3, como a qualidade da imagem, a fonte escolhida ou mesmo limitações da biblioteca *OCR Tesseract*, que não possui um grau de precisão de 100%. Para além disso, poderá haver situações onde o caracter ':' é mal identificado, o que leva a uma decisão, por parte do algoritmo de indicar que o placard não está presente. Foi uma decisão de implementação tomada pois trata-se de uma característica que existe em todos os tipos de placards, independentemente das equipas ou do resultado corrente. Apesar destes obstáculos, muitos dos resultados incompletos podem ser posteriormente acertados consultando os resultados obtidos em *frames* vizinhos.

Em suma os resultados tiveram uma taxa de precisão elevada e encontraram-se dentro do esperado.

Tabela 4.10: Resultado do algoritmo aplicado ao placard de resultados, a 100 *frames*.

Nível	1	2	3
Resultados	70	15	15

4.2 Análise processamento global

Ao realizar-se uma análise dos dados presentes na Tabela 4.11, é possível concluir que, em média, são processados dois *frames* por segundo, o valor poderá variar um pouco consoante o tipo de *frames* que estão a ser analisados no intervalo de tempo.

Frames do tipo *Close-up* são muito dispendiosos, devido ao algoritmo de reconhecimento e identificação facial, logo se existir um grande volume de *frames Close-up*, num dado intervalo é normal que existam algumas variações nos valores de tempo.

Na implementação efetuada, devido à qualidade dos vídeos analisados, onde, em média, apresentavam 50 FPS, decidiu-se realizar uma análise de 25 em 25 *frames*, pois a informação presente em cada momento não varia assim tanto que justi-

fique uma análise *frame* a *frame*, não sendo por isso necessário processar todos os *frames* de um determinado vídeo.

Sabendo do grande volume de dados e algoritmos que estão a ser aplicados durante o processamento global, os valores temporais obtidos encontram-se dentro do esperado, tendo em conta que não foi implementado nenhum mecanismo de concorrência e/ou paralelismo.

Tabela 4.11: Resultado temporal do processamento global.

Frames (#)	Tempo (s)
100	54.46
200	105.67
1000	536.45

Se forem considerados os tempos médios apresentados na Tabela 4.11, um jogo (vídeo) de 90 minutos com 50 FPS, no total seriam processados 10800 *frames*, o que corresponderia a um tempo total de processamento próximo dos 90 minutos. No fundo o tempo de processamento corresponde em média ao mesmo tempo que o vídeo possui.

5

Conclusão e Trabalho Futuro

5.1 Conclusão

Neste projeto foi desenvolvida uma aplicação que permite extrair, a partir de um jogo de futebol, um conjunto de dados que possibilitam saber quais foram os momentos mais relevantes durante o evento. Com a informação extraída, pretende-se dar ao utilizador a possibilidade de fazer pesquisas sobre o vídeo analisado, fornecendo para tal *queries* pré-definidas, mas também dando a liberdade ao utilizador de criar as suas próprias.

Este sistema foi desenvolvido utilizando as tecnologias C++ e Qt, bem como as bibliotecas OpenCV, para todo o processamento de imagem, pois trata-se de uma biblioteca bastante rica no que toca a funções e implementações com o objetivo de extrair o máximo de informação presente numa determinada imagem ou vídeo.

No capítulo referente ao trabalho relacionado, foram apresentados um conjunto de algoritmos desenvolvidos por outros investigadores, dos quais se retiraram algumas ideias principais, que posteriormente foram adaptadas e implementadas neste projeto. Não se utilizaram todos os conceitos estudados, pois muitos dos algoritmos apresentados tinham como objetivo a deteção de lances de golo ou similares, que apesar de serem momentos importantes num jogo de futebol não eram o principal foco deste projeto, porém algumas ideias base utilizadas foram aproveitadas e adaptadas para este sistema.

Foram desenvolvidos um conjunto de algoritmos com o objetivo de extrair um conjunto de informações de cada *frame* processado. Cada um desses algoritmos foi corretamente implementado e submetido a um extenso período de testes, onde se obtiveram taxas de precisão bastante elevadas.

Com os algoritmos implementados tornou-se possível atingir o objetivo proposto para este projeto. Após se processar todo o vídeo dum jogo de futebol, obteve-se o ficheiro XML, que representa a base de dados dos vários momentos e características que representam o jogo em si. Através desse ficheiro poderão ser executadas um conjunto de *queries* de forma a obter resultados mais específicos e do interesse do utilizador.

Os algoritmos desenvolvidos foram criados de modo a obter os melhores resultados possíveis dentro do tempo de desenvolvimento deste projeto. Porém, existe sempre espaço a melhorar como será apresentado na subsecção seguinte.

Em suma, pode concluir-se que todas as ideias e desafios inicialmente propostos foram atingidos com sucesso e com resultados bastante favoráveis.

5.2 Trabalho Futuro

Quando se desenvolve um projeto da área da Informática, nomeadamente em processamento de imagem, existe sempre a possibilidade de melhorar algo, seja por melhorias em termos de *hardware* ou de *software*. Por isso são referidos alguns pontos no projeto que poderão ser melhorados no futuro:

- Conversão do processo heurístico na deteção da cor para um processo de *machine learning*. Atualmente existe uma função heurística que determina o tipo de câmara de cada *frame*. Seria interessante adaptar um processo semelhante ao utilizado na deteção das zonas do campo para este processo. Seria necessário criar um classificador novo e uma extensa base de dados com diversos exemplos dos tipos de câmara.
- Criação de mecanismos de treino automatizado para o classificador das zonas do campo. O classificador criado foi treinado recorrendo a 250 imagens das várias zonas do campo. Seria interessante incluir uma interface e respetivos mecanismos de modo a treinar o classificador sempre que ele processa um resultado certo, utilizando um *input*

positivo ou negativo fornecido pelos utilizadores do sistema, permitindo assim melhorar o grau de precisão do classificador.

• Criação de um detetor facial dos jogadores. No projeto foi utilizado um detetor facial desenvolvido pela biblioteca OpenCV, porém, como demonstrado, os resultados nem sempre são os esperados. Assim, é possível desenvolver um detetor adaptado e treinado para detetar especificamente as faces dos jogadores, ou seja, ser possível detetar faces com diferentes ângulos e não só uma perspetiva mais frontal como é implementado no OpenCV.

- Detetor e classificador dos números e nomes presentes nos equipamentos dos jogadores. Para além do reconhecimento facial, seria interessante arranjar mecanismos que reconhecessem e identificassem os números e nomes dos jogadores presentes num determinado *frame*. Neste projeto ainda existiu uma tentativa de implementar tal mecanismo com o auxilio do *OCR Tesseract* só que os resultados não foram os melhores, daí esta componente não ter sido desenvolvida.
- Criação de um centro de dados com a informação relativa a equipamentos de equipas e faces dos vários jogadores. Com a criação deste centro de dados, os utilizadores deixariam de ter de fornecer as faces e equipamentos dos vários jogadores e equipas, bastando para tal só fornecer o vídeo do jogo e o local em que cada equipa inicia o jogo.
- Extensão dos conceitos para outro tipo de desportos e eventos. Seria interessante aplicar alguns dos mecanismos desenvolvidos a outro tipo de eventos desportivos e não só.

Bibliografia

[1] Guilherme Fião, Teresa Romão, Nuno Correia, Pedro Centeiro, A. Eduardo Dias*, "Automatic Generation of Sport Video Highlights Based on Fan's Emotions and Content"*, em Proceedings of the 13th International Conference on Advances in Computer Entertainment Technology, 2016.

[2] Serhiy Moskovchuk, Nuno Correia, *"Video Metadata Extraction in a Video-Mail System"*, em FCT-UNL, 2015.

[3] Insights into the #WorldCup conversation on Twitter. URL: https://blog.twitter.com/2014/insights-into-the-worldcup-conversation-on-twitter (acedido em 02/07/2017).

[4] Emotion: Theory, Research and Experience. Volume 1. Theories of Emotion. Edited by R. Plutchik and H. Kellerman. (Pp. 399; illustrated; £19.00.) Academic Press: London.1980." In: Psychological Medicine 11.01 (Feb. 1981), pp. 207–207. ISSN: 1469-8978. DOI: 10.1017/S0033291700053769. URL: http://journals.cambridge.org/article_S0033291700053769 (acedido em 02/07/2017).

[5] B. Cardoso, T. Romão, and N. Correia. "CAAT: A Discrete Approach to Emotion Assessment". In: CHI '13 Extended Abstracts on Human Factors in Computing Systems. CHI EA '13. Paris, France: ACM, 2013, pp. 1047–1052. ISBN: 978-1-4503-1952-2. DOI: 10.1145/2468356.2468543. URL: http://doi.acm.org/10.1145/2468356.2468543.

[6] Adam Coates, Blake Carpenter, Carl Case, Sanjeev Satheesh, Bipin Suresh, Tao Wang, David J. Wu, Andrew Y. Ng, *"Text Detection and Character Recognition in Scene Images with Unsupervised Feature Learning", em* 12th International Conference on Document Analysis and Recognition, 2011.

[7] Bradski,G., Kaehler,A.(2008). "Histograms and Matching", in Bradski,G., Kaehler,A.(2008). "Learning OpenCV". Gravenstein Highway Noth, Sebastopol, CA, O'Reilly Media Inc. cap.7, p.193-219.

[8] Bradski,G., Kaehler,A.(2008). "Image Transform", in Bradski,G., Kaehler,A.(2008). "Learning OpenCV". Gravenstein Highway Noth, Sebastopol, CA, O'Reilly Media Inc. cap.6, p.153-162.

[9] Bradski,G., Kaehler,A.(2008). "Image Parts and Segmentation", in Bradski,G., Kaehler,A.(2008). "Learning OpenCV". Gravenstein Highway Noth, Sebastopol, CA, O'Reilly Media Inc. cap.9, p.265-295.

[10] Gonzales, R., Woods, R. (2007). "Image Segmentation", in Gonzales, R., Woods, R. (2007). "Digital Image Processing". Upper Saddle River, NJ, USA, Pearson cap.10, p.711-784.

[11] D. Tjondronegoro, Y.P.P Chen, B. Pham, "*Sports Video Summarization using Highlights and Play-Breaks*", em MIR '03 Proceedings of the 5th ACM SIGMM international workshop on Multimedia information retrieval 2003.

[12] Jurgen Assfalg, Marco Bertini, Walter Nunziati, *"Automatic extraction and annotation of soccer video highlights"*, em International Conference on Image Processing, 2003.

[13] Cunxun Zang, Qingshan Liu, Xiaofeng Tong, Hanqing Lu, *"A Framework for Providing Adaptative Sports Video to Mobile Devices",* em MobiMedia'06, 2006.

[14] Youness Tabii, Oulad Haj Thami Rachid, *"A Framework for Soccer Video Processing and Analysis Based on Enhanced Algorithm for Dominant Color Extraction"*, em International Journal of Image Processing, Volume (3), Issue (4), 2009.

[15] Ying Yang, Shouxun Lin, Yongdong Zhang, Sheng Tang, "*Highlights Extraction in Soccer Videos based on Goal-Mouth Detection"*, em Signal Processing and Its Applications, 2007.

[16] Júnior, S. F. S, Araújo, A.A. and Menotti, D., "An overview of automatic event detection in soccer matches", em IEEE Workshop on Applications of Computer Vision (WACV 2011), 5-7 January, Kona, HI, USA, 2011, pp. 31-38.

[17] Ahmet Ekin, A. Murat Tekalp, *"Automatic Soccer Video Analysis and Summarization"*, em IEEE Transactions on Image Processing, 2003.

[18] Anil Kokaram, Niall Rea, Rozenn Dahyot, A. Murat Tekalp, Patrick Bouthemy, Patrick Gros, Ibrahim Sezan, *"Browsing Sports Video",* IEEE Signal Processing Magazine, 2006.

[19] H.M. Zawbaa, Nashwa El-Bendary, A. H. Hassanien, Tai-hoon Kim, *"Event Detection Based Approach for Soccer Video Summarization Using Machine learning"*, em International Journal of Multimedia and Ubiquitous Engineering, 2012.

[20] A. Raventós, R. Quijada, Luis Torres, Francesc Tarrés*, "Automatic summarization of soccer highlights using audio-visual descriptors",* em SpringerPlus, 2014.

[21] Sunghoon Choi, Yongduek Seo, Hyunwoo Kim, Ki-Sang Hong, *"Where are the ball and players? : Soccer Game Analysis with Color-based Tracking and Image Mosaick"*, em Compression, Hardware & Software, Image Database, Neural Networks, Object Recognition & Reconstruction, 2005.

[22] Wei-Ta Chu, Yung-Chieh Chou, *"Event Detection and Highlight Detection of Broadcasted Game Videos"*, em HCMC'15, 2015.

[23] Kongwah Wan, Xin Yan, Xinguo Yu, Changsheng Xu, *"Real-time Goal-Mouth Detection in MPEG Soccer Video",* em MULTIMEDIA'03, 2003.

[24] Young Rui, Ziyou Xiong, Regunathan Radhakrishnan, Ajay Divakaran, Thomas S. Huang, *"A Unified Framework for Video Summarization, Browsing and Retrieval"*, em Mitsubishi Electric Research Laboratories, 2004.

[25] OpenCv Library URL: http://opencv.org/, acedido em 03/03/2018.

[26] Visual Studio URL: http://visualstudio.com/, acedido em 03/03/2018.

[27] Fião, G., Romão, T., Correia, N., Centieiro, P. and Dias, A.E., Automatic Generation of Sport Video Highlights Based on Fan's Emotions and Content. In Proceedings of ACE 2016, Osaka, Japan, 9-12 November 2016, n.29.

[28] Romão, T, Centieiro, P., Fião, G., Correia, N. and Dias, A.E. Designing a System for the Automatic Generation of Sport Video Summaries. In Proc. Of EICS 2017, Lisbon, Portugal, 26-29 June 2017 (in press).

[29] S. Lee, B. Heere, and L. Chalip. "Identifying Emotions Associated with Professional Sport Team Brands". In: North American Society for Sport Management Conference 2013, pp. 390–391.

[30] P. Centieiro, B. Cardoso, T. Romão, and A. E. Dias. “If You Can Feel It, You Can Share It!: A System for Sharing Emotions During Live Sports Broadcasts”. In: Proceedings of the 11th Conference on Advances in Computer Entertainment Technology.ACE ’14. Funchal, Portugal: ACM, 2014, 15:1–15:8. ISBN: 978-1-4503-2945-3.DOI: 10.1145/2663806.2663846. URL: http://doi.acm.org/10.1145/2663806.2663846.